(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2016-0130785
(43) 공개일자 2016년11월14일

(51) 국제특허분류(Int. Cl.)
G06F 9/455 (2006.01)
(52) CPC특허분류
G06F 9/45533 (2013.01)
(21) 출원번호 10-2016-7026551
(22) 출원일자(국제) 2014년10월17일
심사청구일자 없음
(85) 번역문제출일자 2016년09월26일
(86) 국제출원번호 PCT/US2014/061165
(87) 국제공개번호 WO 2015/130349
국제공개일자 2015년09월03일
(30) 우선권주장
61/945,534 2014년02월27일 미국(US)

(71) 출원인
오픈모바일 월드 와이드, 인크
미국 매사추세츠 01701 프레이밍햄 스위트 114 스펀 스트리트 111
(72) 발명자
리우, 시앙하이
미국 매사추세츠 01760 프레이밍햄 스위트 114 스펀 스트리트 111
바즈파이, 찬드라
미국 매사추세츠 01760 네이틱 파멜라 로드 21
(뒷면에 계속)
(74) 대리인
김해중

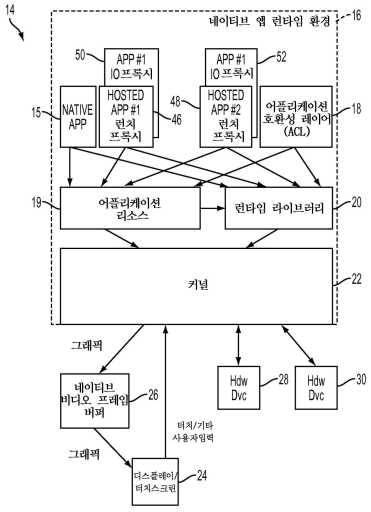
전체 청구항 수 : 총 14 항

(54) 발명의 명칭 **멀티 오퍼레이팅 시스템 환경에서 모바일 장치 상에서 실행되는 호스트 어플리케이션에서 서비스 대체를 위한 인-프로세싱 트랩핑**

(57) 요 약

본 발명은 네이티브 소프트웨어 어플리케이션과 네이티브 런타임 환경을 포함하는 네이티브 운영 체제를 실행하는 중앙 처리 장치를 포함하는 컴퓨팅 장치를 제공한다. 각각의 네이티브 소프트웨어 어플리케이션은 네이티브 운영 체제하에서 의 실행을 위한 명령어를 구비한다. 호스트 런타임 환경은 네이티브 런타임 환경 내에서 실행하고, 호스트된 런타임 환경의 각각은 네이티브 운영 체제와 상이한 호스트된 운영 체제 하에서의 실행을 위한 명령어를 구비한 호스트된 소프트웨어 어플리케이션을 실행한다. 호스트된 런타임 환경의 제1 프로세스 내에서 실행되는 제1 호스트된 소프트웨어 어플리케이션은 객체-지향 프로그래밍(OOP) 클래스(class)에 의해 정의된 객체의 멤버를 참조하는 명령어를 포함한다. 프로세스는 참조 멤버로서 참조 클래스에 의해 특정되는 것 이외의 데이터 및/또는 코드를 이용하여 그 명령어를 실행한다.

대 표 도 - 도2



(72) 발명자

멘나이스, 케빈

미국 매사추세츠 01760 프레이밍햄 스위트 114 스핀 스트리트 111

괴츠, 자렛

미국 매사추세츠 02118-3100 보스턴 #3 콜럼버스 에비뉴 482

명 세 서

청구범위

청구항 1

중앙 처리 유닛 - 중앙 처리 유닛은 하나 이상의 네이트브 소프트웨어 어플리케이션이 실행되는 하나 이상의 네이티브 런타임 환경을 포함하는 네이티브 운영 체제를 실행하고, 여기서 각각의 네이티브 소프트웨어 어플리케이션은 네이티브 운영 체제 하에서의 실행을 위한 명령어를 포함함 - ;

하나 이상의 네이티브 런타임 환경 내에서 실행되는 하나 이상의 호스트된 런타임 환경 - 호스트된 런타임 환경의 각각은 네이티브 운영 체제와 상이한 호스트된 운영 체제 하에서의 실행을 위한 명령어를 구비한 호스트된 소프트웨어 어플리케이션을 실행함 - ;

호스트된 런타임 환경의 제1 프로세스 내에서 실행되는 제1 호스트된 소프트웨어 어플리케이션 - 제1 호스트된 소프트웨어 어플리케이션은, 객체-지향 프로그래밍(OOP) 클래스(class)(이하 "참조 클래스(referenced class)")에 의해 정의된 객체(object)의 멤버(member)(이하 "참조 멤버(referenced member)")를 참조하는 명령어를 포함하고, 참조 멤버는 방법 멤버(method member) 및 데이터 멤버(data member) 중 어느 하나임 - ;

참조 멤버로서 참조 클래스에 의해 특정되는 것 이외의 데이터 및/또는 코드(이하 "대체 멤버")를 이용하여 명령어를 실행하는 프로세스;

를 포함하는 컴퓨팅 장치.

청구항 2

제1항에 있어서,

참조 멤버와 대체 멤버는 유사한 기능을 제공하지만 상이한 메커니즘을 이용하는 컴퓨팅 장치.

청구항 3

제2항에 있어서,

상이한 메커니즘은 장치에 대해 외적인(external) 상이한 서비스인 컴퓨팅 장치.

청구항 4

제3항에 있어서,

상이한 메커니즘은 상이한 웹 서비스인 컴퓨팅 장치.

청구항 5

제4항에 있어서,

참조 멤버는 제1 클래스에 의해 정의되는 방법 멤버이고, 방법 멤버는 맵핑(mapping), 네비게이션, 광고, 디지털 월렛, 계정 인증(account authorization) 및/또는 제1 호스트된 소프트웨어 어플리케이션에 의해 요구되는 다른 서비스를 지원하기 위해 제1 외부 웹 서비스(external web service)를 이용하는 컴퓨팅 장치.

청구항 6

제5항에 있어서,

대체 멤버는 전술한 서비스를 지원하기 위해 상이한 제2 웹 서비스를 이용하는 컴퓨팅 장치.

청구항 7

제3항에 있어서,

상이한 메커니즘은 컴퓨팅 장치 및/또는 그 런타임 환경에 고유한 상이한 서비스인 컴퓨팅 장치.

청구항 8

제7항에 있어서,

참조 멤버는 제1 클래스에 의해 정의되는 방법 멤버이고, 방법 멤버는 전화통화(telephony), 카메라, 주소록/연락처, 문서 미리보기, 음악/멀티미디어 재생 또는 호스트된 런타임 환경에 고유한 다른 서비스를 이용하고,

대체 멤버는 전술한 서비스를 이용하기 위해 네이티브 런타임 환경 및/또는 네이티브 운영 체제의 서비스를 이용하는 컴퓨팅 장치.

청구항 9

제8항에 있어서,

참조 멤버 및/또는 참조 클래스에 대한 제1 소프트웨어 어플리케이션 내의 리퍼런스(references)는 장치 상의 소프트웨어 어플리케이션의 로딩 동안 대체 멤버로 분해(resolved)되는 컴퓨팅 장치.

청구항 10

컴퓨팅 장치 상에서 네이티브 운영 체제를 실행하는 단계 - 네이티브 운영 체제는 하나 이상의 네이티브 소프트웨어가 실행되는 하나 이상의 네이티브 런타임 환경을 포함하고, 각각의 네이티브 소프트웨어 어플리케이션은 네이티브 운영 체제 하에서의 실행을 위한 명령어를 구비함 - ;

하나 이상의 네이티브 런타임 환경 내에서 하나 이상의 호스트된 런타임 환경을 실행하는 단계 - 호스트된 런타임 환경의 각각은 네이티브 운영 체제와 상이한 호스트된 운영 체제하에서의 실행을 위한 명령어를 구비한 호스트된 소프트웨어 어플리케이션을 실행함 - ;

호스트된 런타임 환경의 제1 프로세스 내에서, 제1 호스트 소프트웨어 어플리케이션을 실행하는 단계 - 제1 호스트된 소프트웨어 어플리케이션은 객체-지향 프로그래밍(OOP) 클래스(class)(이하 "참조 클래스(referenced class)")에 의해 정의된 객체(object)의 멤버(member)(이하 "참조 멤버(referenced member)")를 참조하는 명령어를 포함하고, 참조 멤버는 방법 멤버(method member) 및 데이터 멤버(data member) 중 어느 하나임 - ;

프로세스에 의해 참조 멤버로서 참조 클래스에 의해 특정되는 것 이외의 데이터 및/또는 코드(이하 "대체 멤버")를 이용하여 명령어를 실행하는 단계를 포함하는

디지털 데이터 프로세싱 방법.

청구항 11

제10항에 있어서,

참조 멤버와 대체 멤버는 유사한 기능을 제공하지만 상이한 메커니즘을 이용하는 디지털 데이터 프로세싱 방법.

청구항 12

제11항에 있어서,

상이한 메커니즘은 장치에 대해 외적인(external) 상이한 서비스인 디지털 데이터 프로세싱 방법.

청구항 13

제12항에 있어서,

상이한 메커니즘은 상이한 웹 서비스인 디지털 데이터 프로세싱 방법.

청구항 14

제11항에 있어서,

대체 맴버에 대한 리퍼런스를 클래스 로더(class loader)를 이용하여 제1 호스트된 소프트웨어로 대체하는 디지털 데이터 프로세싱 방법.

발명의 설명

기 술 분 야

[0001] 본 출원은 2014년 2월 27일자에 "IN-PROCESS TRAPPING FOR SERVICE SUBSTITUTION IN HOSTED APPLICAITONS EXECUTING ON MOBILE DEVICES WITH MULTI OPERATING SYSTEM ENVIRONMENT"의 명칭으로 출원되고 본 명세서에서 참조로서 내포된 미국특허출원 NO.61/945,534호를 우선권 주장한다.

[0002] 본 발명은 디지털 데이터 처리에 관한 것으로, 구체적으로는 여러 상이한 플랫폼 상에서 실행되도록 만들어진 단일 하드웨어/소프트웨어 플랫폼 어플리케이션("앱스(apps)") 상에서의 실행을 위한 방법 및 장치에 관한 것이다. 본 발명은 예를 들면, 비제한적인 예로서, 스마트 폰, 태블릿 컴퓨터, 텔레비전에 연결되는 셋톱 박스, 차량 인포테인먼트 시스템, 또는 비행중 오락 시스템 등과 같은 스마트 모바일 기기용의 앱들 사이의 크로스-플랫폼 호환성을 지원하는 애플리케이션을 구비한다.

