(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2019년12월26일
(11) 등록번호 10-2059705
(24) 등록일자 2019년12월19일

(51) 국제특허분류(Int. Cl.)
G06F 9/445 (2018.01)
(21) 출원번호 10-2014-7035504
(22) 출원일자(국제) 2013년05월28일
심사청구일자 2018년04월23일
(85) 번역문제출일자 2014년12월17일
(65) 공개번호 10-2015-0024842
(43) 공개일자 2015년03월09일
(86) 국제출원번호 PCT/US2013/042791
(87) 국제공개번호 WO 2013/191852
국제공개일자 2013년12월27일
(30) 우선권주장
13/525,356 2012년06월18일 미국(US)
(56) 선행기술조사문헌
KR1020010072477 A
KR1020070062606 A
KR1020110113196 A

(73) 특허권자
마이크로소프트 테크놀로지 라이센싱, 엘엘씨
미국 워싱턴주 (우편번호 : 98052) 레드몬드 원 마이크로소프트 웨이
(72) 발명자
트로핀 미르체아
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
크왈리나 크르지스초프
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
더서드 패트릭 에이치
미국 워싱턴주 98052-6399 레드몬드 원 마이크로소프트 웨이 엘씨에이 - 인터내셔널 페이턴츠 마이크로소프트 코포레이션
(74) 대리인
제일특허법인(유)

전체 청구항 수 : 총 10 항 심사관 : 임재우

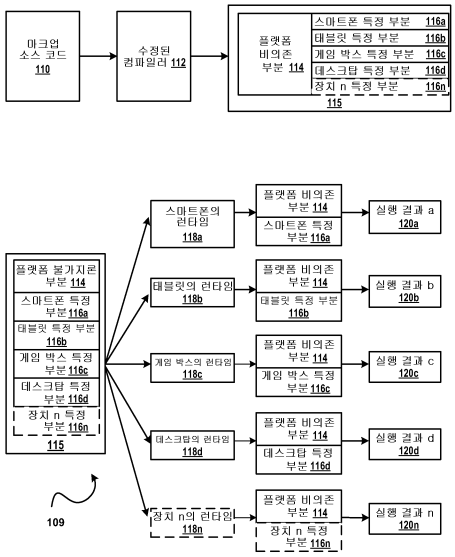
(54) 발명의 명칭 **적응식 이식가능 라이브러리**

(57) 요 약

플랫폼 비의존 소스 코드는 상이한 플랫폼 및/또는 버전에 대해 어노테이션이 달리는 소스 코드의 하나 이상의 부분을 이용하여 확장될 수 있다. 플랫폼 비의존 및 플랫폼 특정적 및/또는 버전 특정적 부분을 포함하는 소스 코드는 다수의 장치 플랫폼에 및/또는 하나 이상의 플랫폼의 여러 버전에 분배될 수 있고 또한 그 상에서 실행될

(뒷면에 계속)

대 표 도 - 도1b



수 있는 단일 실행 파일을 생성하기 위해 한번 컴파일링될 수 있다. 플랫폼 특정 또는 버전 특정 실행 파일(예를 들어, 어셈블리, 바이너리 등)은 리소스(예를 들어, 데이터)로서 바이너리 또는 실행 파일에 내장될 수 있다. 컴파일링된 코드가 장치상에서 실행되는 경우, 런타임은 실행 파일이 실행되는 장치에 대응하는 플랫폼 특정 데이터를 추출할 수 있고, 이 추출된 데이터를 실행 파일에 결합시킬 수 있으며, 실행 파일을 로딩하여 실행할 수 있다. 따라서, 동일한 바이너리는 바이너리가 실행되는 플랫폼의 성능에 따라 다른 결과 또는 기능을 생성할 수 있다.

명 세 서

청구범위

청구항 1

컴퓨팅 장치의 적어도 하나의 프로세서와,

상기 컴퓨팅 장치의 상기 적어도 하나의 프로세서에 통신가능하게 결합된 메모리와,

상기 메모리에 로딩된 적어도 하나의 모듈

을 포함하되,

상기 적어도 하나의 모듈은 상기 적어도 하나의 프로세서로 하여금

소스 코드의 플랫폼 비의존 부분(platform-agnostic portion)을 수신하고,

소스 코드의 복수의 플랫폼 특정 부분을 수신하고- 상기 소스 코드의 복수의 플랫폼 특정 부분은 리소스로서 지정됨 -,

상기 소스 코드의 플랫폼 비의존 부분과 상기 소스 코드의 복수의 플랫폼 특정 부분을 단일 실행 파일로 컴파일링하며,

범용으로 이식가능한 상기 단일 실행 파일을 복수의 상이한 플랫폼에 배치하게 하는

시스템.

청구항 2

제1항에 있어서,

상기 컴퓨팅 장치는 스마트폰, 태블릿, 게임 박스, 데스크탑 컴퓨터 또는 노트북인

시스템.

청구항 3

제1항에 있어서,

상기 메모리에 로딩되는 경우 상기 적어도 하나의 프로세서로 하여금 인터페이스를 포함하는 플랫폼 특정 플랫폼 추상화 계층을 생성하게 하는 적어도 하나의 모듈을 더 포함하는

시스템.

청구항 4

제1항에 있어서,

상기 소스 코드의 복수의 플랫폼 특정 부분 중 하나의 플랫폼 특정 부분은 소프트웨어의 버전을 포함하는

시스템.

청구항 5

제1항에 있어서,

상기 소스 코드 내의 어노테이션은 상기 소스 코드의 플랫폼 비의존 부분과 상기 소스 코드의 복수의 플랫폼 특정 부분을 상기 단일 실행 파일로 컴파일링하는 컴파일러를 결정하는

시스템.

청구항 6

컴퓨팅 장치의 프로세서에 의해, 플랫폼 비의존 부분과 복수의 플랫폼 특정 부분을 포함하는 단일 실행 파일을 수신하는 단계- 상기 복수의 플랫폼 특정 부분은 제1 플랫폼에 대한 플랫폼 특정 부분과 제2 플랫폼에 대한 플랫폼 특정 부분을 포함함 -와,

상기 컴퓨팅 장치의 프로세서의 플랫폼을 결정하는 단계- 상기 프로세서의 플랫폼은 상기 제1 플랫폼을 포함함 -와,

상기 단일 실행 파일의 상기 복수의 플랫폼 특정 부분으로부터 상기 제1 플랫폼에 대한 상기 플랫폼 특정 부분을 추출하는 단계와,

상기 제1 플랫폼에 대한 상기 플랫폼 특정 부분을 바이너리의 상기 플랫폼 비의존 부분에 결합하는 단계와,

상기 결합된 바이너리를 로딩하는 단계

를 포함하는 방법.

청구항 7

제6항에 있어서,

상기 제1 플랫폼은 스마트폰, 태블릿, 게임 박스, 데스크탑 컴퓨터 또는 노트북을 포함하는

방법.

청구항 8

제6항에 있어서,

상기 단일 실행 파일은 복수의 플랫폼 특정 플랫폼 추상화 계층 부분을 더 포함하고, 상기 복수의 플랫폼 특정 플랫폼 추상화 계층 부분은 상기 제1 플랫폼에 대한 플랫폼 특정 플랫폼 추상화 계층과 상기 제2 플랫폼에 대한 플랫폼 특정 플랫폼 추상화 계층을 포함하는

방법.

청구항 9

제8항에 있어서,

상기 단일 실행 파일의 상기 복수의 플랫폼 특정 플랫폼 추상화 계층 부분으로부터 상기 제1 플랫폼에 대한 상기 플랫폼 특정 플랫폼 추상화 계층을 추출하는 단계와,

상기 제1 플랫폼에 대한 상기 플랫폼 특정 플랫폼 추상화 계층 부분을 바이너리의 상기 플랫폼 비의존 부분에 결합하는 단계와,

상기 제1 플랫폼에 대한 상기 플랫폼 특정 플랫폼 추상화 계층 부분이 바이너리의 상기 플랫폼 비의존 부분에 결합된 바이너리를 로딩하는 단계를

더 포함하는 방법.

