(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2021년10월05일
(11) 등록번호 10-2308140
(24) 등록일자 2021년09월27일

(51) 국제특허분류(Int. Cl.)
H04L 29/08 (2006.01) *H04L 12/26* (2006.01)
(52) CPC특허분류
H04L 67/02 (2013.01)
H04L 43/10 (2013.01)
(21) 출원번호 10-2017-7029007
(22) 출원일자(국제) 2016년03월09일
심사청구일자 2021년03월09일
(85) 번역문제출일자 2017년10월11일
(65) 공개번호 10-2017-0128447
(43) 공개일자 2017년11월22일
(86) 국제출원번호 PCT/US2016/021627
(87) 국제공개번호 WO 2016/145126
국제공개일자 2016년09월15일
(30) 우선권주장
62/131,619 2015년03월11일 미국(US)
(56) 선행기술조사문헌
CN103945003 A*
KR1020100070563 A*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
파세토, 인크.
미국 85251 애리조나주 스코츠데일 노스 스코츠데일 로드 4110 스위트 315
(72) 발명자
말패스, 루크
영국 스토크-온-트렌트 에스티2 7엘더블유 배그놀 로드 156
(74) 대리인
양영준, 정은진, 백만기

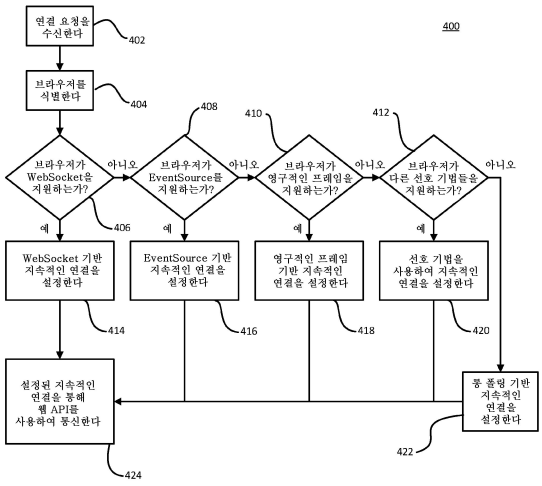
전체 청구항 수 : 총 25 항 심사관 : 황철규

(54) 발명의 명칭 **웹 API 통신을 위한 시스템 및 방법**

(57) 요 약

디바이스들 간의 통신 방법이 제공된다. 이 방법은 연결 요청을 수신하는 단계, 연결 요청에 응답하여 연결을 설정하는 단계, 및 연결을 통해 제1 웹 API 요청을 수신하는 단계를 포함한다. 이 방법은 제1 웹 API 요청에 응답하여 제1 프로시저를 실행하는 단계, 연결을 통해 제2 웹 API 요청을 수신하는 단계, 및 제2 웹 API 요청에 응답하여 제1 프로시저 또는 제2 프로시저 중 적어도 하나를 실행하는 단계를 추가로 포함한다.

대 표 도 - 도4



(52) CPC특허분류
H04L 67/141 (2013.01)
H04L 67/32 (2013.01)

명 세 서

청구범위

청구항 1

디바이스들 간의 통신 방법으로서,

서버에 의해, 제1 디바이스로부터 연결 요청을 수신하는 단계;

상기 서버에 의해, 상기 연결 요청을 하고 있는 상기 제1 디바이스상에서 실행중인 애플리케이션에 대한 식별 정보를 획득하는 단계;

상기 서버에 의해 그리고 상기 식별 정보를 체크하는 것에 의해, 상기 제1 디바이스가 Web Socket 기반 연결을 포함하는 연결의 제1 타입과 호환되지 않는다고 결정하는 단계;

상기 서버에 의해, 상기 제1 디바이스가 상기 Web Socket 기반 연결과 호환되지 않는 것에 응답하여 그리고 상기 식별 정보를 체크하는 것에 의해, 상기 제1 디바이스가 연결의 제2 타입과 호환된다고 결정하는 단계 - 상기 연결의 제2 타입은 EventSource 기반 연결, 영구적인 프레임 기반 연결(forever frame based connection), 또는 롱 폴링 지속 연결(long polling persistent connection) 중 적어도 하나를 포함함 -;

상기 서버에 의해, 상기 제1 디바이스가 상기 연결의 제2 타입과 호환된다고 결정하는 것에 응답하여, 상기 연결의 제2 타입의 연결을 설정하는 단계;

상기 서버에 의해, 상기 연결을 통해 제1 웹 API 요청을 수신하는 단계;

상기 제1 웹 API 요청에 응답하여, 상기 서버에 의해, 제1 프로시저를 실행하는 단계;

상기 서버에 의해, 상기 연결을 통해 제2 웹 API 요청을 수신하는 단계; 및

상기 제2 웹 API 요청에 응답하여, 상기 서버에 의해, 상기 제1 프로시저 또는 제2 프로시저 중 적어도 하나를 실행하는 단계를 포함하는, 통신 방법.

청구항 2

제1항에 있어서, 상기 연결의 제2 타입은 상기 EventSource 기반 연결을 포함하는, 통신 방법.

청구항 3

제2항에 있어서, 상기 연결의 제2 타입은 상기 연결의 제2 타입과 상기 연결 요청을 송신하는 제1 디바이스와의 호환성에 적어도 부분적으로 기초하여 상기 서버에 의해 선택되는, 통신 방법.

청구항 4

제2항에 있어서, 상기 연결은 상기 제1 웹 API 요청과 상기 제2 웹 API 요청 사이에 지속되는, 통신 방법.

청구항 5

제4항에 있어서, 상기 연결은 상기 제2 웹 API 요청 후에 지속되는, 통신 방법.

청구항 6

제1항에 있어서, 상기 제1 프로시저를 실행하는 것에 응답하여, 상기 서버에 의해, 상기 연결을 유지하는 단계를 추가로 포함하는, 통신 방법.

청구항 7

제1항에 있어서, 상기 연결이 미리 결정된 지속 시간 동안 유휴 상태인 것에 응답하여, 상기 서버에 의해, 상기 연결을 데이터 저장 시스템으로 스위칭하는 단계를 추가로 포함하는, 통신 방법.

청구항 8

제7항에 있어서, 상기 연결을 통해 상기 제2 웹 API 요청을 수신하는 것에 응답하여, 상기 서버에 의해, 상기 연결을 활성 메모리로 스위칭하는 단계를 추가로 포함하는, 통신 방법.

청구항 9

제7항에 있어서, 상기 데이터 저장 시스템은 데이터베이스, 데이터베이스 팜, 해시 테이블, 또는 룩업 테이블 중 적어도 하나를 포함하는, 통신 방법.

청구항 10

컴퓨터-기반 시스템으로서,

프로세서;

상기 프로세서와 통신하도록 구성된 유형의(tangible) 비일시적인 메모리를 포함하고, 상기 유형의 비일시적인 메모리는, 상기 프로세서에 의한 실행에 응답하여, 상기 프로세서로 하여금 동작들을 수행하게 하는 명령어들을 저장하고 있고, 상기 동작들은:

상기 프로세서에 의해, 통신 디바이스로부터 연결 요청을 수신하는 동작;

상기 프로세서에 의해, 상기 연결 요청을 하고 있는 상기 통신 디바이스상에서 실행중인 애플리케이션에 대한 식별 정보를 획득하는 동작;

상기 프로세서에 의해 그리고 상기 식별 정보를 체크하는 것에 의해, 상기 통신 디바이스가 Web Socket 기반 연결을 포함하는 연결의 제1 타입과 호환되지 않는다고 결정하는 동작;

상기 프로세서에 의해, 상기 통신 디바이스가 상기 Web Socket 기반 연결과 호환되지 않는 것에 응답하여 그리고 상기 식별 정보를 체크하는 것에 의해, 상기 통신 디바이스가 연결의 제2 타입과 호환된다고 결정하는 동작 - 상기 연결의 제2 타입은 EventSource 기반 연결, 영구적인 프레임 기반 연결, 또는 롱 폴링 지속 연결 중 적어도 하나를 포함함 -;

상기 프로세서에 의해, 상기 통신 디바이스가 상기 연결의 제2 타입과 호환된다고 결정하는 것에 응답하여, 상기 연결의 제2 타입에서의 상기 통신 디바이스와의 연결을 설정하는 동작;

상기 프로세서에 의해, 상기 연결을 통해 상기 통신 디바이스로부터 제1 웹 API 요청을 수신하는 동작;

상기 제1 웹 API 요청에 응답하여, 상기 프로세서에 의해, 제1 프로시저를 실행하는 동작;

상기 프로세서에 의해, 상기 제1 프로시저부터의 결과를 상기 연결을 통해 상기 통신 디바이스로 반환하는 동작;

상기 제1 프로시저부터의 결과를 상기 통신 디바이스로 반환하는 것에 응답하여, 상기 프로세서에 의해, 상기 연결을 열린 상태로 유지하는 동작;

상기 프로세서에 의해, 상기 연결을 통해 상기 통신 디바이스로부터 제2 웹 API 요청을 수신하는 동작; 및

상기 제2 웹 API 요청에 응답하여, 상기 프로세서에 의해, 상기 제1 프로시저 또는 제2 프로시저 중 적어도 하나를 실행하는 동작을 포함하는, 컴퓨터-기반 시스템.

청구항 11

제10항에 있어서, 상기 연결의 제2 타입은 상기 EventSource 기반 연결을 포함하는, 컴퓨터-기반 시스템.

청구항 12

제11항에 있어서, 상기 프로세서에 의해, 상기 통신 디바이스 상에서 실행 중인 애플리케이션과 상기 연결의 제2 타입과의 호환성에 적어도 부분적으로 기초하여 상기 연결의 제2 타입을 선택하는 동작을 추가로 포함하는, 컴퓨터-기반 시스템.

청구항 13

제11항에 있어서, 상기 연결은 상기 제1 웹 API 요청과 상기 제2 웹 API 요청 사이에 지속되는, 컴퓨터-기반 시스템.

청구항 14

제13항에 있어서, 상기 연결은 상기 제2 웹 API 요청 후에 지속되는, 컴퓨터-기반 시스템.

청구항 15

제10항에 있어서, 상기 연결이 미리 결정된 지속 시간 동안 유휴 상태인 것에 응답하여, 상기 프로세서에 의해, 상기 연결을 데이터 저장 시스템으로 스위칭하는 동작을 추가로 포함하는, 컴퓨터-기반 시스템.

청구항 16

제15항에 있어서, 상기 연결을 통해 상기 제2 웹 API 요청을 수신하는 것에 응답하여, 상기 프로세서에 의해, 상기 연결을 활성 메모리로 스위칭하는 동작을 추가로 포함하는, 컴퓨터-기반 시스템.