배 경 기 술

[0003] 스마트 모바일 기기 시장은 분석가에 따르면, 지난 해에 40% 가까이 성장했다. 이것은 오픈 소스 리눅스 및 안드로이드 운영 체제의 변형을 실행하는 장치의 판매에 의해 큰 영향을 받았다. 이는 시장에 요긴하지만, 이러한 장치들을 위해 개발된 앱들의 상호-호환성의 부족으로 인한 문제점이 있다. 따라서, 예를 들면, 미고(Meego) 운영 체제를 실행하는 장치를 위해 개발된 앱은 타이젠(Tizen) 또는 안드로이드(Android) 운영 체제를 실행하는 장치상에서 실행되지 않는다. 전체적으로 상이한 계통(lineage)으로 변경될 때 이 문제는 더 악화된다. 예를 들어, 타이젠을 위해 개발된 앱은 웹(Web) OS 또는 Windows OS를 실행하는 장치에서는 실행되지 않는다.

[0004] 이는 오래된 앱과의 호환성이 부족한 새로운 모바일 장치를 구매한 소비자에 대한 문제만은 아니다. 또한 이는 새로운 하드웨어/소프트웨어 플랫폼을 제공하기 위해 노력하는 공급자 체인 내의 제조업자, 캐리어(carrier) 및 기타업자에 대한 문제이며, 이들은 이용가능한 앱의 큰 생태계의 부족으로 인해 방해를 받는다. 앱 개발자 역시 제품 생존가능성(viability)을 설정하거나 유지하기 위해 다양한 플랫폼에 대한 앱을 포팅(port)하도록 강제되기 때문에 시장에서의 세분화(fragmentation)로 고통받는다.

[0005] 상호 호환성 문제를 해결하기 위한 소수의 종래 기술만이 제한된 성공을 거두었다. 예를 들면, 에이서(Acer)의

애스피어(Aspire)는 윈도우 OS와 안드로이드 OS 를 지원하는 이중 부트 모드를 지원한다. 그러나, 이 장치는 두 개의 운영 체제 모두에 있어서 하나의 모드로 어플리케이션을 실행할수는 없다.

발명의 내용

해결하려는 과제

[0006] 전술한 관점에서, 본 발명의 목적은 디지털 데이터 프로세싱을 위한 향상된 시스템 및 방법을 제공하는 것을 목적으로 한다.

[0007] 본 발명은 여러 상이한 하드웨어/소프트웨어 플랫폼 상에서 실행을 위해 만들어진 하나의 하드웨어/소프트웨어 플랫폼 어플리케이션(앱) 상에서 실행을 지원하기 위한 시스템 및 방법을 제공하는 것을 다른 목적으로 한다.

[0008] 본 발명은 또한 예를 들면 비제한적인 예로서, 스마트 폰, 태블릿 컴퓨터, 텔레비전에 연결되는 셋톱 박스, 차량 인포테인먼트 시스템, 또는 비행중 오락 시스템 등과 같은 스마트 모바일 기기용의 앱들 사이의 크로스-플랫폼 호환성을 지원하기 위한 시스템 및 방법을 제공하는 것을 다른 목적으로 한다.

[0009] 본 발명의 이들 목적 및 다른 목적은 도면과 이어진 설명을 통해 명확하게 된다.

과제의 해결 수단

[0010] 전술한 목적들은 본 발명에 의해 달성될 수 있다. 본 발명에 따르면 컴퓨팅 장치를 제공하고 컴퓨팅 장치는 중앙 처리 유닛을 포함하고, 중앙 처리 유닛은 하나 이상의 네이티브 소프트웨어 어플리케이션이 실행되는 하나 이상의 네이브트 런타임 환경을 포함하는 네이티브 운영체제를 실행하고, 네이티브 소프트웨어 어플리케이션의 각각은 네이티브 운영 체제 하에서의 실행을 위한 명령어를 포함한다. 하나 이상의 호스트된 런타임 환경은 하나 이상의 네이트브 런타임 환경 내에서 실행되고, 호스트된 런타임 환경의 각각은 네이티브 운영 체제와 상이한 호스트된 운영 체제 하에서의 실행을 위한 명령어를 가진 호스트된 소프트웨어 어플리케이션을 실행한다.

[0011] 호스트된 런타임 환경의 제1 프로세스 내에 실행되는 제1 호스트된 소프트웨어 어플리케이션은 객체지향 프로그래밍(OOP) 클래스(이하 "기준 클래스")에 의해 정의된 객체의 맴버(이하 "기준 멤버")를 참조하는 명령어를 포함한다. 프로세스는 기준 멤버로서 기준 클래스에 의해 특정되는 것 이외의 데이터 및/또는 코드(이하 "대체 멤버")를 이용하는 명령어를 실행한다. 본 명세서에서, 객체의 "멤버"는 방법 멤버 및 데이터 멤버 중 어느 하나일 수 있다.

[0012] 본 발명의 연관된 양태는 전술한 바와 같은 컴퓨팅 장치를 제공하고, 여기서 기준 멤버와 대체 맴버는 이들에 대해 상이한 메커니즘을 이용하는 제1 소프트웨어 어플리케이션의 관점으로부터 유사한 기능성을 제공한다.

[0013] 이들 상이한 메커니즘은, 본 발명의 일부 양태에 따르면, 상이한 웹(또는 회부) 서비스일 수 있다. 즉, 예를 들면, 기준 멤버는 제1 클랙스에 의해 정의도니 객체의 방법 멤버일 수 있으며, 이 방법 멤버는 맵핑(mapping), 광고(advertisement), 디지털 월렛(wallet), 계정 허가(account authorization) 및/또는 제1 호스트된 소프트웨어 어플리케이션에 의해 요구되는 다른 서비스를 지원하기 위해 제1 외부 웹 서비스를 이용하는 한편, 대체 멤버는 전술한 서비스를 지원하기 위해 제2의 상이한 웹 서비스를 이용하는 코드를 포함한다.

[0014] 이들 상이한 메커니즘은 또한 상이한 네이티브 기능 또는 서비스일 수도 있다. 즉, 예를 들면, 기준 멤버가 그

렇지 않고 전화통신술(telephony), 카메라, 주소/연락처, 문서 미리보기, 음악/멀티미디어 재생 또는 호스트된 런타임 환경 본래의 다른 기능/서비스를 이용하는 경우, 대체 멤버는 대신에 네이티브 운영체제의 유사한 기능/서비스를 이용할 수도 있다.

[0015] 본 발명의 다른 양태에서는 전술한 컴퓨팅 장치를 제공하고, 여기에서 기준 멤버 및/또는 기준 클래스에 대한 제1 소프트웨어 어플리케이션 내의 기준은 장치 상에 소프트웨어 어플리케이션을 로딩ㅎ는 동안 대체 부재로 분해(resolved)될 수도 있다.

[0016] 본 발명의 다른 양태에서는 전술한 바와 같은 컴퓨팅 장치 상에서 제1 소프트웨어 어플리케이션을 실행하기 위한 오퍼레이션을 병렬화(paralleling)하기 위한 바업을 제공한다.

[0017] 본 발명의 전술한 양태 및 다른 양태는 첨부된 도면을 참조한 발명의 상세한 설명을 통해 보다 명확하게 이해될 수 있다.

발명의 효과

[0018] 본 발명에 따르면 전술한 목적을 달성할 수 있다.

도면의 간단한 설명

[0019] 도 1의 (A) 내지 (C)는 본 발명을 포함하는 타입의 컴퓨팅 장치를 나타낸 도면.

도 2는 도 1의 장치에서 실행되는 타입의 네이티브 운영 체제를 도시한 도면.

도 3은 도 1의 장치에서 호스트된 소프트웨어 어플리케이션의 실행을 위해 네이티브 소프트웨어 어플리케이션에 의해 정의되는 하나 이상의 호스트된 런타임 환경을 도시한 도면.

도 4는네이티브 런타임 환경에서 실행중인 어플리케이션의 런치 프록시(proxy)과의 사용자 상호작용(interaction)에 기반한 예시적인 호스트된 소프트웨어 어플리케이션의 런칭, 어플리케이션의 IO 프록시를 통해 호스트된 소프트웨어 어플리케이션의 오퍼레이션을 나타내는 어플리케이션 윈도우의 디스플레잉, 그 프록시로부터 호스트된 어플리케이션으로의 사용자 입력의 전송시에 있어서의 구성요소의 상호작용을 도시함 도면.

도 5는 객체의 선택된 멤버를 참조하는 명령어가 대체 멤버에 대한 기준으로 대체되는 것을 나타낸 본 발명에 따른 시스템에서의 호스트된 어플리케이션을 나타낸 도면.

발명을 실시하기 위한 구체적인 내용

[0020] 아키텍처(Architecture)

[0021] 도 1의 (A)는 본 발명을 포함하는 타입의 컴퓨팅 장치(10)를 나타낸다. 도시된 장치(10)는 일반적으로 컴퓨팅 장치에 제공되는 중앙 처리 유닛(CPU), 입력/출력(I/O), 메모리(RAM) 및 비휘발성 저장소(MEM)의 세부항목을 포함하고 이들은 시장에서 상업적으로 이용가능하며, 본 발명의 범주 내에서 모두 적용가능하다. 도시된 실시예에서, 장치(10)는 스마트 폰, 테블릿 컴퓨터 등과 같은 모바일 컴퓨팅 장치를 포함하고, 그러나 다른 실시예에서 장치는 다른 컴퓨팅 장치, 모바일 또는 텔레비전에 연결되는 셋탑 박스, 차량용 인포 시스템 또는 비행중 오락 시스템 등과 같은 다른 장치를 포함할 수도 있다.

[0022] 장치(10)는 영구적으로, 간헐적으로 또는 다른 방식으로 하나 이상의 다른 컴퓨팅 장치, 서버 또는 클라우드(12)로 지시된 네트워크에 의해 디지털 통신가능한 다른 장치에 접속될 수 있으며, 네트워크는 인터넷, 대도시 통신 네트워크(metropolitan area network), 광역 네트워크(wide area network), 근거리 통신망(local area network), 위성 통신망, 셀룰러 통신망, 포인트-투-포인트 네트워크, 및/또는 이들 중 하나 이상의 조합을 포함

할 수 있으며, 본 발명의 범위 내에서 종래의 기술들이 이용될 수도 있다.

