(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2007년11월05일
(11) 등록번호 10-0773004
(24) 등록일자 2007년10월29일

(51) Int. Cl.
G06F 9/44(2006.01) *G06F 15/177*(2006.01)
(21) 출원번호 10-2005-7008333
(22) 출원일자 2005년05월11일
심사청구일자 2005년12월02일
번역문제출일자 2005년05월11일
(65) 공개번호 10-2005-0086494
공개일자 2005년08월30일
(86) 국제출원번호 PCT/GB2004/000068
국제출원일자 2004년01월09일
(87) 국제공개번호 WO 2004/077292
국제공개일자 2004년09월10일
(30) 우선권주장
10/339,762 2003년01월09일 미국(US)
(56) 선행기술조사문헌
US 2002/069353 A1
KR1020000063253 A
"Smartinstall for PCs: allow multiple pre-load images to be restored from a single Recovery CD", IBM
전체 청구항 수 : 총 8 항

(73) 특허권자
인터내셔널 비지네스 머신즈 코포레이션
미국 10504 뉴욕주 아몬크 뉴오차드 로드
(72) 발명자
포스터 로버트
미국 텍사스주 78759 오스틴 시에라 옥스 10604
그로쉬 마크
미국 텍사스주 78734-4665 레이크웨이 레어 이글 코트 104
(뒷면에 계속)
(74) 대리인
김태홍, 신정건

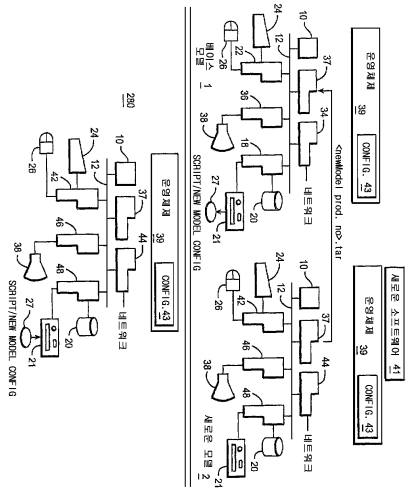
심사관 : 퇴-김성배

(54) 시스템 부팅시 하드웨어를 구성하는 동안 사용자의 상호작용을 없애기 위한 시스템 및 장치

(57) 요 약

사용자가 초기화 또는 구성자 소프트웨어 유틸리티 또는 프로그램과 상호작용할 필요 없이 컴퓨터 시스템을 업데이트할 수 있도록 하는 시스템 및 방법이 제공된다. 더욱 구체적으로 본 발명은 시스템 제공자, 제조업체, 또는 서비스 제공자가 인스톨 파일을 생성할 수 있게 하는데, 이 인스톨 파일은 사용자에게 제공될 때, 새로이 추가된/제거된 하드웨어를 자동적으로 수용하도록 시스템을 구성할 것이다. 시스템의 베이스 모델과 새로운 모델이 유지된다. 새로운 모델은 새로운 그래픽 어댑터, 통신 어댑터, I/O 제어기 등과 같은 하나 이상의 상이한 구성을 포함할 것이다. 제공자는 사용자가 초기화 또는 구성자 소프트웨어 유틸리티 또는 프로그램과 상호작용할 필요 없이 컴퓨터 시스템을 업데이트할 수 있도록 하는 시스템 및 방법을 제공할 것이다. 더욱 구체적으로, 본 발명은 시스템 제공자, 제조업체, 또는 서비스 제공자가 인스톨 파일을 생성할 수 있게 하는데, 이 인스톨 파일은 사용자에게 제공될 때, 새로이 추가된/제거된 하드웨어를 자동적으로 수용하도록 시스템을 구성할 것이다. 시스템의 베이스 모델과 새로운 모델이 유지된다. 새로운 모델은 새로운 그래픽 어댑터, 통신 어댑터, I/O 제어기 등과 같은 하나 이상의 상이한 구성을 포함할 것이다. 제공자는 베이스 모델과 함께 사용될 복구/인스톨 이미지를 생성할 것이다. 구성자 프로그램을 포함하는 운영 체제는 베이스 모델 시스템과 새로운 모델 시스템 모두에서 실행되고 있을 것이다. 장치 드라이버와 같이 새로운 모델의 변화된 하드웨어 구성과 함께 사용될 소프트웨어는, 새로운 모델 시스템 상에 인스톨되고, 그 구성자 프로그램을 이용하여 초기화된다. 베이스 모델 시스템에 대한 중요한 제품 데이터를 이용하여 새로운 모델에 대한 파일이 생성된다. 그 후 이러한 새로운 모델 구성 파일이 베이스 모델 기계에 인스톨되고, 그 구성자 프로그램이 실행된다. 그 후 (새로운 모델 구성 데이터를 포함하는) 구성 정보는 시스템 이미지로서 저장되고 스크립트 파일이 이 시스템 이미지에 부가된다. 새로운 모델 시스템을 가진 최종 사용자들이 CD 등에 의해 이러한 새로운 모델 이미지를 이용할 수 있게 된다. 새로운 모델 이미지 내의 스크립트 파일은 시작 우선순위를 특정하는데, 이 시작 우선순위는 구성자 프로그램 전에 호출되어, 시스템 제공자의 새로운 모델 기계로부터 캡쳐된 적당한 소프트웨어로 시스템을 초기화할 것이다.

대표도 - 도5



(72) 발명자

엔구옌 밍

미국 텍사스주 78726 오스틴 너트우드 코브 11303

왕 크리스틴 이주

미국 텍사스주 78660 플루거빌 리피 코브 17706

특허청구의 범위

청구항 1

데이터 프로세싱 시스템을 자동적으로 구성하는 방법에 있어서,

베이스 데이터 프로세싱 시스템에 대하여 복구 파일을 생성하는 단계;

상기 복구 파일 및 새로운 소프트웨어를 상기 베이스 데이터 프로세싱 시스템과 상이한 적어도 하나의 장치를 구비한 변경된 데이터 프로세싱 시스템 상에 인스톨하는 단계;

상기 변경된 데이터 프로세싱 시스템 상에 포함된 상기 새로운 소프트웨어를 이용하여, 상기 변경된 데이터 프로세싱 시스템의 적어도 하나의 장치에 대한 구성 파일을 생성하는 단계;

상기 구성 파일을 상기 베이스 데이터 프로세싱 시스템 상에 인스톨하고, 상기 구성 파일에 기초하여 상기 베이스 데이터 프로세싱 시스템을 구성하고, 상기 구성된 베이스 데이터 프로세싱 시스템의 이미지를 캡쳐하고, 상기 구성 파일, 상기 베이스 데이터 프로세싱 시스템의 이미지 및 하나 이상의 명령의 추가에 기초하여 변경된 구성 파일을 생성하여, 상기 변경된 상기 구성 파일을 자동적으로 호출하는 단계; 및

상기 변경된 구성 파일을 자동적으로 호출함으로써 상기 적어도 하나의 상이한 장치가 구성되도록, 제2의 변경된 데이터 프로세싱 시스템 상에 상기 변경된 구성 파일을 인스톨하는 단계를 포함하는 데이터 프로세싱 시스템 자동 구성 방법.

청구항 2

제1항에 있어서,

상기 구성 파일은 상기 베이스 데이터 프로세싱 시스템에 대한 베이스 제품 정보를 저장하는 것인 데이터 프로세싱 시스템 자동 구성 방법.

청구항 3

제2항에 있어서,

상기 제2의 변경된 데이터 프로세싱 시스템 상에 상기 변경된 구성 파일을 인스톨하는 단계는,

상기 제2의 변경된 데이터 프로세싱 시스템과 관련된 제2의 제품 정보를 판독하는 단계;

상기 베이스 제품 정보와 상기 제2의 제품 정보를 비교하는 단계; 및

상기 베이스 제품 정보와 상기 제2의 제품 정보가 일치하지 않을 때, 상기 변경된 구성 파일을 자동적으로 호출하는 단계를 포함하는 것인 데이터 프로세싱 시스템 자동 구성 방법.

청구항 4

제3항에 있어서,

상기 변경된 구성 파일을 자동적으로 호출하는 단계는,

상기 적어도 하나의 상이한 장치가 인스톨된 후 처음으로 상기 제2의 변경된 데이터 프로세싱 시스템이 초기화되고 있을 때 상기 변경된 구성 파일을 자동적으로 호출하는 단계를 더 포함하는 것인 데이터 프로세싱 시스템 자동 구성 방법.

청구항 5

제3항에 있어서,

상기 베이스 제품 정보와 상기 제2의 제품 정보가 일치할 때, 사용자가 상기 제2의 변경된 데이터 프로세싱 시스템과 상호 작용하는 것을 가능하게 하는 단계를 더 포함하는 것인 데이터 프로세싱 시스템 자동 구성 방법.

청구항 6

제5항에 있어서,

상기 적어도 하나의 상이한 장치가 인스톨된 후 처음 이후에 상기 제2의 변경된 데이터 프로세싱 시스템이 인스톨되고 있을 때 사용자가 상기 두 번째 변경된 데이터 프로세싱 시스템과 상호작용하는 것을 가능하게 하는 단계를 더 포함하는 것인 데이터 프로세싱 시스템 자동 구성 방법.

청구항 7

제1항 내지 제6항 중 어느 한 항에 의한 방법을 수행하는 수단들을 포함하는, 데이터 프로세싱 시스템의 자동 구성 시스템.

청구항 8

삭제

청구항 9

삭제

청구항 10

삭제

청구항 11

삭제

청구항 12

삭제

청구항 13

제1항 내지 제6항 중 어느 한 항에 의한 방법을 수행하는 명령어들을 포함하는 컴퓨터 프로그램이 저장된 컴퓨터로 판독가능한 기록 매체.

명 세 서

기 술 분 야

<1> 본 발명은 네트워크 장치, 시스템 제어 콘솔 등과 같은 폐쇄 시스템의 하드웨어 및 소프트웨어 구성을 업그레이드하는 것에 관한 것이다. 더욱 구체적으로 본 발명은 운영 체제가 새로운 하드웨어 부품의 존재 또는 이전에 인식된 하드웨어 부품의 부재를 검출할 때 시스템 부팅 동안 사용자의 상호작용을 없애는 기술을 제공한다.

배 경 기 술

<2> 컴퓨터 기술이 끊임없이 발전함에 따라 시스템은 점점 더 대규모화되고 복잡해진다. 예를 들어, 많은 서버 유형의 컴퓨터들이 연결되어, 수백 개의 또는 수천 개의 상호 연결된 컴퓨터 프로세서를 포함하는 다중 노드 시스템을 형성한다. 이러한 프로세서들은 많은 논리적 파티션으로 조직화될 수 있는데, 이러한 파티션들은, 다중 노드 시스템의 다양한 기능을 제어하고 모니터하는 등의 일을 하는 다수의 하드웨어 관리 콘솔에 의해 제어될 것이다. 이러한 컴퓨팅 시스템은 매우 큰 리소스와 구매자들에 의한 재정적 부담을 보여준다. 이러한 컴퓨팅 시스템의 고비용 때문에 구매자들은 기술이 변화하는 만큼 빨리 그들의 전체 시스템과 관리 콘솔을 업그레이드하지는 않을 것이다. 그러나, 주변 장치와 소프트웨어는 기존의 컴퓨터 시스템에 주기적으로 추가되어 그래픽, 비디오 처리, 통신, 네트워킹, 저장 장치 등과 같은 다양한 분야에서 시스템의 성능을 향상시킬 것이다.

<3> 또한, 소위 "폐쇄(closed)" 컴퓨터 시스템이라 불리는 다른 시스템이 있는데 이들은 오늘날 컴퓨터에 의해 작동되는 세상에서 더욱더 일반화되고 있다. "폐쇄" 시스템은 일반적으로 고정된 하드웨어 및 소프트웨어 구성을 가진 시스템이다. 예를 들어, 이러한 시스템은 PDA, 휴대폰, 셋톱 박스, 자동차에서 발견되는 것과 같은 전용 제어기 등을 포함할 수 있다. 또한, 이러한 유형의 시스템은 특정한 유형의 네트워킹 애플리케이션 전용으로서 업데이트가 되지 않는 컴퓨터 시스템인 네트워크 장치를 포함한다. 다른 "폐쇄" 시스템은 다중 프로세싱 또는 클러스터링 환경에서 함께 연결된 다양한 컴퓨터 시스템을 제어하는데 사용되는 하드웨어 관리 콘솔을

포함한다. 하드웨어 관리 콘솔은 또한 종종 데이터 프로세싱 시스템에서 다양한 파티션들을 관리하는데 사용된다. 파티셔닝(partitioning)이란 단일 시스템에서 발견되는 하드웨어 리소스들(프로세서, 메모리, I/O 어댑터, 통신 어댑터 등)을 그 시스템에서 실행중인 다중 운영 체제 인스턴스에 할당하는 것을 말한다. 파티셔닝은 특정한 운영 체제 인스턴스만이 다양한 명령어들과 데이터를 이용할 수 있게 하는 어드레스 변환 기법을 사용하여, 스위치, 라우터 등을 통하여 물리적으로 또는 논리적으로 행해질 수 있다. 마찬가지로, 하드웨어 관리 장치는 본 발명이 생각하는 "폐쇄 시스템"의 또 다른 유형이다. 하드웨어 관리 장치는 컴퓨터 시스템에서 존재하는 다양한 기능을 제어하고/제어하거나 모니터하는데 사용되는데, 이러한 기능으로는 부품들이 적당한 순서로 초기화되는 것 등을 보장하기 위하여, 관리되고 있는 시스템의 전원을 켜고 끄는 것 등이 있다.