청구항 10

컴퓨터 실행가능 명령어들을 포함하는 컴퓨터 판독가능 저장 매체로서,

상기 컴퓨터 실행가능 명령어들은 실행되는 경우 컴퓨팅 장치의 적어도 하나의 프로세서로 하여금,

컴퓨팅 장치의 프로세서를 통해, 코드 부분과 비-코드 부분을 포함하는 단일 실행 파일을 수신하고- 상기 단일 실행 파일은 상기 코드 부분 내의 플랫폼 비의존 부분과 상기 비-코드 부분 내의 복수의 플랫폼 특정 부분을 포함하고, 상기 복수의 플랫폼 특정 부분은 제1 플랫폼에 대한 플랫폼 특정 부분과 제2 플랫폼에 대한 플랫폼 특정 부분을 포함함 -,

상기 컴퓨팅 장치의 플랫폼을 결정하고- 상기 컴퓨팅 장치의 플랫폼은 상기 제1 플랫폼을 포함함 -,

상기 단일 실행 파일의 상기 복수의 플랫폼 특정 부분으로부터 상기 제1 플랫폼에 대한 상기 플랫폼 특정 부분을 추출하고,

상기 제1 플랫폼에 대한 상기 플랫폼 특정 부분을 바이너리의 상기 플랫폼 비의존 부분에 결합하며,

상기 결합된 바이너리를 로딩하게 하는

컴퓨터 판독가능 저장 매체.

발명의 설명

기 술 분 야

배 경 기 술

[0001] 컴퓨터 공학에서 사용되는 "이식성(portability)"이라는 용어는 서로 다른 컴퓨팅 환경에서 동일한 소프트웨어를 사용할 수 있는 능력을 말한다. "이식가능 코드"라는 용어는 플랫폼에 특정되지 않는 코드를 지칭하는데 사용될 수 있다. 즉, 동일한 소프트웨어는 임의의 플랫폼에서 또는 적어도 여러 플랫폼에서 실행될 수 있다. "이식가능 코드"라는 용어는 또한 처음부터 소프트웨어를 작성하기보다는 다른 플랫폼에서 실행되게 비용 효율적으로 변경될 수 있는 코드를 지칭하는데 사용될 수 있다. 같은 용어에 대한 이들 서로 다른 용도를 구분하기 위해, "보편적으로 이식가능한 코드(universally portable code)"라는 용어가 사용되어 임의의 플랫폼에서 실행될 수 있는 플랫폼 비의존 코드(platform-agnostic code)를 지칭하는 데 사용될 것이다. "이식가능 코드"라는 용어는 다른 플랫폼에서 실행될 수 있게 비용 효율적으로 변경될 수 있는 코드를 지칭하는데 사용될 것이다.

[0002] 다른 플랫폼에서 실행되도록 코드의 다른 버전들을 생성하는 한 가지 방법은 조건부 컴파일(conditional compilation)을 통한 것이다. 조건부 컴파일은 컴파일러가 코드를 여러 번 컴파일함으로써 다수의 다른 플랫폼과 연관된 다수의 다른 실행 파일을 생성할 수 있게 해준다. 컴파일러는 한 세트의 파라미터 또는 지시(directives)를 통해 첫 번째 플랫폼을 위한 실행 파일을 생성하도록 동작하고 또 다른 세트의 파라미터 또는 지시를 통해 두 번째 플랫폼을 위한 실행 파일을 생성하도록 동작하고 또한 이러한 방식으로 계속 동작한다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0003] 소스 코드의 플랫폼 비의존 부분은 다른 플랫폼들 및/또는 소프트웨어의 버전들에 대해 어노테이션이 달리는 하나 이상의 부분을 포함하도록 확장될 수 있다. 플랫폼 비의존 및 플랫폼 특정 및/또는 버전 특정 부분을 포함하는 소스 코드는 다수의 장치 플랫폼 및/또는 하나 이상의 플랫폼의 여러 버전에 분배될 수 있고 또한 그 상에서 실행될 수 있는 단일 실행 파일을 생성하기 위해 한번 컴파일링될 수 있다. 개발 환경의 플랫폼 특정 속성은 컴파일러가 플랫폼 특정 자동 완성 및 타입 체킹을 제공할 수 있게 해준다. 플랫폼 특정 또는 버전 특정 실

행 파일(예를 들어, 어셈블리, 바이너리 등)은 리소스(예를 들어, 데이터)로서 보편적으로 이식가능한 (플랫폼 비의존) 실행 파일에 내장될 수 있다. 플랫폼 특정 플랫폼 추상화 계층 및/또는 버전 특정 인터페이스가 또한 생성될 수 있고 리소스 또는 데이터로서 보편적으로 이식가능한 실행 파일에 내장될 수 있다. 플랫폼 특정 플랫폼 추상화 계층은 대응하는 플랫폼 특정 실행 파일에 의해 실행 시간에 구현될 수 있다. 개발 도구, 예를 들어 IDE(integrated development environment)는 실행 파일이 특정 장치 상에서 실행되는 경우, 그 실행 파일이 정확히 실행되도록 그 실행 파일을 설정할 수 있다. 컴파일링된 코드가 장치 상에서 실행되는 경우, 프로그램 실행 관리자(예를 들어, 런타임)는 실행 파일이 실행되는 장치에 대응하는 플랫폼 특정 데이터를 추출할 수 있고, 이 추출된 데이터를 실행 파일에 결합시킬 수 있으며, 이 실행 파일을 로딩하여 실행할 수 있다. 따라서, 동일한 바이너리는 바이너리가 실행되는 플랫폼의 성능에 따라 다른 결과 또는 기능을 생성할 수 있다.

[0004] 본 요약은 이하 발명의 상세한 설명에서 더 기술되는 개념들의 모음을 단순화된 형식으로 소개하기 위해 제공되는 것이다. 본 요약은 청구대상의 주된 사항 또는 핵심 사항을 밝히기 위한 것이 아니며, 청구대상의 범위를 제한하려는 용도도 아니다.

도면의 간단한 설명

[0005] 도 1a는 이식가능한 코드를 조건부 컴파일링하는 당업계에 알려져 있는 시스템(100)의 예를 나타낸다.

도 1b는 본 명세서에 개시된 청구 대상의 측면에 따라 다수의 플랫폼에서 실행될 수 있는 단일 실행 파일을 생성하는 시스템(109)의 예를 나타낸다.

도 1c는 본 명세서에 개시된 청구 대상의 측면에 따라 다수의 플랫폼에서 실행될 수 있는 단일 실행 파일을 생성하는 시스템(121)의 예를 나타낸다.

도 2는 본 명세서에 개시된 청구 대상의 측면에 따라 단일 실행 파일을 생성하고 그 생성된 실행 파일을 실행하는 방법(200)의 예를 나타낸다.

도 3은 본 명세서에 개시된 청구 대상의 측면에 따른 컴퓨팅 환경의 일 예의 블록도를 나타낸다.

도 4는 본 명세서에 개시된 청구 대상의 측면에 따른 통합 개발 환경의 일 예의 블록도를 나타낸다.

발명을 실시하기 위한 구체적인 내용

[0006] **개요**

[0007] 도 1a는 당 업계에 공지된 바와 같이 보편적으로 이식가능한 코드를 생성하기 위한 시스템을 도시한다. 전통적으로, (예를 들면, 소스 코드(101)와 같은) 조건부 컴파일 지시를 갖는 소스 코드는 타겟 장치에서 실행되도록 컴파일된다. 예를 들어, 소스 코드는 스마트폰과 같은 타겟 장치에서 실행되고 스마트폰에 대한 바이너리, 예를 들어 스마트폰(104a)에 대한 바이너리와 같은 스마트폰에 대한 바이너리를 생성하도록 컴파일러(102) 상에서 한번 컴파일링된다. 상이한 지시들을 갖는 소스 코드는 태블릿에서 실행될 수 있는 바이너리(예를 들어, 태블릿(104b)에 대한 바이너리)를 생성하도록 한번 더 컴파일링될 수 있다. 여전히 다른 지시를 갖는 소스 코드는 게임박스에서 실행될 수 있는 바이너리(예를 들어, 게임박스(104c)에 대한 바이너리) 또는 데스크탑에서 실행될 수 있는 바이너리(예를 들어, 데스크탑(104d)에 대한 바이너리) 또는 다른 장치에서 실행될 수 있는 바이너리(예를 들어, 장치 n(104n)에 대한 바이너리)를 생성하도록 또 다시 컴파일링될 수 있다. 따라서, 이식가능한 코드는 여러 번 컴파일링되어 다수의 바이너리를 생성할 수 있는데, 각 바이너리는 특정 플랫폼에서 실행되도록 설계된다.

[0008] 이러한 설계로 인한 하나의 결과는 소프트웨어가 장치에 배치되도록 패키지화되는 경우, 그 장치에 대한 적절한 바이너리가 선택되어야 한다는 것이다. 예를 들어, 스마트폰에 대한 바이너리(104a)는 스마트폰에 배치되어야 하고 스마트폰(106a)의 런타임은 바이너리(104a)를 실행하여 소정의 실행 결과 a(108a)를 생성할 것이다. 유사하게, 태블릿에 대한 바이너리(104b)는 태블릿에 배치되어야 하고 태블릿(106b)의 런타임은 바이너리(104b)를 실행하여 실행 결과 a(108a)와 다를 수 있는 소정의 실행 결과 b(108b)를 생성할 것이다. 유사하게, 게임 박스에 대한 바이너리(104c)는 게임 박스에 배치되어야 하고 게임 박스(106c)의 런타임은 바이너리(104c)를 실행하여 실행 결과 a(108a)와 다를 수 있고 또한 실행 결과 b(108b)와 다를 수 있는 소정의 실행 결과 c(108c)를 생성할 것이다. 데스크탑에 대한 바이너리(104d)는 데스크탑에 배치되어야 하고 데스크탑(106d)의 런타임은 바이너리(104d)를 실행하여 다른 장치들의 실행 결과들과 다를 수 있는 소정의 실행 결과 d(108d)를 생성할 것이다.