청구항 17

제15항에 있어서, 상기 데이터 저장 시스템은 데이터베이스, 데이터베이스 팜, 해시 테이블, 또는 룩업 테이블 중 적어도 하나를 포함하는, 컴퓨터-기반 시스템.

청구항 18

프로세서에 의한 실행에 응답하여 상기 프로세서로 하여금 동작들을 수행하게 하는 명령어들을 저장하고 있는 비일시적인, 유형의 컴퓨터 판독 가능 저장 매체를 포함하는 제조 물품으로서, 상기 동작들은:

상기 프로세서에 의해, 통신 디바이스로부터 연결 요청을 수신하는 동작;

상기 프로세서에 의해, 상기 연결 요청을 하고 있는 상기 통신 디바이스상에서 실행중인 애플리케이션에 대한 식별 정보를 획득하는 동작;

상기 프로세서에 의해 그리고 상기 식별 정보를 체크하는 것에 의해, 상기 통신 디바이스가 Web Socket 기반 연결을 포함하는 연결의 제1 타입과 호환되지 않는다고 결정하는 동작;

상기 프로세서에 의해, 상기 통신 디바이스가 상기 Web Socket 기반 연결과 호환되지 않는 것에 응답하여 그리고 상기 식별 정보를 체크하는 것에 의해, 상기 통신 디바이스가 연결의 제2 타입과 호환된다고 결정하는 동작 - 상기 연결의 제2 타입은 EventSource 기반 연결, 영구적인 프레임 기반 연결, 또는 롱 폴링 지속 연결 중 적어도 하나를 포함함 -;

상기 프로세서에 의해, 상기 통신 디바이스가 상기 연결의 제2 타입과 호환된다고 결정하는 것에 응답하여, 상기 연결의 제2 타입의 상기 통신 디바이스와의 연결을 설정하는 동작;

상기 프로세서에 의해, 상기 연결을 통해 상기 통신 디바이스로부터 제1 웹 API 요청을 수신하는 동작;

상기 제1 웹 API 요청에 응답하여, 상기 프로세서에 의해, 제1 프로시저를 실행하는 동작;

상기 제1 프로시저를 실행하는 것에 응답하여, 상기 프로세서에 의해, 상기 연결을 열린 상태로 유지하는 동작;

상기 프로세서에 의해, 상기 열린 상태의 상기 연결을 통해 상기 통신 디바이스로부터 제2 웹 API 요청을 수신하는 동작; 및

상기 제2 웹 API 요청에 응답하여, 상기 프로세서에 의해, 상기 제1 프로시저 또는 제2 프로시저 중 적어도 하나를 실행하는 동작을 포함하는, 제조 물품.

청구항 19

제18항에 있어서, 상기 연결의 제2 타입은 EventSource 기반 연결을 포함하는, 제조 물품.

청구항 20

제19항에 있어서, 상기 프로세서에 의해, 상기 통신 디바이스 상에서 실행 중인 애플리케이션과 상기 연결의 제2 타입과의 호환성에 적어도 부분적으로 기초하여 상기 연결의 제2 타입을 선택하는 동작을 추가로 포함하는,

제조 물품.

청구항 21

제19항에 있어서, 상기 연결은 상기 제1 웹 API 요청과 상기 제2 웹 API 요청 사이에 지속되는, 제조 물품.

청구항 22

제21항에 있어서, 상기 연결은 상기 제2 웹 API 요청 후에 지속되는, 제조 물품.

청구항 23

제18항에 있어서, 상기 연결이 미리 결정된 지속 시간 동안 유휴 상태인 것에 응답하여, 상기 프로세서에 의해, 상기 연결을 데이터 저장 시스템으로 스위칭하는 동작을 추가로 포함하는, 제조 물품.

청구항 24

제23항에 있어서, 상기 연결을 통해 상기 제2 웹 API 요청을 수신하는 것에 응답하여, 상기 프로세서에 의해, 상기 연결을 활성 메모리로 스위칭하는 동작을 추가로 포함하는, 제조 물품.

청구항 25

제23항에 있어서, 상기 데이터 저장 시스템은 데이터베이스, 데이터베이스 팜, 해시 테이블, 또는 룩업 테이블 중 적어도 하나를 포함하는, 제조 물품.

발명의 설명

기 술 분 야

[0001] 관련 출원에 대한 상호 참조

[0002] 이 출원은 전체가 본 명세서에 참고로 포함되는 2015년 3월 11일자로 출원된 "SYSTEMS AND METHODS FOR WEB API COMMUNICATION"이라는 명칭의 미국 가출원 일련 번호 62/131,619의 우선권을 주장한다.

[0003] 분야

[0004] 본 개시는 일반적으로 웹 API를 사용한 통신에 관한 것으로, 더 상세하게는 지속적인 연결을 통한 웹 API 통신을 위한 시스템 및 방법에 관한 것이다. 본 명세서에서 사용된 "웹 API"라는 문구는 원격 디바이스상의 미리 정의된 프로시저 세트를 호출하기 위해 이용 가능하게 된 URL들(즉, 웹 주소들)의 집합을 지칭한다.

배 경 기 술

[0005] 디바이스들 사이의 통신은 전형적으로 인터넷 또는 근거리 네트워크(local area network)와 같은 네트워크를 통하여 수행된다. 디바이스들에서 실행되는 애플리케이션들도 서로 통신할 수 있다. 상이한 애플리케이션들이 상호 작용할 수 있는 한 가지 방법은 API들을 통한 것이다. 예를 들어, 웹 API는 제3자 애플리케이션이 기존 애플리케이션에 대한 데이터를 호스팅하는 서버와 통신하는 것을 가능하게 할 수 있다.

[0006] 웹 API는 애플리케이션에게 서버에 전송된 각각의 요청에 대해 연결을 열고 닫을 것을 요구할 수 있다. 그와 관련하여, 각각의 요청은 단일의 분리되고 격리된 호출이다. 각각의 요청에 대해 연결을 열고 닫는 것과 관련된 오버헤드로 인해 애플리케이션 성능이 저하될 수 있다. 제3자 애플리케이션과 서버 사이의 통신은 예를 들어 제3자 애플리케이션을 실행하는 클라이언트에게 느리게 보일 수 있다. 통신이 느리고 및/또는 로드 시간이 길어 보이는 것은 애플리케이션을 사용할 소비자들의 외면을 받을 수 있다.

발명의 내용

[0007] 디바이스들 간의 통신 방법은 연결 요청을 수신하는 단계, 연결 요청에 응답하여 연결을 설정하는 단계, 및 연결을 통해 제1 웹 API 요청을 수신하는 단계를 포함한다. 이 방법은 상기 제1 웹 API 요청에 응답하여 제1 프로시저를 실행하는 단계, 상기 연결을 통해 제2 웹 API 요청을 수신하는 단계, 및 상기 제2 웹 API 요청에 응답하여 상기 제1 프로시저 또는 상기 제2 프로시저 중 적어도 하나를 실행하는 단계를 추가로 포함한다.

도면의 간단한 설명

[0008] 도면들과 관련하여 고려될 때, 상세한 설명과 청구항들을 참조함으로써 더 완전한 이해가 얻어질 수 있으며, 여기서 도면들 전체에 걸쳐 동일 참조 번호들은 유사한 요소들을 지칭한다.

도 1은 본 개시의 다양한 실시예에 따른 지속적인 연결을 갖는 웹 API를 구현하기 위한 시스템의 개략도를 예시한다.

도 2는 서버가 클라이언트와의 연결을 설정하고 다양한 실시예에 따른 지속적인 연결을 사용하여 서버와 클라이언트 간에 웹 API 요청들 및 응답들을 송신하는 것에 대한 프로세스를 예시한다.

도 3은 클라이언트가 서버와의 연결을 설정하고 다양한 실시예에 따른 지속적인 연결을 사용하여 서버와 클라이언트 간에 웹 API 요청들 및 응답들을 송신하는 것에 대한 프로세스를 예시한다.

도 4는 다양한 실시예에 따른 연결을 통해 API를 지원하기 위해 서버와의 지속적인 연결의 타입을 선택하기 위한 로직을 예시한다.

도 5는 다양한 실시예에 따른 시간 경과에 따른 2개의 통신 디바이스 사이의 API 통신 시스템을 예시한다.

발명을 실시하기 위한 구체적인 내용

[0009] 본 명세서의 예시적인 실시예들의 상세한 설명은 다양한 실시예를 예시적으로 도시하는 첨부 도면들 및 그림들을 참조한다. 이러한 다양한 실시예가 이 기술 분야의 통상의 기술자들이 본 개시를 실시하기에 충분할 정도로 상세히 기술되지만, 다른 실시예들이 실현될 수 있다는 점, 및 본 개시의 사상 및 범위를 벗어나지 않고 논리적 및 기계적 변경들이 이루어질 수 있다는 점을 이해해야 한다. 따라서, 본 명세서의 상세한 설명은 한정이 아닌 예시를 위해서만 제시된다. 예를 들어, 방법 또는 프로세스 설명들 중 임의의 것에서 언급된 단계들은 임의의 순서로 실행될 수 있고, 제시된 순서로 한정되지 않는다. 또한, 기능들 또는 단계들 중 임의의 것이 하나 이상의 제3자에 아웃소싱될 수 있거나 또는 하나 이상의 제3자에 의해 수행될 수 있다. 또한, 단수에 대한 임의의 언급은 복수의 실시예를 포함하고, 둘 이상의 컴포넌트에 대한 임의의 언급은 단수의 실시예를 포함할 수 있다.

[0010] 시스템들, 방법들, 및 컴퓨터 프로그램 제품들이 제공된다. 본 명세서의 상세한 설명에서, "다양한 실시예," "일 실시예," "실시예," "예시적인 실시예" 등에 대한 언급들은 설명된 실시예가 특정한 특징, 구조, 또는 특성을 포함할 수 있지만, 모든 실시예가 반드시 그 특정한 특징, 구조, 또는 특성을 포함하는 것은 아닐 수 있다는 것을 시사한다. 또한, 이러한 문구들이 반드시 동일한 실시예를 언급하는 것은 아니다. 또한, 특정한 특징, 구조, 또는 특성이 실시예와 관련하여 기술되었을 때, 명백하게 기술되든지 아니든지 간에 그러한 특징, 구조, 또는 특성을 다른 실시예와 관련하여 실행하는 것은 이 기술 분야의 통상의 기술자의 지식 범위 내임을 언급한다. 설명을 읽은 후에, 본 개시를 대안의 실시예들에서 어떻게 구현하는지는 관련 기술(들)의 통상의 기술자에게 명백할 것이다.

[0011] 전자 통신 디바이스들 간의 통신을 위한 시스템들 및 방법들이 본 명세서에 개시된다. 본 명세서에서 사용된, "통신 디바이스"는 다른 디바이스와의 통신이 가능한 임의의 디바이스를 언급할 수 있다. 예를 들어 그리고 제한 없이, 통신 디바이스는 스마트폰, PDA, 랩톱, 데스크톱 컴퓨터, 휴대폰, GPS 디바이스, 자동차 내비게이션 시스템, 무선 프린터, 또는 임의의 다른 디바이스를 언급할 수 있다.