<4> 특정한 유형의 하드웨어 및 소프트웨어와 함께 동작하도록 구성된 데이터 프로세싱 장치는 폐쇄 시스템으로 간주될 수 있다. 모두는 아니지만 대부분의 폐쇄 시스템은 소프트웨어가 업그레이드되도록 할 어떤 유형의 입력/출력(I/O) 장치를 포함할 것이다. 예를 들어, PDA는 데스크탑 PC에 대한 접속을 제공하는 동기화 포트를 포함할 수 있다. 휴대폰은 무선 접속을 통해 액세스될 수 있고, 제어기는 일반적으로 외부 데이터를 액세스하기 위한 어떤 유형의 통신 포트를 포함한다. 또한, 하드웨어 관리 콘솔은 통신 포트 등뿐만 아니라 CD ROM 또는 디스켓 드라이브와 같은 I/O 장치를 포함할 수 있다. 이렇게 해서, 시스템은 그들의 특정한 소프트웨어 구성에 기초하여 "폐쇄"적인 것으로 일반적으로 간주되고, 이는 하드웨어 부품 및 그들의 대응 장치 드라이버의 추가를 제공하지(계획하지) 않는다.

<5> 그러나, 모든 유형의 컴퓨터와 마찬가지로, "폐쇄" 시스템을 새로운 하드웨어 및 소프트웨어 부품으로 업그레이드하는 것이 종종 바람직하다. 이 경우 각 컴퓨터 시스템 모델을 위한 복구/인스톨 이미지가 생성되어야 한다. 복구/인스톨 이미지는 소프트웨어를 특정한 유형의 컴퓨터 시스템상에 미리 로딩하기 위하여 제조업체에 의해 사용된다. 또한, 복구/인스톨 이미지는 하드 드라이버 등과 같은 주요 기계 부품이 고장나는 경우 소프트웨어를 다시 인스톨하기 위하여 사용자에 의해 사용된다. 각 기계 모델을 위하여 별개의 복구/인스톨 이미지를 가지는 것은 비용이 많이 드는 계획이고 유지 문제도 생기는데, 이는 동일한 유형의 시스템의 각 상이한 모델이 상이한 기본 소프트웨어 집합을 가질 것이기 때문이다.

<6> 또한, 많은 운영 체제가 상호작용이 없는(non-interactive) 하드웨어 검출 기능을 포함하고 있으면서도 이 운영 체제들은 새로운 하드웨어가 추가될 때 또는 진단 프로그램이 실행될 필요가 있을 때 시스템 소프트웨어를 클린업(clean up)하기 위하여 사용자가 로그인하여 다양한 명령어를 실행시킬 것을 요구한다. 이러한 유형의 해결책은 하드웨어 관리 콘솔을 포함하는 많은 컴퓨터 장치의 경우 실용적이지 못하다는 것을 알 수 있는데, 이는 이들이 폐쇄 시스템이기 때문이다.

<7> 본 발명과 같은 시스템에 대한 필요성을 보여주는 한가지 예는 새로운 하드웨어 부품이 해제되고(release) 시스템 업그레이드가 요구될 때 발생한다. 만일 이전의 하드웨어를 위하여 생성된 동일한 이미지가 인스톨을 위해 사용된다면, 운영 체제(AIX, Linux)는 비디오 카드, 이더넷 카드 등과 같은 새로운(또는 제거된) 하드웨어를 검출할 것이다. 일반적으로 시스템 시작(부팅) 동안 초기화 프로그램이 호출된다. 이러한 초기화 프로그램은 만일 그래픽 사용자 인터페이스(GUI)와의 사용자 상호작용이 없다면, 일반적으로 미리 정해진 시간 예를 들어, 30초 후에 타임아웃이 될 것이다. GUI는 실제 시스템 명령을 기억하여 도스의 C;프롬프트와 같은 명령 프롬프트에서 실제 시스템 명령을 입력할 것을 요구하지 않고, 사용자가 디스플레이된 다양한 아이콘으로부터 선택하고/선택하거나 정의된 필드에서 직관적인 단어로 된 명령을 타이핑함으로써 사용자가 시스템과 상호작용할 수 있게 하는 통상적으로 사용자에게 친숙한 소프트웨어 애플리케이션이다.

<8> 종래의 시스템의 경우 사용자가 30초 후에 시스템과 상호작용하지 않는다면 초기화 프로그램이 타임아웃된다. 변경되고 있는 하드웨어가 디스플레이 모니터를 제어하는 그래픽 또는 비디오 어댑터 카드일 때 심각한 문제가 존재할 것이라는 것은 당연하다. 더욱 구체적으로 새로운 디스플레이 어댑터 카드가 인스톨될 때, 시스템이 새로운 장치 드라이버로 초기화될 때까지 디스플레이 모니터는 제 기능을 수행하지 못할 것이다. 그러나, 초기화 프로그램이 실행될 때까지 모니터상에 디스플레이된 GUI를 통하여 사용자로부터의 입력을 요구하는 것은 일어날 수 없다. 또한, 초기화 소프트웨어로부터의 GUI가 통신 어댑터, 확장 메모리 등과 같은 새로이 인스톨된 장치로부터의 입력을 기다리고 있을 때 이러한 상황이 발생할 수 있다. 그러나, 초기화 프로그램으로의 사용자 입력에 좌우되는 적합한 소프트웨어가 인스톨되지 않고서 장치는 응답할 수 없다.

<9> 따라서, 사용자의 상호작용 없이도 폐쇄 시스템을 업그레이드하는 것을 지원하는 기술에 대한 필요성이 존재한다는 것을 알 수 있다.

발명의 상세한 설명

<10> 종래 기술과 대조적으로, 본 발명은 사용자가 초기화 프로그램, 구성자(configurator) 소프트웨어 유틸리티 등과 상호작용할 필요 없이 컴퓨터 시스템을 업데이트할 수 있게 하는 메카니즘을 제공한다.

<11> 넓은 의미에서, 본 발명은 인스톨 파일이 사용자에게 제공될 때 새로이 추가된/제거된 하드웨어를 수용하도록 시스템을 자동적으로 구성하는 인스톨 파일을, 제조업체 또는 서비스 제공자가 생성할 수 있도록 한다.

<12> 따라서, 제1 특징에 의하면 본 발명은 데이터 프로세싱 시스템을 자동적으로 구성하는 방법을 제공하는데, 베이스 데이터 프로세싱 시스템을 위한 복구 파일을 생성하는 단계; 상기 베이스 데이터 프로세싱 시스템과 상이한 적어도 하나의 장치를 구비한 변경된 데이터 프로세싱 시스템에 복구 파일을 인스톨하는 단계; 상기 변경된 데이터 프로세싱 시스템에 포함된 소프트웨어를 사용하여 구성 파일을 생성하는 단계; 상기 베이스 데이터 프로세싱 시스템에 상기 구성 파일을 인스톨하고 상기 구성 파일을 자동으로 호출하는 명령을 추가하는 단계; 상기 적어도 하나의 상이한 장치가 상기 구성 파일을 자동으로 호출함으로써 구성되도록, 두 번째 상기 변경된 데이터 프로세싱 시스템에 상기 구성 파일을 인스톨하는 단계를 포함한다.

<13> 제2 특징에 의하면 본 발명은 데이터 프로세싱 시스템을 자동적으로 구성하는 시스템을 제공하는데, 베이스 데이터 프로세싱 시스템을 위한 복구 파일을 생성하는 수단; 상기 베이스 데이터 프로세싱 시스템과 상이한 적어도 하나의 장치를 구비한 변경된 데이터 프로세싱 시스템 상에 상기 복구 파일을 인스톨하는 수단; 상기 변경된 데이터 프로세싱 시스템 상에 포함된 소프트웨어를 이용하여 구성 파일을 생성하는 수단; 상기 베이스 데이터 프로세싱 시스템 상에 상기 구성 파일을 인스톨하고, 상기 구성 파일을 자동적으로 호출하는 명령을 추가하는 수단; 및 상기 적어도 하나의 상이한 장치가 상기 구성 파일을 자동적으로 호출함으로써 구성되도록 두 번째 상기 변경된 데이터 프로세싱 시스템 상에 상기 구성 파일을 인스톨하는 수단을 포함한다.

<14> 제3 특징에 의하면, 본 발명은 제1 특징의 방법에 따라서 데이터 프로세싱 시스템을 자동적으로 구성하는 프로그램 명령어들을 컴퓨터로 판독가능한 매체에 저장한 컴퓨터 프로그램 제품을 제공한다.

<15> 바람직하게는 구성 파일은 사용자와의 상호 작용 없이 두 번째 변경된 데이터 프로세싱 시스템 상에서 호출된다.

<16> 바람직하게는 두 번째 상기 변경된 데이터 프로세싱 시스템 상에 구성 파일을 인스톨하는 것은, 인스톨된 운영체제 소프트웨어를 사용자가 변경하지 않고 두 번째 변경된 데이터 프로세싱을 구성하는 것을 포함한다.

<17> 바람직하게는 구성 파일은 베이스 데이터 프로세싱 시스템을 위한 베이스 제품 정보를 저장한다.

<18> 바람직하게는 두 번째 변경된 데이터 프로세싱 시스템 상에 구성 파일을 인스톨하는 것은, 두 번째 변경된 데이터 프로세싱 시스템과 관련된 두 번째 제품 정보를 판독하는 단계; 베이스 제품 정보와 두 번째 제품 정보를 비교하는 단계; 및 베이스 제품 정보와 두 번째 제품 정보가 일치하지 않을 때, 구성 파일을 자동적으로 호출하는 단계를 포함한다.

<19> 옵션으로서, 적어도 하나의 상이한 장치가 인스톨된 후 처음으로 두 번째 변경된 데이터 프로세싱 시스템이 초기화되고 있을 때 구성 파일이 자동적으로 호출된다.

<20> 바람직하게는 제품 정보와 상기 두 번째 제품 정보가 일치할 때, 사용자가 상기 두 번째 변경된 데이터 프로세싱 시스템과 상호 작용하는 것이 가능해진다.

<21> 옵션으로서, 상기 적어도 하나의 상이한 장치가 인스톨된 후 처음 이후에 상기 두 번째 변경된 데이터 프로세싱 시스템이 인스톨되고 있을 때 사용자가 상기 두 번째 변경된 데이터 프로세싱 시스템과 상호 작용하는 것이 가능해진다.

<22> 바람직한 일실시예에서, 시스템 제공자는 컴퓨터 시스템의 베이스 모델과, 새로운 그래픽 카드, 통신 어댑터, I/O 제어기 등과 같은 하나 이상의 상이한 구성을 포함하는 시스템의 새로운 모델을 둘 다 유지할 것이다. 시스템 제공자는 베이스 모델과 함께 사용될 복구/인스톨 이미지를 생성하였을 것이다. 구성자 프로그램을 포함하는 운영 체제는 베이스 모델 시스템과 새로운 모델 시스템 모두에서 실행되고 있을 것이다.

<23> 다음으로 새로운 모델의 추가된 하드웨어와 함께 사용될 장치 드라이버와 같은 새로운 소프트웨어가 제공자의 새로운 모델 시스템 상에 인스톨되고, 구성자 프로그램을 이용하여 초기화된다. 새로운 모델을 위한 파일이 베이스 모델 시스템을 위한 매우 중요한 제품 데이터를 사용하여 생성된다. 이러한 새로운 모델 구성 파일은 그

후 베이스 모델 기계 상에 인스톨되고, 그 구성 프로그램이 실행된다. (새로운 모델 구성 데이터를 포함하는) 구성 정보가 그 후 저장되고, 이러한 구성 정보에 스크립트 파일이 추가된다. 스크립트 파일은 매크로 또는 배치(batch) 파일에 대한 다른 용어인데, 스크립트는 사용자의 상호 작용 없이 실행될 수 있는 명령의 리스트이다. 스크립트 언어는 스크립트를 작성하는 프로그래머에 의해 사용될 수 있는 간단한 프로그래밍 언어이다.