즉, 장치 특정 바이너리(104n)는 대응하는 장치에 배치되어야 하고 그 장치(106n)의 런타임은 바이너리(104n)를 실행하여 특정 결과(108n)를 생성할 것이다. 코드가 재사용되는 경우, 코드는 장치 특정 부분을 변경하도록 수정되어야 한다.

[0009] 도 1b는 본 명세서에 개시된 청구 대상의 측면에 따른 (참조 라이브러리로도 지칭될 수 있는) 단일 바이너리를 생성하는 시스템(109)을 나타낸다. 본 명세서에 기술된 청구 대상의 측면에 따르면, 소스 코드(110)와 같은 소스 코드는 다수의 타겟 플랫폼에서 실행될 수 있는 단일 바이너리(예를 들어, 바이너리(115))를 생성하도록 수정된 컴파일러(112)에 의해 한번 컴파일링될 수 있다. 당업계에 알려진 바이너리는 리소스 또는 데이터를 나타내는 비-코드 부분 및 코드 부분을 포함할 수 있다. 바이너리(115)는 버전 및/또는 플랫폼 비의존 코드를 포함하는 버전 및/또는 플랫폼 비의존 부분(114)과 같은 하나 이상의 부분과, 버전 특정적인 및/또는 플랫폼 특정적인 하나 이상의 부분(예를 들어, 스마트폰 특정 부분(116a), 태블릿 특정 부분(116b), 게임 박스 특정 부분(116c), 데스크탑 특정 부분(116d) 또는 임의의 장치 특정 부분 n(116n))을 포함할 수 있다. 플랫폼 특정 부분은 리소스 또는 데이터로 라벨링될 수 있다. 단일 바이너리(115)는 다수의 다른 플랫폼에 배치될 수 있다.

[0010] 런타임시, 바이너리(115)는 예를 들어 런타임(예컨대, 스마트폰에 대해서는 런타임(118a), 태블릿에 대해서는 런타임(118b), 게임 박스에 대해서는 런타임(118c), 데스크탑에 대해서는 런타임(118d) 및 일반적으로 장치 n에 대해서는 런타임(118n))과 같은 프로그램 실행 관리자에 의해 실행될 수 있다. 런타임은 (예를 들어, 리소스 또는 데이터로서 라벨링된 바이너리의 부분으로부터) 바이너리의 적절한 (일치하는) 버전 특정 및/또는 플랫폼 특정 부분을 추출할 수 있고 바이너리의 추출된 버전 특정 및/또는 플랫폼 특정 부분을 바이너리의 버전 및/또는 플랫폼 비의존 부분에 결합시킬 수 있다. 즉, 예를 들어, 스마트폰의 런타임(118a)은 바이너리(116a)의 스마트폰 특정 부분을 플랫폼 비의존 부분(114)에 결합시킬 수 있고 그것을 실행시켜 스마트폰에 대한 실행 결과 a(120a)를 생성할 수 있다. 유사하게, 태블릿의 런타임(118b)은 바이너리(116b)의 태블릿 특정 부분을 플랫폼 비의존 부분(114)에 결합시킬 수 있고 그것을 실행시켜 태블릿에 대한 실행 결과 b(120b)를 생성할 수 있다. 게임 박스의 런타임(118c)은 바이너리(116c)의 게임 박스 특정 부분을 플랫폼 비의존 부분(114)에 결합시킬 수 있고 그것을 실행시켜 게임 박스에 대한 실행 결과 c(120c)를 생성할 수 있다. 데스크탑의 런타임(118d)은 바이너리(116d)의 데스크탑 특정 부분을 플랫폼 비의존 부분(114)에 결합시킬 수 있고 그것을 실행시켜 데스크탑에 대한 실행 결과 d(120d)를 생성할 수 있다. 유사하게, 특정 장치의 런타임(118n)은 바이너리(116n)의 장치 특정 부분을 플랫폼 비의존 부분(114)에 결합시킬 수 있고 그것을 실행시켜 장치에 대한 실행 결과 n(120n)을 생성할 수 있다.

[0011] 마찬가지로, 버전 특정 및/또는 플랫폼 특정 플랫폼 추상화 계층(도 1b에는 도시되어 있지 않음)은 바이너리의 데이터 부분으로부터 추출될 수 있고 버전 특정 및/또는 플랫폼 비의존 부분(114)에 결합될 수 있다. 본 명세서에 기술된 청구 대상의 몇몇 측면에 따르면, 단일 바이너리(115)는 코드의 하나 이상의 버전 특정 및/또는 플랫폼 특정 부분이 재사용의 정상 단위로서 런타임에 의해 이해되는 방식으로 생성된다(예를 들어, 바이너리는 런타임이 인식하는 포맷으로 생성된다).

[0012] **적응식 이식가능 라이브러리**

[0013] 도 1c는 본 명세서에 개시된 청구 대상의 측면에 따라 단일 이식가능 실행 파일을 생성하는 시스템(121)의 블록도를 나타낸다. 도 1c는 또한 본 명세서에 개시된 청구 대상의 측면에 따른 이식가능 실행 파일의 실행을 나타낸다. 시스템(121)의 전부 또는 일부분들은 도 3과 관련하여 이하에서 설명되는 컴퓨터와 같은 하나 이상의 컴퓨터 또는 컴퓨팅 장치에 상주할 수 있다. 시스템(121) 또는 이들의 일부분들은 독립형 시스템 또는 플러그 인 또는 애드 인으로서 제공될 수 있다. 시스템(121)은 도 4와 관련하여 설명되는 소프트웨어 개발 컴퓨터와 같은 소프트웨어 개발 컴퓨터 상에서 전체적으로 또는 부분적으로 실행될 수 있다. 시스템(121)의 전부 또는 일부분들은 개발 도구에 의해 동작될 수 있다. 예를 들어, 시스템(121)의 전부 또는 일부분들은 통합 개발 환경(IDE), 예를 들어 도 4와 관련하여 보다 자세히 설명되는 IDE(125) 내에서 실행될 수 있거나 또는 IDE 밖에서 실행될 수 있다.

[0014] 시스템(121)은 컴퓨팅 장치(122)와 같은 하나 이상의 컴퓨팅 장치 또는 컴퓨터를 포함할 수 있다. 컴퓨팅 장치(122)는 프로세서(142) 등과 같은 하나 이상의 프로세서, 메모리(144)와 같은 메모리, 및 단 하나의 보편적으로 이식가능한 실행 파일을 생성하는 모듈(123)과 같은 하나 이상의 모듈을 포함할 수 있다. 모듈(123)로 표현되는 하나 이상의 모듈은 컴파일 체인(124)의 일부를 포함할 수 있다. 모듈(123)은 컴파일러 또는 컴파일러 프리-프로세서 또는 컴파일러 포스트-프로세서 또는 이들의 임의의 조합의 일부일 수 있다. 모듈(123) 등과 같은 하나 이상의 모듈은 메모리(144)에 로딩되어 프로세서(142) 등과 같은 하나 이상의 프로세서로 하여금 모듈

(123)에 부여된 동작들을 수행하게 할 수 있다.

[0015] 시스템(121)은 스마트폰, 태블릿, 데스크탑 컴퓨터, 게임 박스 또는 임의의 종류의 컴퓨팅 장치를 포함하나 이에 국한되지 않는 하나 이상의 다른 컴퓨터 또는 컴퓨팅 장치를 포함할 수 있다. 이들 컴퓨팅 장치 각각은 하나 이상의 프로세서(미도시), 메모리(미도시) 및 당업계에 잘 알려져 있는 다른 컴포넌트들을 포함할 수 있다.

