(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2024년10월30일
(11) 등록번호 10-2723681
(24) 등록일자 2024년10월25일

(51) 국제특허분류(Int. Cl.)
G06F 9/50 (2018.01) *G06F 21/57* (2013.01)
G06F 3/06 (2006.01) *H04L 41/00* (2022.01)
H04L 65/40 (2022.01) *H04L 9/40* (2022.01)
(52) CPC특허분류
G06F 9/5072 (2013.01)
G06F 21/575 (2013.01)
(21) 출원번호 10-2020-7019631
(22) 출원일자(국제) 2018년12월07일
심사청구일자 2021년12월06일
(85) 번역문제출일자 2020년07월07일
(65) 공개번호 10-2020-0106036
(43) 공개일자 2020년09월10일
(86) 국제출원번호 PCT/US2018/064624
(87) 국제공개번호 WO 2019/113553
국제공개일자 2019년06월13일
(30) 우선권주장
62/596,355 2017년12월08일 미국(US)
62/694,846 2018년07월06일 미국(US)
(56) 선행기술조사문헌
미국공개특허 제2010-0115490호(2010.05.06.) 1부.*
미국공개특허 제2017-0317881호(2017.11.02.) 1부.*
미국공개특허 제2013-0297772호(2013.11.07.) 1부.*
*는 심사관에 의하여 인용된 문헌

(73) 특허권자
넷-썬더, 엘엘씨
미국, 일리노이 60090, 휠링, 마켜트 드라이브 111이
(72) 발명자
슈미트 파커 존
미국, 일리노이 60090, 휠링, 마켜트 드라이브 111이
리처드슨 션 마이클
미국, 일리노이 60090, 휠링, 마켜트 드라이브 111이
(뒷면에 계속)
(74) 대리인
강명구, 박윤원

전체 청구항 수 : 총 22 항 심사관 : 김종기

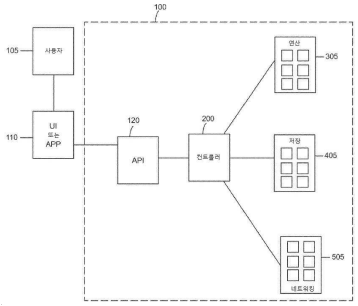
(54) 발명의 명칭 **자동 배포되는 정보 기술(IT) 시스템 및 방법**

(57) 요 약

컨트롤러가 복수의 시스템 규칙, 컴퓨터 시스템에 대한 시스템 상태, 및 복수의 템플릿에 기초하여 컴퓨터 시스템을 위한 물리적 인프라구조를 자동적으로 관리하는, 시스템, 방법, 및 장치가 본 명세서에 개시된다. 컴퓨터, 스토리지, 및/또는 네트워킹 리소스와 같은 리소스를 컴퓨터 시스템에 자동적으로 추가하기 위한 기술이 설명된

(뒷면에 계속)

대 표 도 - 도1



다. 이러한 리소스에 애플리케이션 및 서비스를 자동적으로 배치하기 위한 기술도 설명된다. 이들 기술은 일괄 확장 가능한 사설 클라우드로서 기능할 수 있는 확장 가능한 컴퓨터 시스템을 제공한다.

(52) CPC특허분류
G06F 3/0604 (2013.01)
G06F 3/064 (2013.01)
G06F 3/0659 (2013.01)
G06F 3/067 (2013.01)
H04L 41/12 (2022.05)
H04L 63/10 (2023.05)
H04L 67/1095 (2022.05)
H04L 67/1097 (2022.05)

(72) 발명자
시멜 닐 벤자민
미국, 일리노이 60090, 휠링, 마켜트 드라이브 111이

스프리 카메론 타일러
미국, 일리노이 60090, 휠링, 마켜트 드라이브 111이

명 세 서

청구범위

청구항 1

장치로서,

물리 호스트를 포함하는 컴퓨터 시스템에 사용되는 컨트롤러를 포함하며, 상기 컨트롤러는 복수의 시스템 규칙, 상기 컴퓨터 시스템에 대한 시스템 상태, 및 복수의 템플릿에 기초하여 상기 컴퓨터 시스템을 위한 물리적 인프라구조(physical infrastructure)의 자동화된 관리를 제공하도록 구성되며,

상기 컨트롤러는, 상기 컴퓨터 시스템으로의 새로운 리소스의 연결에 응답하여, (1) 상기 새로운 리소스가 상기 컴퓨터 시스템에 연결됨을 인식하는 것, (2) 상기 연결된 새로운 리소스에 관한 정보를 결정하는 것, (3) 결정된 정보를 상기 컴퓨터 시스템에 대한 시스템 상태에 추가하는 것, (4) 상기 결정된 정보에 기초하여 상기 템플릿 중 하나를 선택하는 것, (5) 상기 선택된 템플릿으로부터 도출된 이미지를 상기 컴퓨터 시스템에 로딩하는 것 - 상기 이미지는 파일 시스템을 포함함 -, 및 (6) 상기 로딩된 이미지의 상기 파일 시스템을 사용하여 부팅하도록 상기 새로운 리소스에 지시하는 것에 의해, 상기 새로운 리소스에 대한 물리적 인프라구조를 상기 컴퓨터 시스템에 자동으로 추가하도록 더 구성되는, 장치.

청구항 2

제1항에 있어서, 상기 자동화된 관리는 상기 시스템 규칙, 상기 시스템 상태, 및 상기 템플릿에 기초하여 상기 컴퓨터 시스템을 위한 물리적 인프라구조의 자동화된 구성을 포함하는, 장치.

청구항 3

제1항 또는 제2항에 있어서, 상기 템플릿은 복수의 상이한 유형의 물리적 인프라구조에 사용되는 복수의 템플릿을 포함하고, 시스템 규칙은 주어진 유형의 물리적 인프라구조를 관리할 때 어느 템플릿이 사용되는지를 제어하는, 장치.

청구항 4

제1항 또는 제2항에 있어서, 상기 컴퓨터 시스템은 복수의 물리 호스트 및 복수의 가상 호스트를 포함하고, 상기 컨트롤러는 상기 물리 호스트 및 상기 가상 호스트에 애플리케이션을 상호 교환 가능하게 배치(deploy)하도록 더 구성되는, 장치.

청구항 5

제1항 또는 제2항에 있어서, 상기 시스템 규칙은 상기 컴퓨터 시스템의 자체 조립(self-assembly)을 위한 글로벌 시스템 규칙을 포함하고, 상기 글로벌 시스템 규칙은 (1) 컴퓨터 시스템에 리소스를 추가하는 것에 관한 완료를 위한 복수의 IT 작업의 사양, (2) 컴퓨터 시스템에 리소스를 추가하기 위해 필요한 하드웨어의 업데이트 가능한 리스트, 및 (3) 컴퓨터 시스템에 리소스를 추가하는 것에 관한 완료를 위한 동작 및 작업의 정렬된 리스트의 사양 중 적어도 하나를 포함하는, 장치.

청구항 6

제1항 또는 제2항에 있어서, 상기 템플릿은 (1) 리소스, (2) 리소스에 로딩된 애플리케이션, 및 (3) 상기 컴퓨터 시스템 상의 리소스에 로딩된 서비스 중 적어도 하나를 생성, 구성, 또는 배치하는 데 사용되는 확립된 정보 세트를 포함하는, 장치.

청구항 7

제6항에 있어서, 상기 템플릿은 베어 메탈 템플릿을 포함하는, 장치.

청구항 8

제6항에 있어서, 상기 복수의 템플릿 각각은 기본 운영 체제 파일 시스템에 대한 베이스 이미지를 포함하는, 장치.

청구항 9

제1항 또는 제2항에 있어서, 상기 컨트롤러는 상기 시스템 규칙에 따라 상기 템플릿을 사용하여 상기 컴퓨터 시스템에 대한 인프라구조를 구축하고 이에 따라 상기 시스템 상태를 업데이트하도록 추가로 구성되며, 컨트롤러는 시스템 규칙, 시스템 상태, 및 템플릿에 기초하여 컴퓨트 자원, 저장 자원, 및 네트워킹 자원 중 적어도 하나를 자동으로 추가하도록 더 구성되는, 장치.

청구항 10

제9항에 있어서, 상기 컨트롤러는,

시스템 규칙을 판독하여 상기 컴퓨터 시스템의 원하는 상태를 달성하기 위해 완료할 작업의 리스트를 개발하는 것;

상기 컴퓨터 시스템의 이용 가능한 리소스에 기초하여 상기 판독 시스템 규칙을 이행하기 위한 명령어를 발행하는 것;

상기 시스템 상태를 사용하여 상기 리스트로부터 작업을 수행하기 위해 상기 컴퓨터 시스템의 사용 가능한 리소스를 찾는 것; 및

상기 리스트로부터 작업에 필요한 리소스가 이용 가능한 것으로 판명된 경우, 상기 이용 가능한 리소스를 사용하여 상기 작업을 실행하는 것을 행하도록 더 구성되는, 장치.

청구항 11

제1항 또는 제2항에 있어서, 상기 컨트롤러는 상기 시스템 규칙, 시스템 상태, 및 템플릿에 기초하여 연산 리소스를 자동으로 추가하도록 더 구성되는, 장치.

청구항 12

제11항에 있어서, 상기 연산 리소스는 베어 메탈 연산 리소스를 포함하는, 장치.

청구항 13

제1항 또는 제2항에 있어서, 상기 컨트롤러는 상기 시스템 규칙, 시스템 상태, 및 템플릿에 기초하여 상기 컴퓨터 시스템에 스토리지 리소스를 자동으로 추가하도록 더 구성되는, 장치.

청구항 14

제13항에 있어서, 상기 스토리지 리소스는 베어 메탈 스토리지 리소스를 포함하는, 장치.

청구항 15

제1항 또는 제2항에 있어서, 상기 컨트롤러는 상기 시스템 규칙, 시스템 상태, 및 템플릿에 기초하여 상기 컴퓨터 시스템에 네트워킹 리소스를 자동으로 추가하도록 더 구성되는, 장치.

청구항 16

제15항에 있어서, 상기 네트워킹 리소스는 베어 메탈 네트워킹 리소스를 포함하는, 장치.

청구항 17

제1항 또는 제2항에 있어서, 상기 컨트롤러는 (1) 상기 물리 호스트를 위한 바이오스를 구성하는 것, (2) 상기 물리 호스트를 위한 부트 옵션을 구성하는 것, (3) 서버를 스토리지 리소스로 포인팅하는 것, 및 (4) 상기 물리 호스트를 부팅하는 것에 의해 인터페이스를 통해 상기 물리 호스트를 관리하도록 더 구성되는, 장치.

청구항 18

삭제

청구항 19

제1항 또는 제2항에 있어서, 상기 컨트롤러는 대역외(out-of-band) 관리에 기초하여 상기 컴퓨터 시스템에 대한 물리적 인프라구조를 자동으로 관리하도록 구성되는, 장치.

청구항 20

제1항 또는 제2항에 있어서, 상기 컨트롤러는 상기 시스템 규칙, 상기 시스템 상태, 및 상기 템플릿에 기초하여 상기 컴퓨터 시스템의 리소스에 애플리케이션 또는 서비스를 자동으로 배치하도록 더 구성되는, 장치.

청구항 21

제20항에 있어서, 상기 컨트롤러는 대역외 관리 연결 또는 대역내 관리 연결을 사용하여, 상기 컴퓨터 시스템의 리소스에 의한 실행을 위한 상기 애플리케이션 또는 서비스를 배치하기 위해 상기 템플릿 중 하나로부터 도출된 애플리케이션 이미지를 부팅하도록 상기 컴퓨터 시스템의 리소스에 지시하도록 구성되는, 장치.

청구항 22

제21항에 있어서, 상기 시스템 규칙은 상기 리소스가 상기 템플릿 중 하나로부터 도출된 이미지로부터 부팅되고, 그 후에 상기 애플리케이션 또는 서비스가 상기 템플릿 중 다른 것으로부터 도출된 이미지로부터 부팅되도록 부팅하기 위한 순서를 지정하는, 장치.

청구항 23

방법으로서,

컨트롤러가 복수의 시스템 규칙, 컴퓨터 시스템에 대한 시스템 상태, 및 복수의 템플릿에 액세스하는 단계; 및

상기 컨트롤러가 상기 액세스된 시스템 규칙, 시스템 상태, 및 템플릿에 기초하여 물리 호스트를 포함하는 컴퓨터 시스템을 위한 물리적 인프라구조를 자동으로 관리하는 단계를 포함하며,

상기 자동으로 관리하는 단계는, 상기 컨트롤러가, 상기 컴퓨터 시스템으로의 새로운 리소스의 연결에 응답하여, (1) 상기 새로운 리소스가 상기 컴퓨터 시스템에 연결됨을 인식하는 것, (2) 상기 연결된 새로운 리소스에 관한 정보를 결정하는 것, (3) 결정된 정보를 상기 컴퓨터 시스템에 대한 시스템 상태에 추가하는 것, (4) 상기 결정된 정보에 기초하여 상기 템플릿 중 하나를 선택하는 것, (5) 상기 선택된 템플릿으로부터 도출된 이미지를 상기 컴퓨터 시스템에 로딩하는 것 - 상기 이미지는 파일 시스템을 포함함 -, 및 (6) 상기 로딩된 이미지의 상기 파일 시스템을 사용하여 부팅하도록 상기 새로운 리소스에 지시하는 것에 의해, 상기 새로운 리소스에 대한 물리적 인프라구조를 상기 컴퓨터 시스템에 자동으로 추가하는 단계를 포함하는, 방법.

청구항 24

삭제

청구항 25

삭제

청구항 26

삭제

청구항 27

삭제

청구항 28

삭제

청구항 29

삭제

청구항 30

삭제

청구항 31

삭제

청구항 32

삭제

청구항 33

삭제

청구항 34

삭제

청구항 35

삭제

청구항 36

삭제

청구항 37

삭제

청구항 38

삭제

청구항 39

삭제

청구항 40

삭제

청구항 41

삭제

청구항 42

삭제

청구항 43

삭제

청구항 44

삭제

청구항 45

삭제

청구항 46

삭제

청구항 47

삭제

청구항 48

삭제

청구항 49

삭제

청구항 50

삭제

청구항 51

삭제

청구항 52

삭제

청구항 53

삭제

청구항 54

삭제

청구항 55

삭제

청구항 56

삭제

청구항 57

삭제

청구항 58

삭제

청구항 59

삭제

청구항 60

삭제

청구항 61

삭제

청구항 62

삭제

청구항 63

삭제

청구항 64

삭제

청구항 65

삭제

청구항 66

삭제

청구항 67

삭제

청구항 68

삭제

청구항 69

삭제

청구항 70

삭제

청구항 71

삭제

청구항 72

삭제

청구항 73

삭제

청구항 74

삭제

청구항 75

삭제

청구항 76

삭제

청구항 77

삭제

청구항 78

삭제

청구항 79

삭제

청구항 80

삭제

청구항 81

삭제

청구항 82

삭제

청구항 83

삭제

청구항 84

삭제

청구항 85

삭제

청구항 86

삭제

청구항 87

삭제

청구항 88

삭제

청구항 89

삭제

청구항 90

삭제

청구항 91

삭제

청구항 92

삭제

청구항 93

삭제

청구항 94

삭제

청구항 95

삭제

청구항 96

삭제

청구항 97

삭제

청구항 98

삭제

청구항 99

삭제

청구항 100

삭제

청구항 101

삭제

청구항 102

삭제

청구항 103

삭제

청구항 104

삭제

청구항 105

삭제

청구항 106

삭제

청구항 107

삭제

청구항 108

삭제

청구항 109

삭제

청구항 110

삭제

청구항 111

삭제

청구항 112

삭제

청구항 113

삭제

청구항 114

삭제

청구항 115

삭제

청구항 116

삭제

청구항 117

삭제

청구항 118

삭제

청구항 119

삭제

청구항 120

삭제

청구항 121

삭제

청구항 122

삭제

청구항 123

삭제

청구항 124

삭제

청구항 125

삭제

청구항 126

삭제

청구항 127

삭제

청구항 128

삭제

청구항 129

삭제

청구항 130

삭제

청구항 131

삭제

청구항 132

삭제

청구항 133

삭제

청구항 134

삭제

청구항 135

삭제

청구항 136

삭제

청구항 137

삭제

청구항 138

삭제

청구항 139

삭제

청구항 140

삭제

청구항 141

삭제

청구항 142

삭제

청구항 143

삭제

청구항 144

삭제

청구항 145

삭제

청구항 146

삭제

청구항 147

삭제

청구항 148

삭제

청구항 149

삭제

청구항 150

삭제

청구항 151

삭제

청구항 152

삭제

청구항 153

삭제

청구항 154

삭제

청구항 155

삭제

청구항 156

삭제

청구항 157

삭제

청구항 158

삭제

청구항 159

삭제

청구항 160

삭제

청구항 161

삭제

청구항 162

삭제

청구항 163

삭제

청구항 164

삭제

청구항 165

삭제

청구항 166

삭제

청구항 167

삭제

청구항 168

삭제

청구항 169

삭제

청구항 170

삭제

청구항 171

삭제

청구항 172

삭제

청구항 173

삭제

청구항 174

삭제

청구항 175

삭제

청구항 176

삭제

청구항 177

삭제

발명의 설명

기 술 분 야

[0001] [관련 특허출원에 대한 상호 참조 및 우선권 주장]

[0002] 본 특허출원은 "Automatically Deployed Information Technology(IT) System and Method"라는 명칭으로 2017년 12월 8일에 출원된 미국 가특허출원 제62/596,355호에 대한 우선권을 주장하며, 그 전체 개시 내용이 참조로 본 명세서에 포함된다.

[0003] 본 특허출원은 또한 "Automatically Deployed Information Technology(IT) System and Method"라는 명칭으로 2018년 7월 6일에 출원된 미국 가특허출원 제62/694,846호에 대한 우선권을 주장하며, 그 전체 개시 내용이 참조로 본 명세서에 포함된다.

배 경 기 술

[0004] 컴퓨팅의 수요, 사용 및 필요성이 지난 수십년 동안 급증해왔다. 이에 따라, 더 큰 스토리지, 속도, 컴퓨팅 능력, 애플리케이션, 접근성에 대한 수요가 다양한 유형 및 크기의 엔티티에 툴을 제공하는 컴퓨팅 분야를 급속하게 변화시켜 왔다. 그 결과, 공용 가상 컴퓨팅 및 클라우드 컴퓨팅 시스템의 사용은 다수의 사용자 및 사용자의 유형에 더 큰 컴퓨팅 리소스를 제공하도록 개발되어 왔다. 이 기하 급수적인 성장은 계속될 것으로 예상된다. 동시에, 더 큰 실패와 보안 위험으로 인해, 인프라구조 설정, 관리, 변경 관리, 업데이트가 더욱 복잡하고 비용이 들게 되었다. 시간 경과에 따라 시스템의 확장성(scalability), 또는 성장은 정보 기술 분야에서도 중요한 도전 과제가 되었다.

[0005] 성능 및 보안과 관련된 대부분의 IT 시스템의 문제는 진단 및 해결이 어려울 수 있다. 시스템의 설정, 구성 및 배치에 허용되는 시간과 리소스에 대한 제약으로 인해 오류가 발생하여 향후 IT 문제를 초래할 수 있다. 시간 경과에 따라 여러 다른 관리자가 사용자, 애플리케이션, 서비스, 보안, 소프트웨어 및 하드웨어를 포함한 IT 시스템의 변경, 패치 또는 업데이트에 관여할 수 있다. 종종 구성 및 변경에 대한 문서 및 기록이 부적절하거나 손실될 수 있어 나중에 특정 시스템이 어떻게 구성되고 작동하는지 이해하기가 어렵다. 이는 향후 변경 또는 문제 해결을 어렵게 할 수 있다. IT 구성 및 설정은 문제 또는 장애가 발생할 때에 복구 및 재현하기가 어려울 수 있다. 또한, 시스템 관리자는 실수, 예를 들어 부정확한 커맨드 또는 다른 실수를 쉽게 만들 수 있어, 종국에는 컴퓨터와 웹 데이터베이스 및 서비스를 중단시킬 수 있다. 또한, 보안 위반의 위험 증가가 아주 흔하지만, 보안 위반을 피하기 위한 변경, 업데이트, 패치는 바람직하지 않은 다운타임을 유발할 수 있다.

[0006] 중요한 인프라구조가 제자리에 있고 작동중이고 운영하고 있으면, 비용 또는 위험이 종종 시스템 변경의 이점을 능가하는 것처럼 보일 수 있다. 운영중인 IT 시스템 또는 환경을 변경하는 데 관련된 문제는 이러한 시스템에 의존하는 사용자 또는 엔터티에 상당한, 때로는 파멸의 문제를 야기할 수 있다. 적어도, 변경 관리 중에 발생하는 장애 또는 문제를 해결하고 수리하는 데 걸리는 시간량은 상당한 시간, 인력 및 비용의 자원이 필요할 수 있다. 실제 환경을 변경할 때에 잠재적으로 발생되는 기술적 문제는 캐스케이딩 현상(cascading effect)이 있을

수 있고, 변경을 취소하여 단독으로 해결되지 않을 수 있다. 이러한 많은 문제는 변경 관리 중 장애가 있는 경우에 시스템을 신속하게 재구축할 수 없는 원인이 된다.

[0007] 또한, IT 시스템 내의 베어 메탈 클라우드 노드 또는 리소스는 보안 문제에 취약하거나, 악의적인 사용자에 의해 침입받거나, 액세스될 수 있다. 해커, 공격자 또는 악의적인 사용자는 해당 노드 또는 리소스를 피벗 오프(pivot off)하여 노드에 결합된 IT 시스템 또는 네트워크의 임의의 다른 부분에 액세스하거나 해킹할 수 있다. IT 시스템의 베어 메탈 클라우드 노드 또는 컨트롤러는 또한 시스템을 보안 위협에 노출시키거나 달리 시스템을 침입할 수 있는 애플리케이션 네트워크에 연결된 리소스를 통해 취약 가능할 수 있다. 본 명세서에 개시된 다양한 예시적인 실시형태에 따르면, IT 시스템은 외부 네트워크에 연결되어 있는지 여부에 관계없이 인터넷과 인터페이스하거나 또는 애플리케이션 네트워크로부터 인터페이스하는 베어 메탈 클라우드 노드 또는 리소스의 보안을 개선하도록 구성될 수 있다.

발명의 내용

과제의 해결 수단

[0008] 예시적인 실시형태에 따르면, IT 시스템은 베어 메탈 클라우드 노드 또는 물리 리소스를 포함한다. 베어 메탈 클라우드 노드 또는 물리 리소스가 턴온, 설정, 관리 또는 사용될 때, 다른 사람이나 고객이 사용 중일 수 있는 노드를 구비한 네트워크에 연결될 수 있는 경우, 대역내(in band) 관리가 컨트롤러로부터 생략, 전환 가능, 연결 해제 가능 또는 필터링될 수 있다.

[0009] 또한, 시스템 내의 애플리케이션 또는 애플리케이션 네트워크는 애플리케이션 네트워크가 컨트롤러에 결합되는 리소스(들)에 의해 컨트롤러로부터 연결 해제, 연결 해제 가능, 전환 가능, 또는 필터링될 수 있다.

[0010] 가상 머신 또는 하이퍼바이저를 구성하는 물리 리소스는 또한 하이퍼바이저가 공유 리소스인 다른 하이퍼바이저에 피벗하는 데 사용될 수 있는 경우에 보안 문제에 취약하거나, 악의적인 사용자에 의해 침입받거나 액세스될 수도 있다. 공격자는 가상 머신으로부터 탈피할 수 있으며 컨트롤러를 통해 경영 및/또는 관리 시스템에 네트워크 액세스할 수 있다. 본 명세서에 개시된 다양한 예시적인 실시형태에 따르면, IT 시스템은 클라우드 플랫폼 상의 가상 리소스를 포함하는 하나 이상의 물리 리소스가 대역내 관리 연결을 통해 컨트롤러에 연결 해제, 연결 해제 가능, 필터링, 필터링 가능 또는 연결되지 않을 수 있는 경우에 보안을 개선하도록 구성될 수 있다.

[0011] 예시적인 실시형태에 따르면, IT 시스템의 물리 리소스는 컨트롤러와 물리 리소스 사이의 대역내 관리 연결이 리소스로부터 생략, 연결 해제, 연결 해제 가능 또는 필터링/필터링 가능할 수 있는 경우에 하나 이상의 가상 머신 또는 하이퍼바이저를 포함할 수 있다.

도면의 간단한 설명

[0012] 도 1은 예시적인 실시형태에 따른 시스템의 개략도이다.

도 2a는 도 1의 시스템을 위한 예시적인 컨트롤러의 개략도이다.

도 2b는 예시적인 스토리지 확장 규칙 세트의 동작의 예시적인 흐름을 도시한다.

도 2c 및 도 2d는 도 2b의 단계 210.1 및 210.2를 수행하기 위한 대안적인 실시예를 도시한다.

도 2e는 예시적인 템플릿을 나타낸다.

도 2f는 템플릿 처리에 대한 컨트롤러 로직의 예시적인 프로세스 흐름을 나타낸다.

도 2g 및 도 2h는 도 2f의 단계 205.11, 205.12 및 205.13에 대한 예시적인 프로세스 흐름을 나타낸다.

도 2i는 다른 예시적인 템플릿을 나타낸다.

도 2j는 템플릿 처리에 대한 컨트롤러 로직의 다른 예시적인 프로세스 흐름을 도시한다.

도 2k는 서비스 의존성을 관리하기 위한 예시적인 프로세스 흐름을 나타낸다.

도 2l은 예시적인 실시형태에 따른 템플릿으로부터 도출된 예시적인 이미지의 개략도이다.

도 2m은 예시적인 시스템 규칙 세트를 도시한다.

도 2n은 도 2m의 시스템 규칙을 처리하는 컨트롤러 로직의 예시적인 프로세스 흐름을 도시한다.

도 2o는 파일시스템 블롭(blob) 또는 파일의 다른 그룹으로부터 스토리지 리소스를 구성하기 위한 예시적인 프로세스 흐름을 도시한다.

도 3a는 연산 리소스가 추가되는 도 2a의 컨트롤러의 개략도이다.

도 3b는 예시적인 실시형태에 따른 템플릿으로부터 도출된 예시적인 이미지의 개략도이다.

도 3c는 연산 리소스, 스토리지 리소스, 및/또는 네트워킹 리소스와 같은 리소스를 시스템에 추가하기 위한 예시적인 프로세스 흐름을 도시한다.

도 4a는 스토리지 리소스가 추가되는 도 2a의 컨트롤러의 개략도이다.

도 4b는 예시적인 실시형태에 따른 템플릿으로부터 도출된 예시적인 이미지의 개략도이다.

도 5a는 JBOD 및 스토리지 리소스가 추가되는 도 2a의 컨트롤러의 개략도이다.

도 5b는 스토리지 리소스 및 스토리지 리소스를 위한 집적 부착된 스토리지를 시스템에 추가하기 위한 예시적인 프로세스 흐름을 도시한다.

도 6a는 네트워킹 리소스가 추가되는 도 2a의 컨트롤러의 개략도이다.

도 6b는 예시적인 실시형태에 따른 템플릿으로부터 도출된 예시적인 이미지의 개략도이다.

도 7a는 예시적인 물리 배치(physical deployment)에서의 예시적인 실시형태에 따른 시스템의 개략도이다.

도 7b는 IT 시스템에 대한 리소스의 추가를 위한 예시적인 처리를 도시한다.

도 7c 및 도 7d는 다수의 연산 리소스, 다수의 서버, 다수의 가상 머신, 및/또는 다수의 사이트에 대한 애플리케이션의 배치에 대한 예시적인 프로세스 흐름을 나타낸다.

도 8a는 예시적인 배치에서의 예시적인 실시형태에 따른 시스템의 개략도이다.

도 8b는 단일의 노드 시스템으로부터 다수의 노드 시스템으로 확장하기 위한 예시적인 프로세스 흐름을 나타낸다.

도 8c는 스토리지 리소스의 새로운 물리 스토리지 리소스로의 마이그레이션을 위한 예시적인 프로세스 흐름을 도시한다.

도 8d는 멀티-테넌트 시스템의 단일 노드에서 가상 시스템, 컨테이너, 및/또는 프로세스를 연산 및 스토리지를 위한 별도의 하드웨어를 가질 수 있는 멀티-노드 시스템으로 마이그레이션하기 위한 예시적인 프로세스 흐름을 나타낸다.

도 8e는 시스템 내의 단일 노드로부터 다수의 노드로 확장하기 위한 다른 예시적인 프로세스 흐름을 나타낸다.

도 9a는 예시적인 물리 배치에서의 예시적인 실시형태에 따른 시스템의 개략도이다.

도 9b는 예시적인 실시형태에 따른 템플릿으로부터 도출된 예시적인 이미지의 개략도이다.

도 9c는 NT 패키지로부터 애플리케이션을 설치하는 일례를 나타낸다.

도 9d는 예시적인 배치에서의 예시적인 실시형태에 따른 시스템의 개략도이다.

도 9e는 가상 연산 리소스 호스트를 IT 시스템에 추가하기 위한 예시적인 프로세스 흐름을 나타낸다.

도 10은 예시적인 배치에서의 예시적인 실시형태에 따른 시스템의 개략도이다.

도 11a는 예시적인 실시형태의 시스템 및 방법을 도시한다.

도 11b는 예시적인 실시형태의 시스템 및 방법을 도시한다.

도 12는 예시적인 실시형태의 시스템 및 방법을 도시한다.

도 13a는 예시적인 실시형태에 따른 시스템의 개략도이다.

도 13b는 예시적인 실시형태에 따른 시스템의 다른 개략도이다.

도 13c 내지 도 13e는 예시적인 실시형태에 따른 시스템의 예시적인 프로세스 흐름을 도시한다.

도 14a는 메인 컨트롤러가 다른 시스템 상에 컨트롤러를 배치한 예시적인 시스템을 나타낸다.

도 14b 및 도 14c는 메인 컨트롤러를 갖는 컨트롤러를 제공하기 위한 가능한 단계를 도시하는 예시적인 흐름을 나타낸다.

도 15a는 메인 컨트롤러가 환경을 생성하는 예시적인 시스템을 나타낸다.

도 15b는 컨트롤러가 환경을 설정하는 예시적인 프로세스 흐름을 도시한다.

도 15c는 컨트롤러가 다수의 환경을 설정하는 예시적인 프로세스 흐름을 도시한다.

도 16a는 컨트롤러가 하나 이상의 컨트롤러를 설정하기 위한 메인 컨트롤러로서 작동하는 예시적인 실시형태를 도시한다.

도 16b 내지 도 16d는 환경이 다른 환경에 기록하도록 구성될 수 있는 예시적인 시스템을 나타낸다.

도 16e는 사용자가 컨트롤러에 의해 생성될 새로운 환경을 구매할 수 있는 예시적인 시스템을 나타낸다.

도 16f는 사용자 인터페이스가 컨트롤러에 의해 생성된 환경으로 인터페이스하기 위해 제공되는 예시적인 시스템을 도시한다.

도 17a 내지 도 18b는 새로운 환경에 대한 변경 관리 작업의 예를 도시한다.

발명을 실시하기 위한 구체적인 내용

[0013] 상기한 당 기술분야의 필요성에 대한 기술적 해결책을 제공하기 위해서, 본 발명은 자동화된 시스템 설정, 구성, 보수관리, 테스팅, 변경 관리 및/또는 업그레이드를 제공하는 정보 기술을 위한 시스템 및 방법에 관한 다양한 발명의 실시형태를 개시한다. 예를 들어, 본 발명자는 복수의 시스템 규칙, 컴퓨터 시스템에 대한 시스템 상태, 및 복수의 템플릿에 기초하여 컴퓨터 시스템을 자동적으로 관리하도록 구성되는 컨트롤러를 개시한다. 예를 들어, 본 발명자는 복수의 시스템 규칙, 컴퓨터 시스템에 대한 시스템 상태, 및 복수의 템플릿에 기초하여 컴퓨터 시스템을 위한 물리적 인프라구조를 자동적으로 관리하도록 구성되는 컨트롤러를 개시한다. 컨트롤러에 의해 수행될 수 있는 자동화 관리의 예는, 애플리케이션 또는 서비스를 실행할 수 있는 컴퓨터에 대한 셋팅 또는 다른 정보를 원격으로 또는 로컬로 액세스 및 변경하는 것, IT 시스템을 구축하는 것, IT 시스템을 변경하는 것, IT 시스템 내에 개별적인 스택을 구축하는 것, 서비스 또는 애플리케이션을 생성하는 것, 서비스 또는 애플리케이션을 로딩하는 것, 서비스 또는 애플리케이션을 구성하는 것, 서비스 또는 애플리케이션을 마이그레이션하는 것, 서비스 또는 애플리케이션을 변경하는 것, 서비스 또는 애플리케이션을 제거하는 것, 스택을 다른 네트워크 상의 다른 스택에 클로닝(cloning)하는 것, 리소스 또는 시스템 구성요소를 생성, 추가, 제거, 설정, 구성, 재구성 및/또는 변경하는 것, 리소스, 서비스, 애플리케이션, IT 시스템, 및/또는 IT 스택을 자동적으로 추가, 제거, 및/또는 반전하는 것, 애플리케이션, 서비스, 스택, 및/또는 다른 IT 시스템 간의 상호 작용을 구성하는 것, 및/또는 IT 시스템 구성요소의 건전성을 모니터링하는 것을 포함할 수 있다. 예시적인 실시형태에서, 컨트롤러는 원격 또는 로컬일 수 있는 물리 또는 가상 컴퓨팅 리소스로서 구현될 수 있다. 채용될 수 있는 컨트롤러의 추가 예는 프로세스, 가상 머신, 컨테이너, 원격 컴퓨팅 리소스, 다른 컨트롤러에 의해 배치된 애플리케이션, 및/또는 서비스 중 하나 또는 이들의 조합 중 어느 하나를 포함하지만 이에 한정되지 않는다. 컨트롤러는 다수의 노드 및/또는 리소스에 걸쳐 분산될 수 있고, 다른 위치 또는 네트워크에 있을 수 있다.

[0014] IT 인프라구조는 대부분 이산 하드웨어 및 소프트웨어 구성요소로부터 구성된다. 일반적으로 사용되는 하드웨어 구성요소는 서버, 랙, 전원 공급 장비, 상호 연결, 디스플레이 모니터, 및 다른 통신 장비로 구성된다. 이러한 이산 구성요소를 선택하고 그 후에 상호 연결하는 방법 및 기술은 다양한 수준의 효율성, 비용 효율성, 성능, 및 보안으로 기능하는 매우 많은 옵션 구성으로 인해 매우 복잡하다. 이러한 인프라구조 구성요소를 연결하는 데 숙련된 개별 기술자/엔지니어는 고용 및 훈련 비용이 많이 든다. 또한, 가능한 매우 많은 수의 하드웨어 및 소프트웨어의 반복이 하드웨어 및 소프트웨어의 유지 관리 및 업데이트의 복잡성을 생기게 한다. 이는 원래 IT 인프라구조를 설치한 개인 및/또는 엔지니어링 회사가 업데이트를 수행할 수 없을 때에 추가적인 도전 과제를 생기게 하였다. 운영 체제와 같은 소프트웨어 구성요소는 일반적으로 광범위한 하드웨어 상에서 작동하도록 설계되거나 특정의 구성요소에 매우 특화되어 있다. 대부분의 경우에, 복잡한 계획, 또는 블루 프린트가 작성되어 실행된다. 변경, 성장, 스케일링, 및 다른 도전 과제는 복잡한 계획이 업데이트되는 것을 필요로 한다.