[0023] 장치(10)의 CPU(I/O, RAM, MEM 세부 항목과 협업함)는 본 발명의 범위 내에서 적응된, 시장에서 상업적으로 이용가능한 타입의 네이티브 운영 체제(14)를 실행한다. 이와 같은 운영 체제의 예는 미고(Meego), 타이젠(Tigen), 안드로이드, 웹OS, 리눅스 운영 체제 등을 포함한다. 보다 일반적으로 및/또는 이에 더하여, 네이티브 운영 체제(14)는 비제한적인 예로서 안드로이드 기반 운영 체제 등의 리눅스 기반 운영 체제를 포함할 수 있다.

[0024] 네이티브 런타임 환경(Native Runtime Environments)

[0025] 도 2는 도 1에 도시된 장치(10) 상에서 실행된는 타입의 네이티브 운영 체제를 나타낸다.

[0026] 도면을 참조하면, 네이티브 운영 체제(14)는 종래 공지된 네이티브 소프트웨어 어플리케이션(본 발명의 교시(teachings)에 따라 적응됨), 예를 들면 네이티브 운영 체제 하에서 실행을 위한 명령어를 구비한 어플리케이션이 실행되어지는 공지된 기술의 하나 이상의 네이티브 런타임 환경(16)을 정의한다. 이와 같은 어플리케이션은 참조번호 15, 18 및 46-52로 표시되었다. 본 명세서에서 용어 "어플리케이션"과 "앱(app)"은 상호교환가능하게 사용되었다.

[0027] 네이티브 런타임 환경(16)은 네이티브 운영체제(14)와 장치(10) 상에서의 구현된 특징(specifics)에 의존해서 하나 이상의 가상 머신 또는 공지되어 있는 기타의 것(본 발명의 교시에 따라 적응됨)을 포함한다. 도시된 네이티브 런타임 환경(16)은 비제한적인 방식의 예로서 본 발명의 교시에 따라 적응되고 모두 공지된 타입의 어플리 어플리케이션 리소스(19)와 런타임 라이브러리(20)를 포함한다. 런타임 환경(16)은 또한 본 발명의 교시에 따라 적응되고 공지된 타입의 커널(24)을 포함한다.

[0028] 커널(24)(또는 대안적인 실시예의 런타임 환경 내에 제공된 대안적인 기능)은 CPU(12)(보다 일반적으로, 장치상에서 실행중인 네이티브 런타임 환경(16) 내에서 실행중인 네이트브 어플리케이션)과 장치(10)에 통합 또는 부착된 하드웨어 장치(24-30) 사이에서, 본 발명의 교시에 따라 적응되어진 종래의 공지된 방식으로, 특히 인터페이스로서 기능한다. 이는 디스플레이/터치 스크린(24)와 본 발명의 교시에 따라 적응되는 종래의 공지된 방식으로 디스플레이를 구동하는 프레임 버퍼(26)를 포함한다. 이는 비제한적인 방식의 예로서 키보드, 트랙볼, 터치 스틱, 기타 사용자 입력 장치 및/또는 공지된 타입의 통합 또는 주변 장치를 포함할 수 있다. 본 발명의 설명에서, 장치(10)와 그 사용자 사이에서의 상호작용을 지지하는 디스플레이/터치 스크린(24), 프레임 버퍼(26), 및 기타 통합/주변 장치는, 이들이 하드웨어, 소프트웨어 또는 그들의 조합(보다 일반적인 경우)을 포함하는지와 관계없이, "하드웨어 인터페이스"로 언급될 수 있다.

[0029] 하나 이상의 네이티브 런타임 환경(16) 에서 실행되는 제한없이 "어플리케이션 호환성 레이어(Applications Compatibility Layer)" 또는 "ACL"로 언급되는 네이티브 소프트웨어 어플리케이션(18)은 호스트된 소프트웨어 어플리케이션이 실행되는 하나 이상의 호스트된 런타임 환경을 정의한다. 각각의 이러한 호스트된 소프트웨어 어플리케이션은 네이티브 운영 체제와 상이한 호스트된 운영 체제 하에서의 실행을 위한 명령어를 포함한다.

[0030] 네이티브 소프트웨어 어플리케이션(46-52)은 호스트된 소프트웨어 어플리케이션(34,36)의 프록시이다. 특히, 일부 실시예에서, 호스트된 런타임 환경(32)에서 실행되는 호스트된 소프트웨어 어플리케이션은 네이트브 런타임 환경(16)에서 실행되는 복수의 대응하는 프록시 - 런치 프록시(launch proxy) 및 IO 프록시를 포함할 수 있다. 여기서, 예시적인 목적을 위해서, 호스트된 소프트웨어 어플리케이션(34)은 런치 프록시(46)와 IO 프록시(50)를 가진 것으로 도시되었다. 호스트된 소프트웨어 어플리케이션(36)은 유사하게 런치 프록시(48)와 IO 프록시(52)를 가진 것으로 도시되었다. 비록 양자의 런치 프록시와 IO 프록시가 도시된 실시예에 이용되고 있지만, 다른 실시예에서, 호스트된 소프트웨어 어플리케이션은 단지 하나의 타입(예를 들면 IO 또는 런치)의 대응하는 프록

시를 가지거나 또는 다른 실시예에서, 호스트된 어플리케이션의 하나 또는 그 이상은 그와 같은 프록시를 가지지 않을 수도 있다.

[0031] 호스트된 런타임 환경(Hosted Runtime Environment(s))

[0032] 호스트된 운영 체제는, 예를 들면, 리눅스-기반 운영 체제이고, 비제한적인 방식의 예로서 안드로이드 기반 운영 체제일 수 있다. 네이티브 운영 체제(14)는 유사하지만 호스트된 운영 체제와는 상이한 변종(flavor)의, 예를 들면, 리눅스-기반 및/또는 안드로이드-기반 운영 체제일 수 있다. 보다 구체적인 방식의 예로, 네이티브 운영 체제(14)는, 본 발명의 교시에 따라 적응되는, 전술한 타이젠, 웹OS, 리눅스 운영 체제 중 하나를 비제한적인 방식의 예로 포함하고, 호스트된 운영 체제는 상업적으로 이용가능한 안드로이드 운영 체제(본 발명의 교시에 따라 적응됨)의 "변종"을 비제한적인 방식의 예로서 포함한다.

[0033] 도 3은 본 발명에 따른 장치(10)에서 호스트된 소프트웨어 어플리케이션(34,36)을 실행하기 위해 네이티브 소프트웨어 어플리케이션(18)(또는 ACL)에 의해 정의되는 하나 이상의 호스트된 런타임 환경(32)을 도시하고 있다. 도시된 호스트된 런타임 환경(32)은 본 발명의 교시에 따라 적응된 공지된 타입이며, 그 내에서 호스트된 운영 체제 하에서의 실행을 위한 명령어를 구비한 소프트웨어 어플리케이션(예를 들면, 호스트된 소프트웨어 어플리케이션)이 작성되고 실행되어진다.

[0034] 호스트된 런타임 환경(32)은, 호스트된 운영 체제의 종류와 런타임 환경(32) 내의 구현된 특징에 의존하여, 하나 이상의 가상 머신 또는 공지된 다른 것(본 발명의 교시에 따라 적응됨)을 포함할 수 있다. 도시된 호스트 런타임 환경(32)은 안드로이드-기반 소프트웨어 어프리케이션(34,36)을 실행하도록 의도되고(다른 실시예는 다른 운영 체제을 위해 설계되고 작성된 어플리케이션을 실행하도록 의도될 수 있음), 비제한적인 방식의 예로 리소스 프레임워크(38), 가상 머신(VMs)(40), 이벤트 핸들러(42) 및 런타임 라이브러리를 포함할 수 있고, 이들은 모두 비제한적인 방식의 예이고, 공지된 타입의 모든 것들이 이용가능하고, 본 발명의 교시에 따라 적응되어 진다.

[0035] 도시된 런타임 환경(32)은 공지된 타입의 보호된 커널 영역에서 오퍼레이션을 실행하는 관점에서 그 자체적으로는 커널을 포함하지 않는다(리눅스-/안드로이드- 기반 운영 체제의 런타임 환경에는 일반적으로 포함됨). 대신 일부 이런 오퍼레이션(예를 들면 리눅스-/안드로이드-기반 운영 체제의 커널 내에 일반적으로 포함될 수 있는 오퍼레이션)은 사용자 공간에서 실행된다.

[0036] 예시적인 방식으로, 이들 커널 스페이스 오퍼레이션은 리소스 프레임워크(34), 가상 머신(36), 이벤트 핸들러(42), 런-타임 라이브러리(44) 및/또는 런타임 환경(32)의 다른 구성요소에 의해 디스플레이 상에 표시를 위해 프레임 버퍼에 그래픽을 로드하는 것에 의존한다. 호스트된 런타임 환경(32)의 커널에서 실행하기 보다는, 도시된 실시예에서 이들 오퍼레이션은 사용자 공간으로 승격되고(elevated) 그런 그래픽을, 네이티브 런타임 환경(16)과 그에 실행되는 어플리케이션 - 특히 I/O 프록시 어플리케이션(50,52) - 에 의해 공유되는 "가상의" 프레임 버퍼(54)로 로딩하도록 채용된다.

[0037] 다른 그러한 커널-스페이스 동작의 실행은 네이티브 운영 체제(14)와 그 런타임 환경(16) 오퍼레이션, 보다 광범위하게는, 그와 달리 런타임 환경(32) 내에서 수행되는 호스트된 소프트웨어 어플리케이션(34,36)의 실행에 요구되는 기능에 대한 패싱-오프(passing-off)에 의해, 구체적으로 예를 들면 그 커널에 의해 회피될 수 있다.

[0038] 도시된 실시예에서 이러한 패싱-오프는 리소스 프레임워크(34), 가상 머신(36), 이벤트 핸들러(42), 런타임 라이브러리(44) 및/또는 그런 기능 또는 그 대안(alternates)을 수행하기 위해 호스트된 소프트웨어 어플리케이션(34,36)의 네이티브 소프트웨어 어플리케이션 프록시(46-52)(런타임 환경(16)에서 실행)과 통신하거나 및/또는

의존하는 런타임 환경(32)의 다른 구성요소에 의해 결과된다.

[0039] 전술한 평가는 이하의 설명들과 참조로서 내포된 참조 출원 내의 설명을 통해 획득될 수 있다.