[0016] 소스 코드 파일(134)과 같은 소스 코드 파일은 플랫폼 비의존 코드(130)와 같은 플랫폼 비의존 코드의 하나 이상의 부분을 포함할 수 있다. 소스 코드 파일(134)은 또한 플랫폼 특정 코드(132) 등과 같은 플랫폼 특정 코드의 하나 이상의 부분을 포함할 수 있다. 플랫폼 특정 코드의 하나 이상의 부분은 장치 1(135), 장치 2(137)...장치 n(139) 등과 같은 장치 상에 실행되도록 특정된 코드를 포함할 수 있다. 장치 1(135), 장치 2(137)...장치 n(139)와 같은 장치는 스마트폰, 태블릿, 게임 박스, 데스크탑 컴퓨터, 랩탑 컴퓨터, 노트북 컴퓨터 또는 현재 알려져 있거나 미래에 생성될 임의의 다른 컴퓨팅 장치와 같은 장치를 포함할 수 있다. 버전 특정 및/또는 플랫폼 특정 코드의 하나 이상의 부분은 해당 장치의 버전 및/또는 타입을 나타내는 식별 어노테이션이 달릴 수 있다. 예를 들어, 제1 어노테이션 또는 어노테이션의 타입은 프로그램 소스 코드의 어노테이션이 부여된 섹션은 스마트폰에 대해 특정적이거나 그 스마트폰의 소프트웨어의 버전에 특정적임을 나타낼 수 있다. 제2 어노테이션 또는 어노테이션의 타입은 프로그램 소스 코드의 어노테이션이 부여된 섹션은 태블릿에 대해 특정적이거나 그 태블릿의 소프트웨어의 버전에 특정적임을 나타낼 수 있다. 사용된 특정 어노테이션은 어떤 컴파일러 또는 컴파일러의 버전이 바이너리를 생성하는데 사용되는지를 제어할 수 있다.

[0017] 모듈(123)은 소스 코드 파일(134)을 수신할 수 있고 컴파일 체인(124)의 다른 컴포넌트들과 연계하여 플랫폼 바이너리(126)와 같은 버전 비의존 및/또는 플랫폼 비의존 코드의 하나 이상의 부분과, 플랫폼 특정 바이너리(128)와 같은 버전 특정 및/또는 플랫폼 특정 코드의 하나 이상의 부분 등을 포함하는 단일 바이너리(129)를 생성할 수 있다. 버전 특정 및/또는 플랫폼 특정 바이너리(128)는 "라이트-업 코드(light-up code)"를 포함할 수 있다. 라이트-업 코드는 하나의 플랫폼 또는 버전에는 존재하나 또 다른 플랫폼 또는 버전에는 존재하지 않고 따라서 하나의 플랫폼 또는 버전의 사용자 인터페이스에서 "라이트 업"하지만 또 다른 플랫폼 또는 버전의 사용자 인터페이스에서는 "라이트 업"하지 않는 하나 이상의 특징을 포함할 수 있다. 예를 들어 플랫폼 특정 바이너리(128) 등과 같은 버전 특정 및/또는 플랫폼 특정 바이너리는 플랫폼 비의존 바이너리(126)와 연관된 데이터와 같은 리소스로서 지정될 수 있다. 모듈(123) 등은 또한 해당 플랫폼과 연관된 코드에 기초하여, 봉착한 플랫폼 특정 코드 부분의 각 타입에 대해 추상화 계층(예를 들어, 플랫폼 특정 인터페이스)을 생성할 수 있다. 예를 들어, 플랫폼 특정 바이너리(128)와 같은 제1 플랫폼 특정 바이너리는 추상화 계층(128a)과 같은 제1 플랫폼 특정 플랫폼 추상화 계층과 연관될 수 있다. 이와 달리, 소스 코드는 단일 멀티-플랫폼 바이너리를 생성하도록 포스트-컴파일 동작에서 결합될 수 있는 다수의 플랫폼 특정 바이너리를 생성하기 위해 표준 조건부 컴파일 기법을 사용하여 여러 번 컴파일링될 수 있다.

[0018] 동일한 바이너리, 예를 들어, 바이너리(129)는 다수의 다른 플랫폼에 배치될 수 있다. 소프트웨어가 배치된 장치상에서 이 소프트웨어가 실행되는 경우, 장치의 런타임(예를 들어, 런타임(136), 런타임(138) 또는 런타임(140) 등)은 적절한 플랫폼 특정 바이너리를 사용하여 그것을 플랫폼 비의존 바이너리에 결합하여 장치 상에서 실행되는 실행 파일을 생성할 수 있다. 예를 들어, 플랫폼 비의존 바이너리, 즉 바이너리 A가 플랫폼 특정 바이너리, 즉 바이너리 B를 사용하는 것으로 가정한다. 실행 시간에서, 장치에 적합한 플랫폼 특정 바이너리 B는 바이너리 A와 결합되도록 런타임에 의해 선택될 수 있고 이후 결과적인 실행 파일이 실행될 수 있다. 상이한 플랫폼 외에, 상이한 버전이 유사하게 수용될 수 있다. 플랫폼에 대한 적절한 플랫폼 추상화 계층은 결합 시간에서 선택된 플랫폼 특정 코드에 기초하여 런타임에 의해 구현될 수 있다. 가상 머신 환경에서 관리되는 언어와 관련하여 설명되었지만, 본 명세서에서 기술된 개념들은 네이티브 코드 환경에서도 적용될 수 있음을 알 수 있을 것이다. 네이티브 코드 환경에서, 플랫폼 비의존 코드는 생성하기가 더 어려울 수 있다.

[0019] 도 2는 다수의 상이한 플랫폼에서 실행될 수 있는 단일 바이너리를 생성할 수 있는 방법(200)을 나타낸다. 바이너리는 하나 이상의 다른 플랫폼에 배치될 수 있다. 바이너리가 실행되는 경우, 바이너리가 배치된 장치의 런타임은 적절한 플랫폼 특정 바이너리를 결정하고, 이 적절한 플랫폼 특정 바이너리를 추출하여 그것을 결합하여 장치에 대한 플랫폼 특정 바이너리를 생성할 수 있다. 따라서, 동일한 바이너리는 상이한 장치에서 실행되는 경우 상이한 결과를 낳을 수 있다. 도 2에 기술된 방법은 예를 들어 도 1b 및 도 1c와 관련하여 기술한 것과 같은 시스템에 의해 실시될 수 있다. 방법(200)은 순차적으로 실행되는 일련의 동작들을 기술하고 있지만, 방법(200)은 그 순서에 국한되지 않는다. 예를 들어, 소정의 동작들은 기술되어 있는 것과는 다른 순서로 실행될 수 있다. 또한, 소정의 동작은 또 다른 동작과 동시에 실행될 수 있다. 일부 예에서는, 모든 동작들이 수행되는 것은 아닐 수 있다. 또한, 상이한 플랫폼들에 대해 기술하고 있지만, 동작들은 또한 소프트웨어의 상이

한 버전들의 생성, 발생, 배치 및 실행에도 유사하게 적용될 수 있음을 알 수 있을 것이다.

[0020] 동작(201)에서, IDE의 소스 코드 편집기 또는 다른 소프트웨어 도구에 의해 플랫폼 비의존 소스 코드가 수신될 수 있다. 동작(202)에서, 어노테이션이 달린 소스 코드가 수신될 수 있다. 플랫폼 특정적인 소스 코드는 소스 코드가 적용되는 플랫폼을 식별하기 위해 어노테이션이 달릴 수 있다. 동작(204)에서, 하나 이상의 플랫폼 비의존 부분 및 하나 이상의 플랫폼 특정 부분 모두를 포함하는 소스 코드는 단일 바이너리 내로 컴파일링될 수 있다. 동작(206)에서, 바이너리의 플랫폼 특정 부분은 데이터 또는 비-코드 리소스로서 라벨링된 또는 표시된 플랫폼 비의존 바이너리에 첨부될 수 있다. 동작(208)에서, 바이너리는 하나 이상의 다른 플랫폼에 배치될 수 있다. 동작(210)에서, 바이너리가 배치된 장치들 중 하나의 런타임은 바이너리를 실행하라는 요청을 수신할 수 있다. 런타임은 플랫폼 비의존 바이너리에 결합할 적절한 플랫폼 특정 부분을 결정 및 추출할 수 있다. 동작(212)에서, 추출된 플랫폼 특정 부분은 플랫폼 비의존 바이너리에 결합될 수 있다. 동작(214)에서, 바이너리는 로딩될 수 있고 동작(216)에서 바이너리는 실행될 수 있다.