[0015] 일부 IT 사용자는 성장하는 산업의 공급자로부터 클라우드 컴퓨팅 서비스를 구매하지만, 이는 인프라구조를 설정하는 문제 및 도전 과제를 해결하는 것이 아니라, 이를 IT 사용자로부터 클라우드 서비스 공급자로 전환하는

것이다. 또한, 대규모 클라우드 서비스 공급자는 유연성, 고객 맞춤화, 확장성 및 새로운 하드웨어 및 소프트웨어 기술의 신속한 채택을 감소시킬 수 있는 방식으로 인프라구조를 설정하는 도전 과제 및 문제를 해결해왔다. 또한, 클라우드 컴퓨팅 서비스는 박스외(out of the box) 베어-메탈 설정, 구성 배치 및 업데이트를 제공하지 않거나, 또는 베어-메탈과 가상 IT 인프라구조 구성요소로의, 이들로부터의 또는 이들 간의 전이를 허용하지 않는다. 클라우드 컴퓨팅 서비스의 이들 및 다른 제한은 다수의 컴퓨팅, 스토리지 및 네트워킹 비효율성을 유발할 수 있다. 예를 들어, 컴퓨팅 및 네트워킹에서의 속도 또는 레이턴시 비효율성은 클라우드 서비스에 의해 또는 클라우드 서비스를 이용하는 애플리케이션 또는 서비스로 제시될 수 있다.

[0016] 예시적인 실시형태의 시스템 및 방법은 신규하고 고유한 IT 인프라 배치, 사용 및 관리를 제공한다. 예시적인 실시형태에 따르면, 리소스 선택, 설치, 상호 연결, 관리 및 업데이트의 복잡성은 코어 컨트롤러 시스템 및 그의 파라미터 파일, 템플릿, 규칙, 및 IT 시스템 상태 내에 근거를 둔다. 시스템은 기술자가 조립, 연결, 및 관리하는 것을 필요로 하는 것이 아니라 구성요소가 자체 조립되도록 자체 조립 규칙 및 작동 규칙 세트를 구성한다. 또한, 예시적인 실시형태의 시스템 및 방법은 현재 전형적인 외부 계획 문서를 필요로 하지 않고 자체 조립 규칙을 사용하여 더 큰 맞춤화, 확장성, 및 유연성을 가능하게 한다. 이들은 또한 효율적인 리소스 사용 및 용도 변경을 가능하게 한다.

[0017] 전체적으로 또는 부분적으로 물리적이든 가상이든 현재 IT 시스템의 많은 이슈와 문제를 개선하는 시스템 및 방법이 제공된다. 예시적인 실시형태의 시스템 및 방법은 유연성을 허용하고, 가변성 및 인적 에러를 감소시키며, 시스템 보안을 향상시킬 가능성이 있는 구조를 제공한다.

[0018] 일부 해결책은 현재 IT 시스템에서 하나 이상의 문제에 대해 개별적으로 존재할 수 있지만, 이러한 해결책은 본 명세서에 설명되는 예시적인 실시형태에 의해 해결되는 다수의 문제를 포괄적으로 해결하지 못한다. 또한, 이러한 기존의 해결책은 특정 문제를 해결할 수 있지만, 다른 문제를 증가시킬 수 있다.

[0019] 현재 해결될 도전 과제 중 일부는 설정, 구성, 인프라구조 배치, 자산 추적, 보안, 애플리케이션 배치, 서비스 배치, 보수관리 및 컴플리언스의 문서화, 보수관리, 스케일링, 리소스 할당, 리소스 관리, 로드 밸런싱, 소프트웨어 장애, 소프트웨어 및 보안 업데이팅/패칭, 테스팅, IT 시스템 복구, 변경 관리, 및 하드웨어 업데이트에 관한 이슈를 포함하지만, 이에 한정되지 않는다.

[0020] 본 명세서에 사용되는 IT 시스템은 서버, 가상 및 물리 호스트, 데이터베이스 및 데이터베이스 애플리케이션을 포함할 수 있지만 이에 한정되지 않으며, 데이터베이스 애플리케이션은 IT 서비스, 비즈니스 컴퓨팅 서비스, 컴퓨터 애플리케이션, 고객 대면 애플리케이션, 웹 애플리케이션, 모바일 애플리케이션, 백-엔드(back-end), 케이스 번호 관리, 고객 추적, 발권, 비즈니스 툴, 데스크톱 관리 툴, 어카운팅, 이메일, 문서화, 컴플라이언스, 데이터 스토리지, 백업, 및/또는 네트워크 관리를 포함하지만 이에 한정되지 않는다.

[0021] IT 시스템을 설정하기 전에 사용자가 당면할 수 있는 하나의 문제는 인프라구조 필요성을 예측하는 것이다. 사용자는 초기 또는 시간 경과에 따라 성장 또는 변경 중에 얼마나 많은 스토리지, 연산력 또는 다른 요구 사항이 필요할 것인지를 알 수 없었다. 예시적인 실시형태에 따르면, IT 시스템 및 인프라구조는 시스템 변경이 필요한 경우, 예시적인 실시형태의 자체 배치 인프라구조(물리 및/또는 가상 모두)가 추후에 인프라구조로부터 자동적으로 추가, 제거, 또는 그 안에 재할당하는 데 사용될 수 있다는 점에서 유연성을 허용한다. 따라서, 시스템을 설정할 때에 제시되는 향후 필요성을 예측하는 도전 과제는 그의 글로벌 규칙, 템플릿, 및 시스템 상태를 사용하여 시스템에 추가할 능력을 제공하고, 이러한 규칙, 템플릿 및 시스템 상태의 변경을 추적함으로써 해결된다.

[0022] 다른 도전 과제는 또한, 예를 들어 시간 경과에 따라 구성된 시스템 요소 또는 그의 구성의 변경으로 인한 향후 비호환성을 포함할 수 있는 정확한 구성, 구성의 균일성, 상호 운용성, 및/또는 상호 의존성에 관한 것일 수 있다. 예를 들어, IT 시스템을 처음에 설정할 때, 요소가 없거나 일부 요소를 구성하지 못할 수 있다. 그리고, 예를 들어 요소 또는 인프라구조 구성요소의 반복이 설정될 때에는 반복 간에 균일성의 부족이 있을 수 있다. 구성은 시스템의 변경이 이루어질 때에 수정될 필요가 있을 수 있다. 최적의 구성과 향후 인프라구조 변경의 유연성 간에 어려운 선택이 제시되어 왔다. 시스템을 처음 배치할 때 예시적인 실시형태에 따르면, 구성은 템플릿으로부터 인프라구조 구성요소로의 글로벌 시스템 규칙을 사용하여 자체 배치되어 구성이 균일하고, 반복 가능하거나 예측 가능하여 최적의 구성을 가능하게 한다. 이러한 초기 시스템 배치는 물리 구성요소에서 수행될 수 있지만, 후속의 구성요소는 추가 또는 수정될 수 있으며 물리적이거나 그렇지 않을 수 있다. 또한, 이러한 초기 시스템 배치는 물리 구성요소에서 수행될 수 있지만, 후속의 환경은 물리 구조로부터 클로닝될 수 있으며 물리적이거나 그렇지 않을 수 있다. 이는 시스템 구성을 최적화하는 것을 가능하게 하면서 지장을 주는 향후 변경을 최소화할 수 있다.

[0023] 배치 단계에서는, 전형적으로 베어-메탈 및/또는 소프트웨어 정의 인프라구조의 상호 운용성의 도전 과제가 있다. 또한, 다른 애플리케이션, 툴 또는 인프라구조와 소프트웨어의 상호 운용성의 도전 과제도 있을 수 있다. 이들은 다른 벤더로부터 시작한 제품 배치로 인한 도전 과제를 포함하지만 이에 한정되지 않는다. 발명자는 베어-메탈, 가상 또는 이들의 조합에 상관없이 인프라구조의 상호 운용성을 제공할 수 있는 IT 시스템을 개시한다. 따라서, 부품의 협력 능력인 상호 운용성은 인프라구조가 자동적으로 구성되고 배치되는 개시된 인프라구조 배치에 내장될 수 있다. 예를 들어, 상이한 애플리케이션이 서로 의존할 수 있으며, 이들이 별도의 호스트에 존재할 수 있다. 이러한 애플리케이션이 서로 상호 작용할 수 있도록 하기 위해서, 본 명세서에 설명되는 컨트롤러 로직, 템플릿, 시스템 상태, 및 시스템 규칙은 애플리케이션의 상호 의존성을 구성하고 상호 의존성을 추적하는 데 사용될 정보 및 구성 명령어를 포함한다. 따라서, 본 명세서에 설명되는 인프라구조 특징은 각각의 애플리케이션 또는 서비스가 서로 대화하는 방법을 관리하는 방법을 제공한다. 예로서, 이메일 서비스가 인증 서비스와 적절하게 통신하는지 확인하는 것; 및/또는 그룹웨어 서비스가 이메일 서비스와 적절하게 통신하는지 확인하는 것이다. 또한, 이러한 관리는 연산 리소스가 예를 들어 스토리지 리소스와 통신하는 방법을 추적할 수 있도록 인프라구조 수준으로 내려갈 수 있다. 그렇지 않으면, IT 시스템의 복잡성이 $O(n^n)$으로 상승할 수 있다.

[0024] 개시된 바에 따르면, 리소스의 자동 배치는 글로벌 시스템 규칙, 템플릿, 및 IT 시스템 상태/시스템 자체 지식에 기초한 컨트롤러의 배치 능력으로 인해 운영 체제 소프트웨어를 사전 구성할 필요가 없다. 예시적인 실시형태에 따르면, 사용자 또는 IT 전문가는 리소스의 추가, 할당 또는 재할당이 상호 운용성을 보장하기 위해 함께 작동하는지 알 필요가 없을 수 있다. 예시적인 실시형태에 따른 추가 리소스는 네트워크에 자동적으로 추가될 수 있다.

[0025] 애플리케이션을 사용하는 것은 전형적으로 연산, 스토리지 및 네트워킹을 포함한 다양한 리소스를 필요로 한다. 또한, 무엇이 제자리에서 실행중인지에 대한 지식 및 다른 애플리케이션과의 상호 운용성을 포함한 리소스 및 시스템 구성요소의 상호 운용성이 필요하다. 애플리케이션은 다른 서비스에 연결하고 구성 파일을 가져와서 모든 구성요소가 함께 적절하게 작동하는지 확인할 필요가 있을 수 있다. 따라서, 애플리케이션 구성은 많은 시간과 리소스를 필요로 할 수 있다. 애플리케이션 구성은 다른 애플리케이션과의 상호 운용성에 문제가 있는 경우에 나머지 인프라구조에 의해 캐스케이딩 현상을 유발할 수 있다. 이는 사고 상태(outage) 또는 침입을 유발할 수 있다. 본 발명자는 이러한 이슈를 해결하기 위해 자동화된 애플리케이션 배치를 개시한다. 따라서, 본 발명자에 의해 개시된 바와 같이, 애플리케이션은 시스템에서 진행되고 있고 지능적으로 구성되는 것에 대한 지식을 사용하여 IT 시스템 상태, 글로벌 시스템 규칙 및 템플릿으로부터 판독함으로써 자체 배치가 이루어질 수 있다. 또한, 예시적인 실시형태에 따르면, 구성의 사전 배치(pre-deployment) 테스트는 본 명세서에 설명되는 변경 관리 특징을 사용하여 수행될 수 있다.

[0026] 예시적인 실시형태에 의해 해결되는 다른 이슈는 상이한 벤더 또는 다른 툴로 전환하는 것이 바람직한 중개 구성과 관련하여 발생할 수 있는 문제에 관한 것이다. 예시적인 실시형태의 양태에 따르면, 템플릿 변환(template translation)은 컨트롤러의 규칙 및 템플릿과 특정 벤더로부터의 애플리케이션 템플릿 간에 제공된다. 이는 시스템이 소프트웨어 또는 다른 툴의 벤더를 자동적으로 변경하는 것을 가능하게 한다.

[0027] 잘못된 구성, 패치의 실패, 및 배치 전에 패칭을 테스트하는 것의 불가능으로 인해 많은 보안 이슈가 발생한다. 보안 이슈는 종종 설정의 구성 단계에서 발생될 수 있다. 예를 들어, 잘못된 구성은 민감한 애플리케이션을 인터넷에 노출된 상태로 방치하거나 이메일 서버로부터 위조된 이메일을 허용할 수 있다. 본 발명자는 자동적으로 구성되는 시스템 설정을 개시하며, 이에 따라 공격자에 대해 보호하여 공격자에게 불필요한 노출을 회피하고 보안 엔지니어 및 애플리케이션 보안 설계자에게 시스템의 많은 지식을 제공한다. 자동화는 인적 에러 또는 잘못된 구성으로 인한 보안 결함을 감소시킨다. 또한, 개시된 인프라구조는 서비스 간의 내성을 제공하고 규칙 기반 액세스를 허용하며 서비스 간의 통신을 실제로 그것을 필요로 하는 것에만 제한할 수 있다. 본 발명자는, 예를 들어 변경 관리와 관련하여 설명된 바와 같이 배치 전에 패치를 안전하게 테스트하는 능력을 갖는 시스템 및 방법을 개시한다.

[0028] 문서화는 종종 IT 관리에서 문제가 되는 영역이다. 설정 및 구성 중에, 기본 목표는 전형적으로 구성요소를 함께 작동시키는 것일 수 있다. 전형적으로, 이는 때때로 시스템이 실제로 무슨 작업을 했는지 파악하기가 어려운 문제 해결과 시행 착오 과정을 포함한다. 실행되는 정확한 커맨드는 전형적으로 문서화되지만, 작동 시스템이 달성했을 수 있는 문제 해결 또는 시행 착오 과정은 종종 잘 문서화되지 않거나 심지어 전혀 문서화되지 않는다. 문서화의 문제 또는 부적절은 감사 추적 및 감사에 문제를 발생시킬 수 있다. 발생하는 문서화 문제는 컴플라이언스를 나타내는 데 문제를 발생시킬 수 있다. 컴플라이언스 이슈는 종종 시스템 또는 그의 구성요소를

구축할 때에 잘 알지 못할 수도 있다. 적용 가능한 컴플라이언스 결정은 IT 시스템의 설정 및 구성 후에만 알게 될 수 있다. 따라서, 문서화는 감사 및 컴플라이언스에 중요하다. 본 발명자는 자동적으로 문서화된 설정 및 구성을 제공하는 글로벌 시스템 규칙 데이터베이스, 템플릿, 및 IT 시스템 상태 데이터베이스를 포함하는 시스템을 개시한다. 발생하는 임의의 구성은 데이터베이스에 기록된다. 예시적인 실시형태에 따르면, 자동적으로 문서화된 구성은 감사 추적을 제공하고, 컴플라이언스를 나타내는 데 사용될 수 있다. 재고 관리는 자동적으로 문서화되고 추적된 정보를 사용할 수 있다.

[0029] IT 시스템 설정, 구성, 및 운영으로부터 발생하는 다른 도전 과제는 하드웨어 및 소프트웨어의 재고 관리를 포함한다. 예를 들어, 전형적으로 얼마나 많은 서버가 있는지, 가동하여 여전히 기능하고 있는지, 그들의 능력은 얼마인지, 각 서버가 어느 랙에 있는지, 전원이 어느 서버에 연결되는지, 각 서버가 무슨 네트워크 카드 및 무슨 네트워크 포트를 사용중인지, 구성요소가 어느 IT 시스템에서 작동되는지 및 많은 다른 중요한 노트를 하는 것이 중요하다. 재고 정보 이외에도, 재고 관리 및 다른 민감한 정보에 사용되는 비밀번호가 효과적으로 관리되어야 한다. 특히 대규모 IT 시스템, 데이터 센터 또는 장비가 자주 변경되는 데이터 센터에서, 이 정보의 수집 및 보존은 종종 수동으로 또는 다양한 소프트웨어 툴을 사용하여 관리되는 데 시간이 걸리는 작업이다. 보안 암호의 준수 보호(compliant protection)는 보안 컴퓨팅 환경을 보장하는 데 중요한 이슈일 수 있는 큰 위험 요인이다. 발명자는 모든 서버 및 다른 구성요소의 재고 및 운영 상태의 수집 및 유지 관리가 IT 시스템 상태, 글로벌 시스템 규칙, 템플릿, 및 컨트롤러의 컨트롤러 로직의 일부로서 자동적으로 업데이트, 저장 및 보호되는 IT 시스템을 개시한다.

[0030] IT 시스템의 설정 및 구성에 관한 문제 이외에도, 본 발명자는 IT 시스템의 보수관리에 나타나는 문제 및 이슈도 해결할 수 있는 IT 시스템을 개시한다. 하드웨어 장애, 예를 들어 특히 전원 장애, 메모리 장애, 네트워크 장애, 네트워크 카드 장애, 및/또는 CPU 장애가 있는 데이터 센터의 지속적인 기능과 관련하여 다수의 문제가 발생한다. 하드웨어 장애 중에 호스트를 마이그레이션할 때에 추가 장애가 발생한다. 따라서, 본 발명자는 동적 리소스 마이그레이션, 예를 들어 호스트가 다운될 때에 하나의 리소스 공급자에서 다른 리소스 공급자로 리소스를 마이그레이션하는 것을 개시한다. 이러한 상황에서, 예시적인 실시형태에 따르면, IT 시스템은 다른 서버, 노드 또는 리소스로, 또는 다른 IT 시스템으로 마이그레이션할 수 있다. 컨트롤러는 시스템의 상태를 보고할 수 있다. 데이터의 복제(duplicate)는 공지된 자동 설정 구성을 갖는 다른 호스트 상에 있다. 하드웨어 장애가 검출된 경우에는, 하드웨어가 제공할 수 있는 임의의 리소스가 장애를 자동적으로 검출한 후에 자동적으로 마이그레이션될 수 있다.

[0031] 많은 IT 시스템에서 중요한 이슈는 확장성이다. 성장하는 비즈니스 또는 다른 조직은 전형적으로 이들이 성장하고 이들의 필요성이 변경됨에 따라 IT 시스템을 추가 또는 재구성한다. 기존 IT 시스템에 더 많은 리소스가 필요할 때, 예를 들어 하드 드라이브 공간, 스토리지 공간, CPU 프로세싱, 더 많은 네트워크 인프라구조; 더 많은 엔드 포인트, 더 많은 클라이언트 및/또는 더 많은 보안을 추가할 때에 문제가 발생한다. 다른 서비스와 애플리케이션 또는 인프라구조의 변경이 필요할 때에 구성, 설정 및 배치에도 문제가 발생한다. 예시적인 실시형태에 따르면, 데이터 센터는 자동적으로 스케일링될 수 있다. 노드 또는 리소스는 동적으로 자동적으로 리소스의 풀에 추가되거나 그로부터 제거될 수 있다. 리소스 풀에 대해 추가 및 제거된 리소스는 자동적으로 할당 또는 재할당될 수 있다. 서비스가 제공되고 새로운 호스트로 신속하게 이동될 수 있다. 컨트롤러는 리소스 풀에 더 많은 리소스를 동적으로 감지 및 추가하고 리소스를 할당/재할당할 위치를 알 수 있다. 예시적인 실시형태에 따른 시스템은 단일 노드 IT 시스템으로부터 다수의 물리 및/또는 가상 노드 또는 다수의 데이터 센터 또는 IT 시스템에 걸친 리소스를 필요로 하는 스케일링된 시스템으로 스케일링될 수 있다.

[0032] 본 발명자는 유연한 리소스 할당 및 관리를 가능하게 하는 시스템을 개시한다.

[0033] 시스템은 리소스 풀에 있을 수 있고 동적으로 할당될 수 있는 연산, 스토리지 및 네트워킹 리소스를 포함한다. 컨트롤러는 네트워크 상의 새로운 노드 또는 호스트를 인식한 다음 이들이 리소스 풀의 일부가 될 수 있도록 구성할 수 있다. 예를 들어, 새로운 서버가 플러그인될 때마다 컨트롤러는 이를 리소스 풀의 일부로서 구성하고 이를 리소스에 추가할 수 있으며 동적으로 사용하기 시작할 수 있다. 노드 또는 리소스는 컨트롤러에 의해 검출되고 다른 풀에 추가될 수 있다. 리소스 요청은, 예를 들어 컨트롤러에 대한 API 요청을 통해 이루어질 수 있다. 그 후, 컨트롤러는 규칙에 따라 풀로부터 필요한 리소스를 배치 또는 할당할 수 있다. 이는 컨트롤러 및/또는 컨트롤러를 통한 애플리케이션이 요청의 필요성에 기초하여 리소스를 로드 밸런싱하고 동적으로 분배하는 것을 가능하게 한다.

[0034] 로드 밸런싱의 예는, 하드웨어 또는 소프트웨어 장애가 발생할 때에 새로운 리소스를 배치하는 것; 증가된 사용

자 로드에 응답하여 동일한 애플리케이션의 하나 이상의 인스턴스를 배치하는 것; 및 스토리지, 컴퓨팅 또는 네트워킹 요구 사항의 불균형에 응답하여 동일한 애플리케이션의 하나 이상의 인스턴스를 배치하는 것을 포함하지만 이에 한정되지 않는다.

[0035] 실제 IT 시스템 또는 환경을 변경하는 데 관여되는 문제는 지속적으로 작동하고 실행 중인 이러한 시스템에 의존하는 사용자 또는 엔터티에게 상당한, 때로는 비극적인 문제를 야기할 수 있다. 이러한 사고 상태는 시스템의 사용에서 잠재적 손실뿐만 아니라, 데이터의 손실, 상당한 시간 리소스로 인한 경제적 손실, 문제 해결에 필요한 인력 및 비용을 나타낸다. 구성의 문서화에 에러가 있거나 시스템의 이해가 부족한 시스템을 재구축하는 어려움에 의해 문제가 악화될 수 있다. 이 문제 때문에, 많은 IT 시스템 사용자는 알려진 보안 위험을 제거하기 위해 IT 리소스를 패치하는 것을 꺼려한다. 따라서, 그들은 보안 위반에 더욱 취약한 상태로 남는다.

[0036] IT 시스템의 보수관리에서 발생하는 많은 문제는 구성이 필요할 수 있는 변경 관리 또는 제어로 인한 소프트웨어 장애와 관련된다. 이러한 장애가 발생할 수 있는 상황은 새로운 소프트웨어 버전으로 업그레이드하는 것, 다른 소프트웨어로 마이그레이션하는 것; 비밀번호 또는 인증 관리 변경; 서비스 간 또는 서비스의 다른 공급자 간의 전환을 포함하지만 이에 한정되지 않는다.

[0037] 수동으로 구성하고 유지 관리되는 인프라구조는 전형적으로 재현하기가 어렵다. 인프라구조를 재현하는 것은 문제가 있는 변경, 정전을 롤링백(rolling back)하는 것 또는 다른 재해 복구를 포함한 여러 가지의 이유 때문에 중요할 수 있다. 수동으로 구성된 시스템의 문제는 진단하기가 어렵다. 수동으로 구성되고 유지 관리되는 인프라구조는 다시 만들기가 어렵다. 또한, 시스템 관리자는 부정확한 커맨드와 같은 실수를 쉽게 할 수 있어, 종국에는 컴퓨터 시스템을 다운시킨 것을 알게 된다.

[0038] 실제 IT 시스템 또는 환경을 변경하는 것은 지속적으로 작동하고 실행 중인 이러한 시스템에 의존하는 사용자 또는 엔터티에게 상당한, 때로는 비극적인 문제를 야기할 수 있다. 이러한 사고 상태는 시스템의 사용에서 잠재적 손실을 나타낼 뿐만 아니라, 이러한 사고 상태는 데이터의 손실, 상당한 시간 리소스로 인한 경제적 손실, 문제 해결에 필요한 인력 및 비용도 야기할 수 있다. 구성의 문서화에 에러가 있거나 시스템의 이해가 부족한 시스템을 재구축하는 어려움에 의해 문제가 악화될 수 있다. 그리고, 많은 경우에는 상당한 또는 중대한 변경 후에 시스템을 이전 상태로 복원하기가 매우 어렵다.

[0039] 또한, 실제 환경을 변경할 때에 잠재적으로 발생될 기술적 문제는 캐스케이딩 현상이 있을 수 있다. 이러한 캐스케이딩 현상은 도전 과제가 될 수 있고, 때로는 변경전(pre-change) 상태로 돌아가는 것이 불가능해질 수 있다. 따라서, 구현된 변경의 문제로 인해 변경이 되돌려져야 될 필요가 있는 경우에도, 시스템 상태는 이미 변경되어 있다. 최근에는 인프라구조 및 시스템 관리 오류뿐만 아니라 생산 환경(production environment)의 잘못된 변경을 취소하는 것이 해결되지 않은 문제인 것으로 언급되고 있다. 또한, 실제 환경으로의 배치 전에 시스템의 변경을 테스트하는 것은 문제가 있는 것으로 알려져 있다.

[0040] 따라서, 본 발명자는 실제 시스템으로의 변경을 변경전 상태로 되돌리도록 구성된 시스템 및 방법에 대한 다수의 예시적인 실시형태를 개시한다. 또한, 본 발명자는 전술한 하나 이상의 문제를 방지 또는 개선할 수 있는 실제 변경을 겪고 있는 시스템 또는 환경의 상태를 실질적으로 되돌릴 수 있도록 구성된 시스템 및 방법을 제공하는 것을 개시한다.

[0041] 예시적인 실시형태의 변형에 따르면, IT 시스템은 글로벌 시스템 규칙, 템플릿, 및 IT 시스템 상태에 대한 완전한 시스템 지식을 갖는다. 인프라구조는 완전한 시스템 지식을 사용하여 클로닝될 수 있다. 시스템 또는 시스템 환경은 소프트웨어 정의 인프라구조 또는 환경으로서 클로닝될 수 있다. 생산 환경이라고 지칭되는, 사용중인 휘발성 데이터베이스를 포함하는 시스템 환경은 개발 및 테스트 프로세스에서 개발 환경으로서 사용될 비휘발성 판독 전용 데이터베이스 내에 기록될 수 있다. 원하는 변경이 개발 환경에서 이루어지고 테스트될 수 있다. 사용자 또는 컨트롤러 로직은 글로벌 규칙을 변경하여 새로운 버전을 생성할 수 있다. 규칙의 버전은 추적될 수 있다. 예시적인 실시형태의 다른 양태에 따르면, 그 후에 새롭게 개발된 환경이 자동적으로 구현될 수 있다. 이전의 생산 환경은 또한 유지 관리되거나 완전히 기능적이어서 데이터의 손실없이 이전 상태 생산 환경으로의 복귀가 가능하다. 그 후, 개발 환경은 새로운 사양, 규칙, 및 템플릿으로 부팅될 수 있고, 데이터베이스 또는 시스템은 생산 데이터베이스와 동기화되고 기록 가능한 데이터베이스로 전환될 수 있다. 그 후, 원래 생산 데이터베이스는 복구가 필요한 경우에 시스템이 되돌릴 수 있는 판독 전용 데이터베이스로 전환될 수 있다.

[0042] 소프트웨어의 업그레이드 또는 패치와 관련하여, 업그레이드 또는 패치가 필요한 서비스가 검출되는 경우에 새로운 호스트가 배치될 수 있다. 새로운 서비스는 업그레이드 또는 패치로 인한 오류가 발생한 경우에 전술한 바

와 같이 변경 복귀가 가능한 동안 배치될 수 있다.

[0043] 하드웨어 업그레이드는 특히 최신 하드웨어가 필수인 많은 상황에서 중요하다. 이러한 유형의 상황의 예는 밀리초의 속도 이점을 갖는 IT 시스템이 사용자가 우수한 거래 결과와 수익을 달성할 수 있게 하는 고빈도 거래 산업에서 발생한다. 특히, 새로운 하드웨어가 프로토콜과 통신하고 기존의 인프라구조와 작업하는 방법을 알 수 있도록 현재 인프라구조와의 상호 운용성을 보장함에 있어서 문제가 발생한다. 구성요소의 상호 운용성을 보장하는 것 이외에도, 구성요소는 기존의 설정과의 통합을 필요로 할 것이다.

[0044] 도 1을 참조하면, 예시적인 실시형태의 IT 시스템(100)이 도시되어 있다. 시스템(100)은 본 명세서에 설명되는 것을 포함하지만 이에 한정되지 않는 하나 이상의 유형의 IT 시스템일 수 있다.

[0045] 사용자 인터페이스(UI)(110)는 독립형 물리 또는 가상 서버에 상주하거나 그렇지 않을 수 있는 애플리케이션 프로그램 인터페이스(application program interface: API) 애플리케이션(120)을 통해 컨트롤러(200)에 결합된 것으로 나타나 있다. 컨트롤러(200)는 본 명세서에 설명되는 임의의 제어 동작을 구현하기 위해 하나 이상의 프로세서 및 하나 이상의 메모리 상에 배치될 수 있다. 이러한 제어 동작을 수행하기 위한 프로세서(들)에 의한 실행을 위한 명령어는 프로세서 메모리와 같은 비일시적 컴퓨터 판독가능 저장 매체에 상주될 수 있다. API(120)는 리던던트일 수 있고 및/또는 병렬로 동작할 수 있는 하나 이상의 API 애플리케이션을 포함할 수 있다. API 애플리케이션(120)은 시스템 리소스를 구성하기 위한 요청을 수신하고, 요청을 파싱(parsing)하여 이를 컨트롤러(200)에 전달한다. API 애플리케이션(120)은 컨트롤러로부터 하나 이상의 응답을 수신하고, 응답(들)을 파싱하여 이를 UI(또는 애플리케이션)(110)에 전달한다. 대안적으로 또는 추가적으로, 애플리케이션 또는 서비스는 API 애플리케이션(120)과 통신할 수 있다. 컨트롤러(200)는 연산 리소스(들)(300), 스토리지 리소스(들)(400) 및 네트워킹 리소스(들)(500)에 결합된다. 리소스(300, 400, 500)는 단일 노드에 상주하거나 그렇지 않을 수 있다. 리소스(300, 400, 500) 중 하나 이상은 가상일 수 있다. 리소스(300, 400, 500)는 다수의 노드에 또는 다수의 노드에 다양한 조합으로 상주하거나 그렇지 않을 수 있다. 물리 디바이스는 연산 리소스(300), 스토리지 리소스(400), 및 네트워킹 리소스(500)를 포함하지만 이에 한정되지 않는 하나 이상 또는 각각의 리소스 유형을 포함할 수 있다. 리소스(300, 400, 500)는 또한 상이한 물리 위치에 있는지 여부와 가상인지 여부에 관계없이 리소스의 풀을 포함할 수 있다. 베어-메탈 연산 리소스는 또한 가상 또는 컨테이너 연산 리소스의 사용을 가능하게 하는 데 사용될 수 있다.

[0046] 알려진 노드 정의 이외에, 본 명세서에 사용되는 노드는 독립형 또는 네트워크 연결형 디바이스에서 기능을 수행하는 네트워크(들) 또는 다른 기능 유닛에 연결된 임의의 시스템, 디바이스 또는 리소스일 수 있다. 노드는 또한, 예를 들어 서버, 물리 또는 가상 호스트 상의 서비스/애플리케이션/복수의 서비스, 가상 서버, 및/또는 멀티-테넌트 서버 상의 또는 컨테이너 내측에서 실행하는 복수 또는 단일의 서비스를 포함할 수 있지만 이에 한정되지 않는다.

[0047] 컨트롤러(200)는, 리던던트일 수도 있고 및/또는 병렬로 동작할 수도 있는 하나 이상의 물리 또는 가상 컨트롤러 서버를 포함할 수 있다. 컨트롤러는 연산 호스트로서 기능하는 물리 또는 가상 호스트에서 실행할 수 있다. 일례로서, 컨트롤러는, 예를 들어 민감한 리소스에 액세스할 수 있기 때문에 다른 목적도 수행하는 호스트에서 실행하는 컨트롤러를 포함할 수 있다. 컨트롤러는 API 애플리케이션(120)으로부터 요청을 수신하고, 요청을 파싱하여 다른 리소스에 대한 적절한 작업을 만들고 다른 리소스를 지시하며; 리소스로부터 정보를 모니터링하고 수신하며; 시스템의 상태 및 변경의 이력을 유지하고; IT 시스템의 다른 컨트롤러와 통신할 수 있다. 컨트롤러는 API 애플리케이션(120)을 포함할 수도 있다.

[0048] 본 명세서에 정의된 연산 리소스는 실제 또는 가상의 단일 연산 노드 또는 하나 이상의 연산 노드를 갖는 리소스 풀을 포함할 수 있다. 연산 리소스 또는 연산 노드는 하나 이상의 서비스를 호스팅하거나 하나 이상의 애플리케이션을 실행할 수 있는 하나 이상의 물리 또는 가상 머신 또는 컨테이너 호스트를 포함할 수 있다. 연산 리소스는 또한 컴퓨팅, 스토리지, 캐싱, 네트워킹, 특수 컴퓨팅을 포함하지만 이에 한정되지 않는 다수의 목적을 위해 설계된 하드웨어 상에 있을 수 있고, 특수 컴퓨팅은 GPU, ASIC, 코프로세서(co-processor), CPU, FPGA 및 다른 특수 컴퓨팅 방법을 포함하지만 이에 한정되지 않는다. 이러한 디바이스는 PCI 익스프레스 스위치 또는 유사한 디바이스와 함께 추가될 수 있으며 이러한 방식으로 동적으로 추가될 수 있다. 연산 리소스 또는 연산 노드는 서비스 또는 애플리케이션을 실행하거나 가상 연산 리소스일 수 있는 복수의 다른 가상 머신을 포함하는 하나 이상의 하이퍼바이저 또는 컨테이너 호스트를 포함할 수 있거나 이를 실행할 수 있다. 연산 리소스의 강조는 연산 기능을 제공하는 것이지만, 데이터 스토리지 및/또는 네트워킹 능력을 포함할 수도 있다.

[0049] 본 명세서에 정의된 스토리지 리소스는 스토리지 노드 또는 풀 또는 스토리지 리소스를 포함할 수 있다. 스토리

지 리소스는 임의의 데이터 스토리지 매체, 예를 들어 고속, 저속, 하이브리드, 캐시 및/또는 RAM을 포함할 수 있다. 스토리지 리소스는 다른 스토리지 리소스에 직접 부착되거나 그렇지 않을 수 있는 하나 이상의 유형의 네트워크, 머신, 디바이스, 노드 또는 이들의 임의의 조합을 포함할 수 있다. 예시적인 실시형태의 양태에 따르면, 스토리지 리소스는 베어-메탈 또는 가상 또는 이들의 조합일 수 있다. 스토리지 리소스의 강조는 스토리지 기능을 제공하는 것일 수 있지만, 연산 및/또는 네트워킹 능력을 포함할 수도 있다.