[0012] 도 1을 참조하면, 다양한 실시예에 따른 지속적인 연결을 통한 웹 애플리케이션 프로그래밍 인터페이스(API) 통신을 위한 시스템(100)이 예시되어 있다. 시스템(100)은 제1 통신 디바이스(102)를 포함한다. 애플리케이션(104)이 제1 통신 디바이스(102)상에서 실행중이다. 시스템(100)은 또한 제2 통신 디바이스(108)를 포함한다. 제2 통신 디바이스(108)는 제2 애플리케이션(106)을 실행할 수 있다. 제1 통신 디바이스(102) 및 제2 통신 디바이스(108)는 네트워크(110)를 통해 통신할 수 있다. 애플리케이션(104)은 제2 애플리케이션(106)에 데이터를 제공하고/하거나 제2 애플리케이션에 의해 제공된 데이터를 사용할 수 있다. 유사하게, 제2 애플리케이션(106)은 제1 애플리케이션(104)에 데이터를 제공하고/하거나 제1 애플리케이션에 의해 제공된 데이터를 사용할 수 있다. 그와 관련하여, 애플리케이션(104) 및 제2 애플리케이션(106)은 서로 통신하기 위해 웹 API를 이용할 수 있다.

[0013] 웹 API의 프로시저들은 네트워크(110)를 통해 제2 통신 디바이스(108)와 통신하는 제1 통신 디바이스(102)에 의해 호출될 수 있다. 예를 들어, 제2 통신 디바이스(108)에 폴더를 요청하는 프로시저는 http://fakeurl.com/api/getfolder처럼 보일 수 있다. 제2 통신 디바이스(108)(예를 들어, 서버)가 제1 통신

디바이스(102)(예를 들어, 클라이언트)로부터 요청을 수신하면, 통신 디바이스는 요청된 데이터, 이 경우 폴더 정보를 미리 결정된 포맷(예를 들어 JSON 또는 XML)으로 제1 통신 디바이스(102)에 제공함으로써 응답할 수 있다.

[0014] 다양한 실시예에서, 애플리케이션(104)은 제2 애플리케이션(106)과 인터페이스하기 위해 웹 API를 사용할 수 있다. 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)은 API 요청을 네트워크(110)를 통해 제2 애플리케이션(106)을 실행중인 제2 통신 디바이스(108)로 전송할 수 있다. 제2 애플리케이션(106)은 API 요청을 수신하고 처리할 수 있다. API 요청을 수신하는 것에 응답하여, 제2 애플리케이션(106)은 수신된 특정 API 요청에 대응하는 프로시저를 실행할 수 있다.

[0015] API 요청들을 송신 및 수신하기 위해, 아래에 더 상세히 설명되는 바와 같이, 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 사이에 연결이 설정된다. 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 사이의 지속적인 연결을 유지함으로써, 시스템(100)은 모든 API 요청에 대해 연결을 열고 닫는 시스템보다 더 빨리 API 요청들을 완료할 수 있다.

[0016] 도 2를 참조하면, 다양한 실시예에 따른 제1 통신 디바이스(102)에 의해 송신되고 제2 통신 디바이스(108)에 의해 수신되는 웹 API 요청들을 처리하기 위한 프로세스(200)가 예시되어 있다. 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)은 제2 통신 디바이스(108)상에서 실행중인 제2 애플리케이션(106)에 데이터를 제공하거나 그로부터 데이터를 취득하도록 프로그래밍될 수 있다. 제2 통신 디바이스(108)는 제1 통신 디바이스(102)로부터 연결 요청을 수신할 수 있다(단계 202). 애플리케이션(104)은 예를 들어 서버로서 기능하는 제2 통신 디바이스(108)에 연결 요청을 제출하는 웹 클라이언트일 수 있다. 요청을 수신하는 것에 응답하여, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)와의 연결을 설정할 수 있다(단계 204). 일단 설정되면, 연결은 무한정 유지될 수 있다.

[0017] 다양한 실시예에서, 제2 통신 디바이스(108)는 전송되는 데이터를 갖는 활성 연결들과 일시적으로 유휴 상태였던 활성 연결들 사이의 컨텍스트 스위칭에 의해 그러한 디바이스들 상에 통상적으로 고유한 한계들을 초과하는 다수의 활성 연결을 유지할 수 있다. 제2 통신 디바이스(108)는 유휴 상태였던 활성 연결들을 SQL 데이터베이스, 데이터베이스 팜, 해시 테이블, 또는 연결 정보를 저장하는 임의의 다른 수단과 같은 데이터 저장 시스템으로 스위칭할 수 있다. 유휴 상태였고 저장소에 있는 활성 연결에 대한 정보가 수신되면, 활성 연결은 저장소로부터 활성 메모리 내로 스위칭될 수 있다.

[0018] 본 명세서에서 논의되는 임의의 데이터베이스들은 관계형, 계층 구조, 그래픽, 또는 개체 지향 구조 및/또는 임의의 다른 데이터베이스 구성들을 포함할 수 있다. 데이터베이스를 구현하는 데 사용할 수 있는 공통 데이터베이스 제품들은 IBM(뉴욕주, 아몽크)에 의한 DB2, Oracle Corporation(캘리포니아주, 레드우스 쇼어스)로부터 출시된 다양한 데이터베이스 제품들, Microsoft Corporation(워싱턴주, 레드몬드)에 의한 Microsoft Access 또는 Microsoft SQL Server, MySQL AB(스웨덴, 웁살라)에 의한 MySQL, 또는 임의의 다른 적합한 데이터베이스 제품을 포함한다. 또한, 데이터베이스들은 임의의 적합한 방식으로, 예를 들어, 데이터 테이블들 또는 룩업 테이블들로서 조직될 수 있다. 각각의 레코드는 단일 파일, 일련의 파일들, 링크된 일련의 데이터 필드들 또는 임의의 다른 데이터 구조일 수 있다. 특정 데이터의 연관(association)은 본 기술 분야에서 공지된 또는 실시된 것들과 같은 임의의 원하는 데이터 연관 기법을 통해 달성될 수 있다. 예를 들어, 연관은 수동으로 또는 자동으로 달성될 수 있다. 자동 연관 기법들은 예를 들어, 데이터베이스 검색, 데이터베이스 병합, GREP, AGREP, SQL, 테이블들에서 키 필드를 이용하여 검색 속력을 높이는 것, 모든 테이블들과 파일들을 통한 순차적인 검색들, 룩업을 간단화하기 위해 공지된 순서에 따라 파일 내의 레코드들을 소팅하는 것, 및/또는 기타 등등을 포함할 수 있다. 연관 단계는 예를 들어, 미리 선택된 데이터베이스들 또는 데이터 섹터들 내의 "키 필드(key field)"를 이용하는 데이터베이스 병합 기능에 의해 달성될 수 있다. 다양한 데이터베이스 튜닝 단계들이 데이터베이스 성능을 최적화하기 위해 고려된다. 예를 들어, 인덱스들과 같은 빈번히 사용되는 파일들이 입력/출력("I/O") 병목 상태를 감소시키기 위해 별도의 파일 시스템들에 배치될 수 있다.

[0019] 이 기술 분야의 통상의 기술자는 또한, 보안의 이유로, 임의의 데이터베이스, 시스템, 디바이스, 서버 또는 시스템의 다른 컴포넌트들이 하나의 위치 또는 다수의 위치에 이들의 임의의 조합으로 이루어질 수 있다는 점을 이해할 것이며, 각각의 데이터베이스 또는 시스템은 방화벽, 액세스 코드, 암호화, 복호화, 압축, 압축 해제, 및/또는 기타 등등과 같은 다양한 적합한 보안 특징들 중 임의의 것을 포함한다.

[0020] 암호화는 본 기술 분야에서 지금 이용 가능한 또는 이용 가능하게 될 수 있는 기법들 중 임의의 기법 - 예를 들어, Twofish, RSA, El Gamal, Schorr 시그니처, DSA, PGP, PKI, GPG(GnuPG), 및 대칭 및 비대칭 암호체계 - 에

의해 수행될 수 있다.

[0021] 다양한 실시예에서, 그 후 제2 통신 디바이스(108)는 설정된 연결을 통해 제1 API 요청을 수신할 수 있다(단계 206). API 요청을 수신하는 것에 응답하여, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)와의 설정된 연결을 통해 데이터를 반환한다(단계 208). 데이터를 반환한 후에, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)와의 설정된 연결을 유지할 수 있다(단계 210). 연결은 도 4를 참조하여 아래에 더 상세히 논의되는 다양한 방법을 사용하여 유지될 수 있다.

[0022] 다양한 실시예에서, 제2 통신 디바이스(108)는 설정된 연결을 통해 제2 웹 API 요청을 수신할 수 있다(단계 212). 제2 웹 API 요청은 제2 통신 디바이스(108)와 제1 통신 디바이스(102) 간에 추가적인 연결이 설정되는 일 없이 수신될 수 있다. 제2 웹 API 요청을 수신하는 것에 응답하여, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)로 설정된 연결을 통해 데이터를 반환한다(단계 214).

[0023] 도 3을 참조하면, 다양한 실시예에 따른 제1 통신 디바이스(102)로부터 제2 통신 디바이스(108)로 API 요청들을 송신하는 프로세스(300)가 예시되어 있다. 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)은 제2 통신 디바이스(108)에 API 요청을 전송함으로써 제2 통신 디바이스(108)상에서 실행중인 제2 애플리케이션(106)에 데이터를 제공하거나 그로부터 데이터를 취득하도록 프로그래밍될 수 있다. 제1 통신 디바이스(102)는 제2 통신 디바이스(108)에 연결 요청을 송신할 수 있다(단계 302). 애플리케이션(104)은 예를 들어 서버로서 기능하는 제2 통신 디바이스(108)에 연결 요청을 제출하는 웹 클라이언트일 수 있다. 요청을 수신하는 것에 응답하여, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)와의 연결을 설정할 수 있다(단계 304). 그 후 제2 통신 디바이스(108)는 설정된 연결을 통해 제1 API 요청을 송신할 수 있다(단계 306). API 요청을 수신하는 것에 응답하여, 제1 통신 디바이스(102)는 설정된 연결을 통해 데이터를 수신한다(단계 308). 데이터를 수신한 후에, 제1 통신 디바이스(102)는 제2 통신 디바이스(108)와의 설정된 연결을 유지할 수 있다(단계 310). 연결은 도 4 및 도 5를 참조하여 아래에 더 상세히 논의되는 다양한 방법을 사용하여 유지될 수 있다.