<24> 많은 가능한 시나리오 중 하나에서 이러한 이미지(새로운 모델 시스템에 대한 구성 데이터 및 스크립트 파일)가 CD로 캡쳐된다. 물론 다른 임의의 유형의 컴퓨터로 판독가능한 저장 매체 또는 파일 전송 방법이 본 발명의 범위 내에서 생각될 수 있다.

<25> 그 후 이러한 새로운 모델 이미지를 최종 사용자들이 이용할 수 있게 되는데, 최종 사용자들은 시스템 제공자에 의해 사용된 새로운 모델 시스템과 동일한 구성을 가지고 있다. 예를 들어, 특정한 그래픽 어댑터 업그레이드가 요구될 때, 새로운 모델 시스템은 새로운 그래픽 어댑터만 제외하고 기본 모델과 동일할 것이다. 시스템 제공자는 그 후 사용자에게 어댑터와 함께 새로운 모델 이미지를 공급할 수 있을 것이고, 사용자는 그 운영 체제 내의 구성자 유틸리티를 호출할 필요 없이 어댑터를 물리적으로 인스톨하고 구성할 수 있다.

<26> 더욱 구체적으로 새로운 모델 이미지 내의 스크립트 파일은 구성자 프로그램 이전에 호출될 수 있는 시작 우선순위를 특정할 것이고, 계속해서 시스템 제공자의 새로운 모델 기계로부터 캡쳐된 적당한 소프트웨어를 가지고 사용자의 새로운 모델 시스템을 초기화할 것이다.

<27> 따라서, 전술한 요약 내용에 따라, 후술하는 상세한 설명, 특허청구범위, 첨부된 도면으로부터 본 발명의 개요, 목적, 특징, 장점이 당업자에게 명백해질 것이다.

실 시 예

<34> 컴퓨터 장치들은 일반적으로 고정된 하드웨어와 소프트웨어 구성을 가지는 폐쇄 시스템이다. 이러한 시스템의 한 예는 IBM 하드웨어 관리 콘솔인데, IBM p690 eServer 데이터 프로세싱 시스템을 관리하는데 사용된다. 일반적으로 하드웨어는 펜티엄 급 마이크로프로세서 등을 포함하는 개인용 컴퓨터이다.

<35> LINUX는 특정한 기준을 만족하는 한, 거의 요금 부과 없이 제공되는 상대적으로 새로운 UNIX와 비슷한 운영 체제이다. 이러한 운영 체제는 상업적인 컴퓨터 시스템 판매자들 간에 사용이 늘어나고 있고 인기가 증가하고 있다. 본 발명의 일실시예에서, 초기화되는 폐쇄 시스템은 LINUX OS를 실행하고 있다. LINUX OS의 배포와 관련된 한가지 가능한 기준은, 만일 LINUX OS가 변경된다면, 특정한 statement와 assertion이 그 프로그램을 변경하는 개인 또는 단체에 의해 만들어진다는 것이다. 새로이 인스톨된 하드웨어를 구성하도록 LINUX를 변경하는 것은 LINUX OS 내에 포함된 장치 드라이버 등과 같은 것이 변경되도록 초래할 수 있다. 이는 실시권(license right)을 유지하기 위하여 변경된 코드에 상대적인 statement를 생성하는 구성 프로그램을 실행하는 사용자에게 부담이 될 수 있다. 따라서, 본 발명의 다른 가능한 장점은 최종 사용자가 LINUX OS에서 헤더 파일을 변경하고 다른 proprietary statement를 변경해야 하는 부담을 없앨 수 있다는 것이다. 이는 LINUX 사용자가 최소한의 영향으로 표준 라이센스 조건 하에 OS의 카피를 유지하는 능력을 유지시킬 것이다.

<36> 일실시예에 따르면, (LINUX를 포함하여) 모든 소프트웨어는 첫 번째 시스템에 인스톨된다. (시스템 상의 모든 소프트웨어의 스냅샷인) 소프트웨어 이미지는 그 후 복구/인스톨 CD로 캡쳐된다. 이 CD는 제조 공정 동안 시스템을 미리 로딩하는데 사용되고 또한 데이터가 하드디스크로부터 손실되었을 때 고객이 시스템을 복구하는데 사용된다.

<37> 종래의 복구/인스톨 CD를 사용함으로써 사용자가 복잡한 재 인스톨 과정을 거쳐야 할 필요성이 없어지고, 사용자는 모든 필요한 소프트웨어가 일관성있는 방식으로 인스톨되는 것이 보장된다. 이러한 종래의 해결 방안은 초기화되고 있는 시스템에 동일한 하드웨어가 부착될 것이라는 것을 가정한다. 즉, 초기화되고 있는 시스템이 첫 번째 또는 베이스 시스템과 동일하다고 가정된다. 그러나, 성능이 향상되고 비용이 감소함에 따라 새로운 하드웨어로 이동할 필요성이 항상 존재하기 때문에, 이러한 종래의 방식은 문제점을 낳는다. 구체적으로 새로운 하드웨어가 해제될 때(release), 만일 이전의 하드웨어(즉, 베이스 시스템)로부터 생성된 동일한 소프트웨어 이미지가 인스톨을 위해 사용된다면, 운영 체제(LINUX)는 새롭고/새롭거나 제거된 하드웨어를 검출하여, 새로운 하드웨어를 구성하고/구성하거나 옛 하드웨어(이더넷, 비디오 카드, 그래픽 어댑터 등)를 제거하기 위하여 사용자와 상호작용할 것이다. LINUX의 경우, "Kudzu"라고 불리는 초기화 프로그램이 시스템 초기화 동안에 호출되고, 사용자가 키보드로 타이핑을 하거나 마우스로 항목을 선택하거나 함으로써 프로그램에 어떤 입력도 제공하지 않으면 30초 후에 타임아웃된다. 이런 일이 발생할 때 만일 예를 들어 새로운 시스템에 새로운 비디오 어댑

터가 존재한다면, 사용자는 모니터 상에 디스플레이된 그래픽 사용자 인터페이스(GUI)를 볼 수 없을 것이다. 이러한 유형의 상황은 초기화 프로그램이 타임아웃되도록 초래하고, 새로운 하드웨어를 가지고 동작하도록 시스템을 재구성하는 것이 불가능할 수 있다.

<38> 본 발명은 운영 체제가 새롭고/새롭거나 제거된 하드웨어를 발견할 때 시스템 부팅 동안 사용자의 상호 작용을 없앤다. 본 발명은 사용자가 운영 체제(본 예에서 LINUX)에 포함된 기존의 소프트웨어 유틸리티 프로그램을 변경할 필요가 없는 방식으로 동작한다. 이는 "무료(free)" 운영 체제로서의 요구 사항으로 인하여 존재하는 LINUX의 라이센싱에 대한 특정한 제한 때문에 LINUX 판매자에게도 장점이 된다.

<39> 본 발명의 바람직한 실시예 및 그 장점은 도면을 참조하면 더 잘 이해할 수 있을 것이고, 도면에 있어서 유사한 도면 부호는 유사하거나 대응하는 구성 요소를 나타낸다.

<40> 도 1을 참조하면, 본 발명과 함께 사용될 수 있는 전형적인 데이터 프로세싱 시스템이 도시되어 있다. CPU(10)는 IBM 사로부터 구입가능한 PowerPC 마이크로프로세서, 인텔사로부터 구입가능한 IA32, IA64 급 마이크로프로세서 등을 포함할 수 있으며, 이들은 시스템 버스(12)에 의해 다양한 다른 시스템 부품들에 상호연결되어 있다. ROM(16)이 버스(12)를 통하여 CPU(10)에 접속되어 있고, 기본 컴퓨터 기능을 제어하는 BIOS(basic input/output system)를 포함한다. RAM(14), I/O 어댑터(18), 통신 어댑터(34)가 시스템 버스(12)에 접속되어 있다. I/O 어댑터(18)는 디스크 저장 장치(20)와 R/W CD(21)와 통신하는 SCSI(small computer system interface) 어댑터가 될 수 있다. 통신 어댑터(34)는 버스(12)를 인터넷과 같은 외부 네트워크와 상호 접속하는 이더넷 카드 또는 DSL(digital subscriber line)과 같은 네트워크 카드가 될 수 있다. 모뎀(40)이 버스(12)에 연결되어 포함되어 있는데, 데이터 프로세싱 시스템이 인터넷 또는 전화선을 경유한 다른 통신 네트워크(LAN, WAN)를 통하여 다른 시스템과 통신할 수 있게 한다. 사용자 입력/출력 장치는 사용자 인터페이스 어댑터(22) 및 디스플레이 어댑터(36)를 통하여 시스템 버스(12)에 접속되어 있다. 키보드(24), 트랙볼(32), 마우스(26), 및 스피커(28)가 모두 사용자 인터페이스 어댑터(22)를 통하여 시스템 버스(12)에 접속되어 있다. 디스플레이 모니터(38)는 디스플레이 어댑터(36)에 의해 시스템 버스(12)에 접속되어 있다. 이러한 방식으로 사용자는 키보드(24), 트랙볼(32), 마우스(26)를 통하여 시스템에 입력할 수 있고, 스피커(28)와 디스플레이(38)를 통하여 시스템으로부터 출력을 수신할 수 있다. 또한, DOS, OS/2, 윈도우 운영 체제 등과 같은 운영 체제(39)가 CPU(10) 상에서 실행되는 것으로 도시되어 있는데, 이는 도 1에 도시된 다양한 부품들의 기능을 조정하는데 사용된다.

<41> 도 2를 참조하면, 본 발명의 바람직한 실시예가 구현될 수 있는 데이터 프로세싱 시스템의 네트워크가 도시되어 있다. 네트워크 데이터 프로세싱 시스템(50)은 본 발명이 구현될 수 있는 컴퓨터들의 네트워크이다. 네트워크 데이터 프로세싱 시스템(50)은 네트워크(52)를 포함하는데, 이는 네트워크 데이터 프로세싱 시스템(50) 내에서 함께 접속된 다양한 장치와 컴퓨터 간에 통신 링크를 제공하는데 사용되는 매체이다. 네트워크(52)는 무선, 유선 통신 링크, 또는 광섬유 케이블과 같은 접속을 포함할 수 있다.

<42> 도시된 예에서 서버(54)가 저장 유닛(56)과 함께 네트워크(52)에 접속된다. 또한, 클라이언트(58, 60, 62)가 네트워크(52)에 접속된다. 네트워크(52)는 무선 또는 광섬유 케이블과 같은 영구적인 접속을 포함하거나, 전화 접속을 통하여 이루어지는 임시 접속을 포함할 수 있다. 통신 네트워크(52)는 또한 다른 공공 및/또는 사설 WAN, LAN, 무선 네트워크, 데이터 통신 네트워크 또는 접속, 인트라넷, 라우터, 위성 링크, 마이크로웨이브 링크, 셀룰러 또는 전화 네트워크, 무선 링크, 광섬유 전송 라인, ISDN 라인, TI 라인, DSL 등을 포함할 수 있다. 일부 실시예에서, 사용자 장치는 본 발명의 범위를 벗어나지 않고 서버(54)에 직접 연결될 수 있다. 또한, 여기서 사용되는 통신은 무선 또는 유선 기술에 의해 가능한 모든 통신을 포함한다.

<43> 클라이언트(58, 60, 62)는 예를 들어, 개인용 컴퓨터, 포터블 컴퓨터, 모바일 또는 고정 사용자 스테이션, 워크스테이션, 네트워크 단말기 또는 서버, 휴대폰, 키오스크, 덤 단말기, PDA, 양방향 페이저(pager), 스마트폰, 정보 장치 또는 네트워크 컴퓨터가 될 수 있다. 본 애플리케이션의 용도를 위하여 네트워크 컴퓨터는 네트워크에 연결된 다른 컴퓨터로부터 프로그램 또는 다른 애플리케이션을 수신하는, 네트워크에 연결된 임의의 컴퓨터가 될 수 있다.