[0050] 네트워킹 리소스(들)(500)는 단일 네트워킹 리소스, 복수의 네트워킹 리소스 또는 네트워킹 리소스의 풀을 포함할 수 있다. 네트워킹 리소스(들)는 물리 또는 가상 디바이스(들), 툴(들), 스위치, 라우터 또는 시스템 리소스 간의 다른 상호 연결, 또는 네트워킹을 관리하기 위한 애플리케이션을 포함할 수 있다. 이러한 시스템 리소스는 물리 또는 가상일 수 있고, 컴퓨팅, 스토리지, 또는 다른 네트워킹 리소스를 포함할 수 있다. 네트워킹 리소스는 외부 네트워크와 애플리케이션 네트워크 간의 연결을 제공할 수 있고, DNS, DHCP, 서브넷 관리, 레이어 3 라우팅, NAT, 및 다른 서비스를 포함하지만 이에 한정되지 않는 호스트 코어 네트워크 서비스일 수 있다. 이들 서비스 중 일부는 물리 또는 가상 시스템의 연산 리소스, 스토리지 리소스, 또는 네트워킹 리소스에서 배치될 수 있다. 네트워킹 리소스는 인피니밴드(Infiniband), 이더넷, RoCE, 파이버 채널 및/또는 옴니패스(Omnipath)를 포함하지만 이에 한정되지 않는 하나 이상의 패브릭 또는 프로토콜을 이용할 수 있고, 복수의 패브릭 간의 상호 연결을 포함할 수 있다. 네트워킹 리소스는 SDN 가능하거나 그렇지 않을 수 있다. 컨트롤러(200)는 IT 시스템의 토폴로지를 구성하기 위해 SDN, VLAN 등을 사용하여 네트워킹 리소스(300)를 직접 변경할 수도 있다. 네트워크 리소스의 강조는 네트워킹 기능을 제공하는 것일 수 있지만, 연산 및/또는 스토리지 능력을 포함할 수도 있다.

[0051] 본 명세서에 사용되는 애플리케이션 네트워크는 애플리케이션, 리소스, 서비스, 및/또는 다른 네트워크를 연결 또는 결합하기 위한, 또는 사용자 및/또는 클라이언트를 애플리케이션, 리소스, 및/또는 서비스에 결합하기 위한 네트워킹 리소스, 또는 그의 임의의 조합을 의미한다. 애플리케이션 네트워크는 서버가 다른 애플리케이션 서버(물리 또는 가상)와 통신하고 클라이언트와 통신하기 위해 사용되는 네트워크를 포함할 수 있다. 애플리케이션 네트워크는 시스템(100) 외부의 머신 또는 네트워크와 통신할 수 있다. 예를 들어, 애플리케이션 네트워크는 웹 프론트엔드를 데이터베이스에 연결할 수 있다. 사용자는 컨트롤러에 의해 관리되거나 그렇지 않을 수 있는 인터넷 또는 다른 네트워크를 통해 웹 애플리케이션에 연결할 수 있다.

[0052] 예시적인 실시형태에 따르면, 연산, 스토리지 및 네트워킹 리소스(300, 400, 500)는 각각 컨트롤러(200)에 의해 자동적으로 추가, 제거, 설정, 할당, 재할당, 구성, 재구성 및/또는 배치될 수 있다. 예시적인 실시형태에 따르면, 추가 리소스가 리소스 풀에 추가될 수 있다.

[0053] 사용자(105)가 액세스하고 시스템과 상호 작용할 수 있는 웹 UI 또는 다른 사용자 인터페이스와 같은 사용자 인터페이스(110)가 나타나 있지만, 대안적으로 또는 추가적으로, 애플리케이션은 API 애플리케이션(들)(120) 또는 다른 것을 통해 컨트롤러(200)와 통신하거나 상호 작용할 수 있다. 예를 들어, 사용자(105) 또는 애플리케이션은, IT 시스템의 구축; IT 시스템 내에 개별 스택의 구축; 서비스 또는 애플리케이션의 생성; 서비스 또는 애플리케이션의 마이그레이션; 서비스 또는 애플리케이션의 변경; 서비스 또는 애플리케이션의 제거; 스택을 다른 네트워크의 다른 스택에 클로닝; 리소스 또는 시스템 구성요소의 생성, 추가; 제거; 설정 또는 구성; 재구성을 포함하지만 이에 한정되지 않는 요청을 송출할 수 있다.

[0054] 도 1의 시스템(100)은 물리 또는 가상 또는 이들의 임의의 조합일 수 있는 다양한 요소, 구성요소 또는 리소스에 대한 연결 또는 다른 통신 인터페이스를 갖는 서버를 포함할 수 있다. 변형에 따르면, 도 1에 도시된 시스템(100)은 연결을 갖는 베어 메탈 서버를 포함할 수 있다.

[0055] 본 명세서에 더욱 상세하게 설명되는 바와 같이, 컨트롤러(200)는, 리소스 또는 구성요소의 전원을 켜는 것, 리소스의 부팅을 자동 설정, 구성, 및/또는 제어하는 것, 리소스를 추가하는 것, 리소스를 할당하는 것, 리소스를 관리하고 사용 가능한 리소스를 업데이트하는 것을 수행하도록 구성될 수 있다. 전원 투입(power up) 프로세스는 부팅되는 디바이스의 순서가 일관되고 디바이스의 전원을 켜는 사용자에 의존하지 않을 수 있도록 컨트롤러 전원을 켜는 것으로 시작할 수 있다. 이 프로세스는 전원 투입된 리소스의 검출을 포함할 수도 있다.

[0056] 도 2a 내지 도 10을 참조하면, 컨트롤러(200), 컨트롤러 로직(205), 글로벌 시스템 규칙 데이터베이스(210), IT 시스템 상태(220), 및 템플릿(230)이 도시되어 있다.

[0057] 시스템(100)은 글로벌 시스템 규칙(210)을 포함한다. 글로벌 시스템 규칙(210)은 특히 연산, 스토리지 및 네트워킹을 포함할 수 있는 리소스를 설정, 구성, 부팅, 할당 및 관리하는 규칙을 선언할 수 있다. 글로벌 시스템 규칙(210)은 시스템(100)이 정확한 또는 원하는 상태에 있어야 하는 최소 요구 사항을 포함한다. 이러한 요구

사항은 완료될 것으로 예상되는 IT 작업 및 원하는 시스템을 예측 가능하게 구축하는 데 필요한 예상되는 하드웨어의 업데이트 가능한 리스트를 포함할 수 있다. 예상되는 하드웨어의 업데이트 가능한 리스트는, (규칙의 시작 또는 템플릿의 사용으로부터, 예를 들어 그 전에) 필요한 리소스가 이용 가능한지를 컨트롤러가 검증하는 것을 가능하게 할 수 있다. 글로벌 규칙은 다양한 작업에 필요한 동작 리스트 및 동작 및 작업 순서에 관한 대응하는 명령어를 포함할 수 있다. 예를 들어, 규칙은 다른 작업을 시작할 때, 예를 들어 애플리케이션을 로딩, 구성, 시작, 재로딩하거나, 또는 하드웨어를 업데이트할 때, 구성요소의 전원 켜기, 리소스, 애플리케이션 및 서비스의 부팅 순서, 종속성을 지정할 수 있다. 규칙(210)은 또한, 예를 들어 애플리케이션 및 서비스에 필요한 리소스 할당의 리스트; 사용될 수 있는 템플릿의 리스트; 로딩될 애플리케이션의 리스트 및 구성 방법; 로딩될 서비스의 리스트 및 애플리케이션 네트워크의 리스트 및 어느 애플리케이션이 어느 네트워크로 진행할 것인지를 구성하는 방법; 다른 애플리케이션에 특정한 구성 변수 및 사용자 특정 애플리케이션 변수의 리스트; 컨트롤러가 시스템 상태를 검사하여 상태가 예상된 바와 같고 각 명령어의 결과가 예상된 바와 같은지를 검증하는 것을 가능하게 하는 예상 상태; 및/또는 규칙 변경의 추적 및 다른 환경에서 다른 규칙으로 테스트 또는 되돌리는 능력을 가능하게 할 수 있는 규칙 변경의 리스트를 포함하는 버전 리스트(예를 들어, 스냅샷) 중 하나 이상을 포함할 수 있다. 컨트롤러(200)는 글로벌 시스템 규칙(210)을 물리 리소스 상의 IT 시스템(100)에 적용하도록 구성될 수 있다. 컨트롤러(200)는 글로벌 시스템 규칙(210)을 가상 리소스 상의 IT 시스템(100)에 적용하도록 구성될 수 있다. 컨트롤러(200)는 글로벌 시스템 규칙(210)을 물리 및 가상 리소스의 조합의 IT 시스템(100)에 적용하도록 구성될 수 있다.

[0058] 도 2m은 글로벌 시스템 규칙의 형태를 취할 수 있는 예시적인 시스템 규칙 세트(210)를 도시한다. 도 2m으로 나타낸 예시적인 시스템 규칙 세트(210)는 컨트롤러(200)에 로딩되거나 시스템 상태를 질의(querying)함으로써 도출될 수 있다(210.1 참조). 도 2m의 예에서, 시스템 규칙(210)은 구성 루틴(210.2)의 형태를 취할 수 있는 명령어 세트를 포함하고, 또한 IT 시스템 또는 환경을 생성 및/또는 재생성하기 위한 데이터(210.3)를 포함한다. 시스템 규칙(210) 내의 구성 규칙은 필요한 템플릿 리스트(210.7)를 통해 템플릿(230)을 탐색하는 방법을 알 수 있다(여기서, 템플릿(230)은 파일시스템, 디스크, 스토리지 리소스에 상주할 수 있거나 또는 시스템 규칙 내부에 배치될 수 있다). 컨트롤러 로직(205)은 또한 템플릿(230)을 프로세싱하기 전에 탐색하여 시스템 규칙(210)을 활성화하기 전에 템플릿이 존재하는지 확인할 수 있다. 시스템 규칙(210)은 시스템 규칙의 서브세트(210.15)를 포함할 수 있고, 이러한 서브세트(210.15)는 구성 루틴(210.2)의 일부로서 실행될 수 있다.

[0059] 또한, 서브시스템 규칙(210.15)은, 예를 들어 통합 IT 애플리케이션의 시스템을 구축하는 툴로서 사용될 수 있다(그 후에 시스템 규칙 실행 루틴(210.16)으로 프로세싱된 다음, 210.15의 추가를 반영한 시스템 상태 및 현재 구성 규칙을 업데이트한다). 서브시스템 규칙(210.15)은 또한 다른 곳에 배치될 수도 있고 사용자 상호 작용에 의해 시스템 상태(220)로 로딩될 수 있다. 예를 들어, 사용자는 또한 서브시스템 규칙(210.15)을 플레이북으로서 사용할 수 있고, 플레이북이 사용 가능하여 실행될 수 있다(그 후에 글로벌 시스템 규칙(210)이 업데이트되어 사용자가 시스템을 클로닝하려는 경우에 플레이북을 재생할 수 있다).

[0060] 구성 루틴(210.2)은 시스템을 구축하는 데 사용되는 명령어 세트일 수 있다. 구성 루틴(210.2)은 또한 실무자가 원하는 경우에 서브시스템 규칙(210.15) 또는 시스템 상태 포인터(210.8)를 포함할 수 있다. 구성 루틴(210.2)을 실행할 때, 컨트롤러 로직(205)은 특정 순서(210.9)로 일련의 템플릿을 프로세싱할 수 있고, 선택적으로 병렬 배치를 허용하지만, 적절한 종속성 핸들링(210.12)을 유지한다. 구성 루틴(210.2)은 210.9에 따라 템플릿을 프로세싱함으로써 구성될 수 있는 애플리케이션에 구성 파라미터(210.5)을 설정할 수 있는 API 호출(210.10)을 선택적으로 호출할 수 있다. 또한, 필요한 서비스(210.11)는 시스템이 API 호출(들)(210.10)을 수행하려는 경우에 시작 및 실행해야 하는 서비스이다.

[0061] 루틴(210.2)은 또한 데이터의 복사, 데이터베이스의 컴퓨팅 리소스로의 전송, 연산 리소스와 스토리지 리소스의 페어링, 및/또는 휘발성 데이터(210.6)의 위치를 갖는 시스템 상태(220)의 업데이트를 포함하지만 이에 한정되지 않는 휘발성 데이터(210.6)에 대한 데이터 로팅(210.13)을 위한 절차, 프로그램, 또는 방법을 포함할 수 있다. 휘발성 데이터에 대한 포인터(210.4 참조)는 다른 곳에 저장될 수 있는 휘발성 데이터를 탐색하기 위해 데이터(210.3)로 유지될 수 있다. 데이터 로딩 루틴(210.13)은 또한 (예를 들어, 데이터베이스에 포함된) 비표준 데이터스토어에 배치하는 경우에 로드 구성 파라미터(210.5)가 사용될 수도 있다.

[0062] 시스템 규칙(210)은 또한, 어느 구성요소가 어느 리소스에 할당되는지를 지시할 수 있고 컨트롤러 로직(205)이 적절한 리소스 및/또는 하드웨어가 이용 가능한지를 결정할 수 있게 하는 리소스 리스트(210.18)를 포함할 수 있다. 시스템 규칙(210)은 또한 대안적인 배치를 위한(예를 들어, 소프트웨어 엔지니어가 실제 테스트를 수행하고 싶지만 전체 데이터센터를 할당하고 싶지 않은 개발 환경을 위한) 대안적인 하드웨어 및/또는 리소스 리스트

(210.19)를 포함할 수 있다. 시스템 규칙은 또한 시스템을 백업하고 리던던시를 위해 스탠바이(standby)를 사용하는 방법에 대한 명령어를 제공하는 데이터 백업/스탠바이 루틴(210.17)을 포함할 수 있다.

[0063] 모든 액션이 수행된 후, 시스템 상태(220)는 업데이트될 수 있고 (기록을 포함할 수 있는) 쿼리가 시스템 상태 쿼리(210.14)로서 저장될 수 있다.

[0064] 도 2n은 도 2m의 시스템 규칙(210)(또는 서브시스템 규칙(210.15))을 프로세싱하는 컨트롤러 로직(205)의 예시적인 프로세스 흐름을 도시한다. 단계 210.20에서, 컨트롤러 로직(205)은 적절한 리소스가 이용 가능한지를 확인하기 위해 검사한다(도 2m의 210.18 참조). 그렇지 않으면, 단계 210.21에서 대안적인 구성이 검사될 수 있다. 제3 옵션은 사용자에게 도 2m의 리스트(210.7)에서 참조되는 템플릿(230)에 의해 지원받을 수 있는 대안적인 구성을 선택하도록 프롬프트되는 것을 포함할 수 있다.

[0065] 단계 210.22에서, 컨트롤러 로직은 그 후에 연산 리소스(또는 적절한 리소스 중 어느 하나)가 휘발성 데이터에 액세스하는지를 확인할 수 있다. 이는 스토리지 리소스에 연결하는 것 또는 스토리지 리소스를 시스템 상태(220)에 추가하는 것을 포함할 수 있다. 단계 210.23에서, 구성 루틴이 그 후에 프로세싱되고, 각 루틴이 프로세싱됨에 따라 시스템 상태(220)가 업데이트된다(단계 210.24). 시스템 상태(220)는 또한 진행하기 전에 특정 단계가 완료되었는지를 검사하기 위해 질의될 수 있다(단계 210.25).

[0066] 도 210.23으로 나타낸 구성 루틴 프로세싱 단계는 210.26의 절차(또는 이들의 조합) 중 어느 하나를 포함할 수 있다. 다른 절차도 포함할 수 있다. 예를 들어, 210.26에서의 프로세싱은 템플릿 프로세싱(210.27), 구성 데이터 로딩(210.28), 정적 데이터 로딩(210.29), 동적 휘발성 데이터 로딩(210.30), 및/또는 서비스, 앱, 서브시스템, 및/또는 환경의 결합(210.31)을 포함할 수 있다. 210.26 내의 이러한 절차는 일부 시스템 구성요소가 독립적 일 수 있고 다른 것이 상호 의존적일 수 있으므로 루프에서 반복되거나 병렬로 실행할 수 있다. 컨트롤러 로직, 서비스 의존성, 및/또는 시스템 규칙은 어느 서비스가 서로 의존할 수 있는지를 지시할 수 있으며, 시스템 규칙으로부터 IT 시스템을 더 구축하기 위해 서비스를 결합할 수 있다.

[0067] 글로벌 시스템 규칙(210)은 또한 스토리지 확장 규칙을 포함할 수 있다. 스토리지 확장 규칙은, 예를 들어 시스템 내의 기존 스토리지 리소스에 스토리지 리소스를 자동적으로 추가하는 규칙 세트를 제공한다. 또한, 스토리지 확장 규칙은 연산 리소스(들) 상에서 실행하는 애플리케이션이 스토리지 확장을 요청하는 시기를 알 수 있는 트리거 포인트를 제공할 수 있다(또는 컨트롤러(200)는 연산 리소스 또는 애플리케이션의 스토리지를 확장할 시기를 알 수 있다). 컨트롤러(200)는 새로운 스토리지 리소스를 할당 및 관리할 수 있으며, 스토리지 리소스를 특정의 실행중인 리소스에 대한 기존 스토리지 리소스와 병합 또는 통합할 수 있다. 이러한 특정의 실행중인 리소스는, 시스템 내의 연산 리소스, 시스템 내에서 컴퓨터 리소스를 실행하는 애플리케이션, 가상 머신, 컨테이너, 또는 물리 또는 가상 연산 호스트 또는 이들의 조합일 수 있지만 이에 한정되지 않는다. 실행중인 리소스는, 예를 들어 스토리지 공간 쿼리를 통해 스토리지 공간이 부족하다는 것을 컨트롤러(200)에 신호할 수 있다. 대역내 관리 연결(270)에서, SAN 연결(280), 또는 컨트롤러(200)에 대한 임의의 네트워킹 또는 커플링이 이러한 쿼리에 사용될 수 있다. 대역외(out of band) 관리 연결(260)도 사용될 수 있다. 이들 스토리지 확장 규칙(또는 이들 스토리지 확장 규칙의 서브세트)은 실행중이 아닌 리소스에도 사용될 수 있다.

[0068] 스토리지 확장 규칙은 시스템 내에서 새로운 스토리지 리소스를 탐색하고, 연결하고, 설정하는 방법을 지시한다. 컨트롤러는 시스템 상태(220) 내에 새로운 스토리지 리소스를 등록하고, 스토리지 리소스가 있는 위치 및 그것에 연결하는 방법을 실행중인 리소스에 알려준다. 실행중인 리소스는 이러한 등록 정보를 사용하여 스토리지 리소스에 연결한다. 컨트롤러(200)는 새로운 스토리지 리소스를 기존 스토리지 리소스와 병합할 수 있거나, 또는 새로운 스토리지 리소스를 볼륨 그룹에 추가할 수 있다.

[0069] 도 2b는 예시적인 스토리지 확장 규칙 세트의 동작의 예시적인 흐름을 도시한다. 단계 210.41에서, 실행중인 리소스는 트리거 포인트 또는 다른 것에 기초하여 스토리지에 부족한 것으로 결정한다. 단계 210.42에서, 실행중인 리소스는 대역내 관리 연결(270), SAN 연결(280), 또는 운영 체제에 보이는 다른 유형의 연결을 통해 컨트롤러(200)에 연결한다. 이 연결을 통해, 실행중인 리소스는 컨트롤러(200)에 스토리지가 부족하다는 것을 통지할 수 있다. 단계 210.43에서, 컨트롤러는 실행중인 리소스에 대한 스토리지 용량을 확장하도록 스토리지 리소스를 구성한다. 단계 210.44에서, 컨트롤러는 새롭게 구성된 스토리지 리소스가 배치되는 위치에 관한 정보를 실행중인 리소스에 제공한다. 단계 210.45에서, 실행중인 리소스는 새롭게 구성된 스토리지 리소스에 연결한다. 단계 210.46에서, 컨트롤러는 새로운 스토리지 리소스 위치의 시스템 상태(220)에 맵을 추가한다. 그 후, 컨트롤러는 새로운 스토리지 리소스를 실행중인 리소스에 할당된 볼륨 그룹에 추가할 수 있거나(단계 210.47), 컨트롤러는 실행중인 리소스에 대한 새로운 스토리지 리소스의 할당을 시스템 상태(220)에 추가할 수 있다(단계 210.48).

[0070] 도 2c는 도 2b의 단계 210.41 및 210.42를 수행하기 위한 대안적인 실시예를 도시한다. 단계 210.50에서, 컨트롤러는 대역외 관리 연결(260)을 통해 키 커맨드를 전송하여 실행중인 리소스에 대한 스토리지 상태 업데이트를 위한 모니터 또는 콘솔을 볼 수 있다. 예를 들어, 모니터는 대역외 연결(260)을 통해 스크린을 검토할 수 있는 ipmi 콘솔일 수 있다. 일례로서, 대역외 연결(260)은 키보드/마우스로서 USB에 그리고 VGA 모니터 포트에 플러그될 수 있다. 단계 210.51에서, 실행중인 리소스는 스크린에 정보를 표시한다. 단계 210.52에서, 그 후에 컨트롤러는 대역외 관리 연결(260) 및 스크린 스크랩 또는 유사한 동작을 통해 모니터 또는 콘솔 상에 제시된 정보를 판독하고; 여기서, 이 판독 정보는 트리거 포인트에 기초하여 낮은 스토리지 상태를 표시할 수 있다. 그 후에 프로세스 흐름이 도 2b의 단계 210.43으로 계속될 수 있다.

[0071] 도 2d는 도 2b의 단계 210.41 및 210.42를 수행하기 위한 다른 대안적인 실시예를 도시한다. 단계 210.55에서, 실행중인 리소스는 컨트롤러에 의해 판독하기 위해 모니터 또는 콘솔에 정보를 자동적으로 표시한다. 단계 210.56에서, 컨트롤러는 자동적으로, 주기적으로 또는 지속적으로 모니터 또는 콘솔을 판독하여 실행중인 리소스를 검사한다. 이 판독에 응답하여, 컨트롤러는 실행중인 리소스가 저장 공간이 부족하다는 것을 알게 된다(단계 210.57). 그 후에 프로세스 흐름이 도 2b의 단계 210.43으로 계속될 수 있다.

[0072] 컨트롤러(200)는 또한 베어 메탈 및/또는 서비스 템플릿을 포함할 수 있는 템플릿의 라이브러리(230)를 포함한다. 이들 템플릿은 이메일, 파일 스토리지, VoIP(Voice over IP), 소프트웨어 어카운팅, 소프트웨어 XMPP, 위키(wiki), 버전 제어, 계정 인증 관리 및 사용자 인터페이스에 의해 구성 가능할 수 있는 제3자 애플리케이션을 포함할 수 있지만 이에 한정되지 않는다. 템플릿(230)은 리소스, 애플리케이션, 또는 서비스와 연관될 수 있으며; 또한 이러한 리소스, 애플리케이션, 또는 서비스가 시스템에 통합되는 방법을 정의하는 레시피로서 기능할 수 있다.

[0073] 이와 같이, 템플릿은 리소스, 또는 리소스에 로딩된 애플리케이션 또는 서비스를 생성, 구성, 및/또는 배치하는 데 사용되는 확립된 정보 세트를 포함할 수 있다. 이러한 정보는 커널, initrd 파일, 파일시스템 또는 파일시스템 이미지, 파일, 구성 파일, 구성 파일 템플릿, 다른 하드웨어 및/또는 연산 백엔드에 대한 적절한 설정을 결정하는 데 사용되는 정보, 및/또는 애플리케이션의 생성, 부팅 또는 실행을 허용 및/또는 용이하게 하는 애플리케이션 및 운영 체제 이미지에 전원을 공급하도록 리소스를 구성하기 위한 다른 사용 가능한 옵션을 포함할 수 있지만, 이에 한정되지 않는다.

[0074] 템플릿은 복수의 물리 서버 유형 또는 구성요소, 복수의 하드웨어 유형에서 실행하는 복수의 하이퍼바이저, 복수의 하드웨어 유형에서 호스팅될 수 있는 컨테이너 호스트를 포함하지만 이에 한정되지 않는 복수의 지원되는 하드웨어 유형 및/또는 연산 백엔드에서 애플리케이션을 배치하는 데 사용될 수 있는 정보를 포함할 수 있다.

[0075] 템플릿은 컴퓨팅 리소스에서 실행하는 애플리케이션 또는 서비스에 대한 부팅 이미지를 도출할 수 있다. 템플릿 및 템플릿으로부터 도출된 이미지는 애플리케이션을 생성하고, 애플리케이션 또는 서비스를 배치하며, 및/또는 애플리케이션의 생성을 허용 및/또는 용이하게 하는 다양한 시스템 기능을 위한 리소스를 배열하는 데 사용될 수 있다. 템플릿은 기본 셋팅 또는 컨트롤러로부터 제공된 셋팅으로부터 구성 옵션으로 오버라이트될 수 있는 파일, 파일 시스템, 및/또는 운영 체제 이미지에서의 가변 파라미터를 가질 수 있다. 템플릿은 애플리케이션 또는 다른 리소스를 구성하는 데 사용되는 구성 스크립트를 가질 수 있고, 구성 변수, 구성 규칙, 및/또는 기본 규칙 또는 변수를 사용할 수 있게 하며; 이들 스크립트, 변수, 및/또는 규칙은 특정 하드웨어 또는 다른 리소스 특정 파라미터, 예를 들어 하이퍼바이저(가상시), 사용 가능한 메모리에 대한 특정 규칙, 스크립트, 또는 변수를 포함할 수 있다. 템플릿은 이진 리소스, 이진 리소스 또는 하드웨어 또는 다른 리소스 특정 파라미터를 생성하는 컴파일 가능한 소스 코드, 특정 하드웨어 또는 다른 리소스 특정 파라미터, 예를 들어 하이퍼바이저(가상시), 사용 가능한 메모리에 대한 컴파일 명령어가 있는 이진 리소스 또는 소스 코드의 특정 세트 형태의 파일을 가질 수 있다. 템플릿은 리소스에서 실행하고 있는 것과 무관한 정보 세트를 포함할 수 있다.

[0076] 템플릿은 베이스 이미지를 포함할 수 있다. 베이스 이미지는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 템플릿은 커널을 포함할 수 있다. 커널 또는 복수의 커널은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 initrd 또는 복수의 커널을 포함할 수 있다. 이미지는 하나 이상의 리소스에 로딩되거나 배치된 템플릿 광고(ad)로부터 도출될 수 있다. 로딩된 이미지는 또한 대응하는 템플릿의 커널 또는 initrd와 같은 부팅 파일을 포함할 수 있다.

[0077] 이미지는 템플릿에 기초하여 리소스에 로딩될 수 있는 템플릿 파일시스템 정보를 포함할 수 있다. 템플릿 파일시스템은 애플리케이션 또는 서비스를 구성할 수 있다. 템플릿 파일시스템은, 예를 들어 파일시스템이 저장되는

스토리지 공간을 절약하거나 판독 전용 파일의 사용을 용이하게 하기 위해, 모든 리소스에, 또는 유사한 리소스에 공통적인 공유 파일시스템을 포함할 수 있다. 템플릿 파일 시스템 또는 이미지는 배치되고 있는 서비스에 공통적인 파일 세트를 포함할 수 있다. 템플릿 파일 시스템은 컨트롤러에 프리로딩되거나 다운로딩될 수 있다. 템플릿 파일시스템은 업데이트될 수 있다. 템플릿 파일 시스템은 재구축이 필요하지 않을 수 있으므로 비교적 빠른 배치가 가능할 수 있다. 다른 리소스 또는 애플리케이션과 파일시스템을 공유하는 것은 파일이 불필요하게 중복되지 않으므로 스토리지의 감소를 가능하게 할 수 있다. 이는 또한 템플릿 파일시스템과 다른 파일만이 복구될 필요가 있으므로 장애로부터 더 용이한 복구를 가능하게 할 수 있다.

[0078] 템플릿 부팅 파일은 부팅 프로세스를 지원하는 데 사용되는 커널 및/또는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부팅 파일은 운영 체제를 부팅하고 템플릿 파일 시스템을 설정할 수 있다. initrd는 부팅할 수 있도록 템플릿을 설정하는 방법에 대한 명령어를 갖는 작은 임시 파일시스템을 포함할 수 있다.

[0079] 템플릿은 템플릿 BIOS 셋팅을 추가로 포함할 수 있다. 템플릿 BIOS 셋팅은 물리 호스트에서 애플리케이션을 실행하기 위한 선택적 셋팅을 설정하는 데 사용될 수 있다. 사용되는 경우에는, 본 명세서의 도 1 내지 도 12와 관련하여 설명되는 바와 같이, 대역외 관리(260)는 리소스 또는 애플리케이션을 부팅하는 데 사용될 수 있다. 물리 호스트는 대역외 관리 네트워크(260) 또는 CDROM을 사용하여 리소스 또는 애플리케이션을 부팅할 수 있다. 컨트롤러(200)는 이러한 템플릿에 정의된 애플리케이션 특정 바이오스 셋팅을 설정할 수 있다. 컨트롤러(200)는 대역외 관리 시스템을 사용하여 특정 리소스에 특정한 API를 통해 직접 바이오스 변경을 행할 수 있다. 셋팅은 콘솔 및 이미지 인식을 통해 검증될 수 있다. 따라서, 컨트롤러(200)는 콘솔 특징을 사용하고 가상 키보드 및 마우스로 바이오스 변경을 행할 수 있다. 컨트롤러는 UEFI 쉘을 사용할 수도 있고, 콘솔에 직접 입력할 수도 있으며, 이미지 인식을 사용하여 성공적인 결과를 검증하고 커맨드를 정확하게 입력하고 성공적인 셋팅 변경을 보장할 수도 있다. BIOS 변경 또는 특정 BIOS 버전으로의 업데이트를 위해 사용할 수 있는 부팅 가능한 운영 체제가 있는 경우, 컨트롤러(200)는 디스크 이미지 또는 ISO 부팅을 원격으로 로딩하여 운영 체제가 BIOS를 업데이트하고 신뢰 가능한 방식으로 구성 변경을 허용하는 애플리케이션을 실행할 수 있다.

[0080] 템플릿은 템플릿 특정 지원 리소스의 리스트 또는 특정 애플리케이션 또는 서비스를 실행하는 데 필요한 리소스의 리스트를 추가로 포함할 수 있다.

[0081] 템플릿 이미지 또는 이미지 또는 템플릿의 일부는 컨트롤러(200)에 저장될 수 있거나, 또는 컨트롤러(200)는 이를 스토리지 리소스(410)로 이동 또는 복사할 수 있다.

[0082] 도 2e는 예시적인 템플릿(230)을 나타낸다. 템플릿은 애플리케이션 또는 서비스를 생성하는 데 필요한 모든 정보를 포함한다. 템플릿(230)은 또한 유사하거나 동일한 기능을 제공하는 다른 하드웨어 유형에 대한 정보, 대안 데이터, 파일, 바이너리를 포함할 수 있다. 예를 들어, 다른 아키텍처에 대해 컴파일링된 바이너리(234)를 갖는 /usr/bin 및 /bin에 대한 파일시스템 블롭(232)이 있을 수 있다. 템플릿(230)은 또한 데몬(daemon)(233) 또는 스크립트(231)를 포함할 수 있다. 데몬(233)은 호스트의 전원이 켜지고 준비될 때 부팅시에 실행할 수 있는 바이너리 또는 스크립트이고; 일부 경우에, 데몬(233)은 컨트롤러에 의해 액세스 가능할 수 있고 컨트롤러가 호스트의 셋팅을 변경 가능하게 할 수 있는 API에 전력을 공급할 수 있다(그리고 컨트롤러는 활성 시스템 규칙을 후속적으로 업데이트할 수 있다). 데몬은 또한 전술 및 후술되는 대역외 관리(260) 또는 대역내 관리(270)를 통해 전원이 차단되었다가 재시작될 수 있다. 이들 데몬은 또한 새로운 서비스에 종속 서비스를 제공하기 위해 범용 API(예를 들어, 엔진박스(nginx) 또는 아파치(apache)를 제어하는 API와 통신하는 범용 웹 서버 API)에 전력을 공급할 수 있다. 스크립트(231)는 이미지를 부팅하는 동안 또는 그 후에 또는 데몬을 시작하거나 서비스를 활성화한 후에 실행할 수 있는 설치 스크립트일 수 있다.

[0083] 템플릿(230)은 또한 커널(235) 및 프리부팅 파일시스템(236)을 포함할 수 있다. 템플릿(230)은 또한 상이한 하드웨어 및 상이한 구성을 위해 복수의 커널(235) 및 하나 이상의 프리부팅 파일시스템(예를 들어, Linux용 initrd 또는 initramf 또는 bsd용 판독 전용 램디스크)을 포함할 수 있다. initrd는 오버레이로서 표시되는 파일시스템 블롭(232)을 마운트하고, 후술하는 바와 같이 SAN 연결(280)을 통해 선택적으로 스토리지 리소스에 연결할 수 있는 initramfs(236)로 부팅함으로써 원격 스토리지에 루트 파일시스템을 마운트하는 데 사용될 수 있다.

[0084] 파일시스템 블롭(232)은 별도의 블롭으로 분할될 수 있는 파일시스템 이미지이다. 블롭은 구성 옵션, 하드웨어 유형, 및 다른 설정 차이에 기초하여 교환 가능할 수 있다. 템플릿(230)으로부터 부팅된 호스트는 복수의 블롭 또는 하나 또는 복수의 파일시스템 블롭으로부터 생성된 이미지를 포함하는 통합 파일 시스템(예를 들어, overlayfs)으로부터 부팅될 수 있다.

[0085] 템플릿(230)은 또한 휘발성 데이터(238) 및/또는 구성 파라미터(239)와 같은 추가 정보(237)를 포함하거나 이와 링크될 수 있다. 예를 들어, 휘발성 데이터(238)는 템플릿(230)에 포함될 수 있거나 외부에 포함될 수 있다. 이는 데이터베이스, 플랫 파일, 디렉토리에 저장된 파일, 파일의 타르볼(tarball), 깃(git) 또는 다른 버전 제어 리포지토리(repository)를 포함하지만 이에 한정되지 않는 파일시스템 블롭(232) 또는 다른 데이터스토어의 형태의 것일 수 있다. 또한, 구성 파라미터(239)는 템플릿(230)의 외부 또는 내부에 포함될 수 있고, 선택적으로 시스템 규칙에 포함되고 템플릿(230)에 적용된다.

[0086] 시스템(100)은 리소스를 포함하지만 이에 한정되지 않는 시스템(100)의 상태를 추적, 유지, 변경 및 업데이트하는 IT 시스템 상태(220)를 추가로 포함한다. 시스템 상태(220)는 이용 가능한 리소스을 추적할 수 있으며, 이는 규칙, 및 템플릿을 구현하기 위해 이용 가능한 리소스가 있는지와 어떤 리소스인지를 컨트롤러 로직에 알려 줄 것이다. 시스템 상태는, 업그레이드 또는 다른 이유 때문에, 예를 들어 효율을 개선하기 위해 또는 우선 순위를 위해 전환할 필요가 있는지 여부에 따라, 컨트롤러 로직(205)이 효율을 검사하고 효율을 이용할 수 있게 하는 사용된 리소스를 추적할 수 있다. 시스템 상태는 실행중인 애플리케이션을 추적할 수 있다. 컨트롤러 로직(205)은 시스템 상태에 따라, 그리고 수정할 필요가 있는지 여부에 따라 실행중인 예상 애플리케이션과 실행중인 실제 애플리케이션을 비교할 수 있다. 시스템 상태(220)는 또한 애플리케이션이 실행중인 위치를 추적할 수 있다. 컨트롤러 로직(205)은 효율 평가, 변경 관리, 업데이트, 문제 해결, 또는 감사 추적을 위해 이 정보를 사용할 수 있다. 시스템 상태는 네트워킹 정보, 예를 들어 어떤 네트워크가 가동중 또는 현재 실행중인지 또는 구성값 및 이력을 추적할 수 있다. 시스템 상태(220)는 변경 이력을 추적할 수 있다. 시스템 상태(220)는 또한 어떤 템플릿이 사용되는지를 규정하는 글로벌 시스템 규칙에 기초하여 어떤 템플릿이 어떤 배치에서 사용되는지를 추적할 수 있다. 이력은 감사, 경고, 변경 관리, 보고서 작성, 하드웨어 및 애플리케이션 및 구성, 구성 변수와 상관된 버전 추적에 사용될 수 있다. 시스템 상태(220)는 감사, 컴플라이언스 테스팅 또는 문제 해결을 위해 구성 히스토리를 유지할 수 있다.