[0021] **적절한 컴퓨팅 환경의 예시**

[0022] 본 명세서에서 기술한 청구 대상의 다양한 측면들에 대한 문맥을 제공하기 위해, 도 3 및 후속하는 설명은 본 명세서에서 기술된 청구 대상의 다양한 실시예들이 구현될 수 있는 적절한 컴퓨팅 환경(510)의 간략하고 개괄적인 설명을 제공하려 한다. 본 명세서에서 기술한 청구 대상은 하나 이상의 컴퓨터 또는 다른 컴퓨팅 장치에 의해 실행되는 프로그램 모듈과 같은 컴퓨터 실행가능 명령어들의 일반적인 문맥에서 설명되고 있지만, 당업자라면, 본 명세서에서 기술된 청구 대상의 부분들은 또한 다른 프로그램 모듈 및/또는 하드웨어와 소프트웨어의 조합과 연계하여 실시될 수 있음을 알 수 있을 것이다. 일반적으로, 프로그램 모듈은 특정 작업을 수행하거나 특정 데이터 유형을 구현하는 루틴, 프로그램, 객체, 물리적 아티팩트, 데이터 구조 등을 포함한다. 전형적으로, 프로그램 모듈의 기능은 다양한 실시예에서 요구에 따라 결합 또는 분배될 수 있다. 컴퓨팅 환경(510)은 적절한 동작 환경의 하나의 예일 뿐이고 본 명세서에 설명된 청구 대상의 사용 또는 기능의 범주를 제한하려는 것은 아니다.

[0023] 도 3을 참조하면, 컴퓨터(512) 형태의 컴퓨팅 장치가 설명되어 있다. 컴퓨터(512)는 적어도 하나의 처리 장치(514), 시스템 메모리(516) 및 시스템 버스(518)를 포함할 수 있다. 적어도 하나의 처리 장치(514)는 예를 들어 시스템 메모리(516)와 같은 메모리 저장된 명령어들을 실행할 수 있다. 처리 장치(514)는 다양한 가용 프로세서 중 임의의 프로세서일 수 있다. 예를 들어, 처리 장치(514)는 그래픽 처리 장치(GPU)일 수 있다. 명령어들은 전술한 하나 이상의 컴포넌트 또는 모듈에 의해 수행되는 기능을 구현하기 위한 명령어들 또는 전술한 방법들 중 하나 이상을 구현하기 위한 명령어들일 수 있다. 듀얼 마이크로프로세서 및 다른 멀티프로세서 아키텍처도 처리 장치(514)로서 채용될 수 있다. 컴퓨터(512)는 디스플레이 스크린 상에서 그래픽의 렌더링을 지원하는 시스템에서 사용될 수 있다. 또 다른 예에서, 컴퓨팅 장치의 적어도 일부분은 그래픽 처리 장치를 포함하는 시스템에서 사용될 수 있다. 시스템 메모리(516)는 휘발성 메모리(520) 및 비휘발성 메모리(522)를 포함할 수 있다. 비휘발성 메모리(522)는 판독 전용 메모리(ROM), 프로그램가능 ROM(PROM), 전기적 프로그램가능 ROM(EPROM) 또는 플래시 메모리를 포함할 수 있다. 휘발성 메모리(520)는 외부 캐시 메모리로서 동작할 수 있는 랜덤 액세스 메모리(RAM)를 포함할 수 있다. 시스템 버스(518)는 시스템 메모리(516)를 포함하는 시스템의 물리적 아티팩트를 처리 장치(514)에 결합한다. 시스템 버스(518)는 메모리 버스, 메모리 제어기, 주변 버스, 외부 버스, 또는 로컬 버스를 포함하는 몇몇 유형들 중 임의의 유형일 수 있고, 이용가능한 버스 아키텍처들 중 임의의 다양한 아키텍처를 이용할 수 있다. 컴퓨터(512)는 시스템 버스(518)를 통해 처리 장치(514)에 의해 액세스가능한 데이터 저장소를 포함할 수 있다. 이 데이터 저장소는 그래픽 렌더링을 위한 실행가능한 명령어들, 3D 모델, 자료, 텍스처 등을 포함할 수 있다.

[0024] 컴퓨터(512)는 전형적으로 휘발성 및 비휘발성 매체, 이동식 및 고정식 매체와 같은 다양한 컴퓨터 판독가능 매체를 포함한다. 컴퓨터 판독가능 매체는 컴퓨터 판독가능 명령어, 데이터 구조, 프로그램 모듈 또는 그 밖의 데이터와 같은 정보를 저장하기 위한 임의의 방법 또는 기술로 구현될 수 있다. 컴퓨터 판독가능 매체는 컴퓨터 판독가능 저장 매체(컴퓨터 저장 매체로도 지칭됨) 및 통신 매체를 포함한다. 컴퓨터 저장 매체는 예를 들어 RAM, ROM, EEPROM, 플래시 메모리 또는 다른 메모리 기술, CDROM, DVD 또는 다른 광학 디스크 저장소, 자기 카세트, 자기 테이프, 자기 디스크 저장소 또는 원하는 데이터를 저장할 수 있고 컴퓨터(512)에 의해 액세스될 수 있는 다른 자기 저장 장치와 같은 물리적 (유형의) 매체를 포함한다. 통신 매체는 예를 들어 통신 신호, 변조된 반송파 또는 원하는 정보를 전달하는데 사용될 수 있고 컴퓨터(512)에 의해 액세스될 수 있는 임의의 다른 일시적 매체와 같은 일시적 매체를 포함한다.

[0025] 도 3은 사용자와 컴퓨터 리소스 간의 중개자로서 동작할 수 있는 소프트웨어를 설명한다는 것을 알 것이다. 이 소프트웨어는 디스크 저장소(524)에 저장될 수 있고, 컴퓨터(512)의 리소스를 할당할 수 있는 운영 체제(528)를 포함할 수 있다. 디스크 저장소(524)는 인터페이스(526)와 같은 고정식 메모리 인터페이스를 통해 시스템 버스(518)에 연결된 하드 디스크 드라이브일 수 있다. 시스템 애플리케이션(530)은 시스템 메모리(516) 또는 디스크 저장소(534)에 저장된 프로그램 모듈(532) 및 프로그램 데이터(534)를 통해 운영 체제(528)에 의해 이루어지는 리소스의 관리를 이용할 수 있다. 컴퓨터는 다양한 운영 체제 또는 운영 체제들의 조합과 함께 구현될 수 있다는 것을 이해할 것이다.

[0026] 사용자는 입력 장치(들)(536)를 통해 컴퓨터(512)에 명령 또는 정보를 입력할 수 있다. 입력 장치(536)는 마우스, 트랙볼, 스타일러스, 터치 패드, 키보드, 마이크로폰, 음성 인식 및 제스처 인식 시스템 등과 같은 포인팅 장치를 포함하나 이에 국한되지 않는다. 이들 및 다른 입력 장치는 인터페이스 포트(들)(538)를 통해 시스템 버스(518)를 거쳐 처리 장치(514)에 연결된다. 인터페이스 포트(들)(538)는 직렬 포트, 병렬 포트, 유니버셜 직렬 버스(USB) 등을 나타낼 수 있다. 출력 장치(들)(540)는 입력 장치와 마찬가지로 동일한 유형의 포트를 사용할 수 있다. 특정 어댑터를 요구하는 모니터, 스피커 및 프린터와 같은 일부 출력 장치(540)가 있음을 나타내기 위해 출력 어댑터(542)가 제공된다. 출력 어댑터(542)는 출력 장치(540)와 시스템 버스(518) 간의 연결을 제공하는 비디오 및 사운드 카드를 포함하나 이에 국한되지 않는다. 원격 컴퓨터(들)(544)와 같은 다른 장치 및/또는 시스템 또는 장치는 입력 기능 및 출력 기능 모두를 제공할 수 있다.

[0027] 컴퓨터(512)는 원격 컴퓨터(들)(544)와 같은 하나 이상의 원격 컴퓨터에 대한 논리적 연결을 사용하여 네트워크형 환경에서 동작할 수 있다. 원격 컴퓨터(544)는 개인용 컴퓨터, 서버, 라우터, 네트워크 PC, 피어 장치 또는 다른 공통 네트워크 노드일 수 있고, 전형적으로 컴퓨터(512)와 관련하여 전술한 요소들 중 다수 또는 전부를 포함할 수 있지만, 도 3에서는 메모리 저장 장치(546)만이 도시되어 있다. 원격 컴퓨터(들)(544)는 통신 연결부(들)(550)를 통해 논리적으로 연결될 수 있다. 네트워크 인터페이스(548)는 근거리 통신망(LAN) 및 광역 네트워크(WAN)와 같은 통신 네트워크를 포함하나 다른 네트워크도 포함할 수 있다. 통신 연결부(들)(550)는 네트워크 인터페이스(548)를 버스(518)에 연결하는데 채용된 하드웨어/소프트웨어를 지칭한다. 통신 연결부(들)(550)는 컴퓨터(512)에 내부적 또는 외부적일 수 있고 모뎀(전화, 케이블, DSL 및 무선) 및 ISDN 어댑터, 이더넷 카드 등과 같은 내부 및 외부 기술을 포함할 수 있다.