[0024] 다양한 실시예에서, 제1 통신 디바이스(102)는 설정된 연결을 통해 제2 웹 API 요청을 송신할 수 있다(단계 312). 제2 웹 API 요청은 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 간에 추가적인 연결을 설정하지 않고 전송될 수 있다. 제2 웹 API 요청을 송신하는 것에 응답하여, 제1 통신 디바이스(102)는 제2 통신 디바이스(108)로부터 설정된 연결을 통해 데이터를 수신한다(단계 314).

[0025] 웹 API 요청들을 수신하고(도 2) 웹 API 요청들을 송신하는(도 3) 방법들은 예를 들어 웹 사이트에 거의 즉각적인 응답 시간을 제공하기 위해 사용될 수 있다. 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 사이의 연결이 설정된 후에, 후속 웹 API 호출들은 더 이상 연결을 설정하는 오버헤드를 부과하지 않는다. 따라서, 웹 API 호출의 결과는 감소된 대기 시간으로 수행된다. 그와 관련하여, 지속적인 연결을 사용하는 웹 API 호출들은 많은 경우에 사용자들에게 외견상 즉각적으로 보일 수 있다.

[0026] 도 4를 참조하면, 도 1의 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 간에 지속적인 연결을 설정하는 프로세스(400)가 도시되어 있다. 제2 통신 디바이스(108)는 웹 클라이언트로서 동작중일 수 있는 제1 통신 디바이스(102)와 상호 작용하면서 서버로서 동작중일 수 있다. 제2 통신 디바이스(108)는 제1 통신 디바이스(102)로부터 연결 요청을 수신할 수 있다(단계 402). 요청에 응답하여, 제2 통신 디바이스(108)는 연결 요청을 하고 있는 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)(예를 들어, 브라우저)을 식별할 수 있다(단계 404). 제2 통신 디바이스(108)는 예를 들어 디바이스 식별자들, 운영 체제 식별자들, 및/또는 애플리케이션 식별자들을 사용하여 브라우저를 식별할 수 있다. 브라우저 식별자 및 디바이스 식별자가 연결 요청과 함께 제공될 수 있다. 예를 들어, 제2 통신 디바이스(108)는 연결 요청과 함께 브라우저 이름, 브라우저 버전, 운영 체제 이름, 및 운영 버전을 수신할 수 있다. 제2 통신 디바이스(108)는 또한 제2 통신 디바이스(108)의 구성에 따라 연결 요청을 수신하는 것에 응답하여 브라우저 이름, 브라우저 버전, 운영 체제 이름, 및 운영 버전을 요청할 수도 있다.

[0027] 일단 제2 통신 디바이스(108)가 식별 정보를 가지면, 제2 통신 디바이스(108)는 식별 정보를 체크하여 어떤 타입의 연결을 설정할지를 결정할 수 있다. 제2 통신 디바이스(108)는 식별된 브라우저가 WebSocket 연결을 지원하는지를 체크함으로써 시작될 수 있다(단계 406). WebSocket은 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 간에 양방향 통신을 제공하기 위한 프로토콜이다. WebSocket 연결은 WebSocket 연결을 처리하도록 코딩되는 최신 브라우저들과만 호환될 수 있다. WebSocket은 지속적인 연결을 위한 깨끗한 전용 솔루션을 제공하지만, 사용중인 많은 디바이스들은 WebSocket 연결과 호환되지 않는다. 따라서, 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 WebSocket 연결과 호환된다면, 제2 통신 디바이스(108)는 제1 통신 디바이스

(102)와의 WebSocket 기반 지속적인 연결을 설정할 것이다(단계 414). 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 WebSocket 연결과 호환되지 않는다면, 제2 통신 디바이스(108)는 다른 호환되는 연결 타입들을 체크할 수 있다.

[0028] 제2 통신 디바이스(108)는 또한 식별된 브라우저가 EventSource(즉, 서버-전송 이벤트) 연결을 지원하는지를 체크할 수 있다(단계 408). EventSource 연결도 서버가 클라이언트에 통신을 푸시하는 것을 가능하게 하여, 양방향 통신을 가능하게 한다. EventSource 연결은 WebSocket 연결과는 상이한 브라우저 및 디바이스의 목록과 호환될 수 있다. 따라서, 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 EventSource 연결과 호환된다면, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)와의 EventSource 기반 지속적인 연결을 설정할 것이다(단계 416). 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 EventSource 연결과 호환되지 않는다면, 제2 통신 디바이스(108)는 다른 호환되는 연결 타입들을 체크할 수 있다.

[0029] 그 후 제2 통신 디바이스(108)는 식별된 브라우저가 영구적인 프레임 연결을 지원하는지를 체크할 수 있다(단계 410). 영구적인 프레임 연결은 서버가 클라이언트에 통신을 푸시하는 것을 가능하게 하여, 양방향 통신을 가능하게 하는 또 다른 기술이다. 영구적인 프레임 연결은 브라우저 내에 숨겨진 iframe 요소를 생성하고 숨겨진 프레임 내부에서 연결을 설정함으로써 설정될 수 있다. 영구적인 프레임 연결은 WebSocket 및 EventSource 연결과는 상이한 브라우저 및 디바이스의 목록과 호환될 수 있다. 따라서, 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 영구적인 프레이밍과 호환된다면, 제2 통신 디바이스(108)는 제1 통신 디바이스(102)와의 영구적인 프레임 기반 지속적인 연결을 설정할 것이다(단계 418). 제1 통신 디바이스(102)에서 실행중인 애플리케이션(104)이 영구적인 프레임 연결과 호환되지 않는다면, 제2 통신 디바이스(108)는 다른 호환되는 연결 타입들을 체크할 수 있다.

[0030] 다양한 실시예에서, 제2 통신 디바이스(108)가 지속적인 연결을 설정하기 위해 어떤 연결 기법을 사용할지를 알기 위해 추가적인 선호 기법들이 체크에 통합될 수 있다. 이를 위해, 제2 통신 디바이스는 식별된 브라우저가 또 다른 선호 연결 기법을 지원하는지를 체크할 수 있다(단계 412). 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 선호 연결 기법과 호환된다면, 제2 통신 디바이스(108)는 선호 기법을 사용하여 제1 통신 디바이스(102)와의 지속적인 연결을 설정할 것이다(단계 420). 제1 통신 디바이스(102)상에서 실행중인 애플리케이션(104)이 선호 연결 기법과 호환되지 않는다면, 제2 통신 디바이스(108)는 디폴트 연결을 사용할 수 있다. 도 4에 예시된 바와 같은 디폴트 연결은 롱 폴링(long-polling) 기반 지속적인 연결을 설정하는 것을 포함한다(단계 422).

[0031] 도 1을 간략히 참조하면, 일단 제1 통신 디바이스(102)와 제2 통신 디바이스(108) 간에 지속적인 연결이 설정되면, 애플리케이션(104)은 감소된 오버헤드로 네트워크(110)를 통해 제2 애플리케이션(106)과 통신할 수 있다. 도 5를 참조하면, 제1 통신 디바이스(102)와 제2 통신 디바이스(108)상에서 각각 실행중인, 애플리케이션(104)과 제2 애플리케이션(106) 간의 지속적인 연결을 사용하는 API 통신 시스템(500)이 시간 경과에 따라 도시되어 있다. 예시된 바와 같이, 타임라인(T)상의 위치가 좌측에서 우측으로 이동함에 따라 시간이 선형적으로 지나간다. 애플리케이션(104)은 연결을 위한 요청(502)을 제2 애플리케이션(106)에 송신할 수 있다. 애플리케이션(104) 및 제2 애플리케이션(106)은 기간 P_0가 경과한 후에 제2 애플리케이션(106)으로부터 애플리케이션(104)으로의 통신(504)으로 연결 프로세스를 완료할 수 있다. 연결을 설정하기 위한 전형적인 기간 P_0는 통신 디바이스들 간의 대기 시간에 따라 약 50ms에서 500ms까지 지속될 수 있다.

[0032] 연결이 설정된 후에, 제1 통신 디바이스(102) 및 제2 통신 디바이스(108)에서 실행중인 애플리케이션들은 설정된 지속적인 연결을 통해 웹 API를 사용하여 통신할 수 있다. 애플리케이션(104)은 API 요청(506)을 제2 애플리케이션(106)에 전송한다. 제2 애플리케이션(106)은 API 요청(506)에 대응하는 기능을 로컬로 실행한 다음 대응하는 데이터(508)를 반환할 수 있다. API 요청(506)을 송신하는 것부터 대응하는 데이터(508)를 반환하는 것까지의 기간 P_1은 예를 들어 대략 12ms일 수 있다. 대응하는 데이터(508)를 반환한 후 10ms 이내에 연결이 닫히면, 애플리케이션(104)과 제2 애플리케이션(106) 간의 단일 API 통신에 대한 오버헤드는 $P_0 + P_1 + 10ms$가 될 것이다. 그러나, (예를 들어, 도 4에 예시된 바와 같이) 지속적인 연결이 설정되었기 때문에, 제2 애플리케이션(106)이 대응하는 데이터(508)를 반환한 후에도 연결은 열린 채로 유지된다.

[0033] 그 후 애플리케이션(106)을 실행중인 제2 통신 디바이스(108)는 정보(510)가 이용 가능하게 됨에 따라 설정된 지속적인 연결을 통해 애플리케이션(104)에 정보(510)를 푸시할 수 있다. 예를 들어, 애플리케이션(104)은 제2 애플리케이션(106)을 통해 채팅 메시지들을 송신 및 수신하는 텍스트 기반 채팅 애플리케이션일 수 있다. 정보

(510)는 사용자가 애플리케이션(104)을 향한 채팅 메시지를 타이핑하고 있음을 나타낼 수 있다. 제2 애플리케이션(106)은 애플리케이션(104)이 사용자가 메시지를 타이핑하고 있다는 표시자를 디스플레이할 수 있도록 애플리케이션(104)에 정보(510)를 제공할 수 있다. 정보(510)는 애플리케이션(104)이 정보(510)를 직접 요청하는 일 없이 애플리케이션(104)에 전달된다. 그와 관련하여, API 통신 시스템(500)은 애플리케이션(104)이 정보(510)를 수신하기 위해 초래하게 되는 오버헤드를 제거할 수 있는데, 그 이유는 애플리케이션(104)이 정보(510)를 요청하기 위해 (매번 연결 오버헤드 기간 P_0를 초래하는) 새로운 연결을 반복적으로 설정하기보다는 설정된 지속적인 연결을 통해 단순히 청취할 수 있기 때문이다.