[0087] 컨트롤러는 시스템 상태, 템플릿, 및 글로벌 시스템 규칙에 포함된 모든 정보를 관리하기 위한 로직(205)을 갖는다. 컨트롤러 로직(205), 글로벌 시스템 규칙 데이터베이스(210), IT 시스템 상태(220), 및 템플릿(230)은 컨트롤러(200)에 의해 관리되며, 컨트롤러(200)상에 상주하거나 그렇지 않을 수 있다. 컨트롤러 로직 또는 애플리케이션(205), 글로벌 시스템 규칙 데이터베이스(210), IT 시스템 상태(220) 및 템플릿(230)은 물리 또는 가상일 수 있고, 분산 서비스, 분산 데이터베이스, 및/또는 파일일 수 있거나 아닐 수 있다. API 애플리케이션(120)은 컨트롤러 로직/컨트롤러 애플리케이션(205)에 포함될 수 있다.

[0088] 컨트롤러(200)는 독립형 머신를 실행할 수 있고 및/또는 하나 이상의 컨트롤러를 포함할 수 있다. 컨트롤러(200)는 컨트롤러 서비스 또는 애플리케이션을 포함할 수 있고, 다른 머신 내부에서 실행할 수 있다. 컨트롤러 시스템은 전체 스택 또는 스택 그룹의 순서 및/또는 일관된 부팅을 보장하기 위해 컨트롤러 서비스를 시동할 수 있다.

[0089] 컨트롤러(200)는 연산, 스토리지, 및 네트워킹 리소스로 하나 이상의 스택을 제어할 수 있다. 각각의 스택은 글로벌 시스템 규칙(210) 내의 규칙의 상이한 서브세트에 의해 제어되거나 그렇지 않을 수 있다. 예를 들어, 시스템 내에서 다른 기능을 갖는 사전 생산, 생산, 개발, 테스트 스택, 병렬, 백업, 및/또는 다른 스택이 있을 수 있다.

[0090] 컨트롤러 로직(205)은 원하는 IT 시스템 상태를 달성하기 위해 글로벌 시스템 규칙을 판독 및 해석하도록 구성될 수 있다. 컨트롤러 로직(205)은 애플리케이션 또는 서비스와 같은 시스템 구성요소를 구축하기 위해 글로벌 규칙에 따라 템플릿을 사용하고, 원하는 IT 시스템 상태를 달성하기 위해 리소스를 할당, 추가, 또는 제거하도록 구성될 수 있다. 컨트롤러 로직(205)은 글로벌 시스템 규칙이 정확한 상태에 도달하기 위한 태스크의 리스트를 개발하고 이용 가능한 동작에 기초하여 규칙을 이행하기 위한 명령어를 발행하는 것을 판독할 수 있다. 컨트롤러 로직(205)은 동작, 예를 들어 시스템 시동, 리소스 추가, 제거, 재구성; 수행 가능한 일의 식별을 실행하기 위한 로직을 포함할 수 있다. 컨트롤러 로직은 시동 시간과 정기적인 간격으로 시스템 상태를 검사하여 하드웨어가 사용 가능한지 그리고 사용 가능한 경우에 작업을 실행할 수 있는지를 알 수 있다. 필요한 하드웨어가 이용 가능하지 않은 경우, 컨트롤러 로직(205)은 시스템 상태(230)로부터 글로벌 시스템 규칙(210), 템플릿(220) 및 이용 가능한 하드웨어를 사용하여 대안적인 옵션을 제시하고, 이에 따라 글로벌 규칙 및/또는 시스템 상태(220)를 수정한다.

[0091] 컨트롤러 로직(205)은 요구되는 변수, 사용자가 계속 입력해야 하는 것 또는 사용자가 시스템에서 기능해야 하는 것을 알 수 있다. 컨트롤러 로직은 글로벌 시스템 규칙으로부터 템플릿의 리스트를 사용하고 시스템 상태에

필요한 템플릿과 비교하여 필요한 템플릿을 사용 가능하게 할 수 있다. 컨트롤러 로직(205)은, 템플릿 특정 지원 리소스의 리스트 상의 리소스가 이용 가능한 경우, 시스템 상태 데이터베이스로부터 식별할 수 있다. 컨트롤러 로직은 리소스를 할당하고, 상태를 업데이트하며 다음 작업 세트로 진행하여 글로벌 규칙을 구현할 수 있다. 컨트롤러 로직(205)은 글로벌 규칙에 지정된 바와 같이 할당된 리소스에서 애플리케이션을 시작/실행할 수 있다. 규칙은 템플릿으로부터 애플리케이션을 구축하는 방법을 지정할 수 있다. 컨트롤러 로직(205)은 템플릿(들)을 취득하고 변수로부터 애플리케이션을 구성할 수 있다. 템플릿은 어떤 커널, 부팅 파일, 파일시스템 및 지원되는 하드웨어 리소스가 필요한지 컨트롤러 로직(205)에 알릴 수 있다. 그 후, 컨트롤러 로직(205)은 애플리케이션 배치에 관한 정보를 시스템 상태 데이터베이스에 추가할 수 있다. 각각의 명령 후에, 컨틀롤러 로직(205)은 시스템 상태 데이터베이스와 글로벌 규칙의 예상 상태를 검사하여 예상 동작이 정확하게 완료되었는지를 검증할 수 있다.

[0092] 컨트롤러 로직(205)은 버전 규칙에 따라 버전을 사용할 수 있다. 시스템 상태(220)는 어떠는 규칙 버전이 상이한 배치에 사용되었는지를 상관시키는 데이터베이스를 가질 수 있다.

[0093] 컨트롤러 로직(205)은 최적화 및 효율적인 순서를 규정하기 위한 효율적인 로직을 포함할 수 있다. 컨트롤러 로직(205)은 리소스를 최적화하도록 구성될 수 있다. 실행중이거나 실행중일 것으로 예상되는 애플리케이션에 관한 시스템 상태, 규칙 및 템플릿의 정보는 리소스에 대한 효율성 또는 우선순위를 구현하기 위해 컨트롤러 로직에 의해 사용될 수 있다. 컨트롤러 로직(205)은 시스템 상태(220)에서 "사용된 리소스"의 정보를 사용하여 효율, 또는 업그레이드, 용도 변경 또는 다른 이유로 리소스를 전환할 필요성을 결정한다.

[0094] 컨트롤러는 시스템 상태(220)에 따라 실행중인 애플리케이션을 검사하고 글로벌 규칙의 실행중인 예상 애플리케이션과 비교할 수 있다. 애플리케이션이 실행중이지 않은 경우에는 그것을 시작할 수 있다. 애플리케이션이 실행하고 있지 않아야 하는 경우에는 그것을 중지하고 필요한 경우에 리소스를 재할당할 수 있다. 컨트롤러 로직(205)은 리소스(연산, 스토리지, 네트워킹) 사양의 데이터베이스를 포함할 수 있다. 컨트롤러 로직은 사용될 수 있는 시스템에 이용 가능한 리소스 유형을 인식하기 위한 로직을 포함할 수 있다. 이는 대역외 관리 네트워크(260)를 사용하여 수행될 수 있다. 컨트롤러 로직(205)은 대역외 관리(260)를 사용하여 새로운 하드웨어를 인식하도록 구성될 수 있다. 컨트롤러 로직(205)은 또한 감사, 보고서 작성 및 변경 관리를 위해 변경 이력, 사용된 규칙 및 버전에 대한 시스템 상태(220)로부터의 정보를 취할 수 있다.

[0095] 도 2f는 이 예의 목적을 위해 호스트라고 지칭될 수 있는, 리소스를 부팅, 전원 공급 및/또는 활성화하기 위해 템플릿(230)을 프로세싱하는 것 및 이미지를 도출하는 것에 관한 컨트롤러 로직(205)의 예시적인 프로세스 흐름을 나타낸다. 이 프로세스는 또한 스토리지 리소스의 구성 및 스토리지 및 연산 호스트 및/또는 리소스의 결합을 포함할 수 있다. 컨트롤러 로직(205)은 시스템(100)에서 이용 가능한 하드웨어 리소스를 알고 있고, 시스템 규칙(210)은 어느 하드웨어 리솟가 이용될 수 있는지를 표시할 수 있다. 컨트롤러 로직(205)은, 단계 205.1에서, 컨트롤러 로직이 도 2e에 의해 나타낸 템플릿(230) 외부에 있는 파일을 수집하도록 실행될 수 있는 명령어 파일을 포함할 수 있는 템플릿(230)을 파싱한다. 명령어 파일은 json 형식일 수 있다. 단계 205.2에서, 컨트롤러 로직은 필요한 파일 버킷의 리스트를 수집한다. 그리고, 단계 205.3에서, 컨트롤러 로직(205)은 필요한 하드웨어 특정 파일을 하드웨어에 의해 그리고 선택적으로 하이퍼바이저(또는 컨테이너 호스트 시스템, 멀티테넌시 유형)에 의해 참조되는 버킷 내에 수집한다. 하이퍼바이저(또는 컨테이너 호스트 시스템 또는 멀티테넌시 유형) 참조는 하드웨어가 가상 머신에서 실행되는 경우에 필요할 수 있다.

[0096] 하드웨어 특정 파일이 있는 경우, 컨트롤러 로직은 단계 205.4에서 하드웨어 특정 파일을 수집할 것이다. 일부 경우에, 파일 시스템 이미지는 커널 모듈(또는 결국 디렉토리 내에 배치되는 커널 모듈)을 포함하는 디렉토리와 함께 커널 및 initramfs를 포함할 수 있다. 그 후, 컨트롤러 로직(205)은 단계 205.5에서 호환 가능한 적절한 베이스 이미지를 선택한다. 베이스 이미지는 템플릿(230)으로부터 도출되는 애플리케이션 또는 이미지에 특정하지 않을 수 있는 운영 체제 파일을 포함한다. 이 맥락에서의 호환성은 베이스 이미지가 템플릿을 작동중인 애플리케이션으로 전환하는 데 필요한 파일을 포함하고 있음을 의미하다. 베이스 이미지는 공간을 절약하기 위한 메커니즘으로서 템플릿 외부에서 관리될 수 있다(그리고 종종 베이스 이미지는 수개의 애플리케이션 또는 서비스에 대해 동일할 수 있다). 또한, 단계 205.6에서, 컨트롤러 로직(205)은 실행 파일, 소스 코드, 및 하드웨어 특정 구성 파일을 갖는 버킷(들)을 선택한다. 템플릿(230)은, 구성 파일, 구성 파일 템플릿(컨트롤러(200)가 구성 템플릿을 구성 파일로 전환할 수 있고 선택적으로 API 엔드포인트를 통해 구성 파일을 변경할 수 있도록 템플릿(230)에 알려질 수 있는 시스템 규칙(210) 내에 변수로 채워지는 플레이스홀더 또는 변수를 포함하는 구성 파일임), 바이너리, 및 소스 코드(이미지가 부팅될 때에 준수될 수 있음)를 포함하지만 이에 한정되지 않는 다른 파일을 참조할 수 있다. 단계 205.7에서, 단계 205.4., 205.5, 및 205.6에서 선택된 요소에 대응하는 하드웨어 특

정 명령어가 부팅되는 이미지의 일부로서 로딩될 수 있다. 컨트롤러 로직(205)은 선택된 구성요소로부터 이미지를 도출한다. 예를 들어, 물리 호스트와 가상 머신에 대해 상이한 사전 설치(preinstall) 스크립트가 있거나, powerpc와 x86에 대해 차이가 있을 수 있다.

[0097] 단계 205.8에서, 컨트롤러 로직(205)은 overlayfs를 마운트하고 대상 파일을 단일 파일 시스템 블롭으로 리패키징한다. 다수의 파일 시스템 블롭이 사용될 때, 이미지는 타르볼의 압축 해제하고 및/또는 git을 패칭하는 다수의 블롭으로 생성될 수 있다. 단계 205.8이 수행되지 않는 경우, 파일시스템 블롭이 분리된 상태로 유지될 수 있고, 이미지가 파일시스템 블롭 세트로서 생성되고 다수의 작은 파일시스템을 함께 마운트할 수 있는 파일 시스템(예를 들어, overlayfs)으로 마운트된다. 그 후, 컨트롤러 로직(205)은 단계 205.9에서 호환 가능한 커널(또는 시스템 규칙(210)에 지정된 커널)을 탐색하고 단계 205.10에서 적용 가능한 initrd를 탐색할 수 있다. 호환 가능한 커널은 템플릿의 종속성과 템플릿을 구현하는 데 사용되는 리소스를 만족시키는 커널일 수 있다. 호환 가능한 initrd는 원하는 연산 리소스에 템플릿을 로딩하는 initrd일 수 있다. 종종, initrd는 (루트 파일시스템이 원격일 수 있으므로) 완전히 부팅하기 전에 스토리지 리소스를 마운트할 수 있도록 물리 리소스에 사용될 수 있다. 커널 및 initrd는 파일시스템 블롭으로 패키징되거나, 직접 커널 부팅에 사용되거나, 예비 운영 체제를 부팅한 후에 실제 시스템에서 커널을 변경하기 위해 kexec를 사용하는 물리 호스트에서 사용될 수 있다.

[0098] 그 후, 컨트롤러는 연산 리소스(들)가 205.11, 205.12, 및/또는 205.13에 의해 나타낸 임의의 기술을 사용하여 애플리케이션(들) 및/또는 이미지(들)에 전력을 공급할 수 있도록 스토리지 리소스(들)를 구성한다. 205.11에 의해, overlayfs 파일이 스토리지 리소스로서 제공될 수 있다. 205.12에 의해, 파일 시스템이 제시된다. 예를 들어, 스토리지 리소스는 연산 리소스가 overlayfs와 유사한 파일시스템을 사용하여 동시에 마운트할 수 있는 조합된 파일시스템 또는 다수의 파일시스템 블롭을 제시할 수 있다. 205.13에 의해, 블롭이 파일시스템을 제시하기 전에 스토리지 리소스에 전송된다.

[0099] 도 2g 및 도 2h는 도 2f의 단계 205.11 및 205.12에 대한 예시적인 프로세스 흐름을 나타낸다. 또한, 시스템은 스토리지 연결 프로세스라고 지칭될 수 있는, 컴퓨터 리소스를 스토리지 리소스에 연결하기 위한 프로세스 및 규칙을 사용할 수 있다. 도 2g 및 도 2h에 나타낸 것 이외에 이러한 스토리지 연결 프로세스의 일례가 본 명세서에 동봉된 부록 A에 제공되어 있다. 도 2g는 스토리지 리소스의 연결을 위한 예시적인 프로세스를 나타낸다. 일부 스토리지 리소스는 판독 전용일 수 있고, 다른 것은 기록 가능할 수 있다. 스토리지 리소스는 자신의 기록 잠금을 관리하여 경쟁 조건을 야기하는 동시 기록이 없도록 하거나 시스템 상태(220)가 어떤 연결이 스토리지 리소스에 기록할 수 있는지 추적할 수 있고(예를 들어, 단계 205.20 참조) 및/또는 고 및/또는 리소스에 대한 다수의 판독-기록 연결을 방지할 수 있다(단계 205.21). 컨트롤러 로직 또는 리소스 자체는 스토리지 리소스의 위치 및 전송 유형(예를 들어, iscsi, iser, nvmeof, 파이버 채널, fcoe, nfs, nfs over rdma, afs, cifs, 윈도우 공유(windows share))에 대해 컨트롤러의 시스템 상태(220)를 질의할 수 있다(단계 205.22). 연산리 소스가 가상인 경우, 하이퍼바이저는 (예를 들어, 하이퍼바이저 데몬을 통해) 스토리지 리소스에 대한 연결을 처리할 수 있다(단계 205.23). 이는 가상 머신이 SAN 280에 대한 지식이 없을 수 있으므로 바람직한 보안 이점을 가질 수 있다.

[0100] 단계 205.24를 참조하면, 연산 리소스 및 스토리지 리소스를 연결하기 위한 프로세스가 시스템 규칙(210)에서 지시될 수 있다. 그 후, 컨트롤러 로직은 시스템 상태(220)를 질의하여 리소스가 이용 가능하고 필요한 경우에 기록 가능한지 확인한다(단계 205.22). 시스템 상태(220)는 SQL 쿼리(또는 다른 유형의 데이터베이스 쿼리), JSON 파싱 등과 같은 다수의 기술 중 어느 것을 통해 질의될 수 있다. 쿼리는 연산 리소스가 스토리지 리소스에 연결하는 데 필요한 정보를 리턴할 것이다. 컨트롤러(200), 시스템 상태(220), 또는 시스템 규칙(210)은 연산 리소스가 시스템 상태에 연결하기 위한 인증 자격증명(authentication credential)을 제공할 수 있다(단계 205.25). 그 후, 연산 리소스가 시스템 상태(220)를 직접 또는 컨트롤러를 통해 업데이트할 것이다(단계 205.26).

[0101] 도 2h는 물리, 가상, 또는 다른 유형의 연산 리소스, 애플리케이션, 서비스, 또는 호스트의 전원을 켜고 스토리지 리소스에 연결하는 부팅 프로세스의 일례를 도시한다. 스토리지 리소스는 선택적으로 퓨전 파일시스템 및/또는 확장 가능한 볼륨을 사용할 수 있게 한다. 컨트롤러 또는 다른 시스템이 물리 호스트를 활성화하는 상황에서, 물리 호스트에는 시스템을 구성하기 위한 운영 체제가 프리로딩될 수 있다. 따라서, 단계 205.31에서, 컨트롤러는 initramfs으로 부트 디스크를 프리로딩할 수 있다. 또한, 컨트롤러(200)는 대역외 관리 연결(260)을 사용하여 예비 운영 체제를 네트워크 부팅하고(단계 205.30), 그 후에 선택적으로 예비 운영 시스템으로 호스트를 프리로딩할 수 있다(단계 205.31). 그 후, initramfs가 단계 205.32에서 로딩되고, 스토리지 리소스가 도 2g에 나타낸 방법을 사용하여 단계 205.33에서 연결된다. 그 후, 확장 가능한 볼륨이 있는 경우,

함께 결합된 서브볼륨 또는 디바이스가 로직 볼륨 관리(LVM)가 사용중인 경우에 단계 205.34에서 볼륨 그룹으로서 선택적으로 조립된다. 또는, 이들은 디스크를 조합하는 다른 방법을 사용하여 단계 205.34에서 결합될 수 있다.

[0102] 퓨전 파일시스템이 사용중인 경우, 파일은 단계 205.36에서 조합될 수 있고, 그 후에 부팅 프로세스가 계속된다(단계 205.46). 일부 알려진 이슈를 수정하기 위해 Linux에서 overlayfs가 사용중인 경우, 다음의 서브프로세스가 실행할 수 있다. /data 디렉토리가 휘발성일 수 있는 각각의 마운트된 파일시스템 블롭에서 만들어질 수 있다(단계 205.37). 그 후, 단계 205.38에서 new_root 디렉토리가 생성될 수 있고, 단계 205.39에서 overlayfs가 디렉토리에 마운트된다. 그 후, initramfs가 /new_root에서 exec_root를 실행한다(단계 205.40).

[0103] 호스트가 가상 머신인 경우 직접 커널 부팅과 같은 추가 툴이 사용 가능할 수 있다. 이 상황에서, 하이퍼바이저는 VM을 부팅하기 전에 스토리지 리소스에 연결할 수 있거나(단계 205.41), 부팅하는 동안에 이를 수행할 수 있다. 그 후, VM이 initramfs를 로딩함과 함께 직접 커널 부팅될 수 있다(단계 205.42). 그 후, 단계 205.43에서 initramfs가 로딩되고, 하이퍼바이저가 이 시점에서 원격일 수 있는 스토리지 리소스에 연결될 수 있다(단계 205.44). 이를 달성하기 위해서, 하이퍼바이저 호스트는 인터페이스를 통과할 필요가 있을 수 있다(예를 들어, inifiniband가 iSER 타깃에 연결하는 데 필요한 경우에는, pci-passhtru를 사용하여 SR-IOV 기반 가상 기능을 통과할 수 있거나 일부 상황에서는 반가상화된(paravirtualized) 네트워크 인터페이스를 사용할 수 있다). 이들 연결은 initramfs에 의해 사용 가능하다. 그 후, 가상 머신은 아직 연결되지 않은 경우에 단계 205.45에서 스토리지 리소스에 연결할 수 있다. 또한, 가상 머신은 하이퍼바이저를 통해(선택적으로 반가상화된 스토리지를 통해) 그의 스토리지 리소스를 수신할 수도 있다. 프로세스는 퓨전 파일 시스템 및 LVM 스타일 디스크를 선택적으로 마운트하고 있는 가상 머신의 경우와 유사할 수 있다.

[0104] 도 2o는 205.13과 같이 파일시스템 블롭 또는 파일의 다른 그룹으로부터 스토리지 리소스를 구성하기 위한 예시적인 프로세스 흐름을 도시한다. 블롭은 단계 205.75에서 수집되고; 205.73에서 스토리지 리소스 호스트에 직접 복사될 수 있다(스토리지 리소스 호스트가 파일 시스템 블롭(232)을 보유하는 디바이스와 다른 경우). 스토리지 리소스가 제 위치에 있으면, 시스템 상태는 205.74에서 스토리지 리소스의 위치 및 사용 가능한 전송(예를 들어, iSER, nvmeof, iSCSI, FcoE, 파이버 채널, nfs, nfs over rdma)으로 업데이트된다. 이들 블롭 중 일부는 판독 전용일 수 있으며, 이 경우에 시스템 상태는 동일하게 유지되며, 새로운 연산 리소스 또는 호스트는 해당 판독 전용 스토리지 리소스에 연결할 수 있다(예를 들어, 베이스 이미지에 연결할 때). 일부 경우에는 205.70롤 나타낸 바와 같이 파일을 단일 파일시스템 이미지에 배치하여 임의의 퓨젼 파일시스템 오버헤드를 피하는 것이 바람직할 수 있다. 이는 블롭을 퓨전 파일시스템으로서 마운트한 후(단계 205.71) 새로운 파일시스템으로 복사하거나 또는 단일 파일 시스템으로서 리패키징한 후(단계 205.72) 선택적으로 새로운 파일시스템 이미지를 스토리지 리소스로서 제시될 새로운 파일시스템 이미지를 위한 적절한 위치에 복사함으로써 달성될 수 있다. 일부 퓨전 파일시스템은 병합이 단계 205.71에서 먼저 마운트하지 않고 달성되고 단일 단계에서 병합하는 것을 가능하게 할 수 있다.

[0105] 도 2i는 도 2e에 나타낸 다른 예시적인 템플릿(230)을 도시한다. 이 예에서, 컨트롤러는 중개 구성 툴와 함께 도 2i에 나타낸 템플릿(230)을 사용하도록 구성될 수 있다. 예시적인 실시형태에 따르면, 중개 구성 툴은 새로운 애플리케이션 또는 서비스를 종속성 애플리케이션 또는 서비스와 결합하는 데 사용되는 공통 API를 포함할 수 있다. 따라서, 템플릿(230)은 템플릿의 서비스를 설정하는 데 필요할 수 있는 종속성(244)의 리스트를 추가로 포함할 수 있다. 템플릿(230)은 또한 종속성의 공통 API에 대한 호출을 포함할 수 있는 연결 규칙(245)을 포함할 수 있다. 템플릿(230)은 또한 하나 또는 복수의 공통 API(243) 및 공통 API 및 버전(242)의 리스트를 포함할 수 있다. 공통 API(243)는 애플리케이션 또는 컨트롤러로부터 호출 가능(또는 불가능)할 수 있는 방법, 기능, 스크립트, 또는 명령어를 가질 수 있으며, 이는 컨트롤러가 종속성 애플리케이션 또는 서비스를 구성할 수 있게 하여, 종속성 애플리케이션 또는 서비스가 템플릿(230)에 의해 구축되는 새로운 애플리케이션에 결합될 수 있도록 한다. 컨트롤러는 공통 API(243)와 통신하고 및/또는 새로운 서비스 또는 애플리케이션과 종속성 서비스 또는 애플리케이션의 결합을 구성하도록 API 호출을 행할 수 있다. 대안적으로, 명령어는 애플리케이션 또는 서비스가 종속성 애플리케이션 또는 서비스에 대해 직접 공통 API(243)와 통신하고 및/또는 그에 호출을 전송 가능하게 할 수 있다. 템플릿(230) 연결 규칙(245)은 새로운 서비스 또는 애플리케이션을 종속성 서비스 또는 애플리케이션과 연결하는 것에 대한 API 호출을 포함할 수 있는 규칙 및/또는 명령어 세트이다.

[0106] 시스템 상태(220)는 실행중인 서비스(246)의 리스트를 추가로 포함할 수 있다. 실행중인 서비스(246)의 리스트는 템플릿(230)으로부터의 종속성(244)을 만족시키려고 컨트롤러 로직(205)에 의해 질의될 수 있다. 컨트롤러는 또한 특정 서비스/애플리케이션 또는 서비스/애플리케이션 유형에 이용 가능한 상이한 공통 API의 리스트(247)

를 포함할 수 있고, 또한 공통 API를 포함하는 템플릿을 포함할 수 있다. 리스트는 컨트롤러 로직(205), 시스템 규칙(210), 시스템 상태(220) 또는 컨트롤러가 액세스할 수 있는 템플릿 스토리지에 상주할 수 있다. 컨트롤러는 또한 모든 기존 또는 로딩된 템플릿으로부터 컴파일된 공통 API(248)의 인덱스를 유지한다.

[0107] 도 2j는 도 2f에 의해 나타냈지만 컨트롤러가 서비스 종속성을 관리하는 단계 255에 의해 템플림(230)을 처리하는 것에 관한 컨트롤러 로직(205)에 대한 예시적인 프로세스 흐름을 도시한다. 도 2k는 도 2j의 단계 255에 대한 예시적인 프로세스 흐름을 나타낸다. 단계 255.1에서, 컨트롤러는 템플릿으로부터 종속성(244)의 리스트를 수집한다. 컨트롤러는 또한 템플릿으로부터 공통 API(243)의 리스트를 수집한다. (A). 단계 255.2에서, 컨트롤러는 템플릿으로부터의 공통 API(243)의 리스트와 공통 API(248)의 인덱스를 비교함으로써, 또한 종속성을 만족시키려고 찾은 애플리케이션 또는 서비스의 유형에 기초하여, 가능한 종속성 애플리케이션 또는 서비스의 리스트를 좁힌다. 단계 255.3에서, 컨트롤러는 시스템 규칙(210)이 종속성을 만족시키는 방법을 지정하는지를 결정한다.

[0108] 단계 255.3에서 "예(yes)"일 경우에는, 컨트롤러는 종속성 서비스 또는 애플리케이션이 실행중인 템플릿의 리스트를 질의함으로써 실행중인지를 결정한다(단계 255.4). 단계 255.4에서 "아니오(no)"일 경우, 서비스 애플리케이션이 실행되고(그리고/또는 설정되고 나서 실행되고), 이는 컨트롤러 로직이 종속성 서비스/애플리케이션의 템플릿을 프로세싱하는 것을 포함할 수 있다(단계 255.5). 종속성 서비스 또는 애플리케이션이 단계 255.4에서 실행중인 것으로 판명되는 경우에는, 프로세스 흐름이 단계 255.6으로 진행한다. 단계 255.6에서, 컨트롤러는 템플릿을 사용하여 종속성 서비스 또는 애플리케이션에 구축되는 새로운 서비스 또는 애플리케이션을 결합한다. 새로운 서비스 또는 애플리케이션과 종속성 애플리케이션/서비스를 결합함에 있어서, 컨트롤러는 프로세싱하고 있는 템플릿을 검토하고, 연결 규칙(245)을 실행할 것이다. 컨트롤러는 종속성(244)을 만족시키고 및/또는 애플리케이션/서비스를 결합하는 방법에 대해 연결 규칙(245)에 기초하여 공통 API(243)에 커맨드를 전송한다. 공통 API(243)는, 서비스의 API 기능을 호출하는 것, 구성을 변경하는 것, 스크립트를 실행하는 것, 다른 프로그램을 호출하는 것을 포함할 수 있지만 이에 한정되지 않는 종속성 애플리케이션 또는 서비스와 새로운 서비스 또는 애플리케이션을 연결하기 위해 컨트롤러부터의 명령어를 번역한다. 단계 255.6에 이어서, 프로세스 흐름은 도 2j의 단계 205.2로 진행한다.

[0109] 단계 255.3이 시스템 규칙(210)이 종속성을 만족시키는 방법을 지정하지 못한다는 결정을 초래하는 경우, 컨트롤러는 적절한 종속성 애플리케이션 또는 서비스가 실행중인지를 알기 위해 단계 255.7에서 시스템 상태(220)를 질의할 것이다. 단계 255.8에서, 컨트롤러는 적절한 종속성 애플리케이션 또는 서비스가 실행중인지에대한 쿼리에 기초하여 그 결정을 내린다. 단계 255.8에서 "아니오(no)"이면, 컨트롤러는 관리자 또는 사용자에게 조치를 통지한다(단계 255.9). 단계 255.8에서 "예(yes)"이면, 프로세스 흐름은 전술한 바와 같이 동작할 수 있는 단계 255.6으로 진행한다. 사용자는 새로운 애플리케이션이 실행중인 종속성 애플리케이션 연결되어야 하는지에 대해 선택적으로 질의받을 수 있고, 이 경우애 컨트롤러는 새로운 애플리케이션 또는 서비스를 단계 255.6에서 다음과 같이 종속성 애플리케이션 또는 서비스에 결합할 수 있다: 컨트롤러는 그것이 프로세싱하고 있는 템플릿(230)을 검토하고, 연결 규칙(245)을 실행할 것이다. 그 후, 컨트롤러는 종속성(244)을 만족시키는 방법에 대해 연결 규칙(245)에 기초하여 공통 API(243)에 커맨드를 전송한다. 공통 API(243)는 컨트롤러로부터의 명령어를 번역하여 새로운 서비스 또는 애플리케이션과 종속성 애플리케이션 또는 서비스를 연결한다.

[0110] 외부 사용자 인터페이스 또는 웹 UI를 통해 사용자, 또는 애플리케이션은 컨트롤러 애플리케이션 또는 로직(205)에 포함될 수도 있는 API 애플리케이션(120)을 통해 컨트롤러(200)와 통신한다.

[0111] 컨트롤러(200)는, 복수의 네트워크, 상호 연결, 또는 컨트롤러가 연산, 스토리지 및 네트워킹 리소스에 작동을 지시할 수 있는 다른 연결 중 하나 이상을 통해, 스택 또는 리소스와 통신한다. 이러한 연결은, 대역외 관리 연결(260); 대역내 관리 연결(270); SAN 연결(280), 및 선택적인 네트워크상 대역내 관리 연결(290)을 포함할 수 있다.

[0112] 대역외 관리는 컨트롤러(200)를 통해 시스템(100)의 구성요소를 검출, 구성, 관리하도록 컨트롤러(200)에 의해 사용될 수 있다. 대역외 관리 연결(260)은, 컨트롤러(200)가 플러그 인되고 이용 가능하지만, 켜져 있지 않은 리소스를 검출하는 것을 가능하게 할 수 있다. 플러그 인되었을 때 리소스는 IT 시스템 상태(220)에 추가될 수 있다. 대역외 관리는 부트 이미지를 로딩하고, 시스템(100)에 속하는리소스를 구성하고, 모니터링하도록 구성될 수 있다. 대역외 관리는 또한 운영 체제의 진단을 위해 임시 이미지를 부팅할 수 있다. 대역외 관리는 BIOS 셋팅을 변경하는데 사용될 수 있고, 실행중인 운영 체제에서 커맨드를 실행하기 위해 콘솔 툴을 또한 사용할 수 있다. 셋팅은 또한 TA 콘솔, 키보드, 및 VGA, DVI 또는 HDMI 포트와 같은 하드웨어 리소스 상의 물리 또는 가상

모니터 포트로부터의 비디오 신호의 이미지 인식을 사용하여, 및/또는 대역외 관리에 의해 제공되는 API, 예를 들어 레드피쉬(Redfish)를 사용하여 컨트롤러에 의해 변경될 수 있다.

[0113] 본 명세서에 사용된 대역외 관리는 운영 체제 및 메인 마더보드에 무과한 리소스 또는 노드에 연결할 수 있는 관리 시스템을 포함할 수 있지만, 이에 한정되지 않는다. 대역외 관리 연결(260)은 네트워크 또는 복수 유형의 직접 또는 간접 연결 또는 상호 연결을 포함할 수 있다. 대역외 관리 연결의 예는 IPMI, 레드피쉬, SSH, 텔넷, 다른 관리 툴, 키보드 비디오 및 마우스(KVM) 또는 KVM over IP, 시리얼 콘솔, 또는 USB를 포함하지만 이에 한정되지 않는다. 대역외 관리는, 네트워크 상에서 사용될 수 있고, 노드 또는 리소스의 전원을 켜고 끌 수 있으며, 온도 및 다른 시스템 데이터를 모니터링할 수 있고; 운영 체제의 제어 외부에 있을 수 있는 BIOS 및 다른 저수준 변경을 행할 수 있으며; 콘솔에 연결하고 커맨드를 전송할 수 있고; 키보드, 마우스, 모니터를 포함하지만 이에 한정되지 않는 입력을 제어할 수 있는 툴이다. 대역외 관리는 물리 리소스의 대역외 관리 회로에 결합될 수 있다. 대역외 관리는 부팅 설치 매체에 사용될 수 있는 디스크로서 디스크 이미지를 연결할 수 있다.

[0114] 관리 네트워크 또는 대역내 관리 연결(270)은 컨트롤러가 리소스가 실행중인 운영 체제와 직접 통신하는 연산, 스토리지, 네트워킹 또는 다른 리소스에 대한 정보를 수집하는 것을 가능하게 할 수 있다. 스토리지 리소스, 연산 리소스 또는 네트워킹 리소스는 연결(260 및/또는 270)과 인터페이스하는 관리 인터페이스를 포함할 수 있어, 컨트롤러(200)와 통신하고, 컨트롤러)에게 실행중인 것과 리소스에 이용 가능한 것을 알려 주고 컨트롤러로부터 커맨드를 수신할 수 있다. 본 명세서에 사용되는 대역내 관리 네트워크는 리소스의 운영 체제에 직접적으로 리소스와 통신할 수 있는 관리 네트워크를 포함한다. 대역내 관리 연결의 예는 SSH, 텔넷, 다른 관리 툴, 시리얼 콘솔, 또는 USB를 포함할 수 있지만 이에 한정되지 않는다.

[0115] 대역외 관리는 본 명세서에서 대역내 관리 네트워크로부터 물리 또는 가상으로 분리된 네트워크로서 설명되지만, 이들은 본 명세서에 더욱 상세하게 설명되는 바와 같이 효율성의 목적에 따라 결합되거나 서로 협업할 수 있다. 따라서 대역외 및 대역내 관리 또는 그의 양태는 컨트롤러의 동일한 포트를 통해 통신하거나 조합된 상호 연결과 결합될 수 있다. 선택적으로, 연결(260, 270, 280, 290) 중 하나 이상은 이러한 네트워크의 다른 것과 분리 또는 조합될 수 있고, 동일한 패브릭을 포함하거나 그렇지 않을 수 있다.