<44> 도시된 예에서, 서버(54)는 부팅 파일, 운영 체제 이미지, 애플리케이션과 같은 데이터를 클라이언트(58-62)에 제공한다. 클라이언트(58, 60, 62)는 서버(14)에 대하여 클라이언트이다. 네트워크 데이터 프로세싱 시스템(50)은 추가적인 서버, 클라이언트, 및 도시되지 않은 다른 장치들을 포함할 수 있다. 도시된 예에서, 네트워크 데이터 프로세싱 시스템(10)은 서로 통신하기 위하여 TCP/IP 프로토콜의 집합을 사용하는 게이트웨이와 네트워크의 세계적인 집합을 나타내는 네트워크(52)와의 인터넷이다. 데이터와 메시지를 라우팅하는 수천 개의 상

업적 컴퓨터 시스템, 정부 컴퓨터 시스템, 교육 컴퓨터 시스템, 및 다른 컴퓨터 시스템들로 구성된 주요 노드 또는 호스트 컴퓨터 사이에, 고속 데이터 통신 라인의 백 본이 인터넷의 중심에 있다. 물론, 네트워크 데이터 프로세싱 시스템(50)이 예를 들어, 인트라넷, LAN, 또는 WAN과 같이 수많은 상이한 유형의 네트워크로서 구현될 수 있다. 도 2는 예로서 도시한 것뿐이며, 본 발명의 아키텍처를 이에 한정하려 한 것이 아니다.

<45> 도 3은 본 발명이 구현될 수 있는 데이터 프로세싱 시스템의 좀 더 상세한 블록도이다. 데이터 프로세싱 시스템(100)은 시스템 버스(106)에 연결된 다수의 프로세서(101, 102, 103, 104)를 포함하는 SMP(symmetric multiprocessor) 시스템이 될 수 있다. 예를 들어, 데이터 프로세싱 시스템(100)은 뉴욕, 아몽크 소재의 IBM 사의 제품인 IBM pSeries, eServer가 될 수 있으며, 네트워크 내의 서버로서 구현된다. 다른 대안으로, 단일 프로세서 시스템이 이용될 수 있다. 또한 시스템 버스(106)에 메모리 제어기/캐시(108)가 접속되는데, 이들은 다수의 로컬 메모리(160-163)에 대한 인터페이스를 제공한다. I/O 버스 브리지(110)는 시스템 버스(106)에 접속되고, I/O 버스(112)에의 인터페이스를 제공한다. 메모리 제어기/캐시(108)와 I/O 버스 브리지(110)가 도시된 바와 같이 통합될 수 있다.

<46> 데이터 프로세싱 시스템(100)은 논리적으로 파티션된 데이터 프로세싱 시스템이다. 따라서, 데이터 프로세싱 시스템(100)은 동시에 실행되는 다수의 이종 운영 체제(또는 단일 운영 체제의 다수의 인스턴스)를 가질 수 있다. 이러한 다수의 운영 체제 각각은 그 내에서 실행되는 임의의 수의 소프트웨어 프로그램을 가질 수 있다. 데이터 프로세싱 시스템(100)은 상이한 I/O 어댑터(120-121, 128-129, 136, 148-149)가 상이한 논리적 파티션에 할당되도록 논리적으로 파티션된다.

<47> 따라서, 예를 들어, 데이터 프로세싱 시스템(100)이 세 개의 논리적 파티션, P1, P2, P3으로 분할된다고 가정하자. I/O 어댑터(120-121, 128-129, 136, 148-149) 각각, 프로세서(101-104) 각각, 로컬 메모리(160-163) 각각은 세 개의 파티션 중 하나에 할당된다. 예를 들어, 프로세서(101), 메모리(160), I/O 어댑터(120, 128, 129)가 논리적 파티션 P1에 할당되고, 프로세서(102-103), 메모리(161), I/O 어댑터(121, 136)가 논리적 파티션 P2에 할당되고, 프로세서(104), 메모리(162-163), I/O 어댑터(148-149)가 논리적 파티션 P3에 할당될 수 있다.

<48> 데이터 프로세싱 시스템(100) 내에서 실행되는 각 운영 체제 이미지는 상이한 논리적 파티션에 할당된다. 따라서, 데이터 프로세싱 시스템(100) 내에서 실행되는 각 운영 체제는 그 논리적 파티션 내에 있는 I/O 유닛만을 액세스할 수 있다.

<49> I/O 버스(112)에 접속된 PCI(peripheral component interconnect) 호스트 브리지(114)는 PCI 로컬 버스(115)에 인터페이스를 제공한다. 다수의 입력/출력 어댑터(120-121)가 PCI 버스(115)에 접속될 수 있다. 전형적인 PCI 버스 구현은 4개 내지 8개 사이의 I/O 어댑터(즉, add-in 커넥터를 위한 확장 슬롯)를 지원할 것이다. 각 I/O 어댑터(120-121)는 데이터 프로세싱 시스템(100)과, 예를 들어 데이터 프로세싱 시스템(100)에 대하여 클라이언트인 다른 네트워크 컴퓨터인 입력/출력 장치 간에 인터페이스를 제공한다.

<50> 추가적인 PCI 호스트 브리지(122)는 추가적인 PCI 버스(123)에 대한 인터페이스를 제공한다. PCI 버스(123)는 PCI 버스(126-127)에 의해 다수의 PCI I/O 어댑터(128-129)에 접속된다. 따라서, 모뎀 또는 네트워크 어댑터 등과 같은 추가적인 I/O 장치는 각 PCI I/O 어댑터(128-129)를 통하여 지원될 수 있다. 이러한 방식으로 데이터 프로세싱 시스템(100)은 다수의 네트워크 컴퓨터에 대한 접속을 가능하게 한다.

<51> 메모리 매핑된 그래픽 어댑터(148)가, 도시된 PCI 버스(144, 145)를 경유하여 PCI 호스트 브리지(140)와 EADS(142)(PCI-PCI 브리지)를 통하여 I/O 버스(112)에 접속될 수 있다. 또한, 하드 디스크(150)가, 도시된 PCI 버스(141, 145)를 경유하여 PCI 호스트 브리지(140)와 EADS(142)를 통하여 I/O 버스(112)에 접속될 수 있다.

<52> PCI 호스트 브리지(130)는 I/O 버스(112)에 접속하기 위하여 PCI 버스(131)에 대한 인터페이스를 제공한다. PCI 버스(131)는 PCI 호스트 브리지(130)를 서비스 프로세서 메일 박스 인터페이스 및 ISA 버스 액세스 통과 로직(194)과 EADS(132)에 접속한다. ISA 버스 액세스 통과 로직(194)은 PCI/ISA 브리지(193)로 예정된 PCI 액세스를 전달한다. NVRAM 저장 장치는 ISA 버스(196)에 접속된다. 서비스 프로세서(135)가 그 로컬 PCI 버스(195)를 통하여 서비스 프로세서 메일박스 인터페이스(194)에 연결된다. 서비스 프로세서(135)는 또한 다수의 JTAG/I^2C 버스(134)를 통하여 프로세서(101-104)에 접속된다. JTAG/I^2C 버스(134)는 JTAG/scan 버스(IEEE 1149.1 참조)와 필립스 I^2C 버스의 조합이다. 그러나, 다른 대안으로 JTAG/I^2C 버스(134) 대신 필립스 I^2C 버스 또는 JTAG/scan 버스를 단독으로 사용할 수 있다. 호스트 프로세서(101, 102, 103, 104)의 모든 SP-ATTN 신호는 서비스 프로세서의 인터럽트 입력 신호에 함께 접속된다. 서비스 프로세서(135)는 자신의 로컬 메모리(19

1)를 가지고 있고, 하드웨어 오피-패널(op-panel, 190)을 액세스한다.

<53> 데이터 프로세싱 시스템(100)이 처음에 전원이 켜질 때, 서비스 프로세서(135)가 JTAG/scan 버스(134)를 이용하여 시스템(호스트) 프로세서(101-104), 메모리 제어기(108), I/O 브리지(110)에 신호를 보낸다. 이 단계가 완료되면, 서비스 프로세서(135)는 데이터 프로세싱 시스템(100)의 목록(inventory) 및 토폴로지를 이해하게 된다. 서비스 프로세서(135)는 또한 시스템 프로세서(101-104), 메모리 제어기(108), I/O 브리지(110)에 신호를 보냄으로써 발견되는 모든 소자에 대하여 BIST(Built-In-Self-Tests), BAT(Basic Assurance Tests), 메모리 테스트를 행한다. 서비스 프로세서(135)가 BIST, BAT, 메모리 테스트 동안 검출된 실패에 대한 모든 에러 정보를 수집해서 보고한다.

<54> 만일 BIST, BAT, 및 메모리 테스트 동안 고장인 것으로 발견된 소자를 빼고서도 시스템 리소스의 의미있는/유효한 구성이 여전히 가능하다면, 데이터 프로세싱 시스템(100)은 계속해서 실행가능한 코드를 로컬(호스트) 메모리(160-163)로 로딩하도록 허용된다. 서비스 프로세서(135)는 그 후 호스트 메모리(160-163)로 로딩된 코드의 실행을 위하여 호스트 프로세서(101-104)를 놓아준다(release). 호스트 프로세서(101-104)가 데이터 프로세싱 시스템(100) 내의 각 운영 체제로부터의 코드를 실행하고 있는 동안, 서비스 프로세서(135)는 에러를 모니터링하고 보고하는 모드로 진입한다. 서비스 프로세서에 의해 모니터링되는 항목의 유형은 예를 들어 냉각 팬 속도 및 동작, 열 센서, 전원 조정기, 및 프로세서(101-104), 메모리(160-163), 버스-브리지 제어기(110)에 의해 보고된 복구가능한 에러와 복구가능하지 않은 에러를 포함한다.

<55> 서비스 프로세서(135)는 데이터 프로세싱 시스템(100) 내의 모든 모니터링되는 항목에 대한 에러 정보를 저장하고 보고할 책임이 있다. 서비스 프로세서(135)는 또한 에러의 유형과 정의된 임계값에 기초하여 액션을 취한다. 예를 들어, 서비스 프로세서(135)는 복구가능한 과도한 에러를 프로세서의 캐시 메모리에 기록하고, 이것이 심각한 고장을 예언한다고 결정할 수 있다. 이러한 결정에 기초하여, 서비스 프로세서(135)는 현재의 실행 세션 및 미래의 IPL(Initial Program Loads) 동안의 재구성을 위하여 그 리소스를 표시할 수 있다. IPL은 때로 "부팅" 또는 "부트스트랩(bootstrap)"이라고도 불린다.

<56> 당업자는 도 3에 도시된 하드웨어가 변화될 수 있다는 것을 인식할 것이다. 예를 들어, 광디스크 드라이브 등과 같은 다른 주변 장치가, 도시된 하드웨어 대신에 사용되거나 도시된 하드웨어와 함께 사용될 수 있다. 도시된 예는 본 발명에 대한 아키텍처에 대한 제한을 의미하려고 한 것이 아니다.

<57> 도 4는 본 발명이 구현될 수 있는 논리적으로 파티션된 전형적인 플랫폼의 블록도이다. 논리적으로 파티션된 프로그램(200)은 파티션된 하드웨어(베이스 하드웨어라고도 불림)(230), 하이퍼바이저라고도 불리는 파티션 관리 펌웨어(210), 파티션(201-204)을 포함한다. 운영 체제(201a-203a)는 파티션(201-203) 내에 존재한다. 운영 체제(201a-203a)는 단일 운영 체제의 다수의 카피일 수도 있고, 플랫폼(200)에서 동시에 실행되는 다수의 이종(heterogeneous) 운영 체제일 수도 있다.

<58> 논리적으로 파티션된 플랫폼(200)은 또한 지정된 서비스 파티션(204)을 포함한다. 서비스 파티션(204)은 파라미터를 설정하고, 펌웨어 업데이트를 인스톨하고, 다른 서비스 기능을 수행하기 위하여 시스템 관리자에 의해 사용될 수 있다.

<59> 하나 이상의 이러한 파티션(201-203)은 시스템 관리자에 의해 원격적으로 사용될 수도 있다. 서비스 파티션(204)은 통상적으로 원격적으로 관리되지는 않는다.

<60> 파티션된 하드웨어(230)는 다수의 프로세서(232-238), 다수의 시스템 메모리 유닛(240-246), 다수의 입력/출력(I/O) 어댑터(248-262), 및 저장 유닛(270)을 포함한다. 프로세서(242-248), 메모리 유닛(240-246), NVRAM 저장 장치(298), I/O 어댑터(248-262) 각각이 다수의 파티션(201-204) 중 하나에 할당될 수 있다.

<61> 파티션된 하드웨어(230)는 또한 서비스 프로세서(290)를 포함한다. DRAM 장치와 같은 비휘발성 메모리 장치(291)가 서비스 프로세서(2910) 내에 포함된다. 다른 정보뿐만 아니라, 여기서 설명된 파티션 테이블과 펌웨어 이미지가 서비스 프로세서 메모리(291) 내에 저장된다.