[0040] 네이티브 및 호스트된 소프트웨어 어플리케이션 설치(Native and Hosted Software Aplication Installation)

[0041] 네이티브 소프트웨어 어플리케이션, 예를 들면 15와 16은, 보다 구체적으로, 운영체제(14)의 타입의 운영 체제 내의 앱의 설치를 위한 종래의 방식으로 네이티브 런타임 환경(16) 내에서의 실행을 위해, 장치(10) 상에 설치된다(사용자 또는 다른 사람의 지시하에). 이러한 설치는 통상적으로 네이티브 운영 체제(14)와 OS(14)에 대해 전통적인 타입의 "인스톨러" 앱(미도시)을 실행하는 런타임 환경(16)의 협동 액션(cooperative action)을 포함하고, 통상적으로는 개발자 사이트 등으로부터 다운로드된 어플리케이션 패키지 파일로부터 인스톨될 어플리케이션의 실행가능한 파일, 아이콘 파일, 기타 지원 파일등을 언패킹하는 단계, 이들은 장치(10)의 정적 저장소(MEM) 내의 지정된 장소에 저장하는 단계를 포함하고, 이들 모두 공지된 방식으로 본 발명의 교시 내에서 적응된다. 이런 어플리케이션 패키지 파일은 본 명세서에서 "네이티브" 어플리케이션 패키지 파일로 언급된다.

[0042] 호스트된 소프트웨어 어플리케이션(34,36)은 호스트된 런타임 환경(32) 하에서의 실행을 위한 ACL(18)의 제어하에서 인스톨된다(사용자 등의 지시에 따라). 이를 위해서, ACL(18)은 비록 본 명세서에서 설명한 것 처럼 변형된 호스트된 운영 체제에 대해 공지된 타입의 인스톨러 앱을 이용하여 어플리케이션 패키지 파일 등을 언팩하거나, 또는 그렇지 않으면 인스톨될 어플리케이션의 실행 파일, 아이콘 파일, 기타 지원 파일들을 장치(10)의 정적 저장소(MEM) 내의 적당한 장소, 예를 들면 호스트된 운영 체제와 일치하는 네이티브 운영 체제(14)에 의해 지정된 장소또는 다른 장소에 언팩한다. 이러한 어플리케이션 패키지 파일은 "호스트된" 어플리케이션 패키지 파일로 언급된다.

[0043] 다른 네이티브 소프트웨어 어플리케이션, 예를 들면 15와 18과 다르게, 각각의 호스트된 소프트웨어 어플리케이션의 ACL(18)에 의한 인스톨과 연관하여, 호스트된 소프트웨어 어플리케이션(34,36)의 프록시인 네이티브 소프트웨어 어플리케이션(46-52)이 ACL(18)로부터의 요청에 의해 네이티브 운영 체제(14)에 설치된다. 각각의 그런 프록시(46-52)는 공지된 방식으로, ACL(18)의 프록시 인스톨러 인터페이스(62)에 의해 생성된 어플리케이션 패키지 파일(또는 다른 것들)로부터 네이티브 운영 체제(14)에 의해 인스톨된다.

[0044] 이런 패키지 파일은 실행가능한 각각의 호스트된 소프트웨어 어플리케이션(34,36)의 대신에

[0045] i) 특히 네이티브 런타임 환경(16) 내에서 네이티브 운영 체제(14) 항의 실행에 적합한

[0046] ii) 런치 프록시 및 IO 프록시에 기인하는 아래 설명되는 기능에 효과적인

[0047] 실행가능한 스터브(stub)를 포함할 수 있다.

[0048] 이들 패키지 파일은 각각의 호스트된 소프트웨어 어플리케이션(34,36)에 대한 어플리케이션 패키지 파일(또는 다른 것)에 최초로 공급된 것들과 동일하거나 변형된 아이콘 파일을 포함할 수 있다. 비록 도시된 실시예에서, 두개의 프록시가 각각의 호스트된 소프트웨어 어플리케이션에 연관되지만, 하나의 아이콘이 도 1의 (A)의 그래픽 데스크 톱 상에서 양자의 플록시와 연관될 수도 있다.

[0049] 다중-운영 체제 모바일 및 기타 계산 장치

[0050] 계산 장치(10)는 다중 운영 체제의 어플리케이션의 끈임없는 실행을 지지하거나, 다른 말로, 이는 사용자 경험을 병합하여 호스트된 런타임 환경 내에 실행된 어플리케이션이 마치 사용자가 네이티브 운영 체제(14) 내에서 실행하는 것처럼 사용자에게 나타난다.

[0051] 즉, 예를 들면, 호스트된 소프트웨어 어플리케이션의 실행을 나타내는 어플리케이션 윈도우는 네이티브 운영 체제(14) 및/또는 네이티브 런타임 환경(16)에 의해 전체 그래픽 사용자 인터페이스의 일부로서 생성되는 "데스크탑"의 일부분을 형성하는 상태 바(status bar)와의 간섭없이 사용자에게 표시되고, 따라서 호스트된 소프트웨어 어플리케이션을 네이티브 소프트웨어 어플리케이션과 유사하게 만든다. 이는 예시적인 방식으로 도 1의 (A) 내지 (C)에 도시되었다.

[0052] 도 1의 (A)를 참조하면, 네이티브 운영 체제(14)는 컴퓨팅 장치를 구동하여 디스플레이/터치스크린(24) 상에, 장치(10)의 사용자에 의한 런칭 또는 기타 활성화를 위해 선택될 수 있는 어플리케이션들을 나타내는 아이콘(58)을 가진 그래픽 데스크탑을 표시시킨다. 도시된 실시예에서, 이들은 네이티브 소프트웨어 어플리케이션, 예를 들면 15 및 호스트된 소프트웨어 어플리케이션, 예를 들면 34, 36일 수 있다.

[0053] 데스크탑 디스플레이는 종래 공지된 타입의, 특히 네이티브 운영 체제(14)에 일반적인 상태 바(56)(일부 실시예는 이와 관련해서 변경될 수도 있음)를 포함한다. 여기서, 상태바(56)는 현재의 일자/시간, 반송파 신호 강도(carrier conductivity signal strength)(예를 들면, Wi-Fi, 셀룰러 등), 활성 앱 등을 나타낼 수 있지만, 다른 실시예에서 이는 다른 것을 나타낼 수도 있다.

[0054] 도 1의 (B)를 참조하면, 네이티브 소프트웨어 어플리케이션, 예를 들면 15가 운영 체제(14) 및/또는 런타임 환경(16)에 의해서 사용자 선택에 응답하여 활성화될 때, 스크린(24) 상에서의 표시를 위해 네이티브 런타임 환경(16)(어플리케이션의 실행을 반영함)에 의해 그에 대해 생성된 어플리케이션 윈도우(60)는 상태 바(56)와 함께 스크린을 점령하는데, 여기서 스크린의 최상부 부분(fraction) 상에 상태바(56)가, 나머지 부분 상에 어플리케이션 윈도우(60)가 표시된다. 다른 말로, 운영 체제(14) 및/또는 런타임 환경(16)은 어플리케이션 윈도우(60)에 상태 바(56)를 겹쳐쓰지 않는다(물론, 이는 운영 체제(14) 및/또는 런타임 환경(16)의 동작의 디폴트 모드(mode)임이 자명하고, 다른 모드, 소위 "풀 스크린 모드"에서 어플리케이션 윈도우(60)는 스크린의 전체를 차지한다).

[0055] 도 1의 (C)를 참조하면, 유사하게, 도시된 실시예에서, 호스트된 소프트웨어 어플리케이션(34,36)이 활성화될 때, 그에 대해 생성되는 어플리케이션 윈도우(호스트된 런타임 환경(32) 내에서의 실행을 반영함)가 네이티브 소프트웨어 어플리케이션의 것과 동일하게 스크린(24) 상에 표시되는데, 즉 상태바(56)를 겹쳐쓰기(overwrite) 없이 표시된다(예를 들면, 적어도 디폴트 모드에서 표시될 때).

[0056] 사용자 경험을 병합하여 호스트된 런타임 환경에서 실행된 어플리케이션이, 마치 그들이 네이티브 운영 체제(14) 내에서 실행되는 것처럼 사용자에게 나타나게 하는 도시된 컴퓨팅 장치(10)의 다른 예는, 이하의 설명이나 참조된 어플리케이션에 내포되어 있는, 예를 들면 네이티브 운영 체제(14) 및/또는 런타임 환경(16)의 것에 대해 일반적인 통지 메커니즘(common notification mechanism)을 사용하는 것이다.

[0057] 다른 실시예에서, 이하의 설명이나 참조된 어플리케이션에 내포되어 있는 것처럼, 이들이 네이티브 어플리케이션, 예를 들면 15이든지 또는 호스트된 소프트웨어 어플리케이션(34,36)이든지 간에, 통지에 대한 사용자 회신에 응답하여 실행중인 소프트웨어 어플리케이션의 일관적인 활성화(consistent activation)를 이용한다.

[0058] 또한 본 발명의 실시예에서, 전술한 바와 같이 호스트된 소프트웨어 어플리케이션과 네이티브 소프트웨어 어플리케이션 사이에서 일관적인 테밍(consistent theming)을 이용한다.

[0059] 다른 실시예들은 이하의 설명으로부터 당업자에게 자명하게 될 것이다.

[0060] 다중 운영 체제 모바일 및 기타 컴퓨팅 장치에서의 호스트된 어플리케이션 디스플레이

[0061] 이와 관련된 장치(10)의 오퍼레이션의 이해는 도 4의 참조에 의해 자명해질 수 있는데, 도 4는 네이티브 런타임 환경(16)에서 실행되는 앱의 런치 프록시(46)("App #1 Launch Stub"로 라벨됨)와의 사용자 상호작용에 기반한 호스트된 런타임 환경(32)내의 호스트된 소프트웨어 어플리케이션(34)(여기서 "App1"로 라벨됨)의 런칭, 앱의 IO 프록시("App #1 IO stub"로 라벨됨)를 통해 호스트된 소프트웨어 어플리케이션(34)의 오퍼레이션을 나타내는 어플리케이션 위도우의 디스플레이, 및 그 프록시(50)로부터의 사용자 입력을 앱(34)으로 전송함에 있어서, 전술하여 설명한 구성요소의 상호 상용을 나타낸다.

[0062] 단계 64를 설명하기에 앞서, 네이티브 런타임 환경(16)(및/또는 네이트브 운영 체제(14))는, 장치(10)의 사용자에 의한 런치 또는 기타 활성화시 선택될 수 있는 네이티브 및 호스트된 소프트웨어 어플리케이션을 나타내는 아이콘(58)을 그래픽 데스크탑(예를 들면 도 1의 (A) 참조)에 표시한다. 전술한 바와 같이, 이들 아이콘은 앱 각각의 인스톨과 연관되어 네이티브 런타임 환경(16) 및/또는 네이티브 운영 체제(14)에 제공될 수 있다.