[0028] 도시되어 있는 네트워크 연결부는 단지 예시이며 컴퓨터들 간의 통신 링크를 수립하는 다른 수단이 사용될 수 있다. 당업자라면, 컴퓨터(512) 또는 다른 클라이언트 장치는 컴퓨터 네트워크의 일부로서 이용될 수 있음을 알 수 있다. 이와 관련하여, 본 명세서에서 기술된 청구 대상은 임의의 개수의 메모리 또는 저장 장치 및 임의의 수의 저장 장치 또는 볼륨에 걸쳐 발생하는 임의의 수의 애플리케이션 및 프로세스를 갖는 임의의 컴퓨터 시스템에 관한 것일 수 있다. 본 명세서에 개시된 청구 대상의 측면들은 네트워크 환경에 배치되고 원격 또는 로컬 저장소를 구비한 서버 컴퓨터 및 클라이언트 컴퓨터를 갖춘 환경에 적용될 수 있다. 본 명세서에 개시된 청구 대상의 측면들은 또한 프로그래밍 언어 기능, 해석 및 실행 기능을 갖는 독립형 컴퓨팅 장치에도 적용될 수 있다.

[0029] 도 4는 통합형 개발 환경(IDE)(600) 및 공통 언어 런타임 환경(602)을 나타낸다. IDE(600)는 사용자가 컴퓨터 시스템에서 프로그램, 프로그램들의 집합, 웹 사이트, 웹 애플리케이션 및 웹 서비스를 설계, 코딩, 컴파일, 테스트, 실행, 편집, 디버깅 또는 빌딩할 수 있도록 해준다. 소프트웨어 프로그램은 하나 이상의 소스 코드 언어(예를 들어, Visual Basic, Visual J#, C++, C#, J#, Java Script, APL, COBOL, Pascal, Eiffel, Haskell, ML, Oberon, Perl, Python, Scheme, Smalltalk 등)로 작성된 소스 코드(컴포넌트(610))를 포함할 수 있다. IDE(600)는 네이티브 코드 개발 환경을 제공할 수 있거나 가상 머신에서 실행되는 관리되는 코드 개발을 제공할 수 있거나 이들의 조합을 제공할 수 있다. IDE(600)는 Microsoft.NET™ 프레임워크를 사용하여 관리되는 코드 개발 환경을 제공할 수 있다. 애플리케이션이 실행되는 경우, 중간 언어 컴포넌트(650)가 모델링 도구(652) 및 모델 저장소(653)를 사용하는 언어 특정 소스 컴파일러(620)를 사용하여 소스 코드 컴포넌트(610) 및 네이티브 코드 컴포넌트(611)로부터 생성될 수 있고, 네이티브 코드 컴포넌트(611)(예를 들어 머신 실행가능 명령어)는 중간 언어 컴파일러(660)(예를 들어 JIT(just-in-time) 컴파일러)를 사용하여 중간 언어 컴포넌트(650)로부터 생성된다. 즉, 중간 언어(IL) 애플리케이션이 실행되는 경우, 이 중간 언어(IL) 애플리케이션은 실행되는 동안 이것이 실행되는 플랫폼에 대한 적절한 머신 언어로 컴파일링되어, 코드가 여러 플랫폼에 걸쳐 이식가능하게 한다. 이와 달리, 다른 실시예에서, 프로그램은 해당 플랫폼에 적절한 네이티브 코드 머신 언어(미도시)로 컴파일링될 수 있다.

[0030] 사용자는 알려져 있는 소프트웨어 프로그래밍 기법 및 특정 소스 언어와 연관된 특정 논리적 및 구문 규칙에 따

라 IDE(600) 내의 사용자 인터페이스(640) 및 소스 코드 편집기(651)를 통해 소스 코드 컴포넌트를 생성 및/또는 편집할 수 있다. 이후, 소스 코드 컴포넌트(610)는 소스 컴파일러(620)를 통해 컴파일링될 수 있고, 그에 따라 프로그램의 중간 언어 표현, 예를 들어 어셈블리(630)가 생성될 수 있다. 어셈블리(630)는 중어 언어 컴포넌트(650) 및 메타데이터(642)를 포함할 수 있다. 애플리케이션 디자인은 배치 전에 검증될 수 있다.

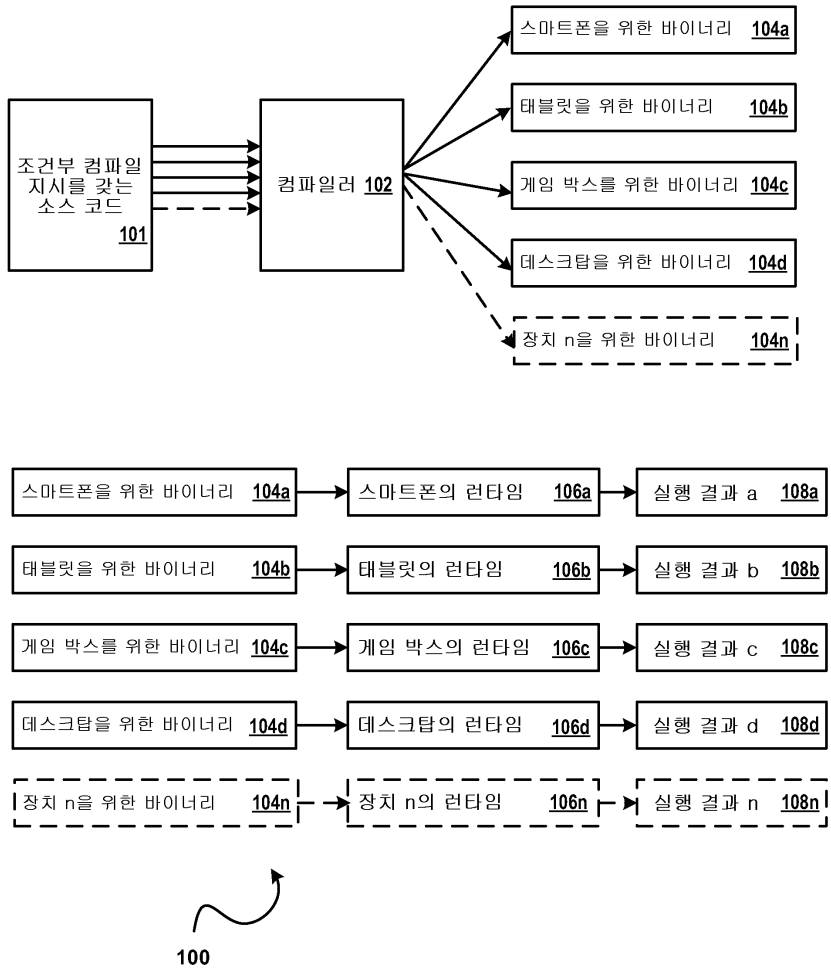
[0031] 본 명세서에서 기술된 다양한 기법들은 하드웨어 또는 소프트웨어와 연계하여, 또는 적절하다면 이들의 조합과 연계하여 구현될 수 있다. 따라서, 본 명세서에서 기술된 방법 및 장치, 또는 이들의 소정의 측면 또는 부분들은 플로피 디스켓, CD-ROM, 하드 드라이브 또는 임의의 다른 머신 판독가능 저장 매체와 같은 유형의 매체에 기록되는 프로그램 코드(즉, 명령어)의 형식을 취할 수 있는데, 이 프로그램 코드가 컴퓨터와 같은 머신에 로딩되어 그 머신에 의해 실행되는 경우, 머신은 본 명세서에서 기술된 청구 대상의 측면들을 실시하는 장치가 된다. 본 명세서에서 사용되는 "머신 판독가능 저장 매체"라는 용어는 임의의 형식의 전파 신호를 제공하는 (즉, 저장 및/또는 전송하는) 임의의 메카니즘을 배제하는 것으로 취급되어야 한다. 프로그램가능 컴퓨터에서 프로그램 코드가 실행되는 경우, 컴퓨팅 장치는 일반적으로 프로세서, (휘발성 및 비휘발성 메모리 및/또는 저장 소자를 포함하는) 프로세서에 의해 판독가능한 저장 매체, 적어도 하나의 입력 장치 및 적어도 하나의 출력 장치를 포함할 것이다. 예를 들어 데이터 처리 API 등을 사용하는 도메인 특정 프로그래밍 모델 측면들의 생성 및/또는 구현을 이용하는 하나 이상의 프로그램은 상위 레벨 절차 또는 객체 지향 프로그래밍 언어로 구현되어 컴퓨터 시스템과 통신할 수 있다. 그러나, 프로그램(들)은 필요하다면 어셈블리 또는 머신 언어로 구현될 수 있다. 어떠한 경우든, 언어는 컴파일링된 또는 해석된 언어일 수 있고 하드웨어 구현과 결합될 수 있다.