[0116] 또한, 연산 리소스, 스토리지 리소스, 및 컨트롤러는 컨트롤러(200)가 스토리지 네트워크를 각 리소스를 부팅하는 데 사용할 수 있는 방식으로 스토리지 네트워크(SAN)(280)에 결합되거나 그렇지 않을 수 있다. 컨트롤러(200)는 부트 이미지 또는 다른 템플릿을 별도의 스토리지 또는 다른 리소스 또는 다른 리소스에 전송할 수 있어 다른 리소스가 스토리지 또는 다른 리소스를 부트 오프할 수 있게 한다. 컨트롤러는 이러한 상황에서 어디로부터 부팅해야 할지를 지시한다. 컨트롤러는 리소스의 전원을 켜고 리소스에 어디로부터 부팅할지와 자체적으로 구성하는 방법을 지시한다. 컨트롤러(200)는 리소스에 부팅하는 방법, 사용할 이미지, 이미지가 다른 리소스에 있다면 이미지의 배치되는 위치를 지시한다. BIOS의 리소스는 사전에 구성될 수 있다. 컨트롤러는 추가적으로 또는 대안적으로 대역외 관리를 통해 BIOS를 구성할 수 있어 스토리지 에어리어 네트워크를 부트 오프할 것이다. 컨트롤러는(200)는 또한 ISO로부터 운영 체제를 부팅하고 리소스가 로컬 디스크에 데이터를 복사할 수 있도록 구성될 수 있다. 그 후, 로컬 디스크는 부팅에 사용될 수 있다. 컨트롤러는 리소스가 부팅할 수 있는 방식으로 다른 컨트롤러를 포함한 다른 리소스를 구성할 수 있다. 일부 리소스는 연산, 스토리지, 또는 네트워킹 기능을 제공하는 애플리케이션을 포함할 수 있다. 또한, 컨트롤러는 스토리지 리소스를 부트 업(boot up)하여 스토리지 리소스가 후속하는 리소스 또는 서비스의 부트 이미지를 공급할 수 있게 하는 것이 가능하다. 스토리지는 또한 다른 목적으로 사용되는 다른 네트워크를 통해 관리될 수 있다.

[0117] 선택적으로, 리소스 중 하나 이상은 네트워크상 대역내 관리 연결(290)에 결합될 수 있다. 연결(290)은 대역내 관리 연결(270)과 관련하여 설명된 하나 이상의 유형의 대역내 관리를 포함한다. 연결(290)은 컨트롤러를 애플리케이션 네트워크에 연결하여 네트워크를 사용하거나, 대역내 관리 네트워크를 통해 그들을 관리할 수 있다.

[0118] 도 21은 리소스 또는 리소스에 로딩된 애플리케이션 또는 서비스를 부팅하기 위해 (다른 리소스 또는 데이터베이스를 통해) 템플릿(230)으로부터 리소스로 직접 또는 간접적으로 로딩될 수 있는 이미지(250)를 도시한다. 이미지(250)는 리소스 유형 및 하드웨어에 대한 부트 파일(240)을 포함할 수 있다. 부트 파일(240)은 배치될 리소스, 애플리케이션 또는 서비스에 대응하는 커널(241)을 포함할 수 있다. 부트 파일(240)은 또한 부팅 프로세스를 지원하는 데 사용되는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부트 시스템(240)은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 복수의 커널 또는 initrd를 포함할 수 있다. 또한, 이미지(250)는 파일시스템(251)을 포함할 수 있다. 파일 시스템(251)은 베이스 이미지(252) 및 대응하는 파일시스템뿐만 아니라 서비스 이미지(253) 및 대응하는 파일시스템 및 휘발성 이미지(254) 및 대응하는 파일시스템을 포함할 수 있다.

로딩되는 파일 시스템 및 데이터는 리소스 유형과 실행할 애플리케이션 또는 서비스에 따라 다를 수 있다. 베이스 이미지(252)는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지(252)는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지(252)는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 서비스 파일시스템(253)은 리소스, 애플리케이션 또는 서비스에 대한 구성 파일 및 사양을 포함할 수 있다. 휘발성 파일시스템(254)은 이진 애플리케이션, 특정 주소 및 다른 정보와 같은 해당 배치에 고유한 정보 또는 데이터를 포함할 수 있으며, 이는 암호, 세션 키 및 개인 키를 포함하지만 이에 한정되지 않는 변수로서 구성되거나 그렇지 않을 수 있다. 파일 시스템은 일부 판독 전용 및 일부 판독-기록 파일시스템을 허용하기 위해 overlayFS와 같은 기술을 사용하여 하나의 단일 파일시스템으로서 마운트될 수 있어, 애플리케이션에 사용되는 중복 데이터의 양을 감소시킨다.

[0119] 상기한 바와 같이, 컨트롤러(200)는 연산, 스토리지, 및/또는 네트워킹 리소스와 같은 리소스를 시스템에 추가하는 데 사용될 수 있다. 도 11a는 시스템(100)에 베어메탈 노드와 같은 물리적 리소스를 추가하는 예시적인 방법을 도시한다. 리소스, 즉 연산, 스토리지 또는 네트워킹 리소스는 네트워크 연결(1110)을 통해 컨트롤러에 플러그된다. 네트워크 연결은 대역외 관리 연결을 포함할 수 있다. 컨트롤러는 리소스가 대역외 관리 연결(1111)을 통해 플러그 인되는 것을 인식한다. 컨트롤러는, 리소스의 유형, 능력 및/또는 속성(1112)을 포함할 수 있지만 이에 한정되지 않는 리소스와 관련된 정보를 인식한다. 컨트롤러는 리소스 및/또는 리소스와 관련된 정보를 그의 시스템 상태(1113)에 추가한다. 템플릿으로부터 도출된 이미지는, 스토리지 리소스와 같은 다른 리소스 상에서, 또는 컨트롤러(1114) 상에서, 리소스를 포함할 수 있지만 이에 한정되지 않는 시스템의 물리 구성요소에 로딩된다. 이미지는 구성 파일을 포함할 수 있는 하나 이상의 파일시스템을 포함한다. 이러한 구성은 BIOS 및 부팅 파라미터를 포함할 수 있다. 컨트롤러는 이미지(1115)의 파일시스템을 사용하여 부팅하도록 물리 리소스에 지시한다. 추가 리소스, 또는 복수의 베어메탈 또는 다른 종류의 물리 리소스가 템플릿의 이미지 또는 그의 적어도 일부를 사용하여 이러한 방식으로 추가될 수 있다.

[0120] 도 11b는 예시적 실시형태의 글로벌 시스템 규칙과 템플릿을 사용하여 리소스를 자동적으로 할당하는 예시적인 방법을 도시한다. 요청(1120)을 만족시키도록 리소스 할당을 필요로 하는 시스템에 대해 요청이 이루어진다. 컨트롤러는 그의 시스템 상태 데이터베이스(1121)에 근거하여, 그의 리소스 풀을 인식한다. 컨트롤러는 필요한 리소스(1122)를 결정하기 위해 템플릿을 사용한다. 컨트롤러는 리소스를 할당하고 시스템 상태(1123) 내에 정보를 저장한다. 컨트롤러는 템플릿(1124)을 사용하여 리소스를 배치한다.

[0121] 도 12를 참조하면, 애플리케이션 또는 서비스를 자동적으로 배치하기 위한 예시적인 방법이 본 명세서에 설명된 시스템(100)을 사용하여 도시되어 있다. 사용자 또는 애플리케이션은 서비스를 요청한다(1210). 요청이 API 애플리케이션으로 번역된다(1220). API 애플리케이션이 요청을 컨트롤러에 라우팅한다(1230). 컨트롤러가 요청을 해석한다(1240). 컨트롤러가 시스템의 상태 및 그의 리소스를 고려한다(1250). 컨트롤러는 서비스 배치를 위해 그의 규칙과 템플릿을 사용한다(1260). 컨트롤러는 요청을 리소스에 전송하고(1270), 템플릿으로부터 도출된 이미지를 배치하고(1280), IT 시스템 상태를 업데이트한다.

[0122] 리소스를 추가하는 것, 리소스를 할당하는 것, 및 애플리케이션 또는 서비스를 배치하는 것과 같은 동작의 추가적이고 더 상세한 예가 이하에서 더욱 상세하게 설명된다.

[0123] *시스템에 연산 리소스의 추가:*

[0124] 도 3a를 참조하면, 시스템(100)에 연산 리소스(310)를 추가하는 것이 도시되어 있다. 연산 리소스(310)가 추가될 때, 이것은 컨트롤러(200)에 결합되고 전원이 꺼질 수 있다. 연산 리소스(310)가 이미지와 함께 프리로딩될 경우, 네트워크 연결 중 어느 것이 리소스와 통신하고, 리소스를 부팅하고, 시스템 상태에 정보를 추가하는 데 사용될 수 있는 대안적인 단계가 후속될 수 있음에 유의한다. 연산 리소스와 컨트롤러가 동일한 노드에 있으면, 연산 리소스를 실행하는 서비스가 해제된다.

[0125] 도 3a에 나타낸 바와 같이, 연산 리소스(310)는 네트워크, 즉 대역외 관리 연결(260), 대역내 관리 연결(270), 및 선택적으로 SAN(280)을 통해 컨트롤러에 결합된다. 연산 리소스(310)는 또한 서비스, 애플리케이션, 사용자 및/또는 클라이언트가 서로 통신할 수 있는 하나 이상의 애플리케이션 네트워크(390)에 결합된다. 대역외 관리 연결(260)은 독립적인 대역외 관리 디바이스(315) 또는 연산 리소스(310)가 플러그 인되었을 때에 켜지는 연산 리소스(310)의 회로에 결합될 수 있다. 디바이스(315)는 디바이스의 전원 켜기/끄기, 콘솔에 대한 부착 및 커맨드의 입력, 온도 및 다른 컴퓨터 건전성 관련 요소의 모니터링, 및 BIOS 셋팅 및 운영 체제로부터의 범주 밖의 다른 특징의 설정을 포함하지만 이에 한정되지 않는 특징을 허용할 수 있다. 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 연산 리소스(310)를 알 수 있다. 또한 연산 리소스의 유형을 식별하고 대역내 관리 또는 대역

외 관리를 사용하여 그의 구성을 식별할 수 있다. 컨트롤러 로직(205)은 추가된 하드웨어에 대해 대역외 관리(260) 또는 대역내 관리(270)를 조사하도록 구성된다. 연산 리소스(310)가 검출된 경우에는, 컨트롤러 로직(205)이 글로벌 시스템 규칙(220)을 사용하여 리소스가 자동적으로 또는 사용자와 상호 작용하여 구성되는지를 결정할 수 있다. 그것이 자동적으로 추가되는 경우, 설정은 컨트롤러(200) 내의 글로벌 시스템 규칙(210)을 따를 것이다. 사용자에 의해 추가된 경우, 컨트롤러(200) 내의 글로벌 시스템 규칙(210)은 사용자가 리소스의 추가 및 사용자가 연산 리소스로 수행하고자 하는 것을 확인하도록 요청할 수 있다. 컨트롤러(200)는 새로운 리소스가 허가되었는지 확인하기 위해 API 애플리케이션(들)에 질의하거나, 그렇지 않으면 스택을 제어하는 사용자 또는 임의의 프로그램에 요청할 수 있다. 허가 프로세스는 또한 새로운 리소스의 적합성을 확인하기 위해 암호화를 사용하여 자동적으로 안전하게 완료될 수 있다. 컨트롤러 로직(205)은 연산 리소스(310)가 플러그되는 스위치 또는 네트워크를 포함하는 IT 시스템 상태(220)에 연산 리소스(310)를 추가한다.

[0126] 연산 리소스가 물리적인 경우, 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 연산 리소스의 전원을 켤 수 있고, 연산 리소스(310)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(350)를 부트 오프할 수 있다. 이미지는 다른 네트워크 연결을 통해 또는 다른 리소스를 통해 간접적으로 로딩될 수 있다. 부팅되면, 연산 리소스(310)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 그 후, 연산 리소스(310)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다.

[0127] 연산 리소스가 가상인 경우, 컨트롤러(200)는 대역내 관리 네트워크(270)를 통해 또는 대역외 관리(260)를 통해 연산 리소스의 전원을 켤 수 있다. 연산 리소스(310)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(350)를 부트 오프할 수 있다. 이미지는 다른 네트워크 연결을 통해 또는 다른 리소스를 통해 간접적으로 로딩될 수 있다. 부팅되면, 연산 리소스(310)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 그 후, 연산 리소스(310)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다.

[0128] 컨트롤러(200)는 글로벌 시스템 규칙에 따라 자동적으로 리소스를 켜고 끌 수 있으며, 전력을 절약하기 위해 리소스를 끄는 것 또는 애플리케이션 성능을 개선하기 위해 리소스를 켜는 것 또는 IT 시스템 사용자가 가질 수 있는 임의의 다른 이유와 같은 IT 시스템 사용자에 의해 결정된 이유로 IT 시스템 상태를 업데이트할 수 있다.

[0129] 도 3b에서, 이미지(350)는 연산 리소스를 부팅하고 및/또는 애플리케이션을 로딩하기 위해 템플릿(230)으로부터 연산 리소스(310)로 직접 또는 간접적으로(다른 리소스 또는 데이터베이스를 통해) 로딩된다. 이미지(350)는 리소스 유형 및 하드웨어에 대한 부트 파일(340)을 포함할 수 있다. 부트 파일(340)은 배치될 리소스, 애플리케이션 또는 서비스에 대응하는 커널(341)을 포함할 수 있다. 부트 파일(340)은 또한 부팅 프로세스를 지원하는 데 사용되는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부트 시스템(340)은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 복수의 커널 또는 initrd를 포함할 수 있다. 또한, 이미지(350)는 파일시스템(351)을 포함할 수 있다. 파일 시스템(351)은 베이스 이미지(352) 및 대응하는 파일시스템뿐만 아니라 서비스 이미지(353) 및 대응하는 파일시스템 및 휘발성 이미지(354) 및 대응하는 파일시스템을 포함할 수 있다. 로딩되는 파일 시스템 및 데이터는 리소스 유형과 실행할 애플리케이션 또는 서비스에 따라 다를 수 있다. 베이스 이미지(352)는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지(352)는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지(352)는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 서비스 파일시스템(353)은 리소스, 애플리케이션 또는 서비스에 대한 구성 파일 및 사양을 포함할 수 있다. 휘발성 파일시스템(354)은 이진 애플리케이션, 특정 주소 및 다른 정보와 같은 해당 배치에 고유한 정보 또는 데이터를 포함할 수 있으며, 이는 암호, 세션 키 및 개인 키를 포함하지만 이에 한정되지 않는 변수로서 구성되거나 그렇지 않을 수 있다. 파일 시스템은 일부 판독 전용 및 일부 판독-기록 파일시스템을 허용하기 위해 overlayFS와 같은 기술을 사용하여 하나의 단일 파일시스템으로서 마운트될 수 있어, 애플리케이션에 사용되는 중복 데이터의 양을 감소시킨다.

[0130] 도 3c는 연산 리소스(310)와 같은 리소스를 시스템(100)에 추가하기 위한 예시적인 프로세스 흐름을 도시한다. 이 예에서는, 주요 리소스가 연산 리소스(310)로서 설명될 것이지만, 도 3c의 프로세스 흐름에 대한 주요 리소스는 또한 스토리지 리소스(410) 및/또는 네트워킹 리소스(510)일 수 있음을 이해해야 한다. 도 3c의 예에서, 추가된 리소스(310)는 컨트롤러(200)와 동일한 노드에 있지 않다. 단계 300.1에서, 리소스(310)는 전원이 꺼진 상태에서 컨트롤러(200)에 결합된다. 도 3c의 예에서, 대역외 관리 연결(260)은 리소스(310)를 연결하는 데 사

용된다. 그러나, 다른 네트워크 연결이, 실무자가 원하는 대로 사용될 수 있음을 이해해야 한다. 단계 300.2 및 300.3에서, 컨트롤러 로직(205)은 시스템의 대역외 관리 연결을 통해 확인하고, 대역외 관리 연결(260)을 사용하여, 추가되는 리소스(310)의 유형 및 그의 구성을 인식하고 식별한다. 예를 들어, 컨트롤러 로직은 유형 및 구성 정보를 얻기 위한 참조로서 BIOS 또는 다른 정보(시리얼 번호 정보 등)를 알 수 있다.

[0131] 단계 300.4에서, 컨트롤러는 글로벌 시스템 규칙을 사용하여 특정 리소스(310)가 자동적으로 추가되어야 하는지를 결정한다. 그렇지 않을 경우, 컨트롤러는 그의 사용이 허가될 때까지 대기할 것이다(단계 300.5). 예를 들어, 사용자는 특정 리소스(310)를 사용하기 원하지 않는다는 쿼리, 또는 단계 300.4에서 사용될 때까지 자동적으로 보류할 수 있다는 쿼리에 응답할 수 있다. 단계 300.4가 리소스(310)가 자동적으로 추가되어야 한다고 결정할 경우, 컨트롤러는 자동 설정(단계 300.6)을 위한 그의 규칙을 사용하고 단계 300.7로 진행할 것이다.

[0132] 단계 300.7에서, 컨트롤러는 리소스를 시스템 상태(220)에 추가하기 위해 리소스와 연관된 템플릿(230)을 선택하고 사용한다. 일부 경우에, 템플릿(230)은 특정 리소스에 고유할 수 있다. 그러나, 일부 템플릿(230)은 복수의 리소스 유형을 포함할 수 있다. 예를 들어, 일부 템플릿(230)은 불가지론적인 하드웨어일 수 있다. 단계 300.8에서, 컨트롤러는, 글로벌 시스템 규칙(210)을 따라서, 그의 대역외 관리 연결(260)을 통해 리소스(310)의 전원을 켠다. 단계 300.9에서, 글로벌 시스템 규칙(210)을 사용하여, 컨트롤러는 선택된 템플릿(들)으부터 리소스를 위한 부트 이미지를 찾고 로당한다. 그 후, 리소스(310)는 주요 템플릿(230)으로부터 도출된 이미지로부터 부팅된다(단계 300.10). 그 후, 리소스(310)와 관련한 추가 정보가 리소스(310)가 부팅된 후에 대역내 관리 연결(270)을 통해 리소스(310)로부터 수신될 수 있다(단계 300.11). 이러한 정보는, 예를 들어 펌웨어 버전, 네트워크 카드, 리소스가 연결될 수 있는 임의의 다른 디바이스를 포함할 수 있다. 새로운 정보가 단계 300.12에서 시스템 상태(220)에 추가될 수 있다. 그 후, 리소스(310)가 리소스 풀에 추가된 것으로 간주될 수 있고, 할당 준비가 된다(단계 300.13).

[0133] 도 3c와 관련하여, 리소스와 컨트롤러가 동일한 노드에 있는 경우, 리소스를 실행하는 서비스는 해당 노드를 벗어날 수 있음을 이해해야 한다. 이러한 경우에, 컨트롤러는 예를 들어 유닉스 소켓, 루프백 어댑터 또는 리소스와 통신하기 위한 다른 프로세스간 통신 기술과 같은 리소스와의 프로세스간 통신 기술을 사용할 수 있다. 시스템 규칙으로부터, 컨트롤러는 가상 호스트, 또는 하이퍼바이저 또는 컨테이너 호스트를 설치하여 컨테이너로부터 알려진 템플릿을 사용하여 애플리케이션을 실행할 수 있다. 그 후, 리소스 애플리케이션 정보가 시스템 상태(220)에 추가될 수 있고, 리소스는 할당 준비가 될 것이다.

[0134] *시스템에 스토리지 리소스의 추가:*

[0135] 도 4a는 시스템(100)에 스토리지 리소스(410)를 추가하는 것을 도시한다. 예시적인 실시형태에서, 도 3c의 예시적 프로세스 흐름은 시스템(100)에 스토리지 리소스(410)를 추가하기 위해 후속될 수 있고, 여기서 추가된 스토리지 리소스(410)는 컨트롤러(200)와 동일한 노드에 있지 않다. 또한, 스토리지 리소스(410)가 이미지와 함께 프리로딩될 경우, 네트워크 연결 중 어느 것이 스토리지 리소스(410)와 통신하고, 스토리지 리소스(410)를 부팅하고, 시스템 상태(220)에 정보를 추가하는 데 사용될 수 있는 대안적인 단계가 후속될 수 있음에 유의해야 한다.

[0136] 스토리지 리소스(410)가 추가될 때, 이것은 컨트롤러(200)에 결합되고 전원이 꺼질 수 있다. 스토리지 리소스(410)는 네트워크, 즉 대역외 관리 네트워크(260), 대역내 관리 연결(270), SAN(280) 및 선택적으로 연결(290)을 통해 컨트롤러에 결합한다. 스토리지 리소스(410)는 또한 서비스, 애플리케이션, 사용자 및/또는 클라이언트가 서로 통신할 수 있는 하나 이상의 애플리케이션 네트워크(390)에 결합되거나 그렇지 않을 수 있다. 애플리케이션 또는 클라이언트는 애플리케이션을 통해 리소스의 스토리지에 직접 또는 간접 액세스할 수 있어, SAN을 통해 액세스되지 않는다. 애플리케이션 네트워크는 스토리지가 내장될 수 있거나 IT 시스템 상태에서 스토리지 리소스로서 액세스되고 식별될 수 있다. 대역외 관리 연결(260)은 독립적인 대역외 관리 디바이스(415) 또는 스토리지 리소스(410)가 플러그인되었을 때에 켜지는 스토리지 리소스(410)의 회로에 결합될 수 있다. 디바이스(415)는 디바이스의 전원 켜기/끄기, 콘솔에 대한 부착 및 커맨드의 입력, 온도 및 다른 컴퓨터 건전성 관련 요소의 모니터링, 및 BIOS 셋팅 및 운영 체제로부터의 범주 밖의 다른 특징의 설정을 포함하지만 이에 한정되지 않는 특징을 허용할 수 있다. 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 스토리지 리소스(410)를 알 수 있다. 또한 스토리지 리소스의 유형을 식별하고 대역내 또는 대역외 관리를 사용하여 그의 구성을 식별할 수 있다. 컨트롤러 로직(205)은 추가된 하드웨어에 대해 대역외 관리(260) 또는 대역내 관리(270)를 조사하도록 구성된다. 스토리지 리소스(410)가 검출된 경우에는, 컨트롤러 로직(205)이 글로벌 시스템 규칙(220)을 사용하여 리소스(410)가 자동적으로 또는 사용자와 상호 작용하여 구성되는지를 결정할 수 있다. 그것이 자동적으로 추가

되는 경우, 설정은 컨트롤러(200) 내의 글로벌 시스템 규칙(210)을 따를 것이다. 사용자에 의해 추가된 경우, 컨트롤러(200) 내의 글로벌 시스템 규칙(210)은 사용자가 리소스의 추가 및 사용자가 스토리지 리소스로 수행하고자 하는 것을 확인하도록 요청할 수 있다. 컨트롤러(200)는 API 애플리케이션(들)에 질의하거나, 그렇지 않으면 새로운 리소스가 허가되었는지 확인하기 위해 스택을 제어하는 사용자 또는 임의의 프로그램에 요청할 수 있다. 허가 프로세스는 또한 새로운 리소스의 적합성을 확인하기 위해 암호화를 사용하여 자동적으로 안전하게 완료될 수 있다. 컨트롤러 로직(205)은 스토리지 리소스(410)가 플러그되는 스위치 또는 네트워크를 포함하는 IT 시스템 상태(220)에 연산 리소스(410)를 추가한다.

[0137] 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 스토리지 리소스(410)의 전원을 켤 수 있고, 스토리지 리소스(410)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(450)를 부트 오프할 것이다. 이미지는 또한 다른 네트워크 연결을 통해 또는 다른 리소스를 통해 간접적으로 로딩될 수 있다. 부팅되면, 스토리지 리소스(410)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 이제 스토리지 리소스(410)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다.

[0138] 스토리지 리소스는 IT 시스템이 독립적으로 또는 동시에 사용 또는 액세스할 수 있는 스토리지 리소스 풀 또는 복수의 스토리지 리소스 풀을 포함할 수 있다. 스토리지 리소스가 추가될 때, IT 시스템 상태에 스토리지 풀, 복수의 스토리지 풀, 스토리지 풀의 일부, 및/또는 복수의 스토리지 풀의 일부를 제공할 수 있다. 컨트롤러 및/또는 스토리지 리소스는 풀의 다양한 스토리지 리소스 또는 풀 내의 이러한 리소스의 그룹을 관리할 수 있다. 스토리지 풀은 복수의 스토리지 리소스에서 실행하는 복수의 스토리지 풀을 포함할 수 있다. 예를 들어, 플래시 스토리지 디스크 또는 어레이가 플래터 디스크 또는 어레이를 캐싱하거나 또는 전용 연산 노드 상의 스토리지 풀이 전용 스토리지 노드 상의 풀과 결합되어 대역폭 및 레이턴시를 동시에 최적화한다.

[0139] 도 4b는 스토리지 리소스를 부팅하고 및/또는 애플리케이션을 로딩하기 위해 템플릿(230)으로부터 스토리지 리소스(410)로 직접 또는 간접적으로(다른 리소스 또는 데이터베이스를 통해) 로딩되는 이미지(450)를 도시한다. 이미지(450)는 리소스 유형 및 하드웨어에 대한 부트 파일(440)을 포함할 수 있다. 부트 파일(440)은 배치될 리소스, 애플리케이션 또는 서비스에 대응하는 커널(441)을 포함할 수 있다. 부트 파일(440)은 또한 부팅 프로세스를 지원하는 데 사용되는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부트 시스템(440)은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 복수의 커널 또는 initrd를 포함할 수 있다. 또한, 이미지(450)는 파일시스템(451)을 포함할 수 있다. 파일 시스템(451)은 베이스 이미지(452) 및 대응하는 파일시스템뿐만 아니라 서비스 이미지(453) 및 대응하는 파일시스템 및 휘발성 이미지(454) 및 대응하는 파일시스템을 포함할 수 있다. 로딩되는 파일 시스템 및 데이터는 리소스 유형과 실행할 애플리케이션 또는 서비스에 따라 다를 수 있다. 베이스 이미지(452)는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지(452)는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지(452)는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 서비스 파일시스템(453)은 리소스, 애플리케이션 또는 서비스에 대한 구성 파일 및 사양을 포함할 수 있다. 휘발성 파일시스템(454)은 이진 애플리케이션, 특정 주소 및 다른 정보와 같은 해당 배치에 고유한 정보 또는 데이터를 포함할 수 있으며, 이는 암호, 세션 키 및 개인 키를 포함하지만 이에 한정되지 않는 변수로서 구성되거나 그렇지 않을 수 있다. 파일 시스템은 일부 판독 전용 및 일부 판독-기록 파일시스템을 허용하기 위해 overlayFS와 같은 기술을 사용하여 하나의 단일 파일시스템으로서 마운트될 수 있어, 애플리케이션에 사용되는 중복 데이터의 양을 감소시킨다.

[0140] 도 5a는 JBOD를 갖는 노드의 형태를 취할 수 있는 다른 스토리지 리소스, 즉 직접 부착된 스토리지(510) 또는 다른 유형의 직접 부착된 스토리지가 시스템을 위한 추가 스토리지 리소스로서 스토리지 리소스(410)에 결합되는 일례를 도시한다. JBOD는 스토리지 리소스를 제공하는 노드에 전형적으로 연결된 외부 디스크 어레이이고, JBOD는 도 5a에서 직접 부착된 스토리지(510)의 예시적인 형태로서 사용될 것이지만, 다른 유형의 직접 부착된 스토리지가 510으로서 사용될 수 있음을 이해해야 한다.

[0141] 컨트롤러(200)는, 예를 들어 도 5a에 관하여 설명된 바와 같이 그의 시스템에 스토리지 리소스(410)와 JBOD(510)를 추가할 수 있다. JBOD(510)는 대역외 관리 연결(260)을 통해 컨트롤러(200)에 결합된다. 스토리지 리소스(410)는 네트워크, 즉 대역외 관리 네트워크(260), 대역내 관리 연결(270), SAN(280) 및 선택적으로 연결(290)에 결합된다. 스토리지 노드(410)는 SAS 또는 다른 디스크 드라이브 패브릭(520)을 통해 JBOD(510)의 스토리지와 통신한다. JBOD(510)는 또한 대역외 관리 연결(260)을 통해 컨트롤러와 통신하는 대역외 관리 디바이스(515)를 포함할 수 있다. 대역외 관리(260)를 통해 컨트롤러(200)는 JBOD(510) 및 스토리지 리소스(410)를 검출

할 수 있다. 컨트롤러(200)는 또한, 예를 들어 다양한 대역외 관리 회로에 대해 본 명세서에 설명된 바와 같이, 운영 체제에 의해 제어되지 않는 다른 파라미터를 검출할 수 있다. 컨트롤러(200) 글로벌 시스템 규칙(210)은 아직 추가되지 않은 JBOD 및 스토리지 노드를 부팅 또는 시동하기 위한 구성 시동 규칙을 제공할 수 있다. 스토리지 리소스를 켜는 순서는 글로벌 규칙(220)을 사용하여 컨트롤러 로직(205)에 의해 제어될 수 있다. 한 세트의 글로벌 시스템 규칙(220)에 따르면, 컨트롤러는 우선 JBOD(510)의 전원을 켤 수 있고, 그 후에 컨트롤러(200)는 도 4와 관련하여 설명된 것과 유사한 방식으로 로딩된 이미지(450)를 사용하여 스토리지 리소스(410)의 전원을 켤 수 있다. 다른 세트의 글로벌 시스템 규칙에서, 컨트롤러(200)는 우선 스토리지 리소스(410)를 켠 다음 JBOD(510)를 켤 수 있다. 다른 글로벌 시스템 규칙에서는 다양한 디바이스의 전원을 켜는 동안의 타이밍 또는 지연이 지정될 수 있다. 컨트롤러 로직(205), 글로벌 시스템 규칙(210) 및/또는 템플릿(230)을 통해, 다양한 리소스의 준비 또는 동작 상태의 검출이 컨트롤러(200)에 의한 디바이스 할당 관리에서 결정되고 사용될 수 있다. IT 시스템 상태(220)는 스토리지 리소스(410)와의 통신에 의해 업데이트될 수 있다. 스토리지 노드(410)는 디스크 패브릭(520)을 통해 JBOD에 액세스함으로써 JBOD(510)의 스토리지 파라미터 및 구성을 인식한다. 스토리지 리소스(410)는 컨트롤러(200)에 정보를 제공한 다음, 이용 가능한 스토리지의 양 및 다른 속성에 관한 정보로 IT 시스템 상태(220)를 업데이트한다. 컨트롤러는 스토리지 리소스(410)가 부팅되고 스토리지 리소스(410)가 시스템(100)의 스토리지 리소스 풀(400)의 일부로서 인식될 때에 IT 시스템 상태(220)를 업데이트한다. 스토리지 노드는 컨트롤러(200)에 의해 설정된 구성을 사용하여 JBOD 스토리지 리소스를 제어하기 위한 로직을 처리한다. 예를 들어, 컨트롤러는 스토리지 노드에 RAID 10 또는 다른 구성으로부터 풀을 생성하기 위해 JBOD를 구성하도록 지시할 수 있다.

[0142] 도 5b는 스토리지 리소스(410) 및 스토리지 리소스(410)를 위한 집적 부착된 스토리지(510)를 시스템(100)에 추가하기 위한 예시적인 프로세스 흐름을 도시한다. 단계 500.1에서, 직접 부착된 스토리지(510)는 대역외 관리 연결(260)을 통해 전원이 꺼진 상태로 컨트롤러(200)에 결합된다. 단계 500.2에서, 스토리지 리소스(410)는 대역외 관리 연결(260)과 대역내 관리 연결(270)을 통해 전원이 꺼진 상태로 컨트롤러(200)에 결합되는 한편, 스토리지 리소스(410)는, 예를 들어 디스크 드라이브 패브릭과 같은 SAS(520)를 통해 직접 부착된 스토리지(510)에 결합된다.

[0143] 그 후, 컨트롤러 로직(205)은 대역외 관리 연결(260)을 조사하여 스토리지 리소스(410) 및 직접 부착된 스토리지(510)를 검출할 수 있다(단계 500.3). 임의의 네트워크 연결이 사용될 수 있지만, 이 예에서는 대역외 관리가 컨트롤러 로직에 사용되어 추가되고 있는 리소스의 유형(이 경우에, 스토리지 리소스(410) 및 직접 부착된 스토리지(510)) 및 이들의 구성을 인식 및 식별할 수 있다(단계 500.4).

[0144] 단계 500.5에서, 컨트롤러(200)는 각 유형의 스토리지 디바이스에 대한 특정 유형의 스토리지를 위한 템플릿(230)을 선택 및 사용하여 시스템 상태(220)에 리소스(410 및 510)를 추가한다. 단계 500.6에서, 컨트롤러는 (대역외 관리 연결(260)을 통해, 직접 스토리지 및 스토리지 노드의 부팅 순서, 전원을 켜는 순서를 이러한 순서대로 지정할 수 있는) 글로벌 시스템 규칙(210)을 따른다(500.6). 글로벌 시스템 규칙(210)을 사용하여, 컨트롤러는 해당 스토리지 리소스(410)에 대한 선택된 템플릿(230)으로부터 스토리지 리소스(410)를 위한 부트 이미지를 찾고 로딩하고, 그 후에 스토리지 리소스가 이미지로부터 부팅된다(단계 500.7). 스토리지 리소스(410)는 디스크 패브릭(520)을 통해 직접 부착된 스토리지(510)에 액세스함으로써 직접 부착된 스토리지(510)의 스토리지 파라미터 및 구성을 인식한다. 스토리지 리소스(410) 및/또는 직접 부착된 스토리지(510)에 관한 추가 정보는 대역내 관리 연결(270)을 통해 스토리지 리소스로부터 컨트롤러에 제공될 수 있다(단계 500.8). 단계 500.9에서, 컨트롤러는 단계 500.8에서 획득된 정보로 시스템 상태(220)를 업데이트한다. 단계 500.10에서, 컨트롤러는 직접 부착된 스토리지(510)를 처리하기 위해 스토리지 리소스(410)에 대한 구성 및 직접 부착된 스토리지를 구성하는 방법을 설정한다. 그 후에 단계 500.11에서, 직접 부착된 스토리지(510)와 조합하는 스토리지 리소스(410)를 포함하는 새로운 리소스가 리소스 풀에 추가될 수 있고 시스템 내에서 할당 준비가 된다.

[0145] 예시적인 실시형태의 다른 양태에 따르면, 컨트롤러는 대역외 관리를 사용하여 컴퓨팅 또는 서비스에 관여되지 않을 수 있는 스택 내의 다른 디바이스를 인식할 수 있다. 예를 들어, 이러한 디바이스는 냉각탑/에어컨디셔너, 조명 온도, 음성, 알람, 전력 시스템, 또는 시스템과 연관된 임의의 다른 디바이스를 포함할 수 있지만 이에 한정되지 않는다.

[0146] *시스템에 네트워킹 리소스의 추가:*

[0147] 도 6a는 시스템(100)에 네트워킹 리소스(610)를 추가하는 것을 도시한다. 예시적인 실시형태에서, 도 3c의 예시적인 프로세스 흐름은 시스템(100)에 네트워킹 리소스(610)를 추가하기 위해 후속될 수 있고, 여기서 추가된 네

트워킹 리소스(610)는 컨트롤러(200)와 동일한 노드에 있지 않다. 또한, 네트워킹 리소스(610)가 이미지와 함께 프리로딩될 경우, 네트워크 연결 중 어느 것이 네트워킹 리소스(610)와 통신하고, 네트워킹 리소스(610)를 부팅하고, 시스템 상태(220)에 정보를 추가하는 데 사용될 수 있는 대안적인 단계가 후속될 수 있음에 유의해야 한다.