[0063] 네이티브 운영 체제(14)의 타입의 운영 체제의 관례에 따라, 런치 프록시(46)인 네이티브 소프트웨어 어플리케이션은 사용자에 의한 활성화의 선택에 의존하여 네이티브 런타임 환경(16) 및/또는 네이티브 운영 체제(14)에 의해 런칭된다. 단계 64를 참조하기 바란다. 프록시(50)는 네이티브 런타임 환경(16) 및/또는 네이티브 운영 체제(14)에 의해 동시적으로 런칭될 수 있다. 대안적으로 프록시(50)는 그 런치 ID에 따라 프록시(46)에 의해 런치될 수도 있다.

[0064] 런치됨에 따라(또는 네이티브 런타임 환경(16) 및/또는 네이티브 운영 체제(14)로부터의 활성화의 다른 통지(notification)), 프록시(46)는 대응하는 호스트된 소프트웨어 어플리케이션(34)의 활성화를 가져온다. 단계 66을 참조 바람.

[0065] 도시된 실시예에서, 프록시(46)는 호스트된 런타임 환경(32)의 일부분을 형성하고 하나 이상의 호스트된 소프트웨어 어플리케이션(34,36)(예를 들면 적당한 호스트된 어플리케이션 또는 호스트된 런타임 환경(32)내에서 실행되거나 또는 호스트된 운영 체제의 일부분으로서 제공된 다른 소프트웨어를 분배하는 이벤트, 하드웨어 인터페이스로의 사용자 입력과 같은 시스템 레벨-이벤트의 일반적인 공유된 수신(recipient)이기 때문)에 대해 일반적인 이벤트 핸들러(42)에 런치 메시지를 전송하는 것으로 이를 수행한다. 인터 프로세스 통신(IPC), 예를들면 APIs, 메일박스(mailboxes) 등의 공지의 메커니즘을 이용하여 프록시(46)에 의해 이벤트 핸들러(42)로 전달될 수 있는 런치 메시지는 프록시(46)의 식별자 및/또는 그 대응하는 호스트된 소프트웨어 어플리케이션과, 호스트된 소트트웨어 어플리케이션의 런치에 기여하는 호스트된 운영 체제 및/또는 호스트된 런타임 환경(32)에 의해 필요한 다른 정보를 포함할 수 있다.

[0066] 단계 68에서, 이벤트 핸들러(42)는 호스트된 운영 체제 및/또는 호스트된 런타임 환경(32)에 요구되는 공지의 방식으로 호스트된 소프트웨어 어플리케이션(34)을 런치한다. 즉, 그 앱(34)은 마치 장치(10)의 사용자에 의해 직접적으로 선택된 것 처럼 런칭된다.

[0067] 호스트 소프트웨어 어플리케이션(34)의 런치에 이어서, 이벤트 핸들러(42)는 예를 들면 전술한 바와 같은 IPC를 이용하여 그 호스트된 소프트웨어 어플리케이션(34)이 실행을 시작되었음을 신호하고, 보다 바람직하게, 네이티브 런타임 환경(16)에서 프록시 어플리케이션(50)의 런치 및 활성화를 보장한다. 단계 70 참조.

[0068] 런치에 이어서, 호스트된 소프트웨어 어플리케이션(34)은 호스트된 런타임 환경(32) 내에서 공지의 방식으로 동작하고, 호스트된 운영 체제의 타입의 하나의 운영 체제를 실행하는 장치 상에 인스톨되어진 것 처럼, 예를 들면 비제한적인 방식의 예로, 호스트된 리소스 프레임워크(38), 호스트된 이벤트 핸들러(42) 및 런타임 라이브러리(44)에 콜(call)을 한다. 이는 소프트웨어 어플리케이션을 다중 운영 체제 환경의 장치(10)에서 실행가능하도록 하기 위해 개발자 또는 배포자에 의해 호스트된 소프트웨어 어플리케이션의 특별한 리코딩(즉, 포팅(porting))을 필요로하지 않는다는 점에서 유리하다.

[0069] 호스트된 리소스 프레임워크(38), 호스트된 이벤트 핸들러(42), 런타임 라이브러리(44) 및 호스트된 런타임 환경(32)의 다른 구성요소는 호스트된 운영 체제의 타입의 운영 체제의 기술에 공지된 방식으로, 본 발명의 교시에 따라 적응되어 그런 콜에 응답한다. 따라서, 예를 들면, 전술한 바와 같이, 호스트된 런타임 환경(32)에 의해 특정된 커널 공간(privileged kernel space)에서 실행될 수 있는 타입의 일부 그러한 오퍼레이션(예를 들면, 프레임 버퍼를 로딩하기 위한 것들)은, 대신에, 사용자 공간에서 실행된다. 그리고 다른 예시적인 방식에 의해, 다른 그런 오퍼레이션(또는 보다 넓게는 "기능")은 프록시 46-52를 통해 네이티브 운영 체제(14)와 그 런타임 환경(16)에 대해 패스-오프(passed-off)된다.

[0070] 예시적인 방식으로, 호스트 소프트웨어 어플리케이션(34)의 실행을 나타내는 어플리케이션 윈도우를 정의하는 그래픽으로 실제 프레임 버퍼를 로딩하는 것 대신에, 호스트된 런타임 환경(32)은 그와 같은 그래픽으로 가상 프레임 버퍼(54)를 로딩한다. 단계 72 참조. 호스트된 런타임 환경(32)는 호스트 런타임 환경(32)의 일부를 형성하고 하나 이상의 호스트된 소프트웨어 어플리케이션(34,36)에 일반적인 윈도우 서브시스템의 사용을 통해 이를 유발한다(이는 장치(10)의 사용자에게 표시하기 위한 어플리케이션 윈도우를 생성하는 호스트된 소프트웨어 어플리케이션에 의해 일반적이고 공유된 시스템이 사용되기 때문임).

[0071] 호스트된 소프트웨어 어플리케이션(34)의 IO 프록시(50)는, 도1의 (C)에 도시된 방식으로 그리고 연관하여 설명한 바와 같이, 호스트된 런타임 환경(32)에 의해 어플리케이션(34)에 대해 생성된 어플리케이션 윈도우의 스크린(24) 상의 표시에 영향을 준다(effect). 단계 74 참조. IO 프록시(50)는 어플리케이션 윈도우를 정의하는 그래픽을 가상 프레임 버퍼(54)로부터 네이티브 프레임 버퍼(26)로, 네이티브 런타임 환경(16)에 의해 제공되는 이런 목적 또는 다른 목적을 위한 API를 이용하여 전송함으로써 이를 수행한다. 일부 실시예에서, 호스트된 런타임 환경(32)은 예를 들면 호스트된 런타임 환경(32)의 윈도우 서브시스템이 호스트된 소프트웨어 어플리케이션(34)에 대한 업데이트된 어플리케이션 윈도우를 생성했을 때, 호스트된 소프트웨어 어플리케이션(34)이 호스트된 런타임 환경(32)에서 앱을 활성(또는 포그라운딩(foreground)할 때와 같이 그러한 전송에 대한 필요성을 IO 프록시(50)에 경보하기 위해 메시징을 이용하지만, 다른 실시예에서 IO 프록시(50)는 주기적인 기준(periodic basis) 또는 다른 방식에 따라 자신의 합의(accord) 상에서 그런 전송에 영향을 준다.

[0072] 다중 운영 체제 모바일 및 기타 컴퓨팅 장치에서의 사용자/호스트된 어플리케이션 상호작용

[0073] IO 프록시(50)는

[0074] 예를 들면 디스플레이/터치 스크린(24), 키보드, 트랙볼, 터치 스틱, 기타 사용자 입력 장치 등의 장치로의 사용에 의해 만들어진 탭(taps) 및 기타 입력을 전송하기 위해서 단계 64-68과 연관하여 설명된 것에 상응하는 메커니즘을 이용한다. 이런 관점에서, 일반적인 이벤트 핸들러(미도시) 또는 네이티브 런타임 환경(16)의 다른 기능은, 터치 스크린(24) 또는 기타 다른 입력 장치를 통해 그들에 대해 이루어진 사용자의 입력을, IO 프록시(50,52)를 포함하여 그들내에서 실행되는 어플리케이션에 통지한다. 그러한 통지는 본 발명의 교시에 따라 적응되는, 네이티브 운영 체제(14)의 타입의 운영 체제의 기술분야에서 공지되어 있다.

[0075] IO 프록시(50)가 이런 통지를 수신하면, 이는 그와 관련된 정보를 단계 66에서 연관하여 설명한 것과 유사한 방식으로 이벤트 핸들러(42)를 통해서 대응하는 호스트 소프트웨어 어플리케이션(34)에 전송한다. 단계 76 참조. 종래의 어떠한 IPC 메커니즘을 사용하여 IO프록시(50)에 의해 이벤트 핸들러(42)로 전달될 수 있는 그런 정보는 IO 프록시(50)의 식별자 및/또는 그 대응하는 호스트 소프트웨어 어플리케이션(34), 입력이 이루어진 장치의 식

별자, 입력의 타입, 그와 관련된 정보(예를 들면, 위치, 시간, 주기, 터치의 타입, 탭된 키, 포인터상의 압력 등등)을 포함할 수 있다. 정보는 이벤트 핸들러(42)에 의해 수신되고, 터치 또는 다른 사용자 입력이 호스트된 소프트웨어 어플리케이션(34)에 직접 이루어진 것 처럼, 호스트된 운영 체제 및/또는 호스트된 런타임 환경(32)에 요구되는 전통적인 방식으로 대응하는 호스트 소프트웨어 어플리케이션(34)에 인가된다. 단계 78 참조.

[0076] 인-프로세스 트랩핑(In-Process Trapping)

[0077] 전술한 바와 같이, 마치 어플리케이션이 호스트된 운영 체제의 타입의 단일 운영 체제를 실행하는 장치상에 인스톨된 것 같이 다르게 참조하는 것 처럼, 호스트된 소프트웨어 어플리케이션(34)은 호스트된 리소스 프레임워크(38), 호스트된 이벤트 핸들러(42) 및 런-타임 라이브러리(44) 등에 그러한 콜(call)을 만든다. 또한 도 5에 도시된 것 처럼, 어플리케이션(34)은 데이터 멤버 및/또는 방법 멤버(35A-35C)를 참조하는 각각이 OOP 개체의 타입의 명령어(34A-34C)를 포함하고, 어플리케이션은 마치 그런 장치 상에 인스톨되어진 것 처럼 참조하게 된다. 그러한 참조된 멤버(35A-35C)는 어플리케이션(34)에 대해 외부인(즉 로컬이 아닌) 데이터, 기능 또는 서비스에 액세스할 수 있다. 여기서 이들은 편리를 위해 로컬 데이터 및 기능으로서 도시되었다.