[0032] 본 발명은 구조적 특징 및/또는 방법론적 동작에 특정한 언어로 기술되었지만, 첨부한 청구항에 정의된 본 발명은 전술한 바와 같은 특정 특징 또는 동작들에 반드시 국한될 필요는 없다. 그 보다, 전술한 특정 특징 및 동작들은 청구 대상 및 실시예를 구현하기 위한 예시적인 형식으로 개시되어 있다.

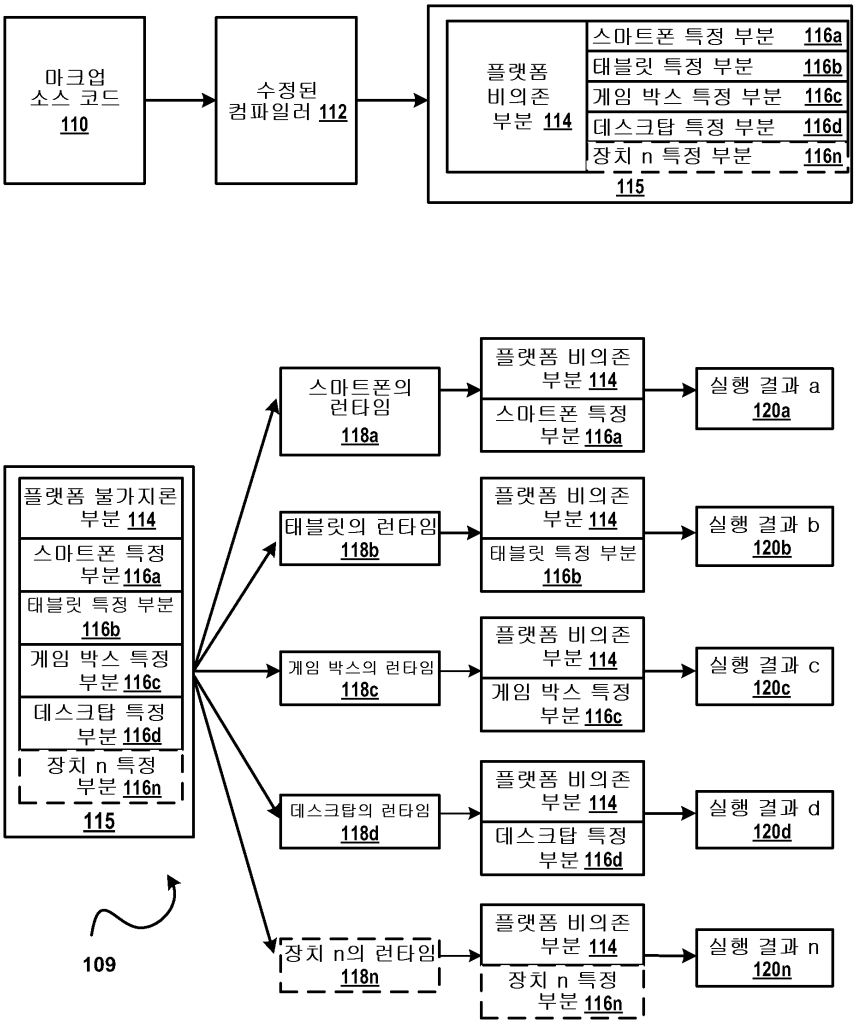
도면

도면1a

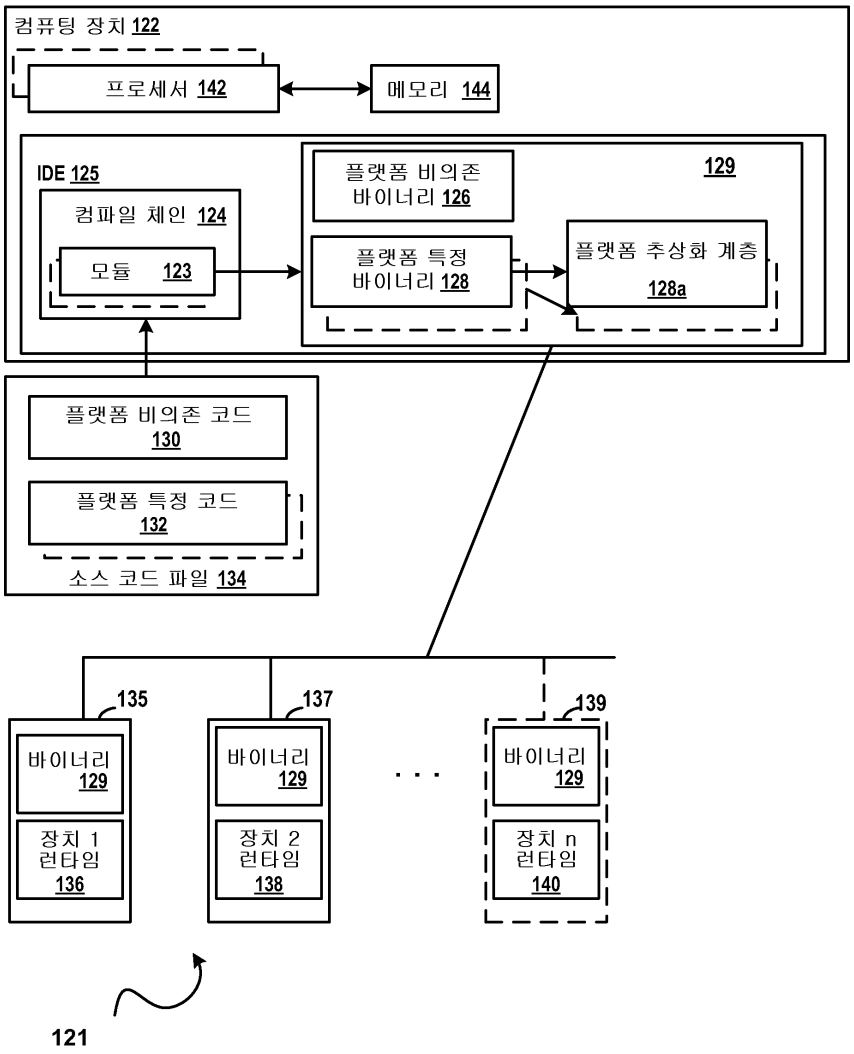
(종래기술)


조건부 컴파일 지시를 갖는 소스 코드
101
컴파일러 102
스마트폰을 위한 바이너리 104a
태블릿을 위한 바이너리 104b
게임 박스를 위한 바이너리 104c
데스크탑을 위한 바이너리 104d
장치 n을 위한 바이너리 104n
스마트폰을 위한 바이너리 104a
스마트폰의 런타임 106a
실행 결과 a 108a
태블릿을 위한 바이너리 104b
태블릿의 런타임 106b
실행 결과 b 108b
게임 박스를 위한 바이너리 104c
게임 박스의 런타임 106c
실행 결과 c 108c
데스크탑을 위한 바이너리 104d
데스크탑의 런타임 106d
실행 결과 d 108d
장치 n을 위한 바이너리 104n
장치 n의 런타임 106n
실행 결과 n 108n
100