<62> 파티션 관리 펌웨어(하이퍼바이저)(210)는 논리적으로 파티션된 플랫폼(200)의 파티셔닝을 생성하고 실행하기 위하여 파티션(201-203)을 위한 많은 기능과 서비스를 수행한다. 하이퍼바이저(210)는 내장된 하드웨어와 동일한, 펌웨어로 구현된 버츄얼 기계(virtual machine)이다. 펌웨어는 예를 들어 ROM(read-only memory), PROM(programmable ROM), EPROM(erasable programmable ROM), EEPROM(electrically erasable programmable ROM), 비휘발성 RAM(random-access memory)와 같이 전기 전력 없이 그 내용을 유지하는 메모리 칩에 저장된 소프트웨어이다. 이렇게 해서 하이퍼바이저(210)는 논리적으로 파티션된 플랫폼(200)의 모든 하드웨어 리소스를

가상화함으로써(virtualize) 독립적인 OS 이미지(201a-203a)를 동시에 실행할 수 있게 한다. 하이퍼바이저(210)는 I/O 장치들을 I/O 어댑터(248-262)를 통하여 OS(201a-203a) 중 하나에 의해 사용되는 배타(exclusive) 모드에서 단일 버츄얼 기계에 부착할 수 있다.

<63> 본 발명의 바람직한 실시예에 따라 하드웨어 관리 콘솔(hardware management console, HMC)(280)과 같은 하드웨어 관리 장치가 논리적으로 파티션된 플랫폼(200)을 포함하는 데이터 프로세싱 시스템(100)에 연결될 수 있다. HMC(280)는 서비스 프로세서(290)에 연결된 별개의 컴퓨터 시스템이고, 서비스 프로세서(290)를 통하여 데이터 프로세싱 시스템(100)의 다양한 기능을 제어하기 위하여 사용자에 의해 사용될 수 있다. HMC(280)는 그래픽 사용자 인터페이스(GUI)를 포함하는데, GUI는 재부팅될 파티션을 선택하기 위하여 사용자에 의해 사용될 수 있다.

<64> HMC(280)는 AIX, LINUX 등과 같은 운영 체제 이미지(282)와 서비스 애플리케이션(284)을 포함한다. 서비스 애플리케이션(284)은 다양한 파티션(201-204)로부터 에러 메시지를 수신한다. 서비스 애플리케이션(284)은 또한 규칙적인 간격으로 하트비트 신호를 생성하고 이 하트비트 신호를 운영 체제(282)를 통하여 서비스 프로세서(290)로 출력한다.

<65> 본 발명의 초기화 기술에 대하여 HMC(280)와 같은 하드웨어 관리 콘솔과 연계하여 설명하겠다. 그러나, 본 발명의 범위가 하드웨어 관리 장치, PDA, 네트워크 장치, 내장된 제어기 등과 같이 임의의 폐쇄 유형 시스템에 똑같이 적용될 수 있음은 당연하다.

<66> 본 발명의 바람직한 실시예의 일예로서 두 개의 컴퓨터 시스템이 고려된다. 첫 번째 시스템을 베이스 모델이라고 부르고, 변경된 시스템을 새로운 모델이라 부른다. 베이스 모델은 원래의 복구/인스톨 이미지가 생성되는 컴퓨터 시스템이고, 새로운 모델은 예를 들어, 새로이 추가되는/제거되는 어댑터 등을 가지는 상이한 하드웨어 구성을 포함하는 새로운 하드웨어 플랫폼이다. 새로운 모델은 복구/인스톨 이미지를 가진 소프트웨어가 인스톨될 시스템이다.

<67> 제1 단계에서 모든 소프트웨어가 새로운 모델 시스템 상에 인스톨되고, 시스템 제공자는 새로운 모델 구성자 프로그램(Linux의 경우에 Kudzu)과 상호작용하여, 새로운 모델 시스템 상에 모든 하드웨어를 구성한다. 모든 구성 파일이 그 후에 캡쳐된다. 이러한 파일들은 시스템 제공자의 새로운 모델 상에 현재 인스톨된, 로딩된 장치 드라이버, 하드웨어 구성, X 윈도우와 같은 X 구성, Motif 구성(유닉스 기반의 시스템에서 GUI를 제공함) 등의 이미지를 포함한다. 새로운 모델 이미지는 <newModel product number>.tar, <currentModel product number>.tar 등으로 불리는 파일에 저장되는데, 여기서 새로운 모델 제품 번호는 새로운 모델 컴퓨터 시스템 BIOS에 저장된 제품 번호이다.

<68> 다음으로, <newModel product number>.tar 파일이 베이스 모델 시스템 상에 인스톨된다. 제공자는 베이스 모델 구성자 프로그램과 상호 작용하여 베이스모델 시스템 상에 모든 하드웨어를 구성한다. 새로운 모델 시스템으로부터 캡쳐된 tar 파일은 베이스 모델 제품 번호와 함께 베이스 모델 시스템 상의 공지된 위치에 복사된다. 구성자 프로그램 이전에 시스템 부팅 시에 실행될 스크립트 파일이 제공된다. 일실시예에서 새로운 모델은 변경된 하드웨어 관리 콘솔이 될 수 있고, 스크립트 파일은 hmcConfig 등과 같은 이름이 될 수 있다. 리눅스 운영 체제 환경의 경우, 유틸리티 chkconfig를 이용함으로써 스크립트 파일이 생성될 수 있고, 스크립트를 /etc/rc.d/ 디렉토리 밑에 추가할 수 있다. 스크립트 파일 내부에 시작 우선 순위가 특정되어 있는데, 이는 구성자 프로그램인 Kudzu 이전에 스크립트가 호출될 수 있게 한다. 새로운 모델에서 시작할 때 스크립트는 컴퓨터 시스템의 BIOS로부터 새로운 모델 기계의 제품 이름을 판독하여, 그 정보를 이용하여 특정한 하드웨어 구성 파일을 업데이트할지 여부를 결정할 필요가 있을 것이다. <newModel product number>.tar 파일과 스크립트 파일을 포함하는 이미지가 CD 상에 캡쳐된다. 당업자는 CD가 단지 이용될 수 있는 많은 적당한 파일 전송 방법론 중의 하나라는 것을 인식할 것이다. 예를 들어, 본 발명의 범위 내에서 테이프, 디스켓, 네트워크 접속 등을 모두 예상할 수 있다.

<69> 캡쳐된 이미지(.tar 파일 및 스크립트)는 최종 사용자의 새로운 모델 기계 상에 CD 드라이버 등을 이용함으로써 인스톨된다. 시스템이 처음으로 재부팅할 때, 스크립트 파일이 호출되어 기계의 모델 번호를 결정할 것이다. 만일 새로운 모델 번호가 베이스 모델과 동일하지 않다면(통상적으로 상이할 것이다), 스크립트 파일은 새로운 모델 기계의 VPD(vital product date)를 이용하여 <newModel product number>.tar 파일을 찾아볼 것이다. 이러한 .tar 파일은 캡쳐된 이미지로부터 하드웨어 구성 파일들을 인스톨할 것이다. 최종 사용자의 새로운 모델 기계 상에서 구성자 프로그램이 호출되지만, 이전에 복구된 필요한 하드웨어 구성 파일들이 모두 새로운 모델 시스템과 일치할 것이므로 사용자의 상호작용이 필요하지 않다. 스크립트 파일은 단지 한번 실행될 것이다.

후속적인 재 부팅에서는 스크립트가 즉시 빠져나가서 사용자에게 구성자 프로그램과 상호작용하는 능력이 주어진다. 즉, 본 발명은 향상된 하드웨어 구성을 인스톨할 수 있는 방법을 제공하면서도 차후의 업그레이드 동안 사용자가 구성자 프로그램과 상호작용하는 것을 금지시키지 않는다. 이렇게 해서, 이미 인스톨된 시스템에 하드웨어를 추가하고 구성자 프로그램이 검출을 수행하도록 하는 것이 여전히 가능하다. 또한, 베이스 모델이 쓸모없어질 때, 해야 될 일이라곤 새로운 베이스 모델의 새로운 중요한(vital) 제품 이름을 가지고 /opt/hsc/bin/baseModel.dat 파일을 업데이트하는 것뿐이다.

<70> 전술한 프로세스의 유사 코드 및 코멘트를 아래와 같이 기술한다.

```
if [!-f <some directory>/executed_for_first_time]
        exit 0

    # Read the VPD model from bios to find the product number
    # of the current machine

    baseModel=`cat /opt/hsc/bin/baseModel.dat`
    currentModel=`readVPD -productNumber`

    # If the model number of the machine is not the same as
    # the base model, use the product number to look up
    # a tar file, whose name is the product number, then
<71> # restore the configuration files needed

    if ! $currentModel != $baseModel ]
       if [ -f <some directory>/$currentModel.tar ]
          tar -xf <some directory>/$currentModel.tar

rm -f <some directory>/executed_for_first_time
    exit 0
```

<72>

<73> 도 5를 참조하여, 본 발명의 바람직한 실시예의 동작에 대하여 설명하겠다. 참조 번호 1은 첫 번째 즉 베이스 모델 데이터 프로세싱 시스템을 가리키며, 전술한 바와 같이, 프로세서(10)를 포함한다. 또한, 도 1에 도시된 것과 유사한 부품이 또한 포함되어 있다. 더욱 구체적으로 버스(12)는 네트워크 어댑터(34), I/O 어댑터(18), 디스플레이 어댑터(36), 사용자 인터페이스 어댑터(22)를 상호연결하는데 사용된다. 직렬 포트와 같은 추가적인 I/O 어댑터(37)가 또한 포함되어, 본 발명에 따라 베이스 모델 소프트웨어를 업데이트하는데 사용될 수 있다. 베이스 모델(1)이 또한 전술한 어댑터를 통하여 버스(12)에 연결되는 다양한 주변 장치를 포함한다는 것을 알 수 있다. 예를 들어, 키보드(24)와 마우스(26)가 사용자 인터페이스 어댑터(22)에 연결되어 있다. 디스플레이 모니터(38)가 디스플레이 어댑터(36)와 연계하여 동작한다. I/O 어댑터(18)는 디스크 저장 장치(20)와 판독/기록 CD 드라이브(21)를 버스(12)에 상호연결하는데 사용된다. 따라서, 버스(12)가 다양한 주변 장치들과 프로세서(10) 간에 데이터를 전송하는데 사용된다는 것을 알 수 있다. 리눅스와 같은 운영 체제(39)가 프로세서(10)에서 실행되는 것으로 도시되어 있다. 이러한 운영 체제는 리눅스의 경우에 Kudzu와 같은 구성자 프로그램(43)을 포함할 것인데, 이는 시스템을 초기화하는데 사용될 것이다.

<74> 전술한 바와 같이, 기능성을 향상시키기 위하여 종종 새로운 부품들을 특정한 데이터 프로세싱 베이스 모델에 추가하는 것이 바람직하다. 이는 다양한 새로운 어댑터 카드 등을 추가함으로써 일어날 수 있다. 참조 번호 2는 새로운 하드웨어 및/또는 소프트웨어를 포함하는 업그레이드된 베이스 모델인 새로운 모델을 도시한다. 물론 본 발명에서 기존에 인스톨된 하드웨어가 없는 것 또는 기존 하드웨어의 향상된 버전을 포함하여, 시스템 구성에 임의의 변화를 주는 것을 생각할 수 있다.

<75> 본 예에서 새로운 모델 시스템(2)에 포함된 어댑터는 베이스 모델(1) 내의 어댑터와 비교할 때 실질적으로 업그레이된 것이다. 즉, 새로운 또는 향상된 디스플레이 어댑터(46), 사용자 인터페이스 어댑터(42), I/O 어댑터(48), 네트워크 어댑터(44)가 새로운 모델(2)에서 모두 제공된다. 새로운 모델(2) 내의 나머지 부품 및 소프트

웨어는 베이스 모델(1)에 포함된 것과 동일하고, 유사한 참조 번호에 의해 식별된다는 것을 알 수 있다. 또한, 새로운 소프트웨어(41)가 제공되는데, 이는 장치 드라이버, 및 새로운 하드웨어와 연계하여 사용될 다른 초기화 유형 프로그램을 포함할 것이다. 도 5에 도시된 바와 같이, 본 발명의 사용자에 독립적인 초기화 프로세스를 구성하기 위하여 새로운 모델 데이터 프로세싱 시스템(2)이 제조업체 또는 다른 시스템 제공자에 의해 이용될 것이다.