[0078] 어플리케이션이 호스트된 운영 체제의 타입의 단일 운영 체제를 실행하는 장치 상에 인스톨되어진 것과 같이 다르게 동작하는 것 처럼, 어플리케이션(34)이 그런 콜을 만들고 그런 객체 및/또는 멤버를 참조하는 것은, 개발자 또는 배포자가 장치(10)의 다중 운영 체제 환경에서 이를 실행하도록 하기 위해 개발자 또는 배포자에 의한 호스트된 소프트웨어 어플리케이션(34)의 특별한 리코딩(즉 포팅)을 필요로하지 않는 점에서 유리하다.

[0079] 호스트된 프레임 워크(38), 호스트된 이벤트 핸들러(42), 및 런타임 라이브러리(44)는 본 발명의 교시에 따라 적응된 바와 같은 호스트된 운영 체제의 타입의 운영 체제의 공지된 방식으로 어플리케이션(34)에 의해 콜에 응답한다.

[0080] 즉, 예를 들면, 전술한 바와 같이, 프레임워크(38), 핸들러(42) 및 라이브러리(44)는 그런 운영 체제에 의해 일부 메커니즘을 통해 통상적으로 실행될 수 있는 일부 오퍼레이션들이, 다른 메커니즘(네이티브 소프트웨어 어플리케이션을 서비스하는데 이용되는 것들과 일치하는 것들)에 의한 실행을 위해, 프록시(46-52) 또는 다른 것을 통해서, 네이티브 운영 체제(14)와 그 런타임 환경(16)에 대해 패스-오프되도록 적응된다. 또한, 전술한 바와 같이, 특정 커널 공간에서 통상적으로 실행될 수 있는 타입의 일부 오퍼레이션(예를 들면 프레임 버퍼를 로딩하기 위한 것들)은 그 대신에 비제한적인 방식에 의해 사용자 공간에서 실행된다.

[0081] 그런 대안적인 메커니즘에 의해 호스트된 소프트웨어 어플리케이션(34)에 의한 그런 콜의 프로세싱은 (a) 호스트된 리소스 프레임워크(38), 호스트된 이벤트 핸들러(42), 런-타임 라이브러리(44) 및/또는 기타 그것(어플리케이션)이 콜하는 프로세스외 코드(out-of-process code)의 것들보다는 개별적인 프로세스로서 실행되고, (b) 이들 콜이 따라서 본 발명의 교시에 적등된 바와 같이 공지된 방식으로 인터셉트 또는 "트랩"될 수 있다는 사실에 의해 가능하게 된다.

[0082] 그러나, 객체의 데이터 및/또는 방법 멤버(35A-35C)를 참조하는 어플리케이션(34)에 의한 명령어(34A-34C)의 실행은 프로세스중(in-process)에 일어나기 때문에, 도시된 장치(10)는 통상적으로 호스트된 리소스 프레임워크(38), 호스트된 이벤트 핸들러(42), 런타임 라이브러리(44) 또는 이들 명령어를 트랩하기 위한 프로세스외 코드에 의존할 수 없다. 이를 수행하기 위해, 도시된 장치(10)는 데이터를 액세스하고 및/또는 그들 명령어에 의해 특정된 이외의 코드를 실행하는 것에 의해 선택된 그런 명령어에 응답한다.

[0083] 보다 구체적으로, 도시된 실시예에서, 장치(10)는, 이들 명령어에 의해 참조된 클래스 및 멤버에 의해 특정된 것 이외의 데이터 및/또는 코드(이하 "대체 멤버(35A'-35C')"라고 함)를 활용하여 이들 명령어를 실행하는 것에

의해, 객체 지향 프로그래밍 클래스("참조 클래스")에 의해 정의된 객체의 적어도 선택된 멤버("참조 멤버")를 참조하는 어플리케이션(34)내의 명령어(34A-34C)에 응답한다. 도시된 실시예에서, 장치(10)는 대체 멤버(35A'-35C')(참조 멤버 35A-35C를 대신하여)를 어플리케이션(34)에 로드하는 것에 의해 이를 달성한다. 이는 로드 타임(load time)에 이루어지는 반면 호스트된 런타임 환경(32)는, 예를 들면 네이티브 어플리케이션(18)(예를 들면 ACL) 및/또는 장치(10)가 부팅될 때 초기화된다. 다른 실시예에서, 이는 어플리케이션(34) 설치 및/또는 실행 시간에서 수행될 수 있다.

[0084] 편리를 위해 단지 하나의 참조 클래스 37이 도시되고 있지만 복수의 참조 클래스가 사용될 수 있다는 것은 자명하다.

[0085] 대체, 전술한 바와 같이 선택된 참조 멤버(35A-35C)를 참조하는 어플리케이션(34)의 명령어에 대한 대체 부재(35A'-35C')의 액세스 및 실행에 의해, 장치(10)는 어플리케이션(34)에 의해 그들 참조된 멤버(35A-35C)를 인터셉트 또는 트랩할 수 있다. 이는 장치(10)로 하여금, 참조된 멤버(35A-35C)를 통해 통해 활용되는 것 보다, 예를 들면 어플리케이션(34)이 데이터, 기능, 및/또는 서비스에 대한 액세스를 제공하도록 상이한 메커니즘을 활용하는 것을 허용한다.

[0086] 참조 클래스(37)는 비제한적인 예의 방식으로 호스트된 운영 체제의 버전을 촉진시키는 기업 또는 엔티티/조직에 의해 공개되는 하나 이상의 클래스일 수 있다. 또한, 선택된 참조 멤버(35A-35C)는 비제한적인 예의 방식으로 데이터, 기능 및/또는 어플리케이션(34)이 실행되어지는 프로세스에 대해 로컬이 아닌 서비스를 액세스하는 클래스의 멤버들, 및/또는 독점 데이터, 기능 및/또는 다른 기업, 엔티티, 조직의 서비스일 수 있다.

[0087] 대체 맴버(35A'-35C')는 호스트된 런타임 환경(32)이 초기화될 때(예를 들면 네이티브 어플리케이션(18) 및/또는 장치(10)의 부팅업), 어플리케이션(34)의 설치시, 장치(10)의 사용자에 의해 그 어플리케이션의 호출시(invocation) 어플리케이션(4)을 로딩하기 적합한 임의의 데이터 및/또는 코드를 포함할 수 있다. 코드-기반 대체 멤버는 NOOP 오퍼레이션(no 오퍼레이션)를 최소한 포함하는 반면, 이들은 그에 대해 상이한 메커니즘을 이용하지만 대체되어지는 참조 멤버와 유사한 기능성을 제공한다.

[0088] 예시적으로, 참조 멤버, 예를 들면 35A가 서드파티(예를 들면, 호스트된 운영 체제의 버전을 촉진하는 기업 또는 기타 엔티티/조직 등)의 독점적인 웹-기반 맵핑 서비스(100)를 호출(invoke)하는 호스트된 어플리케이션(34)의 객체(클래스(37)로부터 예시됨)의 방법 멤버인 경우, 대응하는 대체 멤버, 예를 들면 35A'가 다른 파티의 대안적인 맵핑 서비스(102)를 호출할 수 있고, 대안적으로 또는 그에 더하여, 장치(10)에 대해 로컬인, 보다 구체적으로는 호스트된 런타임 환경(32), 프레임워크(38), 호스트된 핸들러(42), 호스트된 라이브러리(44), 네이티브 런타임 환경(16), 네이티브 런타임 라이브러리(20), 또는 다른 것에 대해 로컬인 기능 및/또는 서비스을 호출한다.

[0089] 다른 예에 있어서, 참조 멤버, 예를 들면 35B가 (객체가)부분적으로 또는 전체적으로 작동불능인 호스트된 어플리케이션의 객체(클래스 37로부터 예시됨)의 데이터 또는 방법 멤버인 경우, 장치(10)에 대해, 보다 구체적으로는 호스트된 런타임 환경(32), 프레임워크(38), 호스트된 핸들러(42), 호스트된 라이브러리(44), 네이티브 런타임 환경(16), 네이티브 런타임 라이브러리(20)에 대해 로컬이거나 로컬이 아닌 대안적이면서 동작가능한 메커니즘을 활용하여대응하는 대체 멤버(35B')는 필수적인 데이터 및/또는 기능/서비스를 그 어플리케이션(34)에 제공할 수 있다.

[0090] 본 발명의 일부 양태에 따른 전술한 상이한 메커니즘은 상이한 웹(또는 다른 외적인) 서비스일 수 있다. 즉, 예를 들면, 참조 멤버는 참조 클래스에 의해 정의된 방법 멤버일 수 있고, 맵핑, 네이게이션, 광고, 디지털 월렛, 계정 인증 및/또는 제1 호스트된 소프트웨어 어플리케이션에서 요구하는 다른 서비스를 지원하기 위해 그 클래

스가 제1 외부 웹 서비스를 활용하는 객체 인스턴스(object instantiations)일 수 있고, 반면 대체 멤버는 그러한 서비스를 지원하기 위해 제2 상이한 웹 서비스를 활용하는 대체 클래스로부터 예시되는 객체의 방법 멤버일 수 있다.

[0091] 그들 각각의 클래스가 대응하는 참조 멤버의 포맷과 동일한 포맷으로 값의 저장 및/또는 검색을 허용하는 데이터 멤버인 참조 멤버를 대체해서, 대체 멤버(35A'-35C')는 장치(10)에 의해 실행된다.

[0092] 유사하게, 기능 또는 서비스를 제공 또는 호출하는 대체 멤버(35A'-35C')는 입력 인수(input arguments)를 수용하고 어플리케이션(34)에 의해 기대되는 형태로 출력 인수를 리턴한다. 그들에 의해 채용된 상이한 메커니즘이 필요하지 않기 때문에(대체시 그들이 실행된는 참조 멤버에 대해), 그들 자체적으로 동일한 포맷으로 인수를 수용하거나 리턴하고, 일부 실시예에서 대체 멤버는 변환(conversions), 리포맷(reformatting) 등에 필요한 코드를 포함할 수 있다.

[0093] 전술한 바와 같이, 도시된 실시예에서, 대체 멤버에 대한 리퍼런스는, 예를 들면 호스트된 런타임 환경(32)이 초기화될 때(네이티브 어플리케이션(18) 및/또는 장치(10)의 부트 업), 어플리케이션(34) 설치 및/또는 실행시, 호스트된 소프트웨어 어플리케이션(34)에서 대응하는 참조 멤버에 대해 대체된다. 이는 도 5에 도시되었는데, 참조 멤버(35A)에 대해 명령어(34A)에 의해 만들어진 리퍼런스는 대체 멤버(35A')에 대한 리퍼런스로 대체되고, 참조 멤버(35B)에 대해 명령어(34B)에 의해 만들어진 리퍼런스는 대체 멤버(35B')에 대한 리퍼런스로 대체되고, 참조 멤버(35C)에 대해 명령어(34C)에 의해 만들어진 리퍼런스는 대체 멤버(35C')에 대한 리퍼런스로 대체된다.