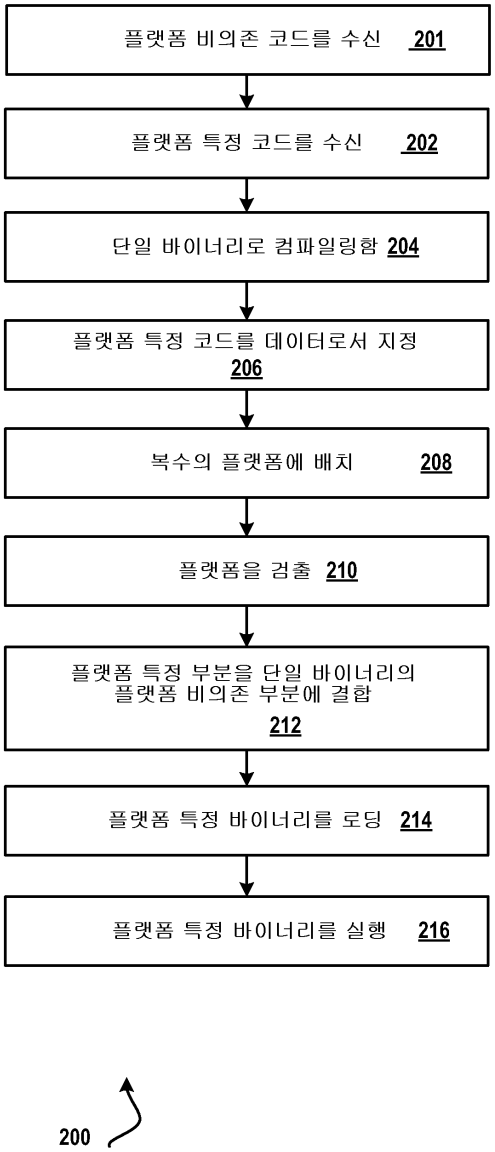
도면1b


마크업
소스 코드
110
수정된
컴파일러 112
플랫폼
비의존
부분 114
스마트폰 특정 부분 116a
태블릿 특정 부분 116b
게임 박스 특정 부분 116c
데스크탑 특정 부분 116d
장치 n 특정 부분 116n
115
플랫폼 불가지론
부분 114
스마트폰 특정
부분 116a
태블릿 특정 부분
116b
게임 박스 특정
부분 116c
데스크탑 특정
부분 116d
장치 n 특정
부분 116n
115
109
스마트폰의
런타임
118a
태블릿의 런타임
118b
게임 박스의 런타임
118c
데스크탑의 런타임
118d
장치 n의 런타임
118n
플랫폼 비의존
부분 114
스마트폰 특정
부분 116a
플랫폼 비의존
부분 114
태블릿 특정 부분
116b
플랫폼 비의존
부분 114
게임 박스 특정
부분 116c
플랫폼 비의존
부분 114
데스크탑 특정
부분 116d
플랫폼 비의존
부분 114
장치 n 특정
부분 116n
실행 결과 a
120a
실행 결과 b
120b
실행 결과 c
120c
실행 결과 d
120d
실행 결과 n
120n

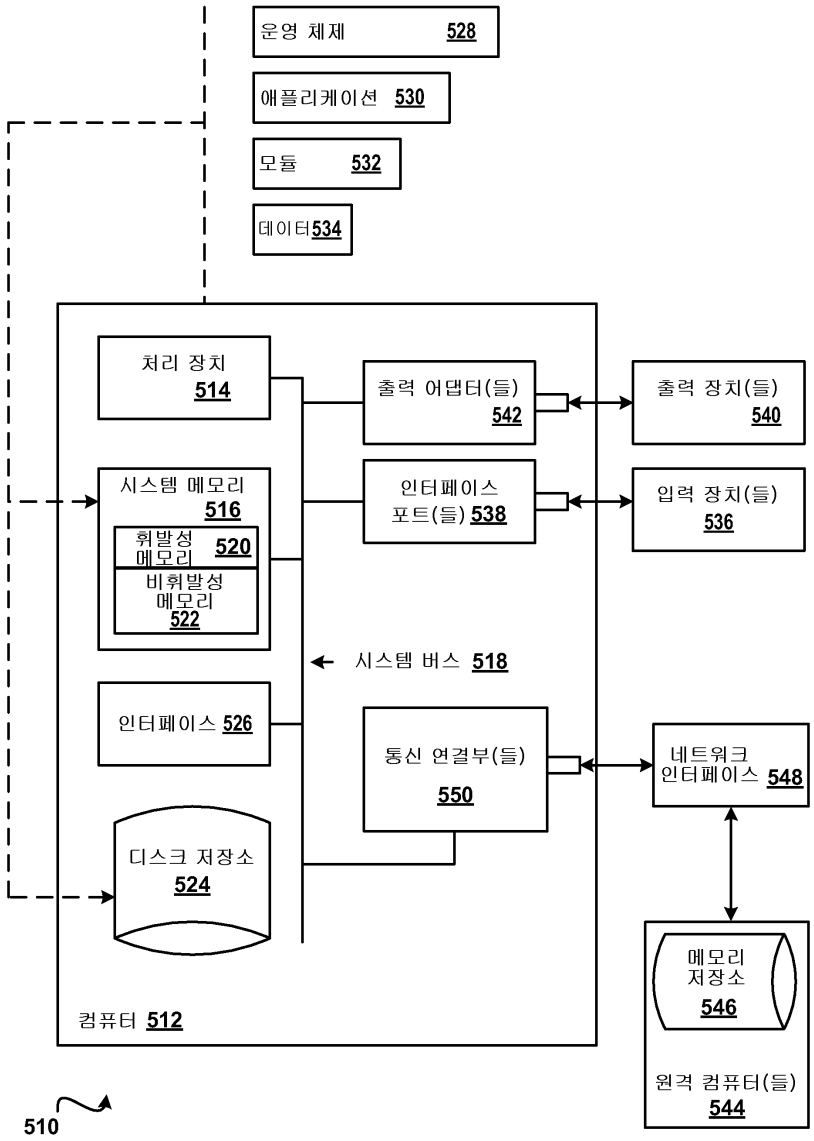
도면1c


컴퓨팅 장치 122
프로세서 142
메모리 144
IDE 125
컴파일 체인 124
모듈 123
플랫폼 비의존 바이너리 126
129
플랫폼 특정 바이너리 128
플랫폼 추상화 계층 128a
플랫폼 비의존 코드 130
플랫폼 특정 코드 132
소스 코드 파일 134
135
137
139
바이너리 129
장치 1 런타임 136
바이너리 129
장치 2 런타임 138
바이너리 129
장치 n 런타임 140
121

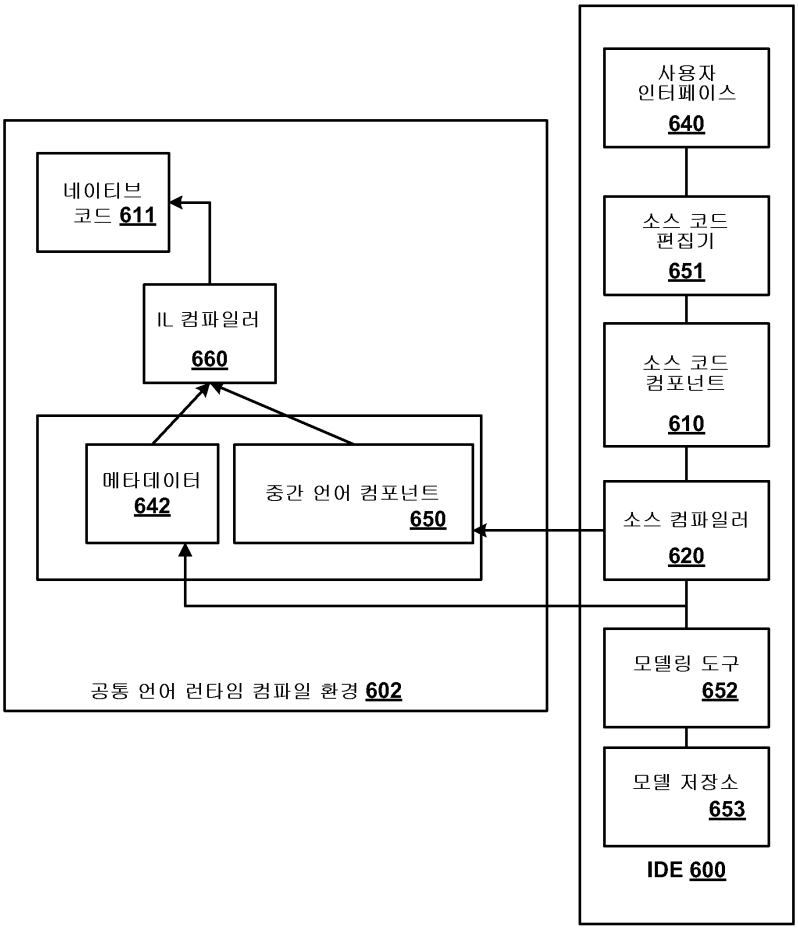
도면2


플랫폼 비의존 코드를 수신 201
플랫폼 특정 코드를 수신 202
단일 바이너리로 컴파일링함 204
플랫폼 특정 코드를 데이터로서 지정 206
복수의 플랫폼에 배치 208
플랫폼을 검출 210
플랫폼 특정 부분을 단일 바이너리의 플랫폼 비의존 부분에 결합 212
플랫폼 특정 바이너리를 로딩 214
플랫폼 특정 바이너리를 실행 216
200

도면3


운영 체제 528
애플리케이션 530
모듈 532
데이터534
처리 장치 514
출력 어댑터(들) 542
출력 장치(들) 540
시스템 메모리 516
휘발성 메모리 520
비휘발성 메모리 522
인터페이스 포트(들) 538
입력 장치(들) 536
시스템 버스 518
인터페이스 526
통신 연결부(들) 550
네트워크 인터페이스 548
디스크 저장소 524
메모리 저장소 546
컴퓨터 512
원격 컴퓨터(들) 544
510

도면4


사용자
인터페이스
640
소스 코드
편집기
651
소스 코드
컴포넌트
610
소스 컴파일러
620
모델링 도구
652
모델 저장소
653
IDE 600
네이티브
코드 611
IL 컴파일러
660
메타데이터
642
중간 언어 컴포넌트
650
공통 언어 런타임 컴파일 환경 602