[0034] 그 후 애플리케이션(104)은 설정된 지속적인 연결을 갖는 API 통신 시스템(500)을 사용하여 제2 API 요청(512)을 제2 애플리케이션(106)에 송신할 수 있다. 애플리케이션(104)은 새로운 연결을 설정하지 않고(그리고 연결을 설정하기 위한 오버헤드 기간 P_0를 초래하지 않고) 제2 API 요청(512)을 송신할 수 있다. 제2 애플리케이션(106)은 제2 API 요청(512)을 처리하고 기간 P_2 후에 결과(514)를 반환할 수 있다. 기간 P_2는 예를 들어 대략 14ms의 지속 시간을 가질 수 있다. 연결은 애플리케이션(104)과 제2 애플리케이션(106) 사이의 추가 통신을 기다리는 동안, 기간 P4 동안 열린 채로 유지될 수 있다. 제2 통신 디바이스(108)는 설정된 연결을 통해 정보를 얻기 위해 애플리케이션(104)에 요청(516)을 송신할 수 있다. 그와 관련하여, 서버는 클라이언트에 직접 정보를 요청함으로써 클라이언트와 통신할 수 있다. 그 후 클라이언트는 기간 P_3 후에 요청된 정보(518)를 서버에 제공할 수 있다. 제2 통신 디바이스(108)에 의해 요청된 정보는 예를 들어 지리적 위치 또는 현지 시간일 수 있다. (도 2 내지 도 4를 참조하여 기술된 바와 같은) 지속적인 연결은 API를 사용하여 서버와 클라이언트 사이의 전형적인 통신에 부수적인 오버헤드를 감소시킴으로써 서버와 클라이언트 사이의 거의 즉각적인 통신을 가능하게 한다. 특히, 지속적인 연결은 적어도 연결을 설정하기 위해 전형적으로 사용되는 기간 P_0만큼 주어진 통신에 대한 시간 오버헤드를 감소시킬 수 있다.

[0035] 다양한 실시예에서, 본 명세서에 설명된 방법들은 본 명세서에 설명된 다양한 특정 머신들을 사용하여 구현된다. 본 명세서에 설명된 방법들은 이 기술 분야의 통상의 기술자에 의해 즉시 이해될 바와 같이, 하기의 특정 머신들 및 이후 개발되는 것들을 사용하여 임의의 적합한 조합으로 구현될 수 있다. 또한, 본 개시로부터 명백한 바와 같이, 본 명세서에 설명된 방법들은 특정 물품들의 다양한 변환들을 야기할 수 있다.

[0036] 간결함을 위해, 시스템들(그리고 시스템들의 개개의 동작 컴포넌트들 중의 컴포넌트들)의 종래의 데이터 네트워킹, 애플리케이션 개발, 및 다른 기능 양태들은 본 명세서에서 상세히 설명되지 않을 수 있다. 또한, 본 명세서에 포함된 다양한 도면에 도시된 연결 라인들은 다양한 요소 사이의 예시적인 기능적 관계 및/또는 물리적 결합을 나타내고자 하는 것이다. 많은 대안적인 또는 추가적인 기능적 관계 또는 물리적 연결이 실제 시스템에 존재할 수 있음에 유의해야 한다.

[0037] 본 명세서에서 논의되는 다양한 시스템 컴포넌트들은 다음 중 하나 이상을 포함할 수 있다: 디지털 데이터를 처리하는 프로세서를 포함하는 호스트 서버 또는 다른 컴퓨팅 시스템들; 디지털 데이터를 저장하기 위해 프로세서에 결합된 메모리; 디지털 데이터를 입력하기 위해 프로세서에 결합된 입력 디지타이저; 메모리에 저장된 그리고 프로세서에 의한 디지털 데이터의 처리를 지시하기 위해 프로세서에 의해 액세스 가능한 애플리케이션 프로그램; 프로세서에 의해 처리된 디지털 데이터로부터 유도된 정보를 표시하기 위해 상기 프로세서 및 메모리에 연결된 디스플레이 디바이스; 및 복수의 데이터베이스. 본 명세서에서 사용된 다양한 데이터베이스는 연결 데이터 및/또는 시스템의 동작에 유용한 유사한 데이터를 포함할 수 있다. 이 기술 분야의 통상의 기술자들이 이해할 것인 바와 같이, 사용자 컴퓨터는 운영 체제(예를 들어, Windows NT, Windows 95/98/2000, Windows XP, Windows Vista, Windows 7, Windows 8, OS2, UNIX, Linux, Solaris, MacOS, 등)뿐만 아니라 컴퓨터들과 전형적으로 관련되는 다양한 종래의 지원 소프트웨어 및 드라이버들을 포함할 수 있다.

[0038] 본 시스템 또는 그것의 임의의 부분(들) 또는 기능(들)은 하드웨어, 소프트웨어, 또는 이들의 조합을 사용하여 구현될 수도 있고, 하나 이상의 컴퓨터 시스템 또는 다른 처리 시스템들에서 구현될 수도 있다. 그러나, 실시예들에 의해 수행되는 조작들은 사람 조작자에 의해 수행되는 정신적 작업들과 통상 관련되는 매칭 또는 선택 등의 용어들로 종종 언급되었다. 본 명세서에 설명된 동작들 중 임의의 동작에 있어서 대부분의 경우에, 이러한 사람 조작자의 능력이 반드시 필요하거나, 바람직한 것은 아니다. 오히려, 동작들은 머신 동작들일 수 있다. 다양한 실시예를 수행하기 위한 유용한 머신들은 범용 디지털 컴퓨터들 또는 유사한 디바이스들을 포함한다.

[0039] 사실, 다양한 실시예에서, 실시예들은 본 명세서에 설명된 기능을 수행할 수 있는 하나 이상의 컴퓨터 시스템에

관한 것이다. 컴퓨터 시스템은 하나 이상의 프로세서를 포함한다. 프로세서는 통신 인프라스트럭처(예를 들어, 통신 버스, 크로스오버 바아, 또는 네트워크)에 연결된다. 다양한 소프트웨어 실시예들이 이 예시적인 컴퓨터 시스템에 관하여 설명된다. 이러한 설명을 읽은 후에, 다양한 실시예를 다른 컴퓨터 시스템들 및/또는 아키텍처들을 사용하여 구현하는 방법은 관련 기술 분야(들)의 통상의 기술자에게 명백해질 것이다. 컴퓨터 시스템은 디스플레이 유닛 상에서의 표시를 위해 통신 인프라스트럭처로부터(또는 도시되지 않은 프레임 버퍼로부터) 그래픽, 텍스트, 및 다른 데이터를 전달하는 디스플레이 인터페이스를 포함할 수 있다.

[0040] 컴퓨터 시스템은 또한 메인 메모리, 예컨대 랜덤 액세스 메모리(RAM)를 포함하고, 보조 메모리를 포함할 수도 있다. 보조 메모리는, 예를 들어, 하드 디스크 드라이브 및/또는 플로피 디스크 드라이브, 자기 테이프 드라이브, 광학 디스크 드라이브, 등을 나타내는 이동식 저장 드라이브를 포함할 수 있다. 이동식 저장 드라이브는 잘 알려진 방식으로 이동식 저장 유닛으로부터 판독 및/또는 이동식 저장 유닛에 기입한다. 이동식 저장 유닛은 이동식 저장 드라이브에 의해 판독되고 기입되는 플로피 디스크, 자기 테이프, 광학 디스크, 솔리드 스테이트 디스크(solid state disk) 등을 나타낸다. 이동식 저장 유닛은 컴퓨터 소프트웨어 및/또는 데이터가 저장되어 있는 컴퓨터 사용 가능 저장 매체를 포함한다는 것을 이해할 것이다.

[0041] 다양한 실시예에서, 보조 메모리는 컴퓨터 프로그램 또는 다른 명령어들이 컴퓨터 시스템으로 로딩되도록 해주는 다른 유사 디바이스들을 포함할 수 있다. 이러한 디바이스들은, 예를 들어, 이동식 저장 유닛 및 인터페이스를 포함할 수 있다. 그러한 것의 예들로는 프로그램 카트리지 및 카트리지 인터페이스(예를 들어 비디오 게임 디바이스들에서 발견되는 것 등), 이동식 메모리 칩(예를 들어 EPROM(erasable programmable read only memory) 또는 PROM(programmable read only memory)) 및 관련 소켓, 및 다른 이동식 저장 유닛들 및 인터페이스들을 포함할 수 있고, 이들은 소프트웨어 및 데이터가 이동식 저장 유닛으로부터 컴퓨터 시스템으로 전송되도록 해준다.

[0042] 컴퓨터 시스템은 또한 통신 인터페이스를 포함할 수 있다. 통신 인터페이스는 소프트웨어 및 데이터가 컴퓨터 시스템과 외부 디바이스들 사이에 전송되도록 해준다. 통신 인터페이스의 예들로는 모뎀, 네트워크 인터페이스(예를 들어 이더넷 카드 등), 통신 포트, PCMCIA(Personal Computer Memory Card International Association) 슬롯 및 카드, 등등을 포함할 수 있다. 통신 인터페이스를 통해 전송되는 소프트웨어 및 데이터는 통신 인터페이스에 의해 수신될 수 있는 전자, 전자기, 광학 또는 다른 신호들일 수 있는 신호들의 형태로 이루어질 수 있다. 이 신호들은 통신 경로(예를 들어, 채널)를 통해 통신 인터페이스에 제공된다. 이 채널은 신호들을 전달하고, 와이어, 케이블, 광 섬유, 전화선, 셀룰러 링크, 무선 주파수(RF) 링크, 무선 및 다른 통신 채널들을 사용하여 구현될 수 있다.

[0043] 용어들 "컴퓨터 프로그램 매체" 및 "컴퓨터 사용 가능 매체"는 일반적으로 이동식 저장 드라이브 등의 매체 및 하드 디스크 드라이브에 설치된 하드 디스크를 언급하기 위해 사용된다. 이러한 컴퓨터 프로그램 제품들은 소프트웨어를 컴퓨터 시스템에 제공한다.

[0044] 컴퓨터 프로그램들(컴퓨터 제어 로직이라고도 언급됨)은 메인 메모리 및/또는 보조 메모리에 저장된다. 컴퓨터 프로그램들은 또한 통신 인터페이스를 통해 수신될 수 있다. 그러한 컴퓨터 프로그램들은, 실행될 때, 본 명세서에 논의된 바와 같은 특징들을 컴퓨터 시스템이 수행할 수 있게 한다. 특히, 컴퓨터 프로그램들은, 실행될 때, 프로세서가 다양한 실시예의 특징들을 수행할 수 있게 한다. 따라서, 그러한 컴퓨터 프로그램들은 컴퓨터 시스템의 제어기들을 나타낸다.