[0148] 네트워킹 리소스(610)가 추가될 때, 이것은 컨트롤러(200)에 결합되고 전원이 꺼질 수 있다. 네트워킹 리소스(610)는 연결, 즉 대역외 관리 연결(260) 및/또는 대역내 관리 연결(270)을 통해 컨트롤러(200)에 결합될 수 있다. SAN(280) 및/또는 연결(290)에 선택적으로 플러그된다. 네트워킹 리소스(610)는 또한 서비스, 애플리케이션, 사용자 및/또는 클라이언트가 서로 통신할 수 있는 하나 이상의 애플리케이션 네트워크(390)에 결합되거나 그렇지 않을 수도 있다. 대역외 관리 연결(260)은 독립적인 대역외 관리 디바이스(615) 또는 네트워킹 리소스(610)가 플러그인 되었을 때에 켜지는 네트워킹 리소스(610)의 회로에 결합될 수 있다. 디바이스(615)는 디바이스의 전원 켜기/끄기, 콘솔에 대한 부착 및 커맨드의 입력, 온도 및 다른 컴퓨터 건전성 관련 요소의 모니터링, 및 BIOS 셋팅 및 운영 체제로부터의 범주 밖의 다른 특징의 설정을 포함하지만 이에 한정되지 않는 특징을 허용할 수 있다. 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 네트워킹 리소스(610)를 알 수 있다. 또한 네트워킹 리소스의 유형 및/또는 네트워크 패브릭을 식별하고 대역내 또는 대역외 관리를 사용하여 구성을 식별할 수 있다. 컨트롤러 로직(205)은 추가된 하드웨어에 대해 대역외 관리(260) 또는 대역내 관리(270)를 조사하도록 구성된다. 네트워킹 리소스(610)가 검출된 경우에는, 컨트롤러 로직(205)이 글로벌 시스템 규칙(220)을 사용하여 네트워킹 리소스(610)가 자동적으로 또는 사용자와 상호 작용하여 구성되는지를 결정할 수 있다. 그것이 자동적으로 추가되는 경우, 설정은 컨트롤러(200) 내의 글로벌 시스템 규칙(210)을 따를 것이다. 사용자에 의해 추가된 경우, 컨트롤러(200) 내의 글로벌 시스템 규칙(210)은 사용자가 리소스의 추가 및 사용자가 리소스로 수행하고자 하는 것을 확인하도록 요청할 수 있다. 컨트롤러(200)는 API 애플리케이션(들)에 질의하거나, 그렇지 않으면 새로운 리소스가 허가되었는지 확인하기 위해 스택을 제어하는 사용자 또는 임의의 프로그램에 요청할 수 있다. 허가 프로세스는 또한 새로운 리소스의 적합성을 확인하기 위해 암호화를 사용하여 자동적으로 안전하게 완료될 수 있다. 그 후, 컨트롤러 로직(205)은 네트워킹 리소스(610)를 IT 시스템 상태(220)에 추가할 수 있다. 컨트롤러에 자신을 식별 불가능한 스위치의 경우, 사용자는 시스템 상태를 수동 또는 추가 할 수 있습니다.

[0149] 네트워킹 리소스가 물리적인 경우, 컨트롤러(200)는 대역외 관리 연결(260)를 통해 네트워킹 리소스(610)의 전원을 켤 수 있고, 연산 리소스(310)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(605)를 부트 오프할 수 있다. 이미지는 또한 다른 네트워크 연결을 통해 또는 다른 리소스를 통해 간접적으로 로딩될 수 있다. 부팅되면, 네트워킹 리소스(610)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 그 후, 네트워킹 리소스(610)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다. 선택적으로, 일부 네트워킹 리소스 스위치는 대역외 관리(260)에 연결된 콘솔 포트를 통해 제어될 수 있고, 전원이 켜질 때에 구성될 수 있거나 부트 로더를 통해, 예를 들어 ONIE를 통해 설치된 스위치 운영 체제를 가질 수 있다.

[0150] 네트워킹 리소스가 가상인 경우, 컨트롤러(200)는 대역내 관리 네트워크(270)를 통해 또는 대역외 관리(260)를 통해 네트워킹 리소스의 전원을 켤 수 있다. 네트워킹 리소스(610)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(650)를 부트 오프할 수 있다. 부팅되면, 네트워킹 리소스(610)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 그 후, 네트워킹 리소스(610)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다.

[0151] 컨트롤러(200)는 물리적이든 가상이든, 다른 물리 또는 가상 리소스, 즉 본 명세서에 정의된 바와 같이 연결, 스토리지, 또는 연산에 연결하기 위해 포트를 할당, 재할당 또는 이동시키도록 네트워킹 리스소에 지시할 수 있다. 이는 SDN, 인피니밴드 파티셔닝, VLAN, vXLAN을 포함하지만 이에 한정되지 않는 기술을 사용하여 수행될 수 있다. 컨트롤러(200)는 가상 스위치 또는 가상 스위치를 호스팅하는 리소스와의 통신을 네트워킹 또는 상호 연결하기 위해 가상 인터페이스를 이동 또는 할당하도록 가상 스위치에 지시할 수 있다. 일부 물리 또는 가상 스위치는 컨트롤러에 결합된 API에 의해 제어될 수 있다.

[0152] 컨트롤러(200)는 또한 이러한 변경이 가능할 때에 패브릭 유형을 변경하도록 연산, 스토리지, 또는 네트워킹 리소스에 지시할 수 있다. 포트는, 예를 들어 하이브리드 인피니밴드/이더넷 인터페이스의 패브릭을 토글링하여

다른 패브릭으로 전환하도록 구성될 수 있다.

[0153] 컨트롤러(200)는 복수의 애플리케이션 네트워크를 전환하는 스위치 또는 다른 네트워킹 리소스를 포함할 수 있는 네트워킹 리소스에 명령어를 제공할 수 있다. 스위치 또는 네트워크 디바이스는 상이한 패브릭을 포함할 수 있거나, 또는 예를 들어 바람직하게는 SDN 능력 및 복수의 패브릭을 갖는 인피니밴드 스위치, ROCE 스위치, 및/또는 다른 스위치에 플러그될 수 있다.

[0154] 도 6b는 네트워킹 리소스를 부팅하고 및/또는 애플리케이션을 로딩하기 위해 템플릿(230)으로부터 네트워킹 리소스(610)로 직접 또는 간접적으로(다른 리소스 또는 데이터베이스를 통해) 로딩되는 이미지(650)를 도시한다. 이미지(650)는 리소스 유형 및 하드웨어에 대한 부트 파일(640)을 포함할 수 있다. 부트 파일(640)은 배치될 리소스, 애플리케이션 또는 서비스에 대응하는 커널(641)을 포함할 수 있다. 부트 파일(640)은 또한 부팅 프로세스를 지원하는 데 사용되는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부트 시스템(640)은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 복수의 커널 또는 initrd를 포함할 수 있다. 또한, 이미지(650)는 파일시스템(651)을 포함할 수 있다. 파일 시스템(651)은 베이스 이미지(652) 및 대응하는 파일시스템뿐만 아니라 서비스 이미지(653) 및 대응하는 파일시스템 및 휘발성 이미지(654) 및 대응하는 파일시스템을 포함할 수 있다. 로딩되는 파일 시스템 및 데이터는 리소스 유형과 실행할 애플리케이션 또는 서비스에 따라 다를 수 있다. 베이스 이미지(652)는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지(652)는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지(652)는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 서비스 파일시스템(653)은 리소스, 애플리케이션 또는 서비스에 대한 구성 파일 및 사양을 포함할 수 있다. 휘발성 파일시스템(654)은 이진 애플리케이션, 특정 주소 및 다른 정보와 같은 해당 배치에 고유한 정보 또는 데이터를 포함할 수 있으며, 이는 암호, 세션 키 및 개인 키를 포함하지만 이에 한정되지 않는 변수로서 구성되거나 그렇지 않을 수 있다. 파일 시스템은 일부 판독 전용 및 일부 판독-기록 파일시스템을 허용하기 위해 overlayFS와 같은 기술을 사용하여 하나의 단일 파일시스템으로서 마운트될 수 있어, 애플리케이션에 사용되는 중복 데이터의 양을 감소시킨다.

[0155] *리소스에 애플리케이션 또는 서비스의 배치:*

[0156] 도 7a는, 컨트롤러(200); 제1 연산 노드(311), 제2 연산 노드(312), 및 제3 연산 노드(313)를 포함하는 물리 및 가상 연산 리소스; 스토리지 리소스(410); 및 네트워크 리소스(610)를 포함하는 시스템(100)을 도시한다. 리소스는 도 1 내지 도 6b와 관련하여 본 명세서에 설명된 방식으로 IT 시스템 상태(220)에 설정되고 추가된 것으로 도시되어 있다.

[0157] 이 도면에는 다수의 연산 노드가 도시되어 있지만, 예시적인 실시형태에 따라 단일 연산 노드가 사용될 수 있다. 연산 노드는 물리 또는 가상 연산 리소스를 호스팅할 수 있으며 물리 또는 가상 연산 노드에서 애플리케이션을 실행할 수 있다. 마찬가지로, 단일 네트워크 제공자 노드 및 스토리지 노드가 도시되어 있지만, 이러한 유형의 다수의 리소스 노드는 예시적인 실시형태의 시스템에 사용될 수 있거나 그렇지 않을 수도 있다.

[0158] 서비스 또는 애플리케이션은 예시적인 실시형태에 따라 임의의 시스템에 배치될 수 있다. 컴퓨팅 노드 상에 서비스를 배치하는 예는 도 7a와 관련하여 설명될 수 있지만, 시스템(100)의 상이한 배열과 마찬가지로 사용될 수 있다. 예를 들어, 도 7a의 컨트롤러(200)는 글로벌 시스템 규칙(210)에 따라 연산 노드(311, 312, 313)의 형태로 연산 리소스(310)를 자동적으로 구성할 수 있다. 그 후, 이들은 또한 IT 시스템 상태(220)에 추가될 수 있다. 따라서, 컨트롤러(200)는 연산 리소스(311, 312, 313)(전원이 꺼질 수도 있고 그렇지 않을 수도 있음) 및 가능하게는 연산 리소스 또는 노드 상에서 실행하는 임의의 물리 또는 가상 애플리케이션을 인식할 수 있다. 컨트롤러(200)는 또한 글로벌 시스템 규칙(210) 및 템플릿(230)에 따라 스토리지 리소스(들)(410) 및 네트워킹 리소스(들)(610)를 자동적으로 구성하고 이를 IT 시스템 상태(220)에 추가할 수 있다. 컨트롤러(200)는 전원이 꺼진 상태로 시작하거나 그렇지 않을 수 있는 스토리지 리소스(410) 및 네트워킹 리소스(610)를 인식할 수 있다.

[0159] 도 7b는 IT 시스템(100)에 대한 리소스의 추가를 위한 예시적인 처리를 도시한다.

[0160] 단계 700.1에서, 새로운 물리 리소스가 시스템에 결합된다. 단계 700.2에서, 컨트롤러는 새로운 리소스를 인식하게 된다. 리소스는 원격 스토리지에 연결될 수 있다(단계 700.4). 단계 700.3에서, 컨트롤러는 새로운 리소스를 부팅하는 방법을 구성한다. 리소스에 이루어지는 모든 연결은 시스템 상태(220)에 로깅될 수 있다(단계 700.5). 전술한 도 3c는 도 7b에 의해 나타낸 것과 같은 프로세스 흐름의 예시적인 실시형태에 대한 추가 상세를 제공한다.

[0161] 도 7c 및 도 7d는 다수의 연산 리소스, 다수의 서버, 다수의 가상 머신, 및/또는 다수의 사이트에 대한 애플리

케이션의 배치에 대한 예시적인 프로세스 흐름을 나타낸다. 이 예의 프로세스는 IT 시스템(100)이 리던던트 및 상관된 애플리케이션 및/또는 서비스를 결합하기 위한 구성요소를 필요로 할 것이라는 점에서 표준 템플릿 배치와 다르다. 컨트롤러 로직은 단계 700.11에서 메타-템플릿을 프로세싱할 수 있고, 여기서 메타-템플릿은 복수의 템플릿(230), 파일시스템 블롭(232), 및 멀티-홈드(multi-homed) 서비스를 구성하는 데 필요한 다른 구성요소(다른 템플릿(230)의 형태일 수 있음)를 포함할 수 있다.

[0162] 단계 700.12에서, 컨트롤러 로직(205)은 이용 가능한 리소스에 대한 시스템 상태(220)를 검사하고; 그러나 리소스가 충분하지 않으면, 컨트롤러 로직은 배치될 수 있는 리던던트 서비스 수를 줄일 수 있다(700.16 참조, 여기서 리던던트 서비스 수가 식별된다). 단계 700.13에서, 컨트롤러 로직(205)은 서비스를 함께 연결하는 데 필요한 네트워킹 리소스 및 상호 연결을 구성한다. 서비스 또는 애플리케이션이 다수의 사이트에 걸쳐 배치되는 경우, 메타-템플릿은 사이트들 간의 데이터 동기화 및 상호 운용성을 허용하는 템플릿으로부터 선택적으로 구성된 서비스를 포함할 수 있다(또는 컨트롤러 로직(205)이 그 서비스를 구성할 수 있다)(700.15 참조).

[0163] 단계 700.16에서, 컨트롤러 로직(205)은 시스템 규칙, 메타 템플릿 데이터, 및 리소스 이용 가능성으로부터 (다수의 호스트에 리던던트 서비스가 있을 경우) 리던던트 서비스 수를 결정할 수 있다. 700.17에서, 다른 리던던트 서비스와의 결합 및 마스터와의 결합이 있다. 다수의 리던던트 호스트가 있는 경우, 컨트롤러 로직(205) 또는 템플릿(바이너리(234), 데몬(232), 또는 운영 체제에서 셋팅을 지시하는 구성 파일을 포함할 수 있는 파일시스템 블롭) 내의 로직은 네트워크 주소와 호스트명 충돌을 방지할 수 있다. 선택적으로, 컨트롤러 로직은 네트워크 주소를 제공하고(700.18 참조) 각각의 리던던트 서비스를 DNS에 등록하고(700.19) 시스템 상태(220)에 등록할 것이다(700.18). 시스템 상태(220)는 리던던트 서비스를 추적할 것이고, 컨트롤러 로직(205)은 호스트명, dns명, 네트워크 주소와 같은 충돌하는 파라미터를 갖는 리던던트 서비스가 시스템 상태(220)에 이미 존재하는 것을 알게 되면 중복 등록을 허용하지 않을 것이다.

[0164] 도 7d에 의해 나타낸 구성 루틴은 메타 템플릿에서 템플릿(들)을 프로세싱할 것이다. 구성 루틴은 모든 리던던트 서비스를 프로세싱하여, 멀티-호스트 또는 클러스터 서비스를 다수의 호스트에 배치하고, 호스트를 결합하기 위해 서비스를 배치할 것이다. 시스템 규칙으로부터 IT 시스템을 배치할 수 있는 임의의 프로세스는 구성 루틴을 실행할 수 있다. 멀티-호스트 서비스의 경우, 예시적인 루틴은700.32에서와 같이 서비스 템플릿을 프로세싱하기, 700.33에서와 같이 스토리지를 제공하기, 700.35에서와 같이 호스트의 전원을 켜기, 700.36에서와 같이 스토리지 리소스와 호스트/연산 리소스를 결합하기(및 시스템 상태(220)에 등록하기)(그 후에 리던던트 서비스의 수에 대해 반복하기(700.38)); 매회 시스템 상태(220)에 등록하기(700.20 참조) 및 개별 서비스를 추적하고 충돌을 방지하는 정보를 로그하기 위해 컨트롤러 로직을 사용하기(700.31 참조)일 수 있다.

[0165] 서비스 템플릿 중 일부는 서비스 및 멀티-호스트 서비스를 결합할 수 있는 툴을 포함할 수 있다. 이들 서비스 중 일부는 종속성으로서 취급될 수 있고(700.39), 그 후에 700.40에서 결합 루틴이 서비스를 결합하고 결합을 시스템 상태(220)에 등록하는 데 사용될 수 있다. 또한, 서비스 템플릿 중 하나는 마스터 템플릿일 수 있으며, 그러면 700.39에서의 종속 서비스 템플릿은 슬레이브 또는 보조 서비스일 것이고; 700.40에서 결합 루틴이 그들을 연결할 것이다. 루틴은 메타-템플릿에서 정의될 수 있고; 예를 들어, 리던던트 dns 구성의 경우, 700.40에서 결합 루틴은 슬레이브 dns와 마스터 dns의 연결 및 dnssec와 함께 영역 전송의 구성을 포함할 수 있다. 일부 서비스는 성능을 향상기키기 위해 물리 스토리지를 사용할 수 있고(700.34 참조), 이는 도 5b에 개시된 예비 OS와 함께 로딩될 수 있다. 서비스를 결합하기 위한 툴은 템플릿 그 자체에 포함될 수 있으며, 서비스들 간의 구성은 컨트롤러 및/또는 멀티노드 애플리케이션/서비스의 다른 호스트에 의해 액세스 가능한 API로 수행될 수 있다.

[0166] 컨트롤러(200)는 사용자 또는 컨트롤러가 애플리케이션에 사용할 적절한 연산 백엔드를 결정할 수 있게 한다. 컨트롤러(200)는 사용자 또는 컨트롤러가 리소스 사용을 결정함으로써 적절한 물리 또는 가상 연산 리소스 상에 애플리케이션을 최적으로 배치할 수 있게 한다. 하이퍼바이저 또는 다른 연산 백엔드가 연산 노드에 배치될 때, 그들은 대역내 관리 연결(270)을 통해 컨트롤러 리소스 이용 통계에 다시 보고할 수 있다. 컨트롤러가 자체 로직 및 글로벌 시스템 규칙으로부터, 또는 사용자 입력으로부터 가상 연산 리소스에서 애플리케이션을 생성하기로 결정하면, 가장 최적의 호스트에서 하이퍼바이저를 자동적으로 선택하고 해당 호스트에서 가상 연산 리소스의 전원을 켤 수 있.

[0167] 예를 들어, 컨트롤러(200)는 템플릿(들)(230)을 사용하여 하나 또는 복수의 연산 리소스에 애플리케이션 또는 서비스를 배치한다. 이러한 애플리케이션 또는 서비스는, 예를 들어 애플리케이션 또는 서비스를 실행하는 가상 머신일 수 있다. 일례에서, 도 7a는 다수의 연산 노드 상의 복수의 가상 머신(VM)의 배치를 도시하고, 도시 된 바와 같이 컨트롤러(200)는 연산 노드(311, 312, 313)의 형태로 그의 연산 리소스 풀에 복수의 연산 리소스

(310)가 있음을 인식할 수 있다. 연산 노드는 속도로 인해 가상 머신의 사용이 바람직하지 않을 수 있는 경우에 예를 들어 하이퍼바이저와 함께 또는 대안적으로 베어메탈에 배치될 수 있다. 이 예에서, 연산 리소스(310)에는 하이퍼바이저 애플리케이션이 로딩되고, VM 1(321) 및 VM 2(322)이 연산 노드(311)에 구성되고 배치된다. 예를 들어, 연산 노드(311)가 추가 VM에 대한 리소스를 갖지 않는 경우 또는 다른 리소스가 선호되는 경우, 특정 서비스에 대해, 컨트롤러(200)는 스택 상태(220)에 기초하여, 연산 노드(311) 상에 이용 가능한 리소스가 없는 것, 또는 다른 리소스에서 새로운 VM을 설정하는 선호도가 있는 것을 인식할 수 있다. 또한, 하이퍼바이저가, 예를 들어 다른 목적에 사용되는 베어메탈 연산 노드일 수 있는 리소스(313)가 아닌 연산 리소스(312)에 로딩되는 것을 인식할 수 있다. 따라서, 설치되는 서비스 또는 애플리케이션 템플릿의 요구 사항, 및 시스템 상태(220)의 상태에 따라, 이 예의 컨트롤러는 다음의 필요한 리소스 VM 3(323)의 배치를 위해 연산 노드(313)를 선택할 수 있다.

[0168] 시스템의 연산 리소스는 스토리지 노드에 대한 스토리지 리소스에 스토리지를 공유하도록 구성될 수 있다.

[0169] 사용자 인터페이스(110) 또는 애플리케이션을 통해 사용자는 시스템(100)에 대해 설정되는 서비스를 요청할 수 있다. 서비스는 이메일 서비스; 웹 서비스; 사용자 관리 서비스; 네트워크 공급자, LDAP, 개발 툴(Dev tool), VOIP, 인증 툴, 어카운팅을 포함할 수 있지만 이에 한정되지 않는다.

[0170] API 애플리케이션(120)은 사용자 또는 애플리케이션 요청을 번역하고 메시지를 컨트롤러(200)에 전송한다. 컨트롤러(200)의 서비스 템플릿 또는 이미지(230)는 서비스에 필요한 리소스를 식별하는 데 사용된다. 그 후, 사용될 리소스가 IT 시스템 상태(220)에 따른 이용 가능성에 기초하여 식별된다. 컨트롤러(200)는 필요한 연산 서비스를 위한 하나 이상의 연산 노드(311, 312 또는 313), 필요한 스토리지 리소스를 위한 스토리지 리소스(410), 및 필요한 네트워킹 리소스를 위한 네트워크 리소스(610)에 요청한다. 그 후, IT 시스템 상태(220)는 할당될 리소스를 식별하여 업데이트된다. 그 후, 서비스는 서비스 또는 애플리케이션을 위한 템플릿(230)에 따라 글로벌 시스템 규칙(210)을 사용하여 할당된 리소스에 설치된다.

[0171] 예시적인 실시형태에 따르면, 다수의 연산 노드가 동일한 서비스 또는 다른 서비스에 관계없이 사용될 수 있는 한편, 예를 들어 스토리지 서비스 및/또는 네트워크 제공자 풀은 연산 노드들 사이에서 공유될 수 있다.

[0172] 도 8a를 참조하면, 시스템(100)은 컨트롤러(200), 및 연산, 스토리지 및 네트워킹 리소스(300, 400, 600)가 단일 노드와 같은 동일하거나 공유된 물리 하드웨어 상에 있는 것으로 도시되어 있다. 도 1 내지 도 10에 나타내고 설명되는 다양한 특징은 단일 노드에 통합될 수 있다. 노드의 전원이 켜지면, 컨트롤러 이미지가 노드에 로딩된다. 연산, 스토리지 및 네트워킹 리소스(300, 400, 600)는 템플릿(230) 및 글로벌 시스템 규칙(210)을 사용하여 구성된다. 컨트롤러(200)는 연산 백엔드(318, 319)를 연산 리소스로서 로딩하도록 구성될 수 있으며, 이는 상기 노드에 또는 다른 노드(들)에 추가되거나 그렇지 않을 수 있다. 이러한 백엔드(318, 319)는 가상 연산, 네트워킹, 및 스토리지 리소스를 생성하기 위한 가상화, 컨테이너, 및 멀티-테넌트 프로세스를 포함할 수 있지만 이에 한정되지 않는다.

[0173] 애플리케이션 또는 서비스(725), 예를 들어 웹, 이메일, 코어 네트워크 서비스(DHCP, DNS 등), 협업 툴은 컨트롤러(200)와 공유되는 노드/디바이스의 가상 리소스에 설치될 수 있다. 이들 애플리케이션 또는 서비스는 컨트롤러(200)와 무관하게 물리 리소스 또는 가상 리소스로 이동될 수 있다. 애플리케이션은 단일 노드의 가상 머신에서 실행할 수 있다.

[0174] 도 8b는 단일의 노드 시스템으로부터 다수의 노드 시스템(예를 들어 도 8a에 의해 나타낸 노드(318 및/또는 319)를 가짐)으로 확장하기 위한 예시적인 프로세스 흐름을 나타낸다. 그래서, 도 8a 및 도 8b를 참조하면, 단일 서버에서 실행중인 컨트롤러(200)를 갖는 IT 시스템을 고려할 수 있고; 여기서 IT 시스템을 멀티-노드 IT 시스템으로 확장하는 것이 바람직하다. 따라서, 확장하기 전에 IT 시스템은 단일 노드 상태에 있다. 도 8a에 의해 나타낸 바와 같이, 컨트롤러(200)는 스토리지 리소스, 연산 리소스, 하이퍼바이저, 및/또는 컨테이너 호스트를 포함할 수 있지만 이에 한정되지 않는 다양한 IT 시스템 관리 애플리케이션 및/또는 리소스에 전력을 공급하기 위해 멀티-테넌트 단일 노드 시스템에서 실행한다.

[0175] 단계 800.2에서, 새로운 물리적 자원이 대역외 관리 연결(260), 대역내 관리 연결(270), SAN(280) 및/또는 네트워크(290)를 통해 새로운 물리 리소스를 연결함으로써 단일 노드 시스템에 결합된다. 이 예의 목적을 위해, 이러한 새로운 물리 리소스는 하드웨어 또는 호스트라고도 지칭될 수 있다. 컨트롤러(200)는 관리 네트워크에서 새로운 리소스를 검출하고, 그 후에 디바이스에 질의할 수 있다. 대안적으로, 새로운 디바이스는 자신을 알리는 메시지를 컨트롤러(200)에 브로드캐스팅할 수 있다. 예를 들어, 새로운 디바이스는 MAC 주소, 대역외 관리, 및/

또는 예비 OS로 부팅하는 것 및 대역내 관리를 사용하는 것 및 이에따라 하드웨어 유형을 식별을하는 것에 의해 식별될 수 있다. 어느 경우에도, 단계 800.3에서, 새로운 디바이스는 그의 노드 유형 및 그의 현재 이용 가능한 하드웨어 리소스 및 소프트웨어 리소스에 관한 정보를 컨트롤러에 제공한다. 그 후, 컨트롤러(200)는 새로운 디바이스 및 그의 능력을 인식한다.

[0176] 단계 800.4에서, 컨트롤러(200)를 실행하는 시스템에 할당되는 작업이 새로운 호스트에 할당될 수 있다. 예를 들어, 호스트에 운영 체제(예를 들어, 스토리지 호스트 운영 체제 또는 하이퍼바이저)가 프리로딩되어 있으면, 컨트롤러(200)는 새로운 하드웨어 리소스 및/또는 능력을 할당한다. 그 후, 컨트롤러는 이미지를 제공하고 새로운 하드웨어를 제공할 수 있거나, 또는 새로운 하드웨어가 컨트롤러로부터 이미지를 요청하고 상기 및 이하에 개시되는 방법을 사용하여 자체 구성할 수 있다. 새로운 호스트가 스토리지 리소스 또는 가상 연산 리소스를 호스팅할 수 있는 경우, 새로운 리소스는 컨트롤러(200)에 이용 가능하게 될 수 있다. 그 후, 컨트롤러(200)는 기존의 애플리케이션을 새로운 리소스에 이동 및/또는 할당하거나 또는 새롭게 생성된 애플리케이션 또는 이후에 생성된 애플리케이션을 위해 새로운 리소스를 사용할 수 있다.

[0177] 단계 800.5에서, IT 시스템은 현재 애플리케이션을 컨트롤러에서 계속 실행할 수 있게 하거나 새로운 하드웨어로 마이그레이션할 수 있다. 가상 연산 리소스를 마이그레이션하는 경우, VM 마이그레이션 기술(예를 들어, qemu+kvm의 마이그레이션 툴)이 사용될 수 있고 새로운 시스템 규칙과 함께 시스템 상태를 업데이트할 수 있다. 후술하는 변경 관리 기술은 이들 변경을 확실하고 안전하게 행하는 데 사용될 수 있다. 더 많은 애플리케이션이 시스템에 추가될 수 있으므로, 컨트롤러는 라운드 로빈 기술, 가중 라운드 로빈 기술, 최소 활용 기술, 가중 최소 활용 기술, 활용에 기초한 보조 훈련에 의한 예측 기술, 스케줄링된 기술, 원하는 용량 기술, 및 최대 크기 기술을 포함하지만 이에 한정되지 않는 시스템의 리소스를 할당하는 방법을 결정하기 위한 다양한 기술 중 어느 것을 사용할 수 있다.

[0178] 도 8c는 스토리지 리소스의 새로운 물리 스토리지 리소스로의 마이그레이션을 위한 예시적인 프로세스 흐름을 도시한다. 스토리지 리소스는 미러링, 마이그레이션 또는 이들의 조합이 될 수 있다(예를 들어, 스토리지가 미러링될 수 있고, 그 후에 원래 스토리지 리소스가 연결 해제된다). 단계 820에서, 스토리지 리소스는 새로운 스토리지 리소스가 컨트롤러와 접촉하거나 컨트롤러가 이를 발견하게 함으로써 시스템에 결합된다. 이는 대역외 관리 연결(260), 대역내 관리 연결(270), SAN 네트워크(280)로, 또는 애플리케이션 네트워크가 사용중일 수 있는 플랫 네트워크에서 또는 이들의 조합으로 수행될 수 있다. 대역내 관리에 따라, 운영 체제가 사전 부팅될 수 있고 새로운 리소스가 컨트롤러에 연결될 수 있다.

[0179] 단계 822에서, 새로운 스토리지 타겟이 새로운 스토리지 리소스에 생성되고; 이는 단계 824에서 데이터베이스에 로깅될 수 있다. 일례에서, 스토리지 타겟은 파일을 복사함으로써 생성될 수 있다. 다른 예에서, 스토리지 타겟은 블록 디바이스를 생성하고 (파일 시스템 블롭(들)의 형태일 수 있는) 데이터를 복사함으로써 생성될 수 있다. 다른 예에서, 스토리지 타겟은 블록 디바이스들 사이에 2개 이상의 스토리지 리소스를 미러링(예를 들어, raid를 생성)하고 선택적으로 iscsi, iser, nvmeof, nfs, nfs over rdma, fc, fcoe, srp 등을 포함하지만 이에 한정되지 않는 원격 스토리지 전송(들)을 통해 연결함으로써 생성될 수 있다. 단계 824에서 데이터베이스 엔트리는 스토리지 리소스가 다른 리소스 또는 호스트와 동일한 디바이스 상에 있는 경우에 원격으로 또는 로컬로 새로운 스토리지 리소스에 연결하기 위해 연산 리소스(또는 다른 유형의 리소스 및/또는 호스트)에 대한 정보를 포함할 수 있다.

[0180] 단계 826에서, 스토리지 리소스가 동기화된다. 예를 들어, 스토리지는 미러링될 수 있다. 다른 예로서, 스토리지는 오프라인으로 취해지고 동기화될 수 있다. raid 1(또는 다른 유형의 raid - 그러나 일반적으로 raid 1 또는 raid 0, 하지만 원하는 경우에 raid 110(미러링된 raid 10)일 수 있음)과 같은 기술(mdadm, zfs, btrfs, hardware raid)이 단계 826에서 이용될 수 있다.

[0181] 그 후, 이전 스토리리 리소스로부터의 데이터는 단계 828에서 데이터베이스 로깅 후에 선택적으로 연결된다(이후에 발생하는 경우, 데이터베이스는 이러한 데이터가 기록되어야 하는 경우에 데이터를 복사하는 상태와 관련된 정보를 포함할 수 있다). 스토리지 타겟이 이전 호스트로부터 멀리 마이그레이션되는 경우(예를 들어, 도 8a 및 8b에 따라 단일 노드 시스템으로부터 멀티-노드 및/또는 분산 IT 시스템으로 이전에 이동하는 것으로 나타낸 바와 같이), 새로운 스토리지 리소스는 단계 830에서 컨트롤러, 시스템 상태, 연산 리소스, 또는 이들의 조합에 의해 1차 스토리지 리소스로서 지정될 수 있다. 이는 이전 스토리지 리소스를 제거하는 단계로서 수행될 수 있다. 일부 경우에, 리소스에 연결된 물리 또는 가상 호스트는 그 후에 업데이트될 필요가 있을 것이고, 일부 경우에는 (물리 또는 가상 호스트의 전원을 켜기 위해 본 명세서에 개시된 기술일 수 있는) 단계 832에서 전이 동

안 전원이 꺼질 수 있다(그 후에 전원이 다시 켜질 것이다).

[0182] 도 8d는 멀티-테넌트 시스템의 단일 노드에서 가상 시스템, 컨테이너, 및/또는 프로세스를 연산 및 스토리지를 위한 별도의 하드웨어를 가질 수 있는 멀티-노드 시스템으로 마이그레이션하기 위한 예시적인 프로세스 흐름을 나타낸다. 단계 850에서, 컨트롤러(200)는 새로운 노드 상에 있을 수 있는 새로운 스토리지 리소스를 생성한다(예를 들어, 도 8a의 노드(318 및 319) 참조). 단계(852)에서, 이전 애플리케이션 호스트는 그 후에 전원이 꺼질 수 있다. 그 후, 단계 854에서, 데이터가 복사 또는 동기화된다. 단계 854에서 복사/동기화하기 전에 단계 852에서 전원을 차단함으로써, 마이그레이션은 단일 노드로부터 VM을 마이그레이션하는 것을 포함하는 경우에 더 안전할 것이다. 전원 끄기는 VM으로부터 물리적으로 전환하는 데 이점이 있을 것이다. 단계 854는 또한 데이터 사전 동기화 단계 862를 통해 전원을 차단하기 전에 달성될 수 있으며, 이는 연관된 다운타임을 최소화하는 것을 도울 수 있다. 또한, 단계 852에서와 같이 호스트의 전원이 차단되지 않을 수 있으며, 이 경우에 새로운 호스트가 준비될 때까지(또는 새로운 스토리지 리소스가 준비될 때까지) 이전 호스트는 온라인을 유지된다. 전원 끄기 단계 852를 피하기 위한 기술은 이하에서 더욱 상세히 설명된다. 단계 854에서, 스토리지 리소스가 핫 스탠바이를 사용하여 미러링되거나 동기화되지 않는 한, 데이터는 선택적으로 동기화될 수 있다.

[0183] 새로운 스토리지 리소스는 이제 동작하고 단계 856에서 데이터베이스에 로깅될 수 있어 컨트롤러(200)는 단계 858에서 새로운 호스트를 새로운 스토리지 리소스에 연결할 수 있다. 다수의 가상 호스트를 갖는 단일 노드로부터 마이그레이션할 때, 이 프로세스는 복수의 호스트에 대해 반복될 필요가 있을 수 있다(단계 860). 부팅 순서는 이들이 추적되는 경우에 애플리케이션의 종속성을 사용하여 컨트롤러 로직에 의해 결정될 수 있다.

[0184] 도 8e는 시스템 내의 단일 노드로부터 다수의 노드로 확장하기 위한 다른 예시적인 프로세스 흐름을 나타낸다. 단계 870에서, 새로운 리소스가 단일 노드 시스템에 결합된다. 컨트롤러는 시스템에 대한 일련의 시스템 규칙 및/또는 확장 규칙 세트를 가질 수 있다(또는 실행중인 서비스, 그들의 템플릿, 및 서로에 대한 서비스의 종속성에 따라 확장 규칙을 도출할 수 있다). 단계 872에서, 컨트롤러는 확장을 용이하게 하는 데 사용하기 위해 이러한 규칙에 대해 검사한다.