[0094] 이런 대체가 호스트된 런타임 환경(32)의 초기화 동안 발생된 경우, 이들 환경(예를 들면, ACL)을 지원하는 네이티브 어플리케이션(18)은, 리퍼런스 대체 클래스로 변형되었다 하더라도, 호스트된 시스템의 타입의 운영 체제에 통상적으로 이용되는 타입의 클래스 로더(class loader)(104)를 이용한다. 이런 대체가 설치 시간에 발생된 경우, 호스트된 운영 체제의 인스톨러 앱은 호스트된 앱 설치와 연관되어 통상적으로 이용되는 타입의 로더(104)를 호출할 수 있으며, 따라서 실행가능한 호스트된 소프트웨어 앱 내의 심볼 리퍼런스는 참조 멤버보다는 대체 맴버를 분해(resolve)한다. 유사하게 대체가 실행 시간에 발생된 경우, 이벤트 핸들러(42)는 호스트된 앱 호출과 관련하여 본 발명의 교시에 따라 적응된 일반적으로 이용되는 타입의 로더(104)를 호출한다. 모든 예에서, 실행가능한 호스트된 소프트웨어 어플리케이션(108)과 함께, 라이브러리 루틴(110)을 로딩하기 위해, 대체 맴버(106)에 대한 코드는 클래스 로더(104)에 모든 공지되어 있는 방식으로 제공될 수 있다.

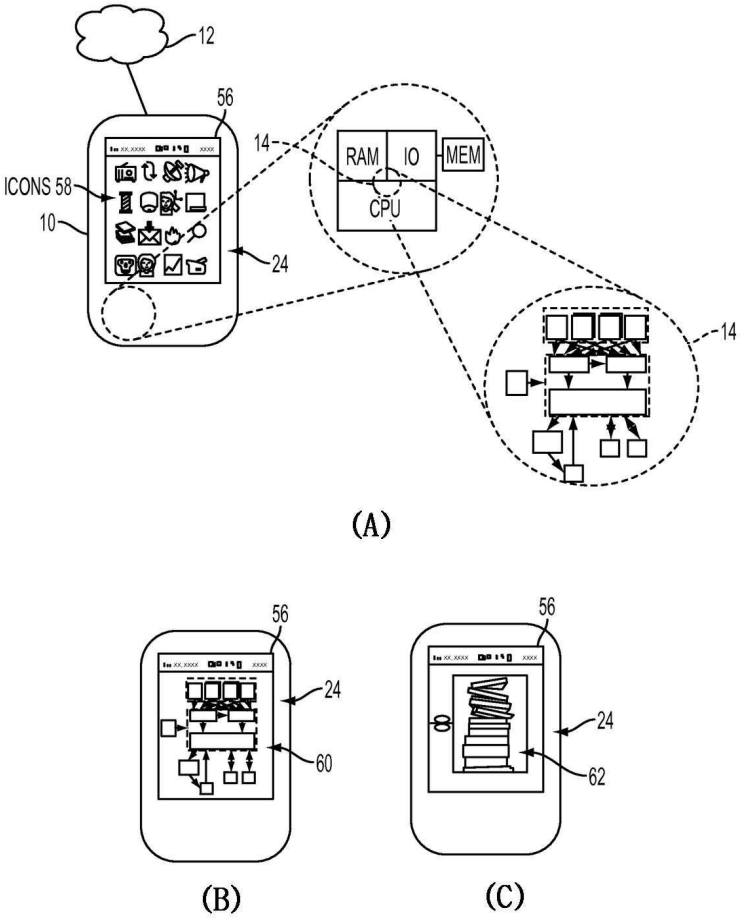
[0095] 본 발명에 따른 시스템의 동작에 대한 보다 완전한 이해는 "MULTI-PLATFORM MOBILE AND OTHER COMPUTING DEVICE AND METHOD"라는 명칭으로, 2013년 10월 23일자에 출원된 미국특허출원 14/061,288호(현재는 미국 공개공보 US2014-0115606 공개됨)와, 동일한 명칭으로 2013년 10월 18일자에 출원된 미국특허출원 61/892,896호를 참조하여 달성될 수 있으며, 양출원 모두 본 명세서에 참조로서 내포된다.

[0096] 결론

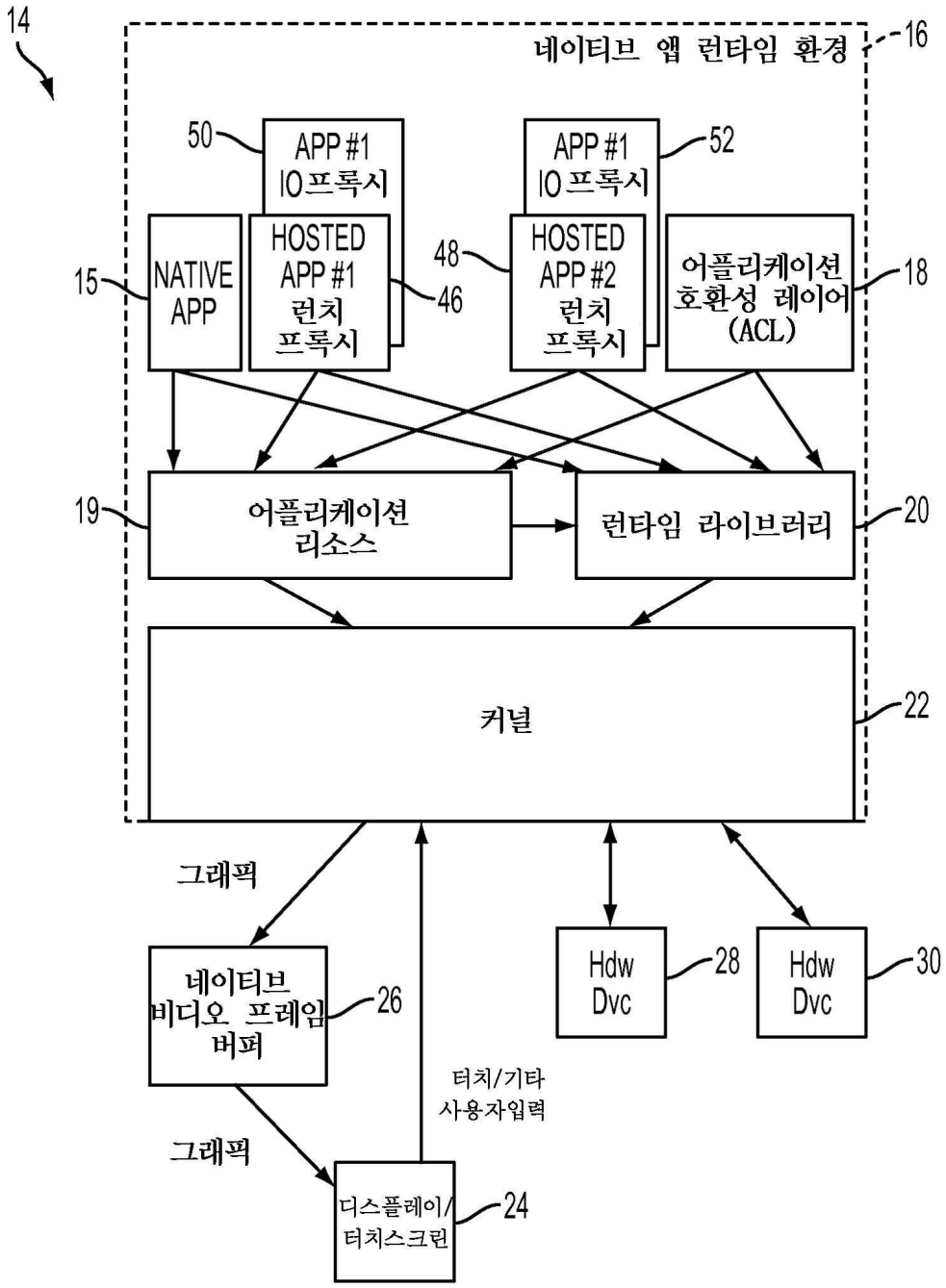
[0097] 도면을 참조하여 전술한 장치 및 방법에 의해 전술한 과제는 달성될 수 있다. 당업자라면 전술하여 설명한 실시예는 본 발명을 실시하기 위한 예일 뿐이며 다른 실시예들이 본 발명의 범위 내에서 이루어질 수 있다는 것을 이해할 수 있을 것이다.