<76> 추가적으로 도 5는 HMC(280)로 식별되는 새로운 모델(2)과 실질적으로 동일한 시스템을 도시한다. 그러나, HMC(280)는 본 발명에서 생각할 수 있는 한가지 유형의 폐쇄 시스템에 불과하다는 것이 당연하다. 전술한 하드웨어 관리 장치, PDA, 셋톱 박스, 내장된 제어기 등도 본 발명의 범위 내에 있는 것으로 생각된다. 당업자들은 HMC(280)가 실지로 베이스 모델(1)이지만, 새로운 모델(2)에 도시된 새로운 부품들을 포함한다는 것을 이해할 것이다. 있음 직한 시나리오에서는 사용자가 베이스 모델(1) 시스템을 구입하고, 그 시스템을 어떤 기간 동안 사용한 다음에 그 시스템을 디스플레이 어댑터(46), 사용자 인터페이스 어댑터(42), I/O 어댑터(48), 및/또는 네트워크 어댑터(44)를 가지고 업그레이드할 것이다. 사용자는 이러한 방식으로 새로운 모델(2) 시스템을 효과적으로 가지게 될 것이다.

<77> 사용자의 상호 작용 없이 HMC(280)의 초기화를 가능하게 하는 본 발명의 동작에 대하여 도 5와 도 6을 참조하여 이제 설명하겠다.

<78> 단계 1에서 사용자에 독립적인 초기화 프로세스가 시작되고, 단계 2에서 베이스 모델(1)의 재 인스톨/복구 이미지가 생성되어 컴팩트 디스크 등에 저장된다. 다음으로, 생성된 재 인스톨/복구 이미지 및 새로운 소프트웨어(41)가 도 5에 도시된 데이터 프로세싱 시스템의 새로운 모델(2) 상에 인스톨된다(단계 3). 구체적으로 새로운 소프트웨어 및 하드웨어 부품들은 새로운 모델(2) 상에 인스톨된다. 단계 4에서 시스템 제공자는 새로운 모델(2) 구성자 프로그램(43)(리눅스의 경우 Kudzu)과 상호작용하여, 새로운 모델 시스템(2)에서 모든 하드웨어를 구성한다. 그 후 구성 파일들이 캡처된다. 이러한 파일들은 시스템 제공자의 새로운 모델에 현재 인스톨되어 있는, 로딩된 장치 드라이버, 하드웨어 구성, X 윈도우와 같은 X 구성, Motif 구성(유닉스 기반의 시스템에서 GUI를 제공함)의 이미지를 포함한다. 단계 5에서 새로운 모델 이미지를 파일 <newModel product number>.tar 로서 저장하는데, 새로운 모델 제품 번호는 새로운 모델 컴퓨터 시스템 BIOS에 저장된 제품 번호이다.

<79> 새로이 생성된 <newModel product number>.tar 파일에서 캡처된(단계 5) 새로운 모델(2)의 구성 파일들은 그 후 단계 6에서 베이스 모델(1) 상에 인스톨된다. 베이스 모델(1)과 새로운 모델(2) 양쪽에 있는 I/O 어댑터(37) 간의 파일 전송이 <newModel product number>.tar 파일을 새로운 모델(2)로부터 베이스 모델(1)로 전송하여 인스톨하는 하나의 메카니즘으로서 도시되어 있다는 것을 도 5로부터 알 수 있다. 파일을 테이프, CD, 또는 다른 저장 매체에 기록한 후에 그 파일을 매체로부터 시스템으로 인스톨하는 것을 포함하여, 이 파일을 베이스 모델(1) 상에 인스톨하는데 사용될 수 있는 많은 다른 파일 전송 기법이 본 발명에서 생각될 수 있음은 당연할 것이다. 일단 <newModel product number>.tar 파일이 베이스 모델 시스템(1) 상에 인스톨되면, 데이터 프로세싱 시스템 제공자가 베이스 모델(1) 구성자 프로그램(43)과 상호작용하여 베이스 모델 시스템(1) 상에 모든 하드웨어를 구성한다(단계 7). 구성 후에, 새로운 모델 시스템(2)으로부터 캡처된 tar 파일이 베이스 모델(1) 상의 알려진 위치에 복사된다(단계 8). 또한, 스크립트 파일이 제공되는데, 이는 다음 시스템 부팅 시에 실행되어, 그 다음의 순차적인 부팅 동작 동안 구성자 프로그램(43)이 실행되는 것을 배제할 것이다. 일실시예에서 새로운 모델(2)은 변경된 하드웨어 관리 콘솔이 될 수 있고, 스크립트 파일은 hmcConfig 등으로 이름 지어질 수 있다. 전술한 바와 같이 리눅스 운영 체제 환경에서는 유틸리티 chkconfig를 이용하고 스크립트를 /etc/rc.d/ 디렉토리 하에 추가함으로써 스크립트 파일이 생성될 수 있다. 스크립트 파일 내에는 구성자 프로그램(43)인 Kudzu 전에 스크립트가 호출될 수 있도록 하는 시작 우선순위가 특정되어 있다.

<80> 단계 9에서 스크립트 파일 및 다른 구성 파일들을 포함하여, 업데이트되거나 변경된 베이스 모델(1) 이미지가 CD(27) 등에 캡처된다. 업데이트된 새로운 모델(1) 이미지를 가진 이러한 CD는 새로운 모델(2) 시스템의 사용자, 구매자 등에게 제공되어, 자동적인 시스템 구성을 제공할 수 있다.

<81> 단계 10에서 변경된 베이스 모델 이미지가 새로운 모델(2) 상에 인스톨된다. 제공된 스크립트 파일은 새로운 모델(2)의 BIOS로부터 기계 제품 이름을 판독하고, 이 정보를 이용하여 특정한 하드웨어 구성 파일들을 업데이트할 것인지 여부를 결정할 것이다. 구체적으로 단계 11은 새로운 모델 시스템(2)이 업데이트된 이후에 처음으로 재부팅되고 있는 것인지 여부를 결정한다. 즉, 새로운 모델(2)이 처음으로 재부팅될 때, CD(27)로부터 스크립트 파일이 호출되어 기계의 모델 번호를 결정할 것이다. 또한, 새로운 모델(2) 번호가 베이스 모델(1)과 동일하지 않다면(일반적으로 상이할 것임), 스크립트 파일은 새로운 모델(2) 기계의 VPD(vital product date)를

이용하여, <newModel product number>.tar 파일을 찾아볼 것이다. 단계 12에서 새로운 모델(2)을 위한 사용자 소프트웨어를 피하고, 사용자의 상호작용 없이 구성을 개시하기 위하여, 스크립트 파일이 호출될 것이다. 이전에 복구된 모든 필요한 하드웨어 구성 파일이 새로운 모델(2) 시스템과 일치할 것이기 때문에, 사용자의 상호작용은 전혀 필요하지 않다. 그러나 단계 11에서 만일 새로운 모델(2)이 새로운 부품들의 인스톨에 이어 최초 인스톨 이후에 재부팅되고 있거나 새로운 모델 제품 번호가 베이스 모델 제품 번호와 일치한다면, 프로세스는 단계 13으로 진행하여 새로운 모델(2) 사용자가 구성자 프로그램과 상호작용하여 시스템을 수동으로 구성하게 될 것이다. 즉, 단계 11은 자동적인 구성이 일어나도록 하기 위하여 두 가지 이벤트가 일어나길 요구한다. 그것은 시스템을 업데이트한 이후의 첫 번째 시스템 재부팅이어야 하고, 새로운 모델(2)의 VPD에 저장된 제품 번호가 .tar 파일에 저장된 베이스 모델 번호와 상이해야만 한다는 것이다. 그렇지 않다면, 사용자에 의한 수동 구성이 일어날 것이다(단계 13). 두 단계(12, 13) 이후에 본 발명의 프로세스는 새로운 모델 시스템(2)이 새로이 추가된 부품들로 구성되면서 단계 14에서 종료된다.

도면의 간단한 설명

<28> 도 1은 폐쇄 시스템으로서 구성될 수 있고 본 발명의 부팅 기술을 구현할 수 있는 클라이언트 컴퓨터 시스템의 블록도.

<29> 도 2는 본 발명의 바람직한 실시예에 따라 구현될 수 있는 데이터 프로세싱 시스템을 도시하는 도면.

<30> 도 3은 본 발명에 따라 구현될 수 있는 데이터 프로세싱 시스템의 좀 더 상세한 블록도.

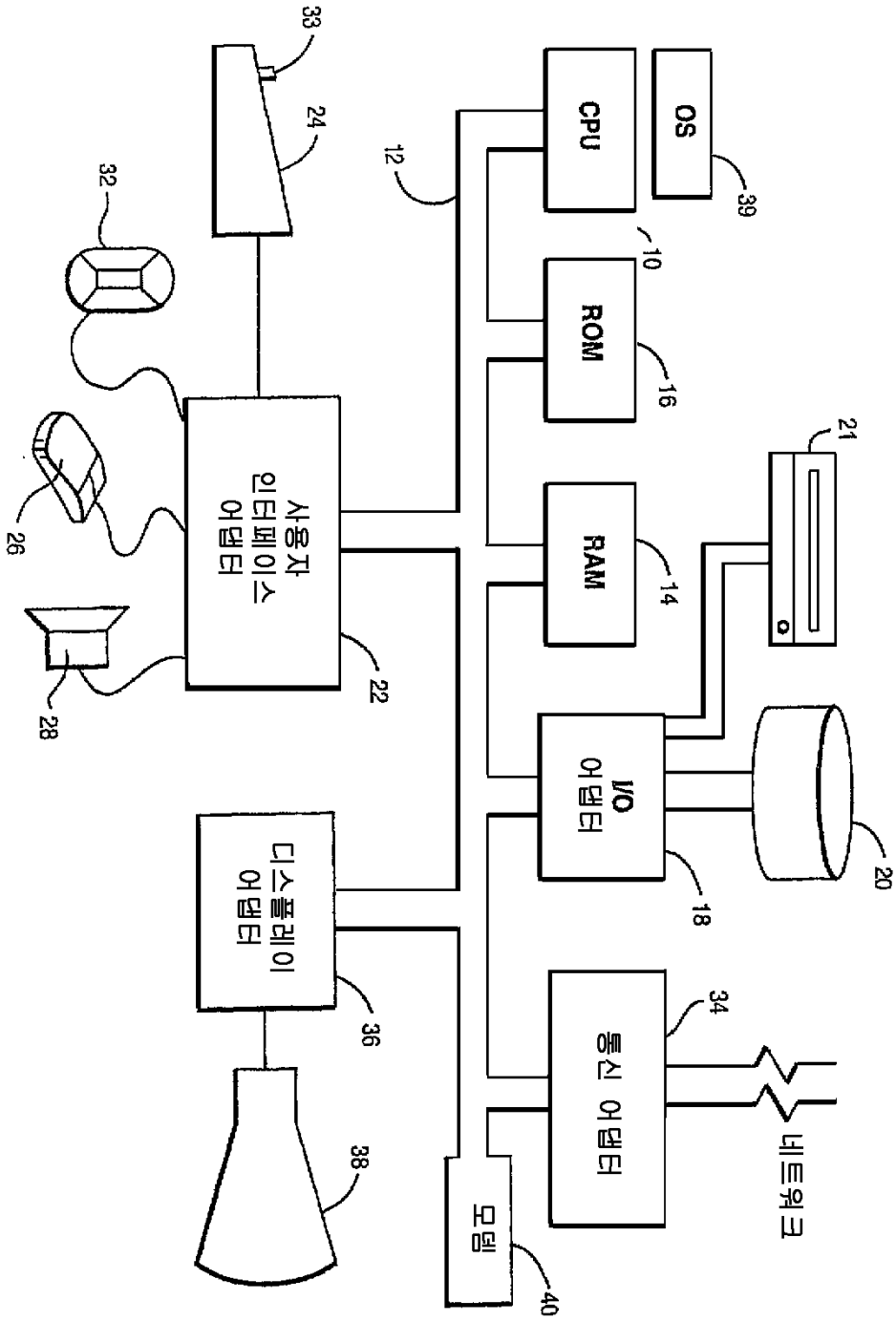
<31> 도 4는 본 발명이 구현될 수 있는 논리적으로 파티션된 전형적인 플랫폼의 블록도.

<32> 도 5는 본 발명의 바람직한 실시예에 따라 사용자의 상호 작용 없이 구성될 수 있는 폐쇄 시스템의 블록도.