[0185] 새로운 물리 리소스가 스토리지 리소스를 포함하고 있다면, 스토리지 리소스는 단계 874에서 단일 노드 또는 다른 형태의 더 단순한 IT 시스템으로부터 제거될 수 있다(또는 스토리지 리소스는 미러링될 수 있다). 스토리지 리소스가 제거된 경우, 연산 리소스 또는 실행중인 리소스는 스토리지 리소스가 제거된 후에 단계 876에서 재로딩 또는 재부팅될 수 있다. 다른 예에서, 연산 리소스는 단계 876에서 미러링된 스토리지 리소르에 연결되고 실행중인 상태를 유지할 수 있는 한펴, 단일 노드 시스템 상의 이전 스토리지 리소스 또는 이전 시스템의 하드웨어 리소스는 연결 해제되거나 비활성화될 수 있다. 예를 들어, 실행중인 서비스가 2개의 미러링된 블록 리소스 - 단일 노드 서버 상의 하나(예를 들어, mdadm raid 1을 사용함) 및 스토리지 리소스 상의 다른 하나 -에 결합될 수 있고; 데이터가 동기화되면, 단일 노드 서버 상의 드라이브가 연결 해제될 수 있다. 이전 하드웨어는 여전히 IT 시스템의 일부를 포함할 수 있고, 이를 혼합 모드에서 컨트롤러와 동일한 노드에서 실행할 수 있다(단계 878). 시스템은 원래 노드가 단지 컨트롤러에 전력을 공급할 때까지 이 마이그레이션 프로세스를 계속 반복할 수 있고, 그 결과 시스템이 분산된다(단계 880). 또한, 도 8e 프로세스 흐름의 각 단계에서, 컨트롤러는 시스템 상태(220)를 업데이트하고 데이터베이스에 시스템에 대한 임의의 변경을 로그할 수 있다(단계 882).

[0186] 도 9a를 참조하면, 애플리케이션(910)이 리소스(900)에 설치된다. 리소스(900)는 본 명세서에 설명된 바와 같이 도 1 내지 도 10에 관한 연산, 스토리지 또는 네트워킹 리소스(310, 410, 610)일 수 있다. 리소스(900)는 물리 리소스일 수 있다. 물리 리소스는 물리 머신 또는 물리 IT 시스템 구성요소를 포함할 수 있다. 리소스(900)는, 예를 들어 물리 연산, 스토리지 또는 네트워킹 리소스일 수 있다. 리소스(900)는 본 명세서에서 도 2a 내지 10과 관련하여 설명된 바와 같이 연산, 네트워킹, 또는 스토리지 리소스 중 다른 하나를 갖는 시스템(100)의 컨트롤러(200)에 연결될 수 있다.

[0187] 리소스(900)는 시작시에 전원이 차단될 수 있다. 리소스(900)는 네트워크, 즉 대역외 관리 연결(260), 대역내 관리 연결(270), SAN(280) 및/또는 네트워크(290)를 통해 컨트롤러에 결합될 수 있다. 리소스(900)는 또한 서비스, 애플리케이션, 사용자 및/또는 클라이언트가 서로 통신할 수 있는 하나 이상의 애플리케이션 네트워크(390)에 결합될 수 있다. 대역외 관리 연결(260)은 독립적인 대역외 관리 디바이스(915) 또는 리소스(900)가 플러그 인되었을 때에 켜지는 리소스(900)의 회로에 결합될 수 있다. 디바이스는 디바이스의 전원 켜기/끄기, 콘솔에 대한 부착 및 커맨드의 입력, 온도 및 다른 컴퓨터 건전성 관련 요소의 모니터링, 및 BIOS 셋팅(195) 및 운영 체제로부터의 범주 밖의 다른 특징의 설정을 포함하지만 이에 한정되지 않는 특징을 허용할 수 있다.

[0188] 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 리소스를 검출할 수 있다. 또한 리소스의 유형을 식별하고

대역내 관리 또는 대역외 관리를 사용하여 그의 구성을 식별할 수 있다. 컨트롤러 로직(205)은 추가 하드웨어에 대해 대역외 관리(260) 또는 대역내 관리(270)를 조사하도록 구성될 수 있다. 리소스(900)가 검출된 경우에는, 컨트롤러 로직(205)이 글로벌 시스템 규칙(220)을 사용하여 리소스(900)가 자동적으로 또는 사용자와 상호 작용하여 구성되는지를 결정할 수 있다. 그것이 자동적으로 추가되는 경우, 설정은 컨트롤러(200) 내의 글로벌 시스템 규칙(210)을 따를 것이다. 사용자에 의해 추가된 경우, 컨트롤러(200) 내의 글로벌 시스템 규칙(210)은 사용자가 리소스의 추가 및 사용자가 연산 리소스로 수행하고자 하는 것을 확인하도록 요청할 수 있다. 컨트롤러(200)는 새로운 리소스가 허가되었는지 확인하기 위해 API 애플리케이션(들)에 질의하거나, 그렇지 않으면 스택을 제어하는 사용자 또는 임의의 프로그램에 요청할 수 있다. 허가 프로세스는 또한 새로운 리소스의 적합성을 확인하기 위해 암호화를 사용하여 자동적으로 안전하게 완료될 수 있다. 그 후, 리소스(900)는 리소스(900)가 플러그되는 스위치 또는 네트워크를 포함하는 IT 시스템 상태(220)에 추가된다.

[0189] 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 리소스(900)의 전원을 켤 수 있다. 컨트롤러(200)는 대역외 관리 연결(260)을 사용하여 물리 리소스의 전원을 켜고 BIOS(195)를 구성할 수 있다. 컨트롤러(200)는 콘솔(190)을 자동적으로 사용하고 원하는 BIOS 옵션을 선택할 수 있으며, 이는 컨트롤러(200)가 이미지 인식으로 콘솔 이미지를 판독하고 대역외 관리를 통해 콘솔(190)을 제어하는 것에 의해 달성될 수 있다. 부트업 상태는 리소스(900)의 콘솔을 통한 이미지 인식, 또는 가상 키보드를 통한 대역외 관리, 리소스에서 수중인 서비스에 질의하는 것, 또는 애플리케이션(910)의 서비스에 질의하는 것에 의해 결정될 수 있다. 일부 애플리케이션은 컨트롤러(200)가 대역내 관리(270)를 사용하여 애플리케이션(910)의 셋팅을 모니터링하는 것, 또는 일부 경우에는 변경하는 것을 가능하게 하는 프로세스를 가질 수 있다.

[0190] 물리 리소스(900)(또는 본 명세서에서 도 1 내지 도 10과 관련하여 설명된 바와 같은 리소스(300, 310, 311,312, 313, 400, 410, 411, 412, 600, 610))의 애플리케이션(910)은 PXE 부트 또는 Flex Boot를 활성화하는 것과 같은 원격 부팅을 구성하도록 BIOS 부트 옵션 또는 다른 방법을 사용하여 SAN(280) 또는 다른 네트워크를 통해 부팅할 수 있다. 추가적으로 또는 대안적으로, 컨트롤러(200)는 이미지(950)에서 애플리케이션 이미지를 부팅하도록 물리 리소스(900)에 지시하기 위해 대역외 관리(260) 및/또는 대역내 관리 연결(270)을 사용할 수 있다. 컨트롤러는 리소스에 부팅 옵션을 구성할 수 있거나 PXE 부트 또는 Flex Boot와 같은 기존의 활성화된 원격 부팅 방법을 사용할 수 있다. 컨트롤러(200)는 선택적으로 또는 대안적으로 대역외 관리(260)를 사용하여 ISO 이미지를 부트 오프하여, 로컬 디스크를 구성한 다음, 리소스가 로컬 디스크(들)(920)로부터 부팅하도록 지시할 수 있다. 로컬 디스크(들)는 로딩된 부트 파일을 가질 수 있다. 이는 대역외 관리(260), 이미지 인식 및 가상 키보드를 사용함으로써 달성될 수 있다. 리소스는 또한 설치된 부트 파일 및/또는 부트 로더를 가질 수 있다. 리소스(900)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(950)를 부트 오프할 수 있다. 글로벌 시스템 규칙(220)은 부팅 순서를 지정할 수 있다. 예를 들어, 글로벌 시스템 규칙(220)은 리소스(900)가 먼저 부팅된 다음 애플리케이션(910)이 부팅되는 것을 요구할 수 있다. 리소스(900)가 이미지(950)를 사용하여 부팅되면, 리소스(900)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 리소스(900)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다. 애플리케이션(910)은 또한 리소스(900) 상에 로딩된 이미지(950) 또는 애플리케이션 이미지(956)를 사용하여 글로벌 시스템 규칙(220)에 의해 지정된 순서로 부팅될 수 있다.

[0191] 컨트롤러(200)는 대역외 관리 연결(260), 또는 다른 연결로, 애플리케이션(910)을 애플리케이션 네트워크(390)에 연결하도록 네트워킹 리소스(610)를 구성할 수 있다. 물리 리소스(900)는 예를 들어 ISER(ISCSI over RDMA), NVMEOF FCOE, FC, 또는 ISCSI를 포함하지만 이에 한정되지 않는 블록 스토리지 리소스와 같은 원격 스토리지 또는 SWIFT, GFUSTER, 또는 CEPHFS와 같은 다른 스토리지 백엔드에 연결될 수 있다. IT 시스템 상태(220)는 서비스 또는 애플리케이션이 가동되어 실행중일 때 대역외 관리 연결(260) 및/또는 대역내 관리 연결(270)을 사용하여 업데이트될 수 있다. 컨트롤러(200)는 대역외 관리 연결(260) 또는 대역내 관리 연결(270)을 사용하여 물리 리소스(900)의 전력 상태, 즉 온 또는 오프 여부를 결정할 수 있다. 컨트롤러(200)는 대역외 관리 연결(260) 또는 대역내 관리 연결(270)을 사용하여 서비스 또는 애플리케이션이 실행중 또는 부트업 상태인지를 결정할 수 있다. 컨트롤러는 수신하는 정보 및 글로벌 시스템 규칙(210)에 기초하여 다른 조치를 취할 수 있다.

[0192] 도 9b는 애플리케이션(910)을 부팅하기 위해 템플릿(230)으로부터 연산 노드로 직접 또는 간접적으로(예를 들어, 다른 리소스 또는 데이터베이스를 통해) 로딩되는 이미지(950)를 도시한다. 이미지(950)는 애플리케이션(910)을 위한 커스텀 커널(941)을 포함할 수 있다.

[0193] 이미지(950)는 리소스 유형 및 하드웨어에 대한 부트 파일(940)을 포함할 수 있다. 부트 파일(940)은 배치될 리

소스, 애플리케이션 또는 서비스에 대응하는 커널(941)을 포함할 수 있다. 부트 파일(940)은 또한 부팅 프로세스를 지원하는 데 사용되는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부트 시스템(940)은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 복수의 커널 또는 initrd를 포함할 수 있다. 또한, 이미지(450)는 파일시스템(951)을 포함할 수 있다. 파일 시스템(951)은 베이스 이미지(952) 및 대응하는 파일시스템뿐만 아니라 서비스 이미지(953) 및 대응하는 파일시스템 및 휘발성 이미지(954) 및 대응하는 파일시스템을 포함할 수 있다. 로딩되는 파일 시스템 및 데이터는 리소스 유형과 실행할 애플리케이션 또는 서비스에 따라 다를 수 있다. 베이스 이미지(952)는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지(952)는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지(952)는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 서비스 파일시스템(953)은 리소스, 애플리케이션 또는 서비스에 대한 구성 파일 및 사양을 포함할 수 있다. 휘발성 파일시스템(594)은 이진 애플리케이션, 특정 주소 및 다른 정보와 같은 해당 배치에 고유한 정보 또는 데이터를 포함할 수 있으며, 이는 암호, 세션 키 및 개인 키를 포함하지만 이에 한정되지 않는 변수로서 구성되거나 그렇지 않을 수 있다. 파일 시스템은 일부 판독 전용 및 일부 판독-기록 파일시스템을 허용하기 위해 overlayFS와 같은 기술을 사용하여 하나의 단일 파일시스템으로서 마운트될 수 있어, 애플리케이션에 사용되는 중복 데이터의 양을 감소시킨다.

[0194] 도 9c는 템플릿(230)의 유형일 수 있는 NT 패키지로부터 애플리케이션을 설치하는 일례를 나타낸다. 단계 900.1에서, 컨트롤러는 패키지 블롭이 설치될 필요가 있는지 결정한다. 단계 900.2에서 컨트롤러는 블롭 유형(블록, 파일, 파일시스템)에 대한 기본 데이터스토어에 스토리지 리소스를 생성한다. 단계 900.3에서, 컨트롤러는 스토리지 리소스 유형에 대한 이용 가능한 스토리지 전송을 통해 스토리지 리소스에 연결한다. 단계 900.4에서, 컨트롤러는 패키지 블롭을 연결된 스토리지 리소스에 복사한다. 그 후, 컨트롤러는 스토리지 리소스로부터 연결 해제하고(단계 900.5) 스토리지 리소스를 판독 전용으로 설정한다(단계 900.6). 그 후, 패키지 블롭이 성공적으로 설치된다(단계 900.7).

[0195] 다른 예에서, 본 명세서에 동봉된 부록 B는 시스템이 연산 리소스를 overlayfs에 연결하는 방법에 관한 예시적인 상세를 설명한다. 이러한 기술은 도 9a에 따라 리소스에 애플리케이션을 설치하는 것 또는 도 2f로부터의 단계 205.11에 따라 스토리지 리소스로부터 연산 리소스를 핫팅(hotting)하는 것을 용이하게 하는 데 사용될 수 있다.

[0196] 도 9d는 리소스(900) 상에 배치된 애플리케이션(910)을 도시한다. 리소스(900)는, 예를 들어 하이퍼바이저(920), 하나 이상의 가상 머신(921, 922) 및/또는 컨테이너를 포함할 수 있는 가상 연산 리소스를 포함할 수 있는 연산 노드를 포함할 수 있다. 리소스(900)는 리소스(900) 상에 로딩된 이미지(950)를 사용하여 도 1 내지 도 10과 관련하여 본 명세서에 설명된 것과 유사한 방식으로 구성될 수 있다. 이 예에서, 리소스(920)는 가상 머신(921, 922)을 관리하는 하이퍼바이저로 나타나 있다. 컨트롤러(200)는 대역내 관리(270)를 사용하여 하이퍼바이저(920)를 호스팅하는 리소스(900)와 통신하여 리소스를 생성하고 리소스를 구성하고 CPU RAM, GPU, 원격 GPU(RDMA를 사용하여 다른 호스트에 원격으로 연결할 수 있음), 네트워크 연결, 네트워크 패브릭 연결, 및/또는 구획 및/또는 세그먼트화된 네트워크에 대한 가상 및 물리 연결을 포함하지만 이에 한정되지 않는 적절한 하드웨어 리소스를 할당할 수 있다. 컨트롤러(200)는 리소스(900) 및 하이퍼바이저(920)를 제어하기 위해 가상 콘솔(190)(예를 들어, SPICE 또는 VNC를 포함하지만 이에 한정되지 않음) 및 이미지 인식을 사용할 수 있다. 추가적으로 또는 대안적으로 컨트롤러(200)는 글로벌 시스템 규칙(210)을 사용하여 템플릿(230)으로부터 애플리케이션 이미지(950)를 부팅하도록 하이퍼바이저(920)에 지시하기 위해 대역외 관리(260) 또는 대역내 관리 연결(270)을 사용할 수 있다. 이미지(950)는 컨트롤러(200)에 저장될 수 있거나 컨트롤러(200)는 이들을 스토리지 리소스(410)로 이동 또는 복사할 수 있다. VM(921, 922)에 대한 부트 이미지는, 예를 들어 이미지(950), 또는 블록 디바이스 또는 원격 호스트 상에 파일로서 로컬로 저장될 수 있고, 예를 들어 qcow2 또는 raw와 같은 이미지 유형을 사용하는 NFS over RDMA/NFS와 같은 파일 공유를 통해 공유될 수 있거나 또는 ISCSI, ISER, NVMEOF, FC, FCOE를 사용하는 원격 블록 디바이스를 사용할 수 있다. 이미지(950)의 일부는 스토리지 리소스(410) 또는 연산 노드(310)에 저장될 수 있다. 글로벌 규칙 및/또는 템플릿을 사용하는 컨트롤러(200)는 대역외 관리 연결(260), 또는 다른 연결로 애플리케이션을 지원하기 위해 네트워킹 리소스(610)를 적절히 구성할 수 있다. 리소스(900) 상의 애플리케이션(910)은 BIOS 부트 옵션을 사용하여 SAN(280) 또는 다른 네트워크에 의해 로딩된 이미지(950)를 사용하는 것 또는 리소스(900) 상의 하이퍼바이저(920)가 예를 들어 ISER(ISCSI over RDMA), NVMEOF FCOE, FC 또는 ISCSI를 포함하지 만 이에 한정되지 않는 블록 스토리지 리소스 또는 SWIFT, GFUSTER, 또는 CEPHFS와 같은 다른 스토리지 백엔드에 연결할 수 있게 하는 것에 의해 부팅할 수 있다. 스토리지 리소스는 스토리지 리소스의 템플릿 타겟으로부터 복사될 수 있다. IT 시스템 상태(220)는 정보를 위해 하이퍼 바이저(920)에 질의함으로써 업데이트될 수 있다. 대역내 관리 연결(270)은 하이퍼바이저(920)와 통신할 수 있고, 리소스

의 전력 상태, 즉 온 또는 오프 여부를 결정하거나 부트업 상태를 결정하는 데 사용될 수 있다. 하이퍼바이저(920)는 또한 가상화 애플리케이션(910)에 가상 대역내 연결(923)을 사용하고 대역외 관리와 유사한 기능을 위해 하이퍼바이저(920)를 사용할 수 있다. 이 정보는 전력 공급 또는 부팅 여부에 기인하여 서비스 또는 애플리케이션이 가동되어 실행중인지 여부를 나타낼 수 있다.

[0197] 부트업 상태는 리소스(900)의 콘솔(190)을 통한 이미지 인식, 또는 가상 키보드를 통한 대역외 관리(260), 리소스에서 수신중인 서비스에 질의하는 것, 또는 애플리케이션(910) 그 자체의 서비스에 질의하는 것에 의해 결정될 수 있다. 일부 애플리케이션은 컨트롤러(200)가 대역내 관리(270)를 사용하여 애플리케이션(910)의 셋팅을 모니터링하는 것, 또는 일부 경우에는 변경하는 것을 가능하게 하는 프로세스를 가질 수 있다. 일부 애플리케이션은 가상 리소스 상에 있을 수 있고 컨트롤러(200)는 대역내 관리(270)(또는 대역외 관리(260))를 사용하여 하이퍼바이저(920)와 통신함으로써 모니터링할 수 있다. 애플리케이션(910)은 모니터링하고 및/또는 입력을 추가한 느이러한 프로세스를 갖지 않을 수 있고(또는 이러한 프로세스가 리소스를 절약하기 위해 해제될 수 있고); 이러한 경우에 컨트롤러(200)는 대역외 관리 연결(260)을 사용하고 이미지 프로세싱 및/또는 가상 키보드를 사용하여 시스템에 로그온하여 관리 프로세스를 변경 및/또는 설정할 수 있게 한다. 가상 연산 리소스와 마찬가지로, 가상 머신 콘솔(190)이 사용될 수 있다.

[0198] 도 9e는 IT 시스템(100)에 가상 연산 리소스 호스트를 추가하기 위한 예시적인 프로세스 흐름을 나타낸다. 단계 900.11에서, 가상 연산 리소스로서 가능한 호스트가 시스템에 추가된다. 컨트롤러는 도 15b 프로세스 흐름에 따라 베어메탈 서버를 구성할 수 있고(단계 900.12); 또는 운영 체제가 프리로딩될 수 있고 및/또는 호스트가 사전 구성될 수 있다(단계 900.13). 그 후, 리소스는 가상 연산 리소스 풀로서 시스템 상태(220)에 추가되고(단계 900.14), 리소스는 컨트롤러(200)로부터 API에 의해 액세스 가능해진다(단계 900.15). API는 일반적으로 대역관리 연결(270)을 통해 액세스되고; 그러나 대역내 관리 연결(270)은 가상 키보드로 선택적으로 활성화 및/또는 비활성화될 수 있으며; 컨트롤러는 대역외 관리 연결(260) 및 가상 키보드 및 모니터를 사용하여 대역외 연결(260)을 통해 통신할 수 있다(단계 900.16). 단계 900.17에서, 컨트롤러는 이제 새로운 리소스를 가상 연산 리소스로서 사용할 수 있다.

[0199] *예시적인 멀티-컨트롤러 시스템:*

[0200] 도 10을 참조하면, 시스템(100)은, 복수의 물리 연산 노드(311, 312, 313)를 포함하는 본 명세서에 도 1 내지 도 10과 관련하여 설명된 연산 리소스(300, 310); 복수의 스토리지 노드(411, 412) 및 JBOD(413)의 형태로 본 명세서에 설명된 스토리지 리소스(400, 410); 구성 요소(205, 210, 220, 230)(도 1 내지 도 9C)을 포함하고 본 명세서에 설명된 컨트롤러(200)로서 구성되는 복수의 컨트롤러(200a, 200b); 복수의 패브릭(611, 612, 613)을 포함하는 본 명세서에 설명된 네트워킹 리소스(600, 610); 및 애플리케이션 네트워크(390)를 갖는 것으로 도시되어 있다.

[0201] 도 10은 예시적인 실시형태의 시스템(100)의 구성요소의 가능한 배열을 도시하지만, 시스템(100)의 구성요소의 가능한 배열을 한정하지는 않는다.

[0202] 사용자 인터페이스 또는 애플리케이션(110)은 컨트롤러(200a 또는 200b) 중 어느 하나 또는 둘 다와 통신할 수 있는 API 애플리케이션(120)과 통신한다. 컨트롤러(200a, 200b)는 대역외 관리 연결(260), 대역내 관리 연결(270), SAN(280), 또는 네트워크 대역내 관리 연결(290)에 결합될 수 있다. 도 1 내지 도 9c를 참조하여 본 명세서에 설명된 바와 같이, 컨트롤러(200a, 200b)는 연결(260, 270, 280 및 선택적으로 290)을 통해 연산 노드(311, 312, 313), JBOD(413)를 포함하는 스토리지(411, 412), 및 네트워킹 리소스(610)에 결합된다. 애플리케이션 네트워크(390)는 연산 노드(311, 312, 313), 스토리지 리소스(411, 412, 413) 및 네트워킹 리소스(610)에 결합된다.

[0203] 컨트롤러(200a, 200b)는 병렬로 동작할 수 있다. 어느 컨트롤러(200a 또는 200b)는 본 명세서에 도 1 내지 도 9c와 관련하여 설명된 바와 같이 초기에 마스터 컨트롤러(200)로서 동작할 수 있다. 컨트롤러(들)(200a, 200b)는 전체 시스템(100)을 전원 차단 상태로부터 구성하도록 배열될 수 있다. 컨트롤러(200a, 200b) 중 하나는 또한 대역외 및 대역내 연결(260, 270)을 통해 다른 컨트롤러를 프로브함으로써 기존의 구성으로부터 시스템 상태(220)를 채울 수 있다. 컨트롤러(200a, 200b)는 하나 이상의 연결(260, 270)을 통해 리소스 또는 다른 컨트롤러로부터 리소스 상태 및 관련 정보에 액세스하거나 수신할 수 있다. 컨트롤러 또는 다른 리소스는 다른 컨트롤러를 업데이트할 수 있다. 따라서, 추가 컨트롤러가 시스템에 추가될 때에는 시스템(100)을 시스템 상태(220)로 다시 복구하도록 구성될 수 있다. 컨트롤러 중 하나 또는 마스터 컨트롤러의 장애의 경우, 다른 컨트롤러가 마스터 컨트롤러로서 지정될 수 있다. IT 시스템 상태(220)는 또한 이용 가능한 또는 리소스 상에 저장된 상태 정

보로부터 재구성 가능할 수 있다. 예를 들어, 애플리케이션은 시스템 상태가 저장되거나 복제되는 가상 연산 리소스를 생성하도록 애플리케이션이 구성되는 연산 리소스에 배치될 수 있다. 글로벌 시스템 규칙(210), 시스템 상태(220), 및 템플릿(230)은 또한 리소스 또는 리소스들의 조합에 저장되거나 복사될 수 있다. 따라서, 모든 컨트롤러가 오프라인 상태가 되고 새로운 것이 추가되면, 시스템은 새로운 컨트롤러가 시스템 상태(220)를 복구할 수 있도록 구성될 수 있다.

[0204] 네트워킹 리소스(610)는 복수의 네트워크 패브릭을 포함할 수 있다. 예를 들어, 도 10에 나타낸 바와 같이, 복수의 네트워크 패브릭은 SDN 이더넷 스위치(611), ROCE 스위치(612), Infiniband 스위치(613), 또는 다른 스위치 또는 패브릭(614) 중 하나 이상을 포함할 수 있다. 연산 노드 상에 가상 머신을 포함하는 하이퍼바이저는 원하는 하나 이상의 패브릭을 이용하여 물리 스위치 또는 가상 스위치에 연결될 수 있다. 네트워킹 배열은, 예를 들어 보안 또는 다른 리소스 최적화 목적을 위해, 예를 들어 세그먼트화된 네트워킹을 통해 물리 네트워크의 제한을 허용할 수 있다.

[0205] 본 명세서에서 도 1 내지 도 10에 설명된 컨트롤러(200)를 통한 시스템(100)은 서비스 또는 애플리케이션을 자동적으로 설정할 수 있다. 사용자 인터페이스(110) 또는 애플리케이션을 통해 사용자는 시스템(100)에 대해 설정되는 서비스를 요청할 수 있다. 서비스는 이메일 서비스; 웹 서비스; 사용자 관리 서비스; 네트워크 공급자, LDAP, 개발 툴(Dev tool), VOIP, 인증 툴, 어카운팅 소프트웨어를 포함할 수 있지만 이에 한정되지 않는다. API 애플리케이션(120)은 사용자 또는 애플리케이션 요청을 번역하고 메시지를 컨트롤러(200)에 전송한다. 컨트롤러(200)의 서비스 템플릿 또는 이미지(230)는 서비스에 필요한 리소스를 식별하는 데 사용된다. 필요한 리소스가 시스템 상태(220)에 따른 이용 가능성에 기초하여 식별된다. 컨트롤러(200)는 필요한 연산 서비스를 위한 연산 리소스(310) 또는 연산 노드(311, 312 또는 313), 필요한 스토리지 리소스를 위한 스토리지 리소스(410), 및 필요한 네트워킹 리소스를 위한 네트워크 리소스(610)에 요청한다. 그 후, 시스템 상태(220)는 할당될 리소스를 식별하여 업데이트된다. 그 후, 서비스는 서비스 템플릿에 따라 글로벌 시스템 규칙(210)을 사용하여 할당된 리소스에 설치된다.

[0206] *향상된 시스템 보안:*

[0207] 도 13a를 참조하면, 시스템(100)이 리소스(1310)를 포함하고, 리소스(1310)가 베어 메탈 또는 물리 리소스일 수 있는 IT 시스템(100)이 나타나 있다. 도 13a는 시스템(100)에 연결된 단일 리소스(1310)만을 나타내지만, 시스템(100)은 복수의 리소스(1310)를 포함할 수 있음을 이해해야 한다. 리소스(들)(1310)는 베어 메탈 클라우드 노드이거나 이를 포함할 수 있다. 베어 메탈 클라우드 노드는 물리 호스트 또는 가상 머신에 대한 원격 액세스를 허용하고, 가상 머신의 생성을 허용하고, 외부 사용자가 리소스(들)에 대한 코드를 실행할 수 있게 하는 외부 네트워크(1380)에 연결되는 리소스를 포함할 수 있지만 이에 한정되지 않는다. 리소스(들)(1310)는 외부 네트워크(1380) 또는 애플리케이션 네트워크(390)에 직접 또는 간접적으로 연결될 수 있다. 외부 네트워크(1380)는 IT 시스템(100)의 컨트롤러(200) 또는 컨트롤러들에 의해 관리되지 않는 인터넷 또는 다른 리소스(들)일 수 있다. 외부 네트워크(1380)는 인터넷, 인터넷 연결(들), 컨트롤러에 의해 관리되지 않는 리소스(들), 다른 광역 네트워크(예를 들어, Stratcom, 피어 투 피어 메시 네트워크, 또는 공개적으로 액세스 가능하거나 그렇지 않을 수 있는 다른 외부 네트워크) 또는 다른 네트워크를 포함할 수 있지만 이에 한정되지 않는다.

[0208] 물리 리소스(1310)가 IT 시스템(100a)에 추가될 때, 이것은 컨트롤러(200)에 결합되고 전원이 꺼질 수 있다. 리소스(1310)는 하나 이상의 네트워크, 즉 대역외 관리(OOBM) 연결(260), 선택적으로 대역내 관리(IBM) 연결(270), 및 선택적으로 SAN 연결(280)을 통해 컨트롤러(200a)에 결합된다. 본 명세서에서 사용된 SAN(280)은 구성 SAN을 포함하거나 그렇지 않을 수 있다. 구성 SAN은 물리 리소스의 전원을 켜거나 이를 구성하는 데 사용되는 SAN을 포함할 수 있다. 구성 SAN은 SAN(280)의 일부일 수 있거나 SAN(280)과 별개일 수 있다. 대역내 관리는 또한 본 명세서에 나타낸 바와 같이 SAN(280)이거나 아닐 수 있는 구성 SAN을 포함할 수 있다. 구성 SAN은 또한 리소스가 사용될 때에 비활성화되거나, 연결 해제되거나, 또는 이용 가능하지 않을 수 있다. OOBM 연결(260)은 시스템(100)의 OS에 보이지 않지만, IBM 연결(270) 및/또는 구성 SAN은 시스템(100)의 OS에 보일 수 있다. 도 13a의 컨트롤러(200)는 본 명세서에서 도 1 내지 도 12b를 참조하여 설명된 컨트롤러(200)와 유사한 방식으로 구성될 수 있다. 리소스(1310)는 내부 스토리지를 포함할 수 있다. 일부 구성에서, 컨트롤러(200)는 스토리지를 채우고 데이터 및/또는 정보를 페치하기 위해 SAN에 연결하도록 리소스를 일시적으로 구성할 수 있다. 대역외 관리 연결(260)은 독립적인 대역외 관리 디바이스(315) 또는 리소스(1310)가 플러그 인되었을 때에 켜지는 리소스(1310)의 회로에 결합될 수 있다. 디바이스(315)는 디바이스의 전원 켜기/끄기, 콘솔에 대한 부착 및 커맨드의 입력, 온도 및 다른 컴퓨터 건전성 관련 요소의 모니터링, 및 BIOS 셋팅 및 운영 체제로부터의 범주 밖의 다른 특징의 설정을 포함하지만 이에 한정되지 않는 특징을 허용할 수 있다. 컨트롤러(200)는 대역외 관리 네트워

크(260)를 통해 리소스(1310)를 알 수 있다. 또한 리소스의 유형을 식별하고 대역내 관리 또는 대역외 관리를 사용하여 그의 구성을 식별할 수 있다. 후술되는 도 13c 내지 도 13e는 물리 리소스(1310)를 IT 시스템(100a)에 추가하고 및/또는 시스템 보안을 향상시키는 방식으로 시스템(100)을 시동 또는 관리하기 위한 다양한 프로세스 흐름을 도시한다.

[0209] 네트워크, 네트워킹 리소스, 네트워크 디바이스 및/또는 네트워킹 인터페이스와 관련하여 본 명세서에서 사용되는 용어 "비활성화(disable)"는 이러한 네트워크, 네트워킹 리소스, 네트워크 디바이스 및/또는 네트워킹 인터페이스가 네트워크, 가상 네트워크(VLAN, VXLAN, infiniband 파티션을 포함하지만 이에 한정되지 않음)로부터 (수동 또는 자동적으로), 전원이 차단되고, 물리적으로 연결 해제되고, 및/또는 가상 또는 일부 다른 방식(예를 들어, 필터링)으로 연결 해제되는 조치를 지칭한다. 용어 "비활성화"는 또한 (소스로부터 데이터를 수신 또는 판독하는 능력을 여전히 가지면서) 리소스가 데이터를 목적지로 전송 또는 기록하는 것을 방지하는 것, (데이터를 목적지로 전송 또는 기록하는 능력을 여전히 가지면서) 리소스가 소스로부터 데이터를 수신 또는 판독하는 것을 방지하는 것과 같은 운용성의 일방 또는 단방향 제한을 강조한다. 이러한 네트워크, 네트워킹 리소스, 네트워크 디바이스 및/또는 네트워킹 인터페이스는 추가 네트워크, 가상 네트워크, 또는 리소스의 결합으로부터 연결해제될 수 있고, 이전에 연결된 네트워크, 가상 네트워크, 또는 리소스의 결합에 연결된 상태를 유지한다. 또한, 이러한 네트워킹 리소스 또는 디바이스는 하나의 네트워크, 가상 네트워크 또는 리소스와 다른 하나의 결합으로부터 전환될 수 있다.

[0210] 네트워크, 네트워킹 리소스, 네트워크 디바이스 및/또는 네트워킹 인터페이스와 관련하여 본 명세서에서 사용되는 용어 "활성화(enable)"는 이러한 네트워크, 네트워킹 리소스, 네트워크 디바이스 및/또는 네트워킹 인터페이스가 네트워크, 가상 네트워크(VLAN, VXLAN, infiniband 파티션을 포함하지만 이에 한정되지 않음)에 (수동 또는 자동적으로), 전원이 켜지고, 물리적으로 연결되고, 및/또는 가상 또는 일부 다른 방식으로 연결되는 조치를 지칭한다. 이러한 네트워크, 네트워킹 리소스, 네트워크 디바이스 및/또는 네트워킹 인터페이스는 다른 시스템 구성요소에 이미 연결된 경우에 추가 네트워크, 가상 네트워크, 또는 리소스의 결합에 연결될 수 있다. 또한, 이러한 네트워킹 리소스 또는 디바이스는 하나의 네트워크, 가상 네트워크 또는 리소스와 다른 하나의 결합으로부터 전환될 수 있다. 용어 "활성화"는 또한 (소스로부터 데이터를 제한하는 능력을 여전히 가지면서) 리소스가 목적지에 대해 데이터를 전송, 기록, 또는 수신 가능하게 하는 것, (목적지로부터 데이터를 제한하는 능력을 여전히 가지면서) 리소스가 소스로부터 데이터를 전송, 수신 또는 판독 가능하게 하는 것과 같은 운용성의 일방 또는 단방향 허용을 강조한다.

[0211] 컨트롤러 로직(205)은 추가된 하드웨어에 대해 대역외 관리 연결(260) 또는 대역내 관리 연결(270) 및/또는 추가된 하드웨어에 대한 구성 SAN(280)을 조사하도록 구성된다. 리소스(1310)가 검출된 경우에는, 컨트롤러 로직(205)이 글로벌 시스템 규칙(220)을 사용하여 리소스가 자동적으로 또는 사용자와 상호 작용하여 구성되는지를 결정할 수 있다. 그것이 자동적으로 추가되는 경우, 설정은 컨트롤러(200) 내의 글로벌 시스템 규칙(210)을 따를 것이다. 사용자에 의해 추가된 경우, 컨트롤러(200) 내의 글로벌 시스템 규칙(210)은 사용자가 리소스의 추가 및 사용자가 리소스(1310)로 수행하고자 하는 것을 확인하도록 요청할 수 있다. 컨트롤러(200)는 새로운 리소스가 허가되었는지 확인하기 위해 API 애플리케이션(들)에 질의하거나, 그렇지 않으면 스택을 제어하는 사용자 또는 임의의 프로그램에 요청할 수 있다. 허가 프로세스는 또한 새로운 리소스의 적합성을 확인하기 위해 암호화를 사용하여 자동적으로 안전하게 완료될 수 있다. 그 후, 컨트롤러 로직(205)은 리소스(1310)가 플러그되는 스위치 또는 네트워크를 포함하는 IT 시스템 상태(220)에 리소스(1310)를 추가한다.