도면

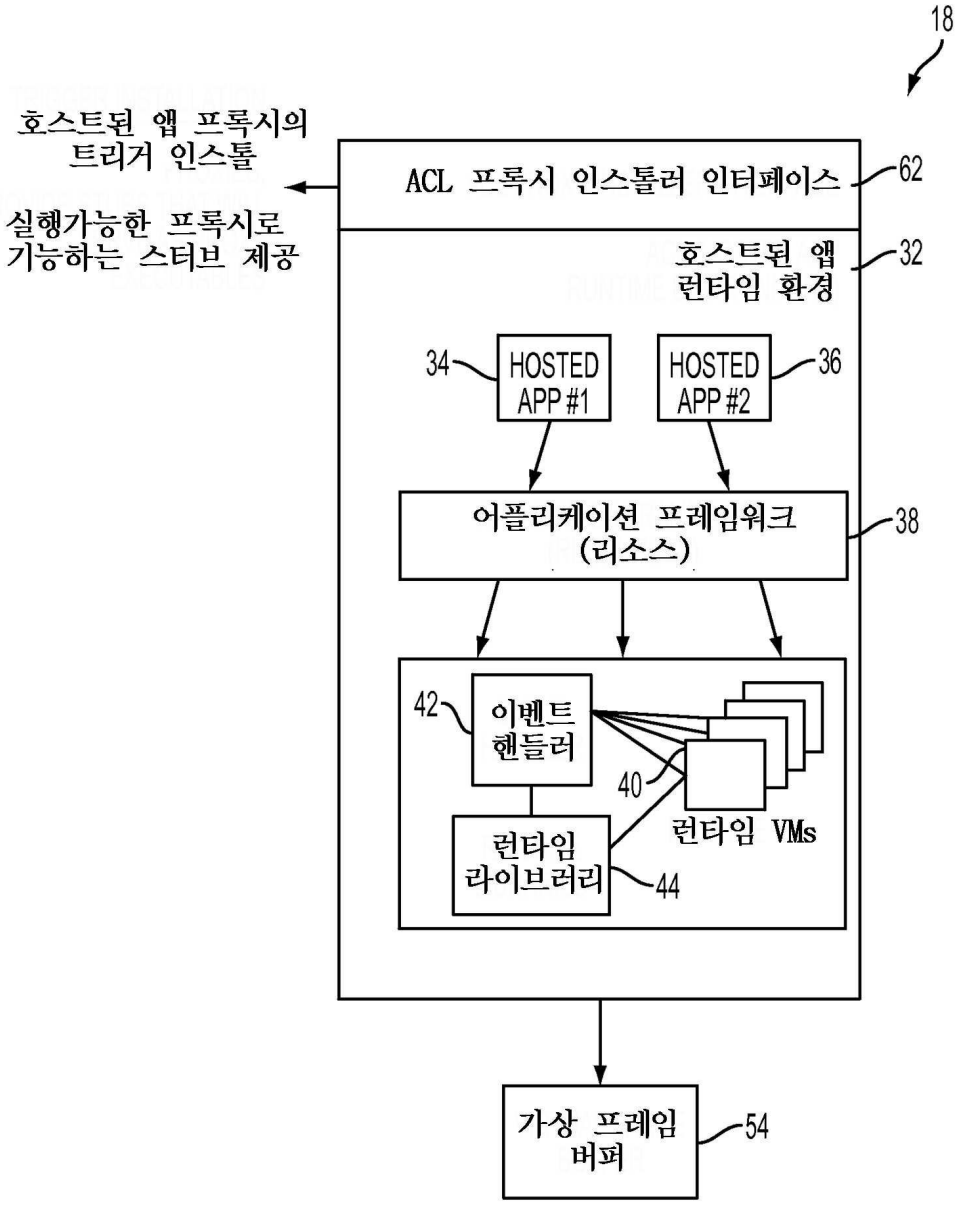
도면1



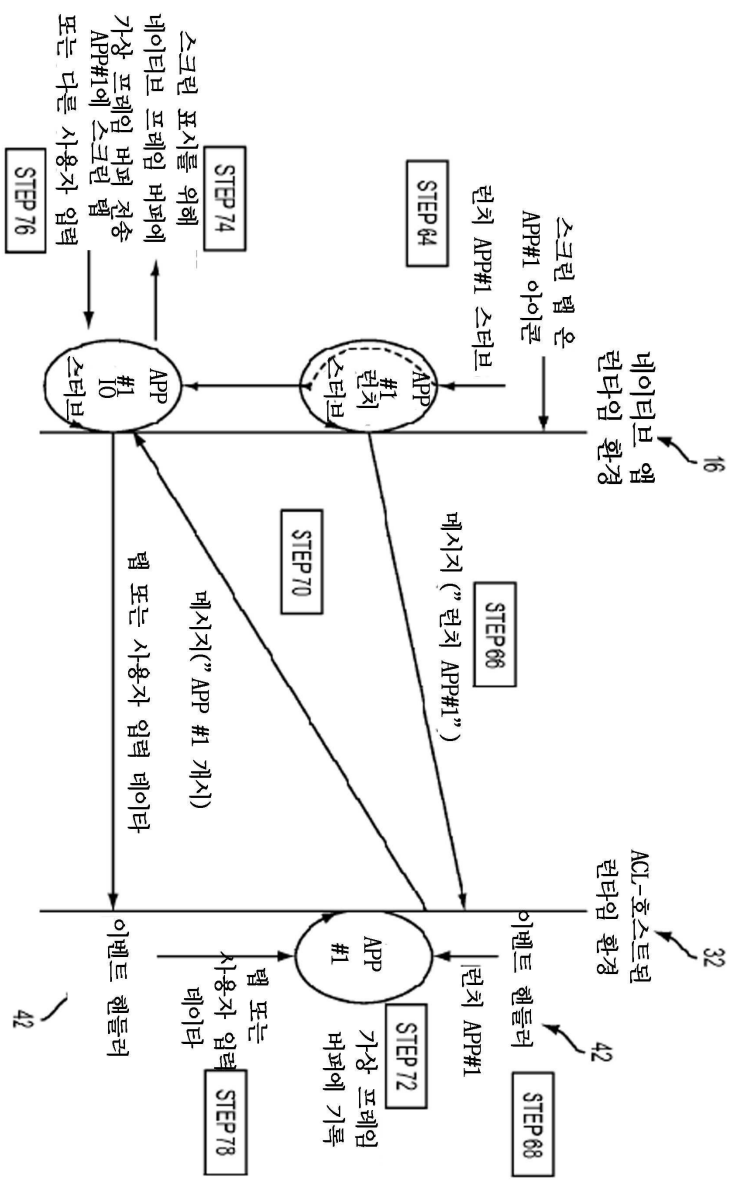
도면2


14
네이티브 앱 런타임 환경
16
50
APP #1
IO프록시
52
APP #1
IO프록시
15
NATIVE
APP
HOSTED
APP #1
런치
프록시
46
48
HOSTED
APP #2
런치
프록시
어플리케이션
호환성 레이어
(ACL)
18
19
어플리케이션
리소스
런타임 라이브러리
20
커널
22
그래픽
네이티브
비디오 프레임
버퍼
26
Hdw
Dvc
28
Hdw
Dvc
30
터치/기타
사용자입력
그래픽
디스플레이/
터치스크린
24

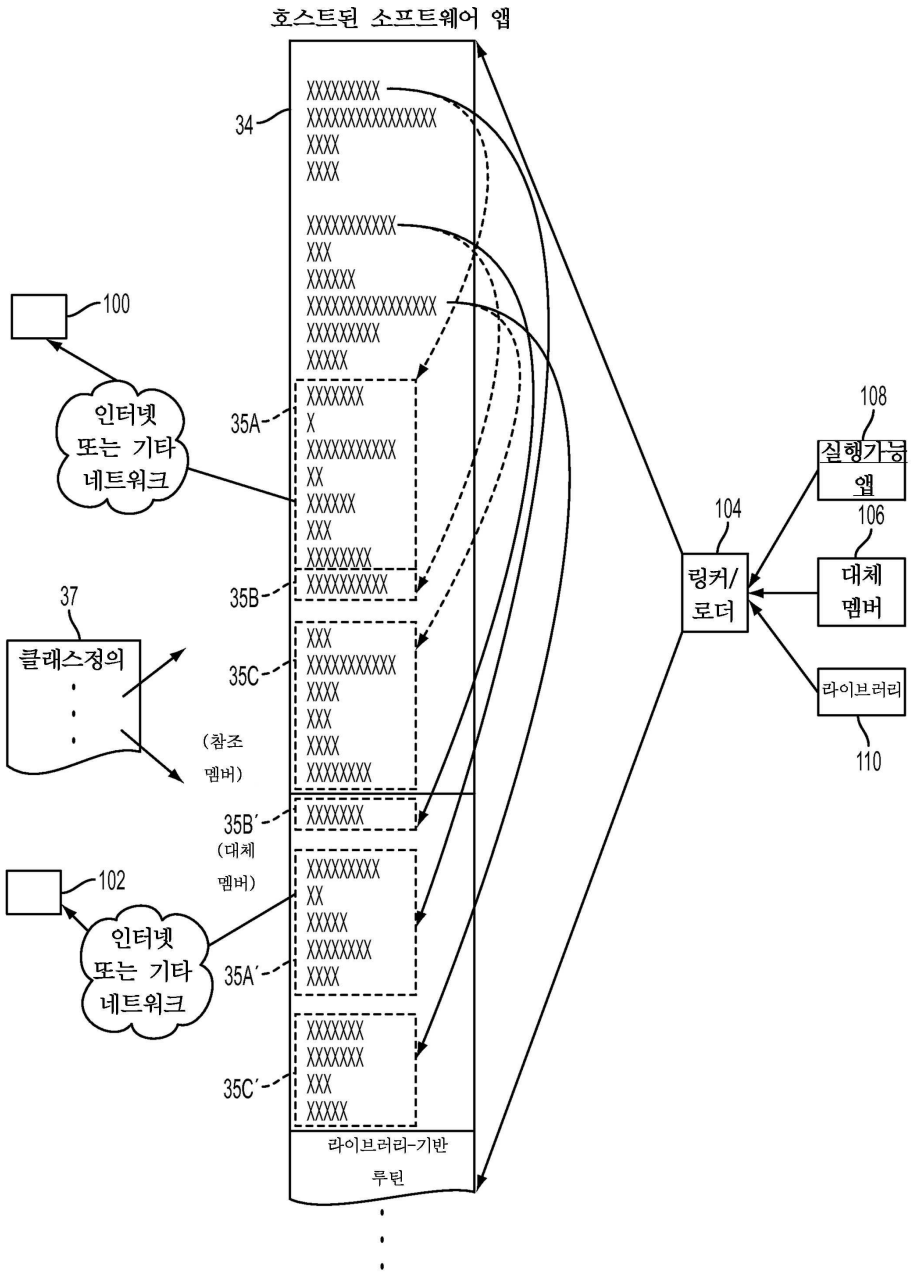
도면3


18
호스트된 앱 프록시의 트리거 인스톨
실행가능한 프록시로 기능하는 스터브 제공
ACL 프록시 인스톨러 인터페이스
62
호스트된 앱 런타임 환경
32
34
HOSTED APP #1
HOSTED APP #2
36
어플리케이션 프레임워크 (리소스)
38
42
이벤트 핸들러
40
런타임 VMs
런타임 라이브러리
44
가상 프레임 버퍼
54

도면4


네이티브 앱
런타임 환경
16
스크린 탭 온
APP#1 아이콘
런치 APP#1 스타브
STEP 64
APP
#1
런치
스타브
APP
#1
10
스타브
STEP 66
메시지 (" 런치 APP#1")
STEP 70
메시지(" APP #1 개시)
탭 또는 사용자 입력 데이타
ACL-호스트된
런타임 환경
32
이벤트 핸들러
42
런치 APP#1
STEP 68
APP
#1
STEP 72
가상 프레임
버퍼에 기록
탭 또는
사용자 입력
데이타
STEP 78
이벤트 핸들러
42
스크린 표시를 위해
네이티브 프레임 버퍼에
가상 프레임 버퍼
APP#1에 스크린 탭
또는 다른 사용자 입력
STEP 74
STEP 76

도면5


호스트된 소프트웨어 앱
34
100
인터넷 또는 기타 네트워크
35A
37
클래스정의
35B
35C
(참조 멤버)
35B′
(대체 멤버)
102
인터넷 또는 기타 네트워크
35A′
35C′
라이브러리-기반 루틴
104
링커/로더
108
실행가능 앱
106
대체 멤버
라이브러리
110