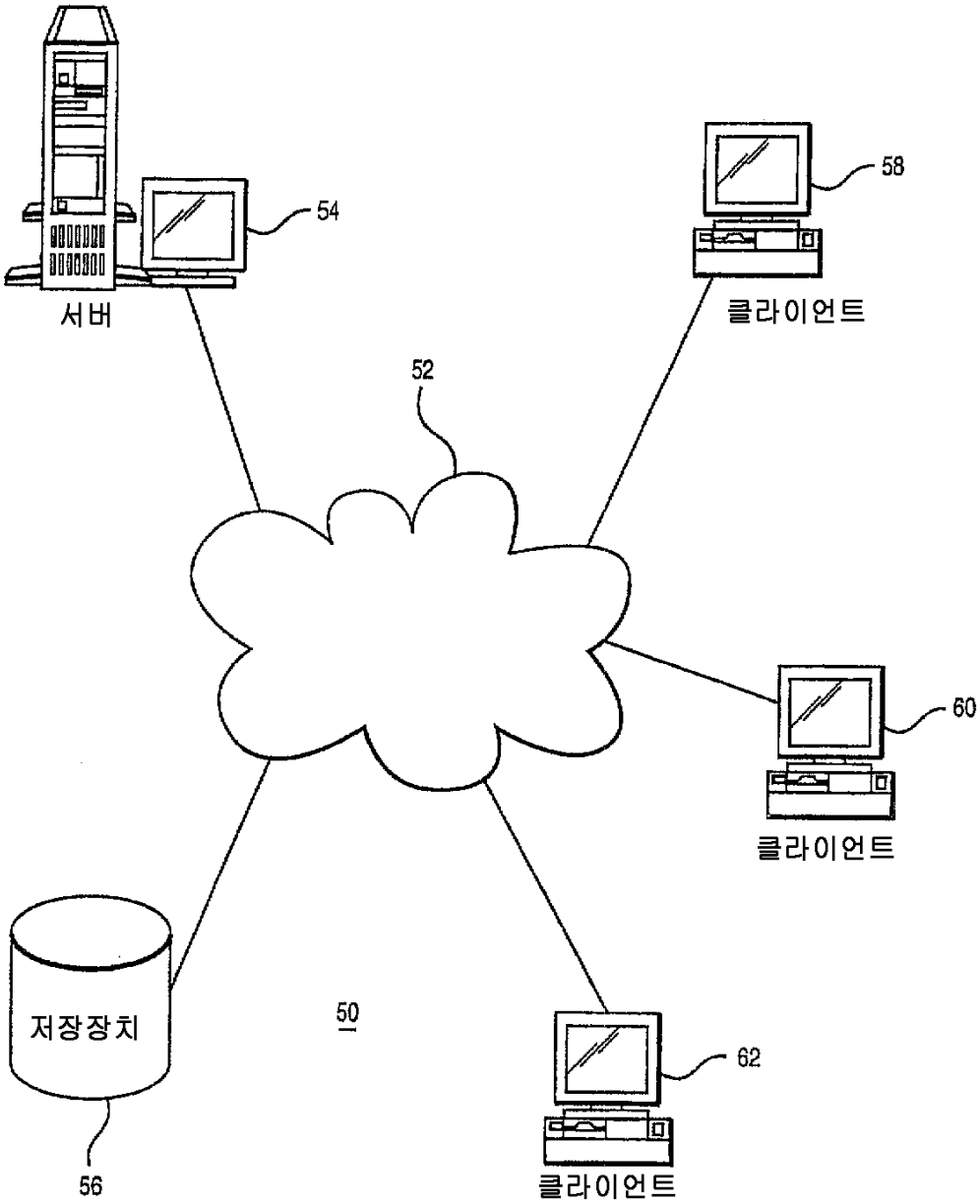
<33> 도 6은 사용자의 상호 작용 없이 폐쇄 시스템의 초기화를 용이하게 하기 위하여 본 발명에 의해 이용되는 단계들을 도시하는 흐름도.

도면

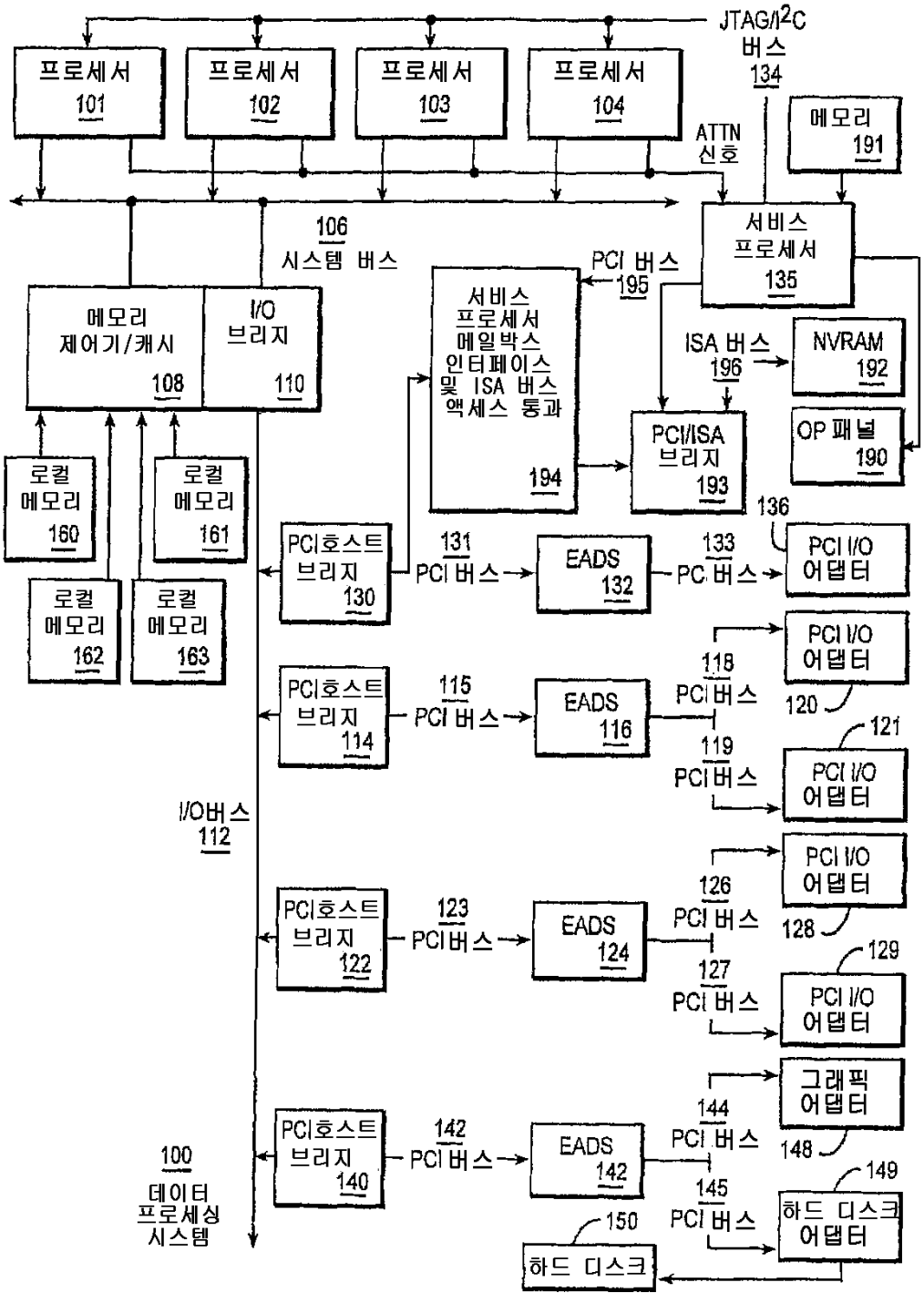
도면1


CPU
OS
39
12
10
ROM
16
RAM
14
21
사용자
인터페이스
어댑터
22
I/O
어댑터
18
20
디스플레이
어댑터
36
38
통신 어댑터
34
네트워크
모뎀
40
33
24
32
26
28