[0045] 다양한 실시예에서, 소프트웨어는 이동식 저장 드라이브, 하드 디스크 드라이브, 또는 통신 인터페이스를 이용하여 컴퓨터 프로그램 제품에 저장될 수 있고 컴퓨터 시스템에 로딩될 수 있다. 제어 로직(소프트웨어)은, 프로세서에 의해 실행될 때, 프로세서가 본 명세서에 설명된 다양한 실시예의 기능들을 수행하게 한다. 다양한 실시예에서, ASIC(application specific integrated circuit)들과 같은 하드웨어 컴포넌트들. 관련 기술 분야(들)의 통상의 기술자들에게는 본 명세서에 설명된 기능들을 수행하도록 하드웨어 상태 머신을 구현하는 것이 명백할 것이다.

[0046] 다양한 실시예에서, 서버는 애플리케이션 서버들(예를 들어, WEB SPHERE, WEB LOGIC, JBOSS)를 포함할 수 있다. 다양한 실시예에서, 서버는 웹 서버들(예를 들어, APACHE, IIS, GWS, SUN JAVA SYSTEM WEB SERVER)을 포함할 수 있다.

[0047] 이 기술 분야의 통상의 기술자들이 이해할 것인 바와 같이, 디바이스는 운영 체제뿐만 아니라 컴퓨터들과 전형적으로 관련되는 다양한 종래의 지원 소프트웨어 및 드라이버들을 포함할 수 있지만, 이에 한정되지 않는다.

디바이스는 임의의 적합한 퍼스널 컴퓨터, 네트워크 컴퓨터, 워크스테이션, 개인용 정보 단말기, 셀룰러폰, 스마트폰, 미니컴퓨터, 메인프레임, 또는 그와 유사한 것 등을 포함할 수 있지만 이에 한정되지 않는다. 디바이스는 네트워크에 액세스할 수 있는 홈 또는 비즈니스 환경에 있을 수 있다. 다양한 실시예에서, 액세스는 상업적으로 입수 가능한 웹-브라우저 소프트웨어 패키지를 통해 네트워크 또는 인터넷을 통해 이루어진다. 디바이스는 SSL(Secure Sockets Layer) 및 TLS(Transport Layer Security) 등과 같은 보안 프로토콜들을 구현할 수 있다. 디바이스는 http, https, ftp, 및 sftp를 포함한 여러 애플리케이션 계층 프로토콜들을 구현할 수 있다.

[0048] 다양한 실시예에서, 시스템(100)의 컴포넌트들, 모듈들, 및/또는 엔진들은 마이크로-애플리케이션들 또는 마이크로-앱들로서 구현될 수 있다. 마이크로-앱들은 전형적으로 예를 들어, Palm 모바일 운영 체제, Windows 모바일 운영 체제, Android 운영 체제, Apple iOS, Blackberry 운영 체제, 및 기타 등등을 포함한, 모바일 운영 체제의 정황에서 배치된다. 마이크로-앱은 다양한 운영 체제들과 하드웨어 리소스들의 동작들을 통제하는 일련의 미리 정해진 규칙들을 통해 더 큰 운영 체제 및 관련된 하드웨어의 리소스들을 활용하도록 구성될 수 있다. 예를 들어, 마이크로-앱이 모바일 디바이스 또는 모바일 운영 체제 외의 디바이스 또는 네트워크와 통신하기를 원하는 경우에, 마이크로-앱은 모바일 운영 체제의 미리 정해진 규칙들 하에서 운영 체제 및 관련된 하드웨어의 통신 프로토콜을 활용할 수 있다. 또한, 마이크로-앱이 사용자로부터의 입력을 원하는 경우에, 마이크로-앱은 운영 체제에 응답을 요청하도록 구성될 수 있고, 운영 체제는 다양한 하드웨어 컴포넌트들을 모니터한 다음 하드웨어로부터 검출된 입력을 마이크로-앱으로 전달한다.

[0049] "클라우드" 또는 "클라우드 컴퓨팅"은 최소의 관리 노력 또는 서비스 제공자 상호작용으로 빨리 준비되고 릴리스될 수 있는 구성 가능한 컴퓨팅 리소스들(예를 들어, 네트워크들, 서버들, 저장소, 애플리케이션들, 및 서비스들)의 공유 풀에 대한 편리한 온-디맨드 네트워크 액세스를 가능하게 하기 위한 모델을 포함한다. 클라우드 컴퓨팅은 위치-의존적 컴퓨팅을 포함할 수 있고, 그에 의해 공유되는 서버들은 온 디맨드로 컴퓨터들 및 다른 디바이스들에 리소스들, 소프트웨어, 및 데이터를 제공한다. 클라우드 컴퓨팅에 관한 추가의 정보는, http://csrc.nist.gov/groups/SNS/cloud-computing/cloud-def-v15.doc (2011년 2월 4일에 마지막 방문함)에서 NIST(National Institute of Standards and Technology: 미국 국립 표준 기술 연구소)의 클라우드 컴퓨팅의 정의를 참조하고, 그 전체 내용이 본 명세서에 참고로 포함된다.

[0050] 본 명세서에서 사용된, "송신(transmit)"은 하나의 시스템 컴포넌트로부터 또 다른 컴포넌트로 전자 데이터를 송신하는 것을 포함할 수 있다. 또한, 본 명세서에서 사용된, "데이터"는 명령들, 쿼리들, 파일들, 저장을 위한 데이터, 및 디지털 또는 임의의 다른 형태의 유사한 것 등과 같은 포괄적 정보를 포함할 수 있다.

[0051] 시스템은 웹 서비스들, 유틸리티 컴퓨팅, 편재형 및 개별화된 컴퓨팅, 보안성 및 아이덴티티 솔루션, 자동 컴퓨팅, 클라우드 컴퓨팅, 상품 컴퓨팅, 이동 및 무선 솔루션, 오픈 소스, 바이오메트릭, 그리드 컴퓨팅 및/또는 메시 컴퓨팅과 관련되는 사용들을 고려한다.

[0052] 본 명세서에서 논의되는 컴퓨터들은 사용자들에 의해 액세스 가능한 적합한 웹사이트 또는 다른 인터넷 기반 그래픽 사용자 인터페이스를 제공할 수 있다. 다양한 실시예에서, Microsoft IIS(Internet Information Server), MTS(Microsoft Transaction Server), 및 Microsoft SQL 서버가 Microsoft 운영 체제, Microsoft NT 웹 서버 소프트웨어, Microsoft SQL 서버 데이터베이스 시스템, 및 Microsoft 상거래 서버와 결합하여 사용된다. 또한, Access 또는 Microsoft SQL 서버, Oracle, Sybase, Informix MySQL, Interbase 등과 같은 컴포넌트들이 사용되어 ADO(Active Data Object) 준수 데이터베이스 관리 시스템을 제공할 수 있다. 다양한 실시예에서, Apache 웹 서버는 Linux 운영 체제, MySQL 데이터베이스, 및 Perl, PHP 및/또는 Python 프로그래밍 언어들과 함께 사용된다.

[0053] 본 명세서에서 논의되는 통신들, 입력들, 저장, 데이터베이스들 또는 디스플레이들 중 임의의 것은 웹 페이지들을 갖는 웹사이트를 통해 용이해질 수 있다. 용어 "웹 페이지"는 본 명세서에 사용되는 바와 같이, 사용자와 상호작용하는 데 사용할 수 있는 문서들 및 애플리케이션들의 타입을 제한하고자 의도되지 않는다. 예를 들어, 전형적인 웹사이트는 표준 HTML 문서들 외에도 다양한 형태들, Java 애플릿, JavaScript, ASP(active server page), CGI(common gateway interface) 스크립트, XML(extensible markup language), 동적 HTML, CSS(cascading style sheet), AJAX(Asynchronous JavaScript And XML), 헬퍼 애플리케이션, 및 플러그인 등을 포함할 수 있다. 서버는 URL(http://yahoo.com/stockquotes/ge) 및 IP 주소(123.56.789.234)를 포함하는 요청을 웹 서버로부터 수신하는 웹 서비스를 포함할 수 있다. 웹 서버는 적절한 웹 페이지들을 검색하고, 웹 페이지들에 대한 데이터 또는 애플리케이션을 IP 주소로 송신한다. 웹 서비스들은 인터넷과 같은 통신 수단을 통해 다른 애플리케이션들과 상호작용할 수 있는 애플리케이션들이다. 웹 서비스들은 전형적으로 XML, SOAP, AJAX,

WSDL, 및 UDDI와 같은 표준 또는 프로토콜에 기초한다. 웹 서비스 방법들은 이 기술 분야에 잘 알려져 있고, 많은 표준 텍스트들에서 다루어진다. 예를 들어, 본 명세서에 참고로 포함되는 ALEX NGHIEM, IT WEB SERVICES: A ROADMAP FOR THE ENTERPRISE (2003)를 참조한다.

[0054] 전문가들은 또한 브라우저 기반 문서 내에 데이터를 표시하기 위한 다수의 방법들이 있다는 것을 이해할 것이다. 데이터는 표준 텍스트로서, 또는 고정된 리스트, 스크롤 가능 리스트, 드롭-다운 리스트, 편집 가능 텍스트 필드, 고정 텍스트 필드, 팝업 창, 및 기타 등등 내에 표현될 수 있다. 마찬가지로, 예를 들어, 키보드를 이용하여 자유로운 텍스트 입력, 메뉴 항목들의 선택, 체크 박스들, 옵션 박스들, 및 기타 등등과 같이, 웹 페이지에서 데이터를 수정하기 위해 이용 가능한 다수의 방법들이 있다.

[0055] 시스템 및 방법은 기능 블록 컴포넌트들, 스크린 샷들, 옵션 선택들, 및 다양한 처리 단계들의 관점에서 본 명세서에서 설명될 수 있다. 그러한 기능 블록들이 특정 기능들을 수행하기 위해 구성된 임의의 수의 하드웨어 및/또는 소프트웨어 컴포넌트들에 의해 실현될 수 있다는 것을 이해할 것이다. 예를 들어, 시스템은 하나 이상의 마이크로프로세서 또는 다른 제어 디바이스들의 제어 하에 다양한 기능들을 수행할 수 있는 다양한 집적 회로 컴포넌트들, 예를 들어, 메모리 요소들, 처리 요소들, 로직 요소들, 및 검색 테이블들, 및 등등을 채택할 수 있다. 유사하게, 시스템의 소프트웨어 요소들은 C, C++, C#, Java, JavaScript, VBScript, Macromedia Cold Fusion, COBOL, Microsoft Active Server Pages, 어셈블리, PERL, PHP, 오크(awk), Python, Visual Basic, SQL Stored Procedures, PL/SQL, 임의의 UNIX 쉘 스크립트 및 XML(extensible markup language) 등과 같은 임의의 프로그래밍 또는 스크립팅 언어로 구현될 수 있고, 다양한 알고리즘들은 데이터 구조들, 오브젝트들, 프로세스들, 루틴들, 또는 다른 프로그래밍 요소들의 임의의 조합으로 구현된다. 또한, 시스템이 데이터 송신, 시그널링, 데이터 처리, 네트워크 제어, 및 등등을 위한 임의의 수의 종래의 기법을 채택할 수 있다는 것에 유의해야 한다. 또한, 시스템은 JavaScript, VBScript 또는 기타 등등과 같은 클라이언트 측 스크립팅 언어로 보안 문제를 검출 또는 예방하기 위해 사용될 수 있다. 암호법 및 네트워크 보안의 기초적 소개를 위해, 다음의 참조 문헌들 중 임의의 것을 참조한다: (1) "Applied Cryptography: Protocols, Algorithms, And Source Code In C," by Bruce Schneier, published by John Wiley & Sons (second edition, 1995); (2) "Java Cryptography" by Jonathan Knudson, published by O'Reilly & Associates (1998); (3) "Cryptography & Network Security: Principles & Practice" by William Stallings, published by Prentice Hall; 이들 모두는 본 명세서에 참고로 포함된다.

[0056] 이 기술 분야의 통상의 기술자에 의해 이해될 것인 바와 같이, 시스템은 기존의 시스템의 커스터마이제이션, 애드온(add-on) 제품, 업그레이드된 소프트웨어를 실행하는 처리 장치, 독립형 시스템, 분산형 시스템, 방법, 데이터 처리 시스템, 데이터 처리를 위한 디바이스, 및/또는 컴퓨터 프로그램 제품으로서 구현될 수 있다. 따라서, 시스템의 임의의 부분 또는 모듈은 코드를 실행하는 처리 장치, 인터넷 기반 실시예, 전적으로 하드웨어 실시예, 또는 인터넷, 소프트웨어 및 하드웨어의 양태들을 조합하는 실시예의 형태를 취할 수 있다. 또한, 시스템은 저장 매체 내에 구현된 컴퓨터 판독가능 프로그램 코드 수단을 갖는 컴퓨터 판독가능 저장 매체상의 컴퓨터 프로그램 제품의 형태를 취할 수 있다. 하드 디스크, CD-ROM, 광학 저장 디바이스, 자기 저장 디바이스, 솔리드 스테이트 저장 디바이스, 및/또는 기타 등등을 포함하는 임의의 적합한 컴퓨터 판독 가능 저장 매체가 이용될 수 있다.

[0057] 시스템 및 방법은 다양한 실시예에 따른 방법들, 장치(예를 들어, 시스템들), 및 컴퓨터 프로그램 제품들의 스크린 샷들, 블록도들 및 흐름도들을 참조하여 설명된다. 블록도들 및 흐름도들의 각각의 기능 블록과, 블록도들 및 흐름도들에서의 각각의 기능 블록들의 조합들은 각각 컴퓨터 프로그램 명령어들에 의해 구현될 수 있다는 것을 이해할 것이다.

[0058] 이러한 컴퓨터 프로그램 명령어들은 범용 컴퓨터, 특수 목적 컴퓨터, 또는 다른 프로그램 가능 데이터 처리 장치에 로딩되어 머신을 생성할 수 있고, 그에 따라 컴퓨터 또는 다른 프로그램 가능한 데이터 처리 장치상에서 실행되는 명령어들이 흐름도 블록 또는 블록들에서 지정된 기능들을 구현하기 위한 수단을 생성하게 된다. 이러한 컴퓨터 프로그램 명령어들은 또한 컴퓨터 또는 다른 프로그램 가능한 데이터 처리 장치에게 특정한 방식으로 기능하도록 지시할 수 있는 컴퓨터 판독가능 메모리에 저장될 수 있으며, 그에 따라 컴퓨터 판독가능 메모리에 저장된 명령어들이 흐름도 블록 또는 블록들에서 지정된 기능을 구현하는 명령 수단을 포함하는 제조 물품을 생성하게 된다. 컴퓨터 프로그램 명령어들은 또한 컴퓨터 또는 다른 프로그램 가능한 데이터 처리 장치에 로딩되어 일련의 동작 단계들이 컴퓨터 또는 다른 프로그램 가능한 장치상에서 수행되게 하여 컴퓨터 구현 프로세스를 생성할 수 있으며, 그에 따라 컴퓨터 또는 다른 프로그램 가능한 장치상에서 실행되는 명령어들이 흐름도 블

록 또는 블록들에서 지정된 기능들을 구현하기 위한 단계들을 제공하게 된다.

[0059] 따라서, 블록도들 및 흐름도들의 기능 블록들은 특정 기능들을 수행하는 수단들의 조합들, 특정 기능들을 수행하는 단계들의 조합들, 특정 기능들을 수행하는 프로그램 명령어 수단들을 지원한다. 또한 블록도들 및/또는 흐름도들의 각각의 기능 블록, 및 블록도들 및 흐름도들의 기능 블록들의 조합들이 특정 기능들 또는 단계들을 수행하는 특수 목적의 하드웨어 기반 컴퓨터 시스템들, 또는 특수 목적의 하드웨어 및 컴퓨터 명령어들의 적합한 조합들에 의해 구현될 수 있다는 것이 이해될 것이다. 또한, 프로세스 흐름들의 예시들 및 그에 대한 설명들은 사용자 창들, 웹페이지들, 웹사이트들, 웹 양식들, 프롬프트들, 등등을 참조할 수 있다. 전문가들은 본 명세서에 설명된 예시된 단계들이 창들, 웹페이지들, 웹사이트들, 웹 양식들, 프롬프트들 및 기타 등등의 사용을 포함하는 임의의 수의 구성들로 포함할 수 있다는 것을 이해할 것이다. 또한 예시되고 설명된 다수의 단계들은 단일 웹페이지들 및/또는 창들로 조합될 수 있지만 간결성을 위해 확장되었다는 것이 더 이해되어야 한다. 다른 경우들에서, 단일 프로세스 단계들로서 예시되고 설명된 단계들은 다수의 웹페이지 및/또는 창으로 분리될 수 있지만, 간결성을 위해 결합되었다.

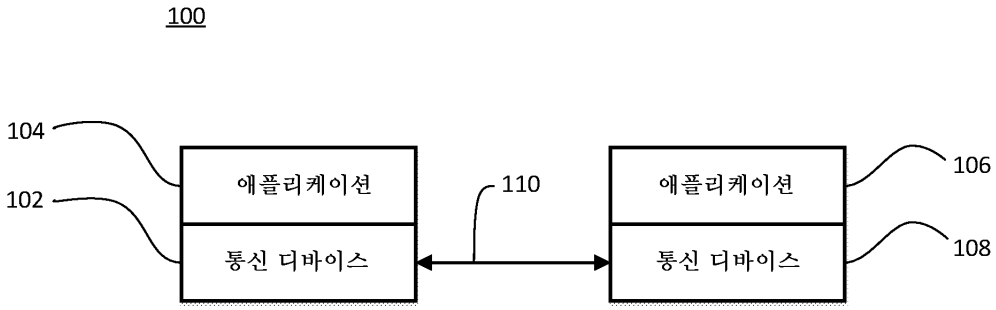
[0060] "비일시적"이라는 용어는 청구 범위로부터 일시적 신호들의 전파 자체만을 제거하는 것이고, 일시적 신호들의 전파 자체만은 아닌 모든 표준 컴퓨터 판독가능 매체에 대한 권리들을 포기하는 것은 아닌 것으로 이해되어야 한다. 달리 말하면, "비일시적 컴퓨터 판독가능 매체" 및 "비일시적 컴퓨터 판독가능 저장 매체"라는 용어의 의미는, 35 U.S.C.§101에 따른 특허 가능 주제의 범위 밖에 있는 것으로 In Re Nuijten에서 발견된 일시적 컴퓨터 판독가능 매체의 타입들만을 배제하는 것으로 해석되어야 한다.

[0061] 이점들, 다른 장점들 및 문제들에 대한 해결책들이 특정한 실시예에 관하여 본 명세서에 설명되었다. 그러나, 이점들, 장점들, 및 문제들에 대한 해결책들, 및 임의의 이점들, 장점들, 및 문제들에 대한 해결책들을 발생시킬 수 있거나 또는 더 현저하게 되도록 할 수 있는 임의의 요소들이 본 개시내용의 중요한, 필요한, 또는 본질적인 특징들 또는 요소들로서 해석되지 말아야 한다. 단수의 요소에 대한 언급은 명시적으로 언급되지 않는한 "하나 및 단 하나"를 의미하는 것이 아니라 "하나 이상"을 의미한다. 또한, 'A, B 및 C 중 적어도 하나' 또는 'A, B 또는 C 중 적어도 하나'와 유사한 문구가 청구항들 또는 명세서에서 사용되는 경우, 그 문구는 A만이 실시예에 존재할 수 있고, B만이 실시예에 존재할 수 있고, C만이 실시예에 존재할 수 있거나, 요소 A, B 및 C의 임의의 조합; 예를 들어 A와 B, A와 C, B와 C 또는 A와 B와 C가 단일 실시예에 존재할 수 있음을 의미하도록 해석되는 것으로 의도된다.

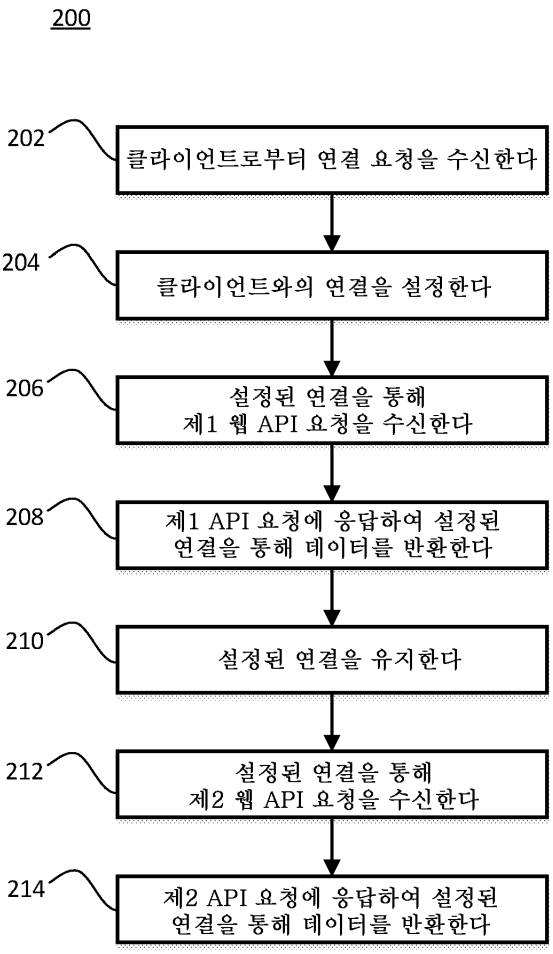
[0062] 본 개시가 방법을 포함하지만, 예를 들어 자기 또는 광학 메모리, 또는 자기 또는 광학 디스크 등과 같은 유형의(tangible) 컴퓨터 판독 가능 캐리어상의 컴퓨터 프로그램 명령어들로서 구현될 수 있다고 생각된다. 이 기술 분야의 통상의 기술자들에게 공지된 전술한 예시적인 실시예들의 요소들과의 모든 구조적, 화학적, 및 기능적 등가물들은 본 명세서에 참고로 명시적으로 포함되고, 본 청구항들에 포함되는 것으로 의도된다. 또한, 디바이스 또는 방법이 본 청구항들에 포함되기 위해, 그 디바이스 또는 방법이 본 개시가 해결하고자 하는 각각의 그리고 모든 문제를 해결하는 것이 필요한 것은 아니다. 또한, 본 개시의 어떤 요소, 컴포넌트, 또는 방법 단계도 그 요소, 컴포넌트, 또는 방법 단계가 청구항들에서 명확하게 언급되는지의 여부에 관계없이 공개되도록 의도되지 않는다. 본 명세서의 어떤 청구항 요소도, 그 요소가 명확히 "~을 위한 수단"이라는 문구를 이용하여 명시되지 않는 한, 35 U.S.C. 112(f)의 조항들에 따라 해석되지 않는다. 본 명세서에서 사용된, "포함하다", "포함하는"이라는 용어들 및 그들의 임의의 다른 변형은 비배타적인 포함을 커버하도록 의도되며, 그에 따라 요소들의 리스트를 포함하는 프로세스, 방법, 물품, 또는 장치가 그러한 요소들만을 포함하는 것이 아니라, 그러한 프로세스, 방법, 물품, 또는 장치에 고유하거나 또는 명시적으로 열거되지 않은 다른 요소들을 포함할 수 있다.