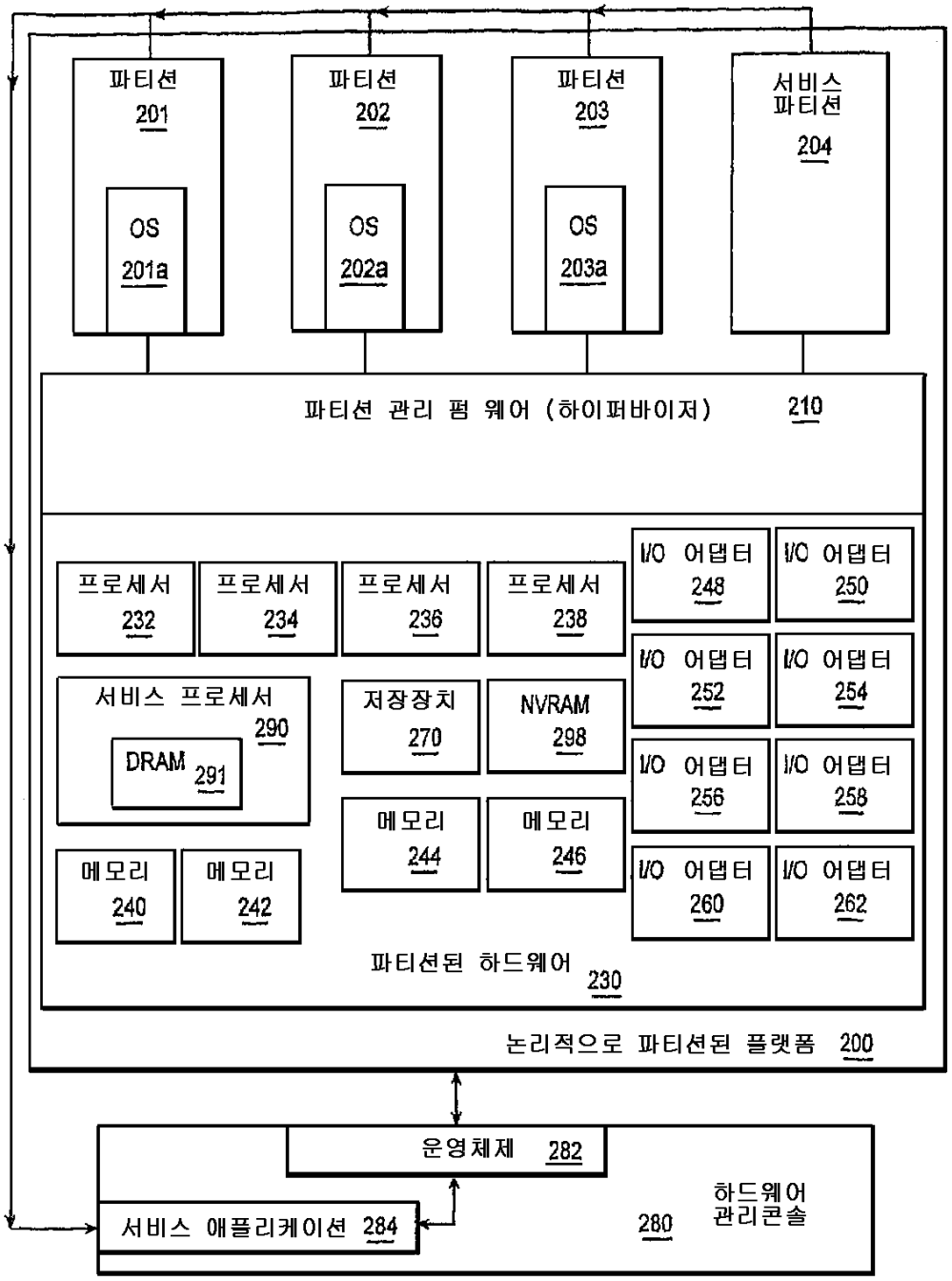
도면2



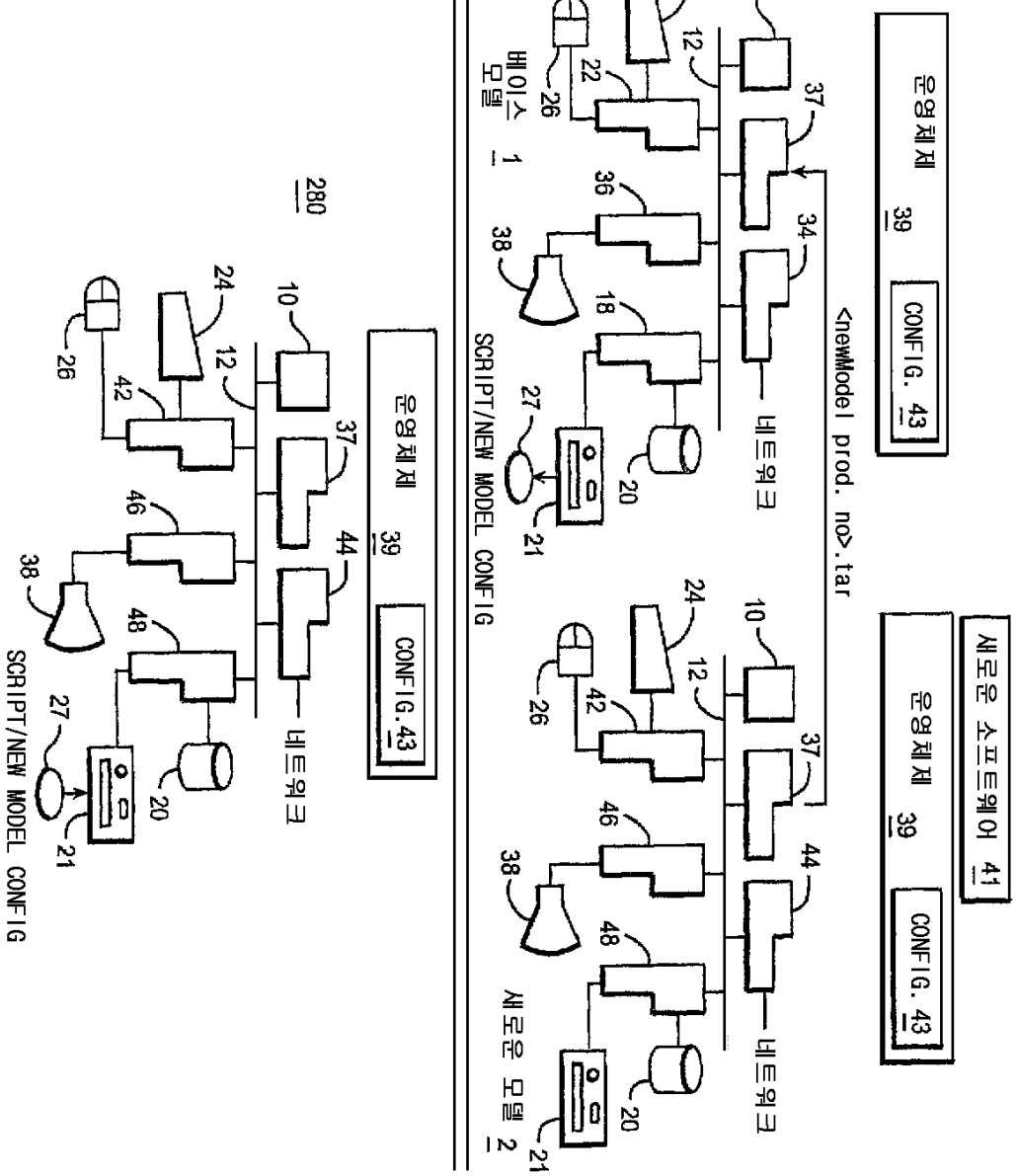
도면3


프로세서 101
프로세서 102
프로세서 103
프로세서 104
JTAG/I2C 버스 134
ATTN 신호
메모리 191
106 시스템 버스
서비스 프로세서 135
PCI 버스 195
메모리 제어기/캐시 108
I/O 브리지 110
서비스 프로세서 메일박스 인터페이스 및 ISA 버스 액세스 통과 194
ISA 버스 196
NVRAM 192
PCI/ISA 브리지 193
OP 패널 190
로컬 메모리 160
로컬 메모리 161
로컬 메모리 162
로컬 메모리 163
136
PCI호스트 브리지 130
131 PCI 버스
EADS 132
133 PCI 버스
PCI I/O 어댑터
118 PCI 버스
120
PCI호스트 브리지 114
115 PCI 버스
EADS 116
119 PCI 버스
121
I/O버스 112
PCI호스트 브리지 122
123 PCI버스
EADS 124
126 PCI 버스
128
127 PCI 버스
129
그래픽 어댑터
144 PCI 버스
148
PCI호스트 브리지 140
142 PCI 버스
EADS 142
145 PCI 버스
149
하드 디스크 어댑터
150
하드 디스크
100 데이터 프로세싱 시스템

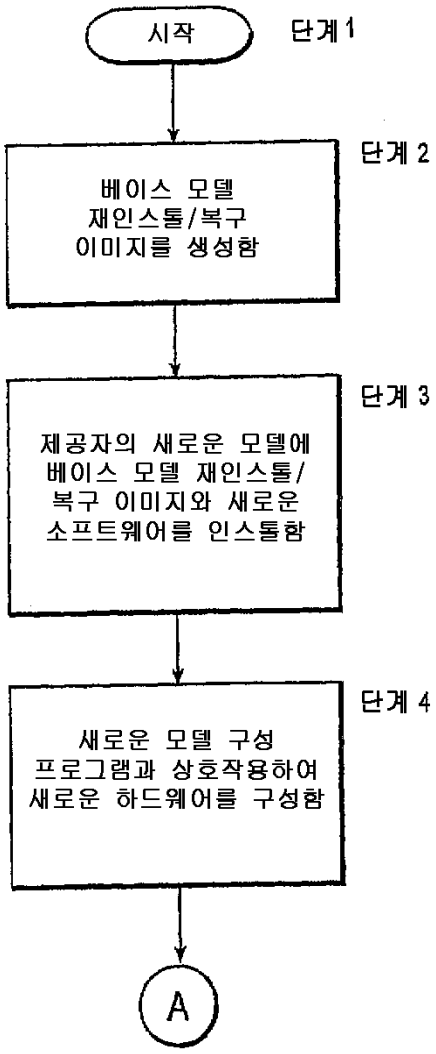
도면4


파티션 201
OS 201a
파티션 202
OS 202a
파티션 203
OS 203a
서비스 파티션 204
파티션 관리 펌 웨어 (하이퍼바이저) 210
프로세서 232
프로세서 234
프로세서 236
프로세서 238
I/O 어댑터 248
I/O 어댑터 250
I/O 어댑터 252
I/O 어댑터 254
I/O 어댑터 256
I/O 어댑터 258
I/O 어댑터 260
I/O 어댑터 262
서비스 프로세서 290
DRAM 291
저장장치 270
NVRAM 298
메모리 244
메모리 246
메모리 240
메모리 242
파티션된 하드웨어 230
논리적으로 파티션된 플랫폼 200
운영체제 282
서비스 애플리케이션 284
하드웨어 관리콘솔
280

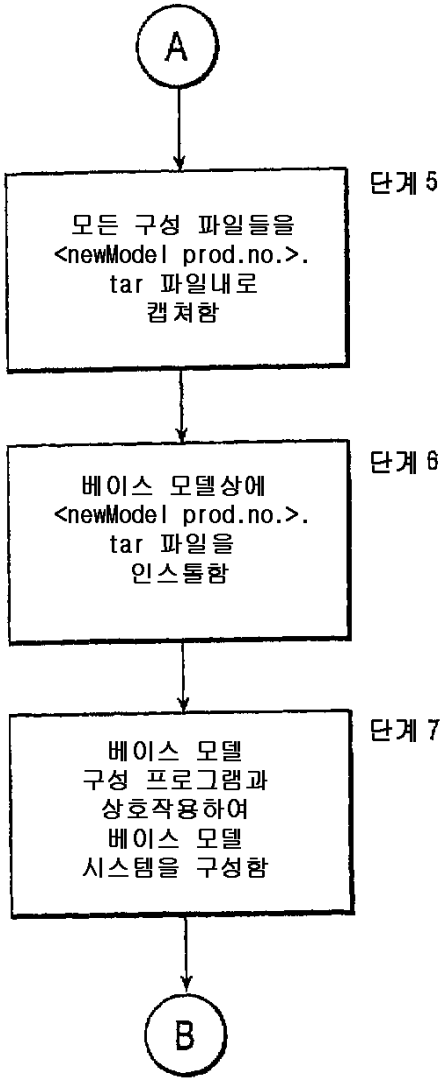
도면5


운영 체제 39
CONFIG. 43
새로운 소프트웨어 41
운영 체제 39
CONFIG. 43
<newModel prod. no>.tar
베이스 모델 1
네트워크
SCRIPT/NEW MODEL CONFIG
새로운 모델 2
네트워크
280
운영 체제 39
CONFIG.43
네트워크
SCRIPT/NEW MODEL CONFIG

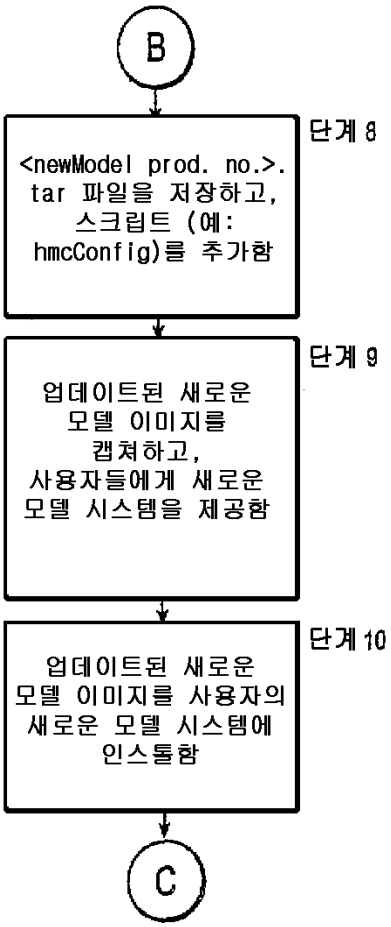
도면6a


시작
단계 1
베이스 모델
재인스톨/복구
이미지를 생성함
단계 2
제공자의 새로운 모델에
베이스 모델 재인스톨/
복구 이미지와 새로운
소프트웨어를 인스톨함
단계 3
새로운 모델 구성
프로그램과 상호작용하여
새로운 하드웨어를 구성함
단계 4
A

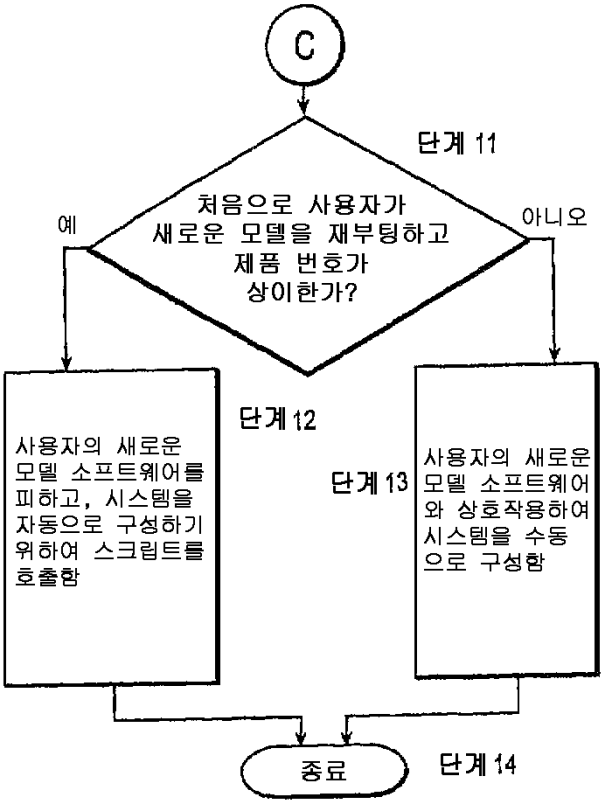
도면6b


A
단계 5
모든 구성 파일들을
<newModel prod.no.>.
tar 파일내로
캡쳐함
단계 6
베이스 모델상에
<newModel prod.no.>.
tar 파일을
인스톨함
단계 7
베이스 모델
구성 프로그램과
상호작용하여
베이스 모델
시스템을 구성함
B

도면6c


B
<newModel prod. no.>.
tar 파일을 저장하고,
스크립트 (예:
hmcConfig)를 추가함
단계 8
업데이트된 새로운
모델 이미지를
캡쳐하고,
사용자들에게 새로운
모델 시스템을 제공함
단계 9
업데이트된 새로운
모델 이미지를 사용자의
새로운 모델 시스템에
인스톨함
단계 10
C

도면6d


C
단계 11
처음으로 사용자가 새로운 모델을 재부팅하고 제품 번호가 상이한가?
예
아니오
사용자의 새로운 모델 소프트웨어를 피하고, 시스템을 자동으로 구성하기 위하여 스크립트를 호출함
단계 12
단계 13
사용자의 새로운 모델 소프트웨어와 상호작용하여 시스템을 수동으로 구성함
종료
단계 14