도면

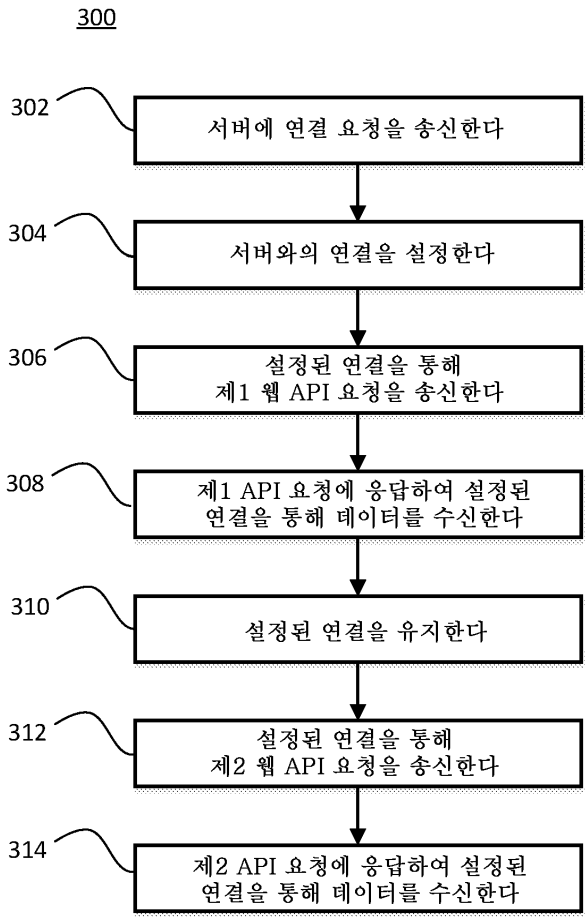
도면1


100
104
102
애플리케이션
통신 디바이스
110
애플리케이션
통신 디바이스
106
108

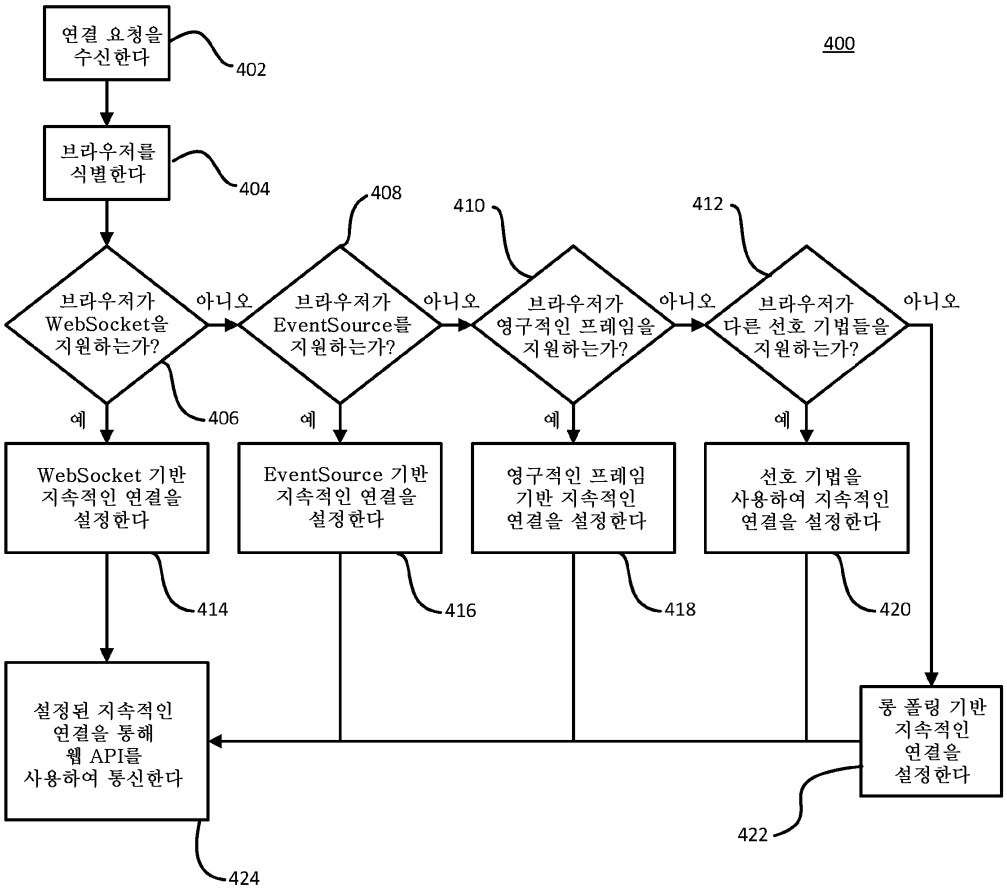
도면2


200
202
클라이언트로부터 연결 요청을 수신한다
204
클라이언트와의 연결을 설정한다
206
설정된 연결을 통해
제1 웹 API 요청을 수신한다
208
제1 API 요청에 응답하여 설정된
연결을 통해 데이터를 반환한다
210
설정된 연결을 유지한다
212
설정된 연결을 통해
제2 웹 API 요청을 수신한다
214
제2 API 요청에 응답하여 설정된
연결을 통해 데이터를 반환한다

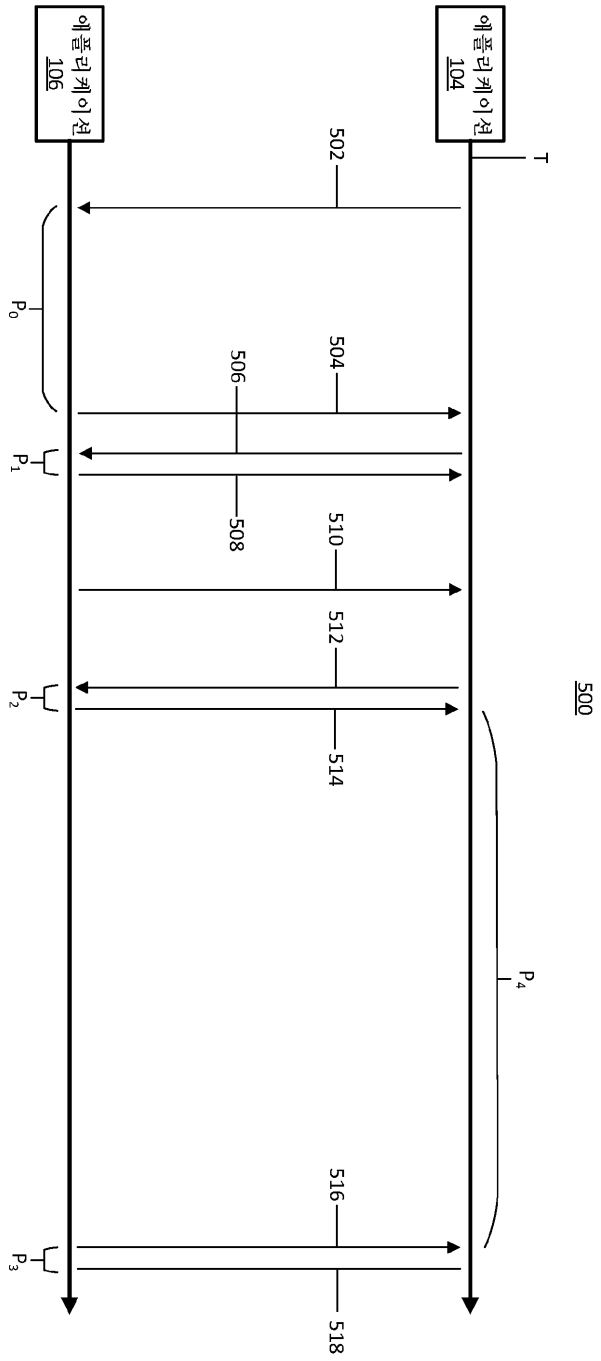
도면3


300
302
서버에 연결 요청을 송신한다
304
서버와의 연결을 설정한다
306
설정된 연결을 통해
제1 웹 API 요청을 송신한다
308
제1 API 요청에 응답하여 설정된
연결을 통해 데이터를 수신한다
310
설정된 연결을 유지한다
312
설정된 연결을 통해
제2 웹 API 요청을 송신한다
314
제2 API 요청에 응답하여 설정된
연결을 통해 데이터를 수신한다

도면4


400
연결 요청을 수신한다
402
브라우저를 식별한다
404
브라우저가 WebSocket을 지원하는가?
406
아니오
예
408
브라우저가 EventSource를 지원하는가?
410
브라우저가 영구적인 프레임을 지원하는가?
412
브라우저가 다른 선호 기법들을 지원하는가?
WebSocket 기반 지속적인 연결을 설정한다
414
EventSource 기반 지속적인 연결을 설정한다
416
영구적인 프레임 기반 지속적인 연결을 설정한다
418
선호 기법을 사용하여 지속적인 연결을 설정한다
420
롱 폴링 기반 지속적인 연결을 설정한다
422
설정된 지속적인 연결을 통해 웹 API를 사용하여 통신한다
424

도면5


애플리케이션 104
애플리케이션 106
T
502
504
506
508
510
512
514
516
518
P0
P1
P2
P3
P4
500