[0212] 리소스가 물리적인 경우, 컨트롤러(200)는 대역외 관리 네트워크(260)를 통해 연산 리소스의 전원을 켤 수 있고, 리소스(1310)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(350)를 부트 오프할 수 있다. 이미지는 다른 네트워크 연결을 통해 또는 다른 리소스를 통해 간접적으로 로딩될 수 있다. 부팅되면, 리소스(1310)에 관한 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 이는 대역내 관리 및/또는 구성 SAN 또는 대역외 관리 연결을 통해 수행될 수 있다. 리소스(1310)는 글로벌 시스템 규칙(210) 및 컨트롤러 로직(205)을 사용하여, 예를 들어 SAN(280)을 통해 템플릿(230)으로부터 로딩된 이미지(350)를 부트 오프할 수 있다. 이미지는 다른 네트워크 연결을 통해 또는 다른 리소스를 통해 간접적으로 로딩될 수 있다. 부팅되면, 연산 리소스(310)에 관한 대역내 관리 연결(270)을 통해 수신된 정보가 또한 수집되고 IT 시스템 상태(220)에 추가될 수 있다. 그 후, 리소스(1310)가 스토리지 리소스 풀에 추가될 수 있고, 컨트롤러(200)에 의해 관리되고 IT 시스템 상태(220)에서 추적되는 리소스가 된다.

[0213] 대역내 관리 및/또는 구성 SAN은 컨트롤러(200)에 의해 리소스(1310)를 설정, 관리, 사용 또는 통신하고 임의의 커맨드 또는 작업을 실행하기 위해 사용될 수 있다. 그러나, 선택적으로, 대역내 관리 연결(270)은 컨트롤러

(200)에 의해 시스템(100) 또는 컨트롤러(200)의 설정, 관리, 사용 또는 동작 중에 또는 언제든지 꺼지거나 비활성화되도록 구성될 수 있다. 대역내 관리는 또한 시스템(100) 또는 컨트롤러(200)의 설정, 관리, 사용 또는 동작 중에 또는 언제든지 켜지거나 활성화되도록 구성될 수 있다. 선택적으로, 컨트롤러(200)는 대역내 관리 연결(270)로부터 컨트롤러(들)(200)로 리소스(1310)를 제어 가능하게 또는 전환 가능하게 연결 해제할 수 있다. 이러한 연결 해제 또는 연결 해제 가능성(disconnectability)은 물리적일 수 있고, 예를 들어 자동화된 물리 스위치 또는 스위치를 사용하여 네트워크로의 리소스의 대역내 관리 연결 및/또는 구성 SAN의 전원을 차단한다. 연결 해제는, 예를 들어 네트워크 스위치가 리소스(1310)의 대역내 관리(270) 및/또는 구성 SAN(280)에 연결된 포트에 대한 전력을 차단함으로써 달성될 수 있다. 이러한 연결 해제 또는 부분 연결 해제는 또한 소프트웨어 정의 네트워킹을 사용하여 달성될 수 있거나, 또는 소프트웨어 정의 네트워킹을 사용하여 컨트롤러에 대해 물리적으로 필터링될 수 있다. 이러한 연결 해제는 대역내 관리 또는 대역외 관리를 통해 컨트롤러에 의해 달성될 수 있다. 예시적인 실시형태에 따르면, 리소스(1310)가 IT 시스템에 추가되기 전, 그 동안 또는 후에 임의의 시점에서, 리소스(1310)는 컨트롤러(200)로부터의 선택적인 제어 명령어에 응답하여 대역내 관리 연결(270)로부터 연결 해제될 수 있다.

[0214] 소프트웨어 정의 네트워킹을 사용하여, 대역내 관리 연결(270) 및/또는 구성 SAN(280)은 일부 기능을 보유하거나 그렇지 않을 수 있다. 대역내 관리(270) 및/또는 구성 SAN(280)은 컨트롤러(200)와의 통신 또는 다른 자원과의 통신을 위해 제한된 연결로서 사용될 수 있다. 연결(270)은 공격자가 컨트롤러(200), 다른 네트워크 또는 다른 리소스에 피벗하는 것을 방지하기 위해 제한될 수 있다. 시스템은 리소스(1310)를 손상시키지 않도록 컨트롤러(200) 및 리소스(1310)와 같은 디바이스가 공개적으로 통신하는 것을 방지하도록 구성될 수 있다. 예를 들어, 대역내 관리(270) 및/또는 구성 SAN(280)에서, 소프트웨어 정의 네트워킹 또는 하드웨어 변경 방법(예컨대, 전자적 제한)을 통해, 대역내 관리 및/또는 구성 SAN만이 데이터를 전송하지만 아무것도 수신하지 못하게 할 수 있다. 대역내 관리 및/또는 구성 SAN은, 물리적으로 또는 컨트롤러로부터 리소스로 기록하는 것만을 가능하게 하는 소프트웨어 정의 네트워킹을 사용하여 일방향 기록 구성요소가 되도록 또는 컨트롤러(200)로부터 리소스(1310)로의 일방향 기록 연결로서 구성될 수 있다. 연결의 일방향 기록 속성은 또한 보안 및 시스템의 동작의 다른 단계 또는 시간에 대한 바람직함에 따라 제어되거나 켜지거나 꺼질 수 있다. 시스템은 또한 리소스로부터 컨트롤러로의 기록 또는 통신이 예를 들어 로그 또는 경보를 통신하기 위해 제한되도록 구성될 수 있다. 인터페이스는 소프트웨어 정의 네트워킹, VLANS, VXLANS 및/또는 infiniband 파티셔닝을 포함하지만 이에 한정되지 않는 기술에 의해 다른 네트워크로 이동되거나 네트워크로부터 추가 및 제거될 수 있다. 예를 들어, 인터페이스는 설정 네트워크에 연결되고, 해당 네트워크로부터 제거되고 런타임에 사용되는 네트워크로 이동될 수 있다. 컨트롤러로부터 리소스로의 통신은 컨트롤러가 리소스(1310)로부터 전송된 임의의 데이터에 물리적으로 응답할 수 없도록 차단되거나 제한될 수 있다. 일례에 따르면, 리소스(1310)가 추가되고 부팅되면, 대역내 관리(270)는 물리적으로 또는 소프트웨어 정의 네트워킹을 사용하여 스위치 오프되거나 필터링될 수 있다. 대역내 관리는 로그 관리 전용의 다른 리소스에 데이터를 전송할 수 있도록 구성될 수 있다.

[0215] 대역내 관리는 대역외 관리 또는 소프트웨어 정의 네트워킹을 사용하여 켜지거나 꺼질 수 있다. 연결 해제된 대역내 관리에 의해, 실행중인 데몬은 필요하지 않을 수 있으며 대역내 관리는 키보드 기능을 사용하여 재활성화될 수 있다.

[0216] 또한, 선택적으로, 리소스(1310)는 대역내 관리 연결을 갖지 않을 수 있고 리소스는 대역외 관리를 통해 관리될 수 있다.

[0217] 대역외 관리는 대안적으로 또는 추가적으로, 예를 들어 키보드, 가상 키보드, 디스크 마운팅 콘솔, 가상 디스크의 부착, 바이오스 셋팅의 변경, 부트 파라미터 및 시스템의 다른 양태의 변경, 부팅 가능한 이미지 또는 설치 CD에 존재할 수 있는 기존의 스크립트의 실행, 또는 컨트롤러(200) 및 리소스(1310)가 리소스(1310)에서 실행하는 운영 체제의 노출이 있거나 없이 통신하는 것을 허용하기 위한 대역외 관리의 다른 특징을 포함하지만 이에 한정되지 않는 것에 의해 시스템의 다양한 양태를 조작하는 데 사용될 수 있다. 예를 들어, 컨트롤러(200)는 대역외 관리(260)에 의해 이러한 툴을 사용하여 커맨드를 전송할 수 있다. 컨트롤러(200)는 또한 리소스(1310)의 제어를 지원하기 위해 이미지 인식을 사용할 수 있다. 따라서, 대역외 관리 연결을 사용하여, 시스템은 대역외 관리 연결을 통해 시스템에 연결되는 리소스의 바람직하지 않은 조작을 방지하거나 피할 수 있다. 대역외 관리 연결은 또한 시스템의 동작 동안 또는 시스템의 동작 동안 선택된 시간에 일방향 통신 시스템으로서 구성될 수 있다.

[0218] 또한, 대역외 관리 연결(260)은 또한 실무자가 원하는 경우 대역내 관리 연결과 동일한 방식으로 컨트롤러(20

0)에 의해 선택적으로 제어될 수 있다.

[0219] 컨트롤러(200)는 글로벌 시스템 규칙에 따라 자동적으로 리소스를 켜고 끌 수 있으며, 전력을 절약하기 위해 리소스를 끄는 것 또는 애플리케이션 성능을 개선하기 위해 리소스를 켜는 것 또는 IT 시스템 사용자가 가질 수 있는 임의의 다른 이유와 같은 IT 시스템 사용자에 의해 결정된 이유로 IT 시스템 상태를 업데이트할 수 있다. 컨트롤러는 또한 대역내 및 대역외 관리 연결에서, SAN 구성을 켜고 끌 수 있거나, 또는 시스템 작동 동안 및 다양한 보안 목적으로(예를 들어, 리소스(1310)가 외부 네트워크(1380) 또는 내부 네트워크(390)에 연결되는 동안 대역내 관리 연결(270) 또는 구성 SAN(280)을 비활성화함) 이러한 연결을 일방향 기록 연결로서 지정할 수 있다. 일방향 대역내 관리는 또한, 예를 들어 시스템의 건전성을 모니터링하기 위해 사용될 수 있고, 이는 운영 체제에 보일 수 있는 로그 및 정보를 모니터링하는 것이다.

[0220] 리소스(1310)는 또한 서비스, 애플리케이션, 사용자 및/또는 클라이언트가 서로 통신할 수 있는 애플리케이션 네트워크와 같은 하나 이상의 내부 네트워크(390)에 결합될 수 있다. 이러한 애플리케이션 네트워크(390)는 또한 외부 네트워크(1380)에 연결되거나 연결 가능할 수 있다. 도 2a 내지 도 12b를 포함하지만 이에 한정되지 않는 본 명세서의 예시적인 실시형태에 따르면, 대역내 관리는 리소스 또는 애플리케이션 네트워크가 외부 네트워크에 연결되는 경우 또는 리소스가 외부 네트워크에 연결되지 않은 애플리케이션 네트워크에 연결되는 경우에 추가 보안을 제공하기 위해, 리소스 또는 애플리케이션 네트워크(390)로부터 연결 해제되거나, 연결 해제 가능할 수 있거나, 또는 컨트롤러로부터 일방향 기록을 제공할 수 있다.

[0221] 도 13a의 IT 시스템(100)은 도 3b에 나타낸 바와 같이 IT 시스템(100)과 유사하게 구성될 수 있으며; 이미지(350)는 연산 리소스 부팅 및/또는 애플리케이션 로딩을 위해 템플릿(230)으로부터 리소스(1310)로 직접 또는 간접적으로(다른 리소스 또는 데이터베이스를 통해) 로딩될 수 있다. 이미지(350)는 리소스 유형 및 하드웨어에 대한 부트 파일(340)을 포함할 수 있다. 부트 파일(340)은 배치될 리소스, 애플리케이션 또는 서비스에 대응하는 커널(341)을 포함할 수 있다. 부트 파일(340)은 또한 부팅 프로세스를 지원하는 데 사용되는 initrd 또는 유사한 파일시스템을 포함할 수 있다. 부트 시스템(340)은 상이한 하드웨어 유형 및 리소스 유형을 위해 구성된 복수의 커널 또는 initrd를 포함할 수 있다. 또한, 이미지(350)는 파일시스템(351)을 포함할 수 있다. 파일 시스템(351)은 베이스 이미지(352) 및 대응하는 파일시스템뿐만 아니라 서비스 이미지(353) 및 대응하는 파일시스템 및 휘발성 이미지(354) 및 대응하는 파일시스템을 포함할 수 있다. 로딩되는 파일 시스템 및 데이터는 리소스 유형과 실행할 애플리케이션 또는 서비스에 따라 다를 수 있다. 베이스 이미지(352)는 기본 운영 체제 파일 시스템을 포함할 수 있다. 기본 운영 체제는 판독 전용일 수 있다. 베이스 이미지(352)는 또한 실행하고 있는 것과 무관한 운영 체제의 기본 툴을 포함할 수 있다. 베이스 이미지(352)는 기본 디렉토리 및 운영 체제 툴을 포함할 수 있다. 서비스 파일시스템(353)은 리소스, 애플리케이션 또는 서비스에 대한 구성 파일 및 사양을 포함할 수 있다. 휘발성 파일시스템(354)은 이진 애플리케이션, 특정 주소 및 다른 정보와 같은 해당 배치에 고유한 정보 또는 데이터를 포함할 수 있으며, 이는 암호, 세션 키 및 개인 키를 포함하지만 이에 한정되지 않는 변수로서 구성되거나 그렇지 않을 수 있다. 파일 시스템은 일부 판독 전용 및 일부 판독-기록 파일시스템을 허용하기 위해 overlayFS와 같은 기술을 사용하여 하나의 단일 파일시스템으로서 마운트될 수 있어, 애플리케이션에 사용되는 중복 데이터의 양을 감소시킨다.

[0222] 도 13b는 하나 이상의 가상 머신을 호스팅하거나 포함하는 하나 이상의 하이퍼바이저(1311)를 각각 포함하는 복수의 리소스(1310)를 도시한다. 컨트롤러(200a)는 베어 메탈 리소스를 각각 포함하는 리소스(1310)에 결합된다. 리소스(1310)는 도 13b를 참조하여 나타내고 설명된 바와 같이 컨트롤러(200a)에 각각 결합된다. 본 명세서의 예시적인 실시형태에 따르면, 대역내 관리 연결(270), 구성 SAN(280) 및/또는 대역외 관리 연결(260)은 도 13a와 관련하여 설명된 바와 같이 구성될 수 있다. 가상 머신 또는 하이퍼바이저 중 하나 이상이 침입받을 수 있거나 침입받게 될 수 있다. 종래 시스템에서는, 그러면 다른 하이퍼바이저 상의 다른 가상 머신이 침입받을 수 있게 된다. 예를 들어, 이는 가상 머신 내에서 하이퍼바이저 익스플로잇 실행(hypervisor exploit run)으로부터 발생할 수 있다. 예를 들어, 피벗팅은 침입받은 하이퍼바이저로부터 컨트롤러 200a)로 진행할 수 있고, 침입받은 컨트롤러(200a)로부터 컨트롤러(200a)에 결합된 다른 하이퍼바이저로 진행할 수 있다. 예를 들어, 둘 다 연결된 네트워크를 사용하여 침입받은 하이퍼바이저와 타겟 하이퍼바이저 간에 피벗팅이 발생할 수 있다. 임의의 것 또는 전부가 컨트롤러(200a)와 리소스(1310) 간의 제공된 링크에서 대역내(또는 구성 SAN) 및 대역외 연결을 비활성화하도록 선택적으로 제어될 수 있는, 도 13b에 도시된 컨트롤러(200a)의 대역내 관리(270), 구성 SAN(280) 또는 대역외 관리(260) 및 리소스(1310)의 배열은 하나의 하이퍼바이저를 벗어나 다른 리소스로 피벗팅하기 위해 침입받은 가상 머신이 사용되는 것을 방지할 수 있다.

[0223] 상기 도 1 내지 도 12와 관련하여 설명된 대역내 관리 연결(270) 및 대역외 관리 연결(260)은 또한 도 13a 및

도 13b와 관련하여 설명된 것과 유사하게 구성될 수 있다.

[0224] 도 13c는 시스템(100)에 베어메탈 노드와 같은 물리 리소스를 추가 또는 관리하기 위한 예시적인 프로세스 흐름을 도시한다. 도 13a 및 도 13b에 나타낸 바와 같이 또는 본 명세서에서 도 1 내지 도 12와 관련하여 나타낸 바와 같이 리소스(1310)는 대역외 관리 연결(260) 및 대역내 관리 연결(270) 및/또는 SAN을 통해 시스템(100)의 컨트롤러에 연결될 수 있다.

[0225] 리소스의 연결의 인스턴스 후에, 단계 1370에서 외부 네트워크 및/또는 애플리케이션 네트워크가 비활성화된다. 위에서 언급한 바와 같이, 다양한 기술 중 임의의 것이 이 비활성화에 사용될 수 있다. 예를 들어, 대역내 관리 연결 또는 구성 SAN을 사용하여, 시스템을 설정하거나, 리소스를 추가하거나, 시스템을 테스트하거나, 시스템을 업데이트하거나, 또는 다른 작업 또는 커맨드를 수행하기 전에, 시스템(100)의 구성요소(또는 공격에 취약한 것만)가 도 13a 및 도 13b와 관련하여 설명된 바와 같이 임의의 외부 네트워크 또는 애플리케이션 네트워크로부터 비활성화, 연결 해제 또는 필터링된다.

[0226] 단계 1370 후에, 대역내 관리 연결 및/또는 구성 SAN은 단계 1371에서 활성화된다. 따라서, 단계 1370 및 1371의 조합은 대역내 관리 및/또는 SAN 연결이 작동하는 동안 외부 네트워크 및/또는 애플리케이션 네트워크로부터 리소스를 격리시킨다. 그 후, 커맨드가 대역내 관리 연결을 통해 컨트롤러(200)의 제어 하에 리소스 상에서 실행될 수 있다(단계 1372 참조). 예를 들어, 도 1 내지 도 13b와 관련하여 본 명세서에 설명된 것을 포함하지만 이에 한정되지 않는 것과 같은 설정 및 구성 단계가 그 후에 대역내 관리 및/또는 구성 SAN을 사용하여 단계 1372에서 수행될 수 있다. 대안적으로 또는 추가적으로, 대역내 관리 및/또는 구성 SAN을 사용하여, 시스템의 동작, 업데이트 또는 관리(임의의 변경 관리 또는 시스템 업데이트를 포함할 수 있지만 이에 한정되지 않음), 테스트, 업데이트, 데이터 전송, 성능 및 건정성에 대한 정보 수집(에러, CPU 사용량, 네트워크 사용량, 파일시스템 정보, 및 스토리지 사용량을 포함하지만 이에 한정되지 않음), 및 로그뿐만 아니라 본 명세서에서 도 1 내지 도 13b에 설명된 바와 같이 시스템(100)을 관리하는 데 사용될 수 있는 다른 커맨드의 수집을 포함하지만 이에 한정되지 않는 다른 작업이 단계 1372에서 수행될 수 있다.

[0227] 리소스를 추가하고, 시스템을 설정하고 및/ 는 이러한 작업 또는 커맨드를 수행한 후, 리소스와 컨트롤러 또는 시스템의 다른 구성요소 사이의 대역내 관리 연결(270) 및/또는 구성 SAN(280)은 도 13a 및 도 13b와 관련하여 본 명세서에 설명된 바와 같이 하나 이상의 방향으로 단계 1373에서 비활성화될 수 있다. 이러한 비활성화는 전술한 바와 같이 연결 해제, 필터링 등을 이용할 수 있다. 단계 1373 후에, 단계 1374에서 외부 네트워크 및/또는 애플리케이션 네트워크로의 연결이 복원될 수 있다. 예를 들어, 컨트롤러는 리소스(1310)가 애플리케이션 네트워크 또는 인터넷에 연결하는 것을 가능하게 하도록 네트워킹 리소스에 전할 수 있다. 시스템이 테스트 또는 업데이트되는 동일한 단계가 후속될 수 있고, 즉, 외부 네트워크 및/또는 애플리케이션에 대한 대역내 관리 연결은 대역내 관리 연결을 리소스에 활성화 또는 연결(일방향 또는 양방향)하기 전에 연결 해제 또는 필터링될 수 있다. 따라서, 단계 1373 및 1374는 리소스가 외부 네트워크 및/또는 애플리케이션 네트워크에 연결되는 동안 대역내 관리 연결 및/또는 구성 SAN을 통해 컨트롤러에 연결하는 것을 격리시키도록 함께 동작한다.

[0228] 대역외 관리는 시스템 또는 리소스를 관리하거나 시스템 또는 리소스를 설정하거나, 시스템 또는 리소스를 구성, 부팅 또는 추가하는 데 사용될 수 있다. 본 명세서의 임의의 실시형태에서 사용되는 대역외 관리는, 부팅 전에 셋팅을 변경하기 위해 가상 키보드를 사용하여 머신에 커맨드를 전송할 수 있고, 또한 가상 키보드에 입력함으로써 운영 체제에 커맨드를 전송할 수 있고; 머신이 로그인되어 있지 않으면, 대역외 관리는 가상 키보드를 사용하여 사용자명과 암호를 입력할 수 있고, 이미지 인식을 사용하여 로그온을 검증하고 입력한 커맨드를 검증하고 실행 여부를 검사할 수 있다. 물리 리소스가 그래픽 콘솔만을 갖는 경우, 가상 마우스도 사용될 수 있으며 이미지 인식은 대역외 관리가 변경을 행할 수 있게 할 것이다.

[0229] 도 13d는 시스템(100)에 베어메탈 노드와 같은 물리 리소스를 추가 또는 관리하기 위한 다른 예시적인 프로세스 흐름을 도시한다. 단계 1380에서, 도 13a 및 도 13b에 나타낸 바와 같이 또는 본 명세서의 도 1 내지 도 12에 나타낸 바와 같이 리소스는 대역외 관리(260)를 통해 시스템 또는 리소스에 연결될 수 있다. 컨트롤러에 의해 용이하게 되는 대역외 관리를 통해 디스크 이미지(예를 들어, ISO 이미지)에 대한 액세스를 제공함으로써 디스크가 가상으로 연결될 수 있다(단계 1381 참조). 그 후. 리소스 또는 시스템은 디스크 이미지로부터 부팅될 수 있고(단계 1382), 그 후에 파일은 디스크 이미지로부터 부팅 가능한 디스크로 복사된다(단계 1383 참조). 이는 또한 대역외 관리를 사용하여 이러한 방식으로 리소스가 설정되는 시스템을 부팅하는 데 사용될 수도 있다. 이는 또한 복수의 리소스가 컨트롤러를 포함하는지 또는 시스템을 구성하는지의 여부에 관계없이 함께 결합될 수 있는 (네트워킹 리소스를 포함하지만 이에 한정되지 않는) 복수의 리소스를 구성 및/또는 부팅하는 데 사용될

수 있다. 따라서, 가상 디스크는 가상 디스크가 리소스에 부착된 것처럼 컨트롤러가 디스크 이미지를 리소스에 연결할 수 있게 하는 데 사용될 수 있다. 대역외 관리는 또한 파일을 리소스로 전송하기 위해 사용될 수 있다. 단계 1383에서 데이터가 가상 디스크로부터 로컬 디스크에 복사될 수 있다. 디스크 이미지는 리소스가 그의 동작에 복사하여 사용할 수 있는 파일을 포함할 수 있다. 파일은 대역외 관리로부터 스케쥴된 프로그램 또는 명령어를 통해 복사되거나 사용될 수 있다. 컨트롤러는 대역외 관리를 통해 가상 키보드를 사용하여 리소스에 로그온하고 커맨드를 입력하여 가상 디스크로부터 그 자체의 디스크 또는 리소스에 액세스 가능한 다른 스토리지로 파일을 복사할 수 있다. 단계 1384에서, 시스템 또는 리소스는 bios, efi 또는 부팅 순서 셋팅을 설정하여 부팅하도록 구성되어, 부팅 가능한 디스크로부터 부팅할 것이다. 부팅 구성은 직접 대역외 관리를 통해 또는 인스톨러 스크립드 내에 포함시킴으로써 실행할 수 있는 efibootmgr과 같은 EFI 관리자를 운영 체제에서 사용할 수 있다(예를 들어, 리소스가 부팅할 때 efibootmgr을 사용하는 스크립트를 자동적으로 실행한다). 또한, 부팅 옵션 및 임의의 다른 bios 변경은 부팅 순서 커맨드를 사용하거나 bios 구성(예를 들어, Supermicro Update Manager가 지원하는 XML BIOS 구성)을 사용하여 Supermicro Boot Manager와 같은 대역외 관리 도구를 통해 설정될 수 있다. bios는 또한 부팅 순서를 포함하여 적절한 bios 셋팅을 설정하기 위해 키보드 및 콘솔로부터의 이미지 인식을 사용하여 구성될 수 있다. 인스톨러는 로딩되는 사전 구성된 이미지에서 실행될 수 있다. 구성은 화면을 보고 이미지 인식을 사용하여 테스트될 수 있다. 구성 후, 리소스는 (예를 들어, 전원을 켜거나, 부팅되거나, 애플리케이션 네트워크에 연결되거나, 또는 이들의 조합으로) 활성화될 수 있다(단계 1385).

[0230] 도 13e는 이 경우에 PXE, Flexboot 또는 유사한 네트워크 부팅을 사용하여, 베어메탈 노드와 같은 물리 리소스를 시스템(100)에 추가 또는 관리하기 위한 다른 예시적인 프로세스 흐름을 도시한다. 단계 1390에서, 도 13a 및 도 13b에 나타낸 바와 같이 또는 본 명세서에서 도 1 내지 도 12와 관련하여 나타낸 바와 같이 리소스(1310)는 (1) 대역내 관리 연결(270) 및/또는 SAN 및 (2) 대역외 관리 연결(260)을 통해 시스템(100)의 컨트롤러에 연결될 수 있다. 그 후, 외부 네트워크 및/또는 애플리케이션 네트워크 연결이 단계 1391에서(단계 1370과 관련하여 전술한 것과 마찬가지로) 비활성화될 수 있다(예를 들어, SDN과 전체 또는 부분적으로, 물리적으로, 또는 가상으로 필터링 또는 연결 해제될 수 있다). 예를 들어, 대역내 관리 연결 또는 SAN을 사용하여, 시스템을 설정하거나, 리소스를 추가하거나, 시스템을 테스트하거나, 시스템을 업데이트하거나, 또는 다른 작업 또는 커맨드를 수행하기 전에, 시스템(100)의 구성요소(또는 공격에 취약한 것만)가 도 13a 및 도 13b와 관련하여 설명된 바와 같이 임의의 외부 네트워크 또는 애플리케이션 네트워크로부터 비활성화, 연결 해제 또는 필터링된다.

[0231] 단계 1392에서, 리소스의 유형이 결정된다. 예를 들어, 리소스 관련 정보는 대역외 관리 툴을 사용하거나 디스크가 리소스에 부착된 것처럼 디스크 이미지(예를 들어, ISO 이미지)를 리소스에 연결하여 mac 주소로부터 수집되어, 리소스 정보를 식별하는 데 사용될 수 있는 툴을 갖는 운영 체제를 일시적으로 부트업할 수 있다. 단계 1393에서, 리소스는 그 후에 PXE 또는 플렉스부팅(flexbooting) 등에 사전 구성되는 것으로 구성되거나 식별된다. 그 후, 단계 1394에서 PXE, Flexboot 또는 유사한 부팅을 수행하기 위해(또는 일시적으로 부팅되고, 재차 전원이 켜지는 경우에) 리소스의 전원이 켜진다. 그 후, 리소스는 단계 1395에서 대역내 관리 연결 또는 SAN으로부터 부팅된다. 단계 1396에서, 데이터는 도 13d의 단계 1383을 참조하여 설명된 것과 유사한 방식으로 리소스에 의해 액세스 가능한 디스크에 복사된다. 그 후, 단계 1397에서, 리소스는 도 13d의 단계 1384와 관련하여 전술한 것과 유사한 방식으로 디스크(들)로부터 부팅하도록 구성된다. 리소스가 PXE, 플렉스부팅 등을 위해 사전 구성된 것으로 식별되는 경우, 파일은 1393 내지 1396의 임의의 단계에서 복사될 수 있다. 대역내 관리가 활성화된 경우, 단계 1398에서 비활성화될 수 있고, 단계 1399에서 애플리케이션 네트워크 또는 외부 네트워크가 재연결되거나 활성화될 수 있다.

[0232] 또한, OOBM 이외의 기술을 사용하여 리소스를 원격으로 활성화(예를 들어 전원 켜기)하고 부팅되었는지 검증할 수 있다. 예를 들어, 시스템은 사용자에게 전원 버튼을 누르고 시스템이 부팅되었다는 것을 컨트롤러에 수동으로 알려주도록(또는 컨트롤러에 키보드/콘솔 연결을 사용하도록) 프롬프트할 수 있다. 또한, 시스템은 부팅되면 IBM을 통해 컨트롤러를 핑(ping)할 수 있고 컨트롤러가 로그온하여 (예를 들어, ssh, telnet 또는 네트워크를 통한 다른 방법과 같은 방법을 통해) 재부팅하도록 알려준다. 예를 들어, 컨트롤러는 ssh에 들어가서 재부팅 커맨드를 전송할 수 있다. PXE를 사용 중이고 OOBM이 없는 경우, 어떤 경우에도 시스템은 원격으로 리소스의 전원을 켜거나 사용자에게 수동으로 전원을 켜도록 지시할 수 있는 방법을 가져야 한다.

[0233] *컨트롤러 및/또는 환경의 배치:*

[0234] 예시적인 실시형태에서, 컨트롤러는 발신 컨트롤러(200)로부터 시스템 내에 배치될 수 있다(여기서, 이러한 발신 컨트롤러(200)는 "메인 컨트롤러"라고 지칭될 수 있다). 따라서, 메인 컨트롤러는 격리되거나 격리 가능한

IT 시스템 또는 환경일 수 있는 시스템 또는 환경을 설정할 수 있다.

[0235] 본 명세서에 설명된 환경은 서로 상호 작용할 수 있는 컴퓨터 시스템 내의 리소스의 집합을 지칭한다. 컴퓨터 시스템은 그 안에 여러 환경을 포함할 수 있지만; 이것이 반드시 그럴 필요는 없다. 환경의 리소스(들)는 환경에서 실행하는 하나 이상의 인스턴스, 애플리케이션 또는 서브애플리케이션을 포함할 수 있다. 또한, 환경은 하나 이상의 환경 또는 하위 환경을 포함할 수 있다. 환경은 컨트롤러를 포함하거나 그렇지 않을 수 있으며, 환경은 하나 이상의 애플리케이션을 운영할 수 있다. 환경의 이러한 리소스는, 예를 들어 네트워킹 리소스, 연산 리소스, 스토리지 리소스 및/또는 환경의 애플리케이션을 포함하는 특정 환경을 실행하는 데 사용되는 애플리케이션 네트워크를 포함할 수 있다. 따라서, 환경은 하나 이상의 애플리케이션의 기능을 제공할 수 있음을 이해해야 한다. 일부 예에서, 본 명세서에 설명된 환경은 다른 환경과 물리적으로 또는 가상으로 분리되거나 분리 가능할 수 있다. 또한, 다른 예에서, 환경은 다른 환경에 대한 네트워크 연결을 가질 수 있으며, 여기서 이러한 연결은 필요에 따라 비활성화 또는 활성화될 수 있다.

[0236] 또한, 메인 컨트롤러는 다양한 환경에서 또는 별도의 시스템으로서 하나 이상의 추가 컨트롤러를 설정, 배치 및/또는 관리할 수 있다. 이러한 추가 컨트롤러는 메인 컨트롤러와 독립적이거나 독립될 수 있다. 이러한 추가 컨트롤러는 메인 컨트롤러와 독립적이거나 준독립적인 경우에도 작동 중에 다양한 시간에 메인 컨트롤러(또는 모니터링 애플리케이션을 통해 별도의 모니터 또는 환경)로부터 지시를 받거나 그에 정보를 전송할 수 있다. 환경은 (예를 들어, 서로 및/또는 메인 컨트롤러로부터 환경을 격리 가능하게 함으로써) 보안 목적 및/또는 다양한 관리 목적을 위해 구성될 수 있다. 환경은 외부 네트워크에 연결될 수 있지만 다른 관련 환경은 외부 네트워크에 연결되거나 연결되지 않을 수 있다.

[0237] 메인 컨트롤러는 별도의 시스템인지 여부와 컨트롤러 또는 서브컨트롤러를 포함하는지 여부에 관계없이 환경 또는 애플리케이션을 관리할 수 있다. 메인 컨트롤러는 또한 글로벌 구성 파일 또는 다른 데이터의 공유 스토리지를 관리할 수 있다. 메인 컨트롤러는 또한 그 기능에 따라 글로벌 시스템 규칙(예를 들어, 시스템 규칙(210)) 또는 그 서브세트를 다른 컨트롤러로 파싱할 수 있다. 각각의 새로운 컨트롤러("서브컨트롤러"라고 지칭될 수 있음)는 메인 컨트롤러의 구성 규칙의 서브세트 일 수 있는 새로운 구성 규칙을 수신할 수 있다. 컨트롤러에 배치된 글로벌 구성 규칙의 서브세트는 설정중인 IT 시스템의 유형에 의존하거나 그에 대응할 수 있다. 메인 컨트롤러는, 예를 들어 운송 또는 배포 또는 다른 것을 위해 메인 컨트롤러로부터 영구적으로 분리되는 새로운 컨트롤러 또는 별도의 IT 시스템을 설정 또는 배치할 수 있다. 글로벌 구성 규칙(또는 그의 서브세트)은 다양한 환경에서 애플리케이션 또는 서브애플리케이션을 설정하기 위한 프레임워크 및 이들이 서로 상호 작용하는 방법을 정의할 수 있다. 이러한 애플리케이션 또는 환경은 메인 컨트롤러에 의해 배치된 글로벌 구성 규칙의 서브세트를 포함하는 서브컨트롤러에서 실행할 수 있다. 일부 예에서, 이러한 애플리케이션 또는 환경은 메인 컨트롤러에 의해 관리될 수 있다. 그러나, 다른 예에서는 이러한 애플리케이션 또는 환경이 메인 컨트롤러에 의해 관리되지 않는다. 애플리케이션 또는 환경을 관리하기 위해 메인 컨트롤러로부터 새로운 컨트롤러가 생성되는 경우, 새로운 컨트롤러에 의한 제어를 용이하게 하도록 다수의 애플리케이션에 걸쳐 애플리케이션을 검사하는 종속성이 있을 수 있다.

[0238] 따라서, 예시적인 실시형태에서, 시스템은 다른 컨트롤러 또는 이러한 다른 컨트롤러를 포함하는 IT 시스템을 배치하도록 구성된 메인 컨트롤러를 포함할 수 있다. 이러한 구현된 시스템은 메인 컨트롤러로부터 완전히 분리되도록 구성될 수 있다. 독립되면, 이러한 시스템은 독립형 시스템으로서 동작하도록 구성될 수 있고; 또는 동작 중에 다양한 이산 또는 연속 시간에 메인 컨트롤러와 같은 다른 컨트롤러(또는 애플리케이션이 있는 환경)에 의해 제어되거나 모니터링될 수 있다.

[0239] 도 14a는 메인 컨트롤러(1401)가 상이한 시스템(1400a 및 1400b)에 컨트롤러(1401a 및 1401b)를 각각 배치한 예시적인 시스템을 나타낸다(여기서 1400a 및 1400b는 서브시스템이라고 지칭될 수 있지만; 서브시스템(1400a 및 1400b)도 환경의 역할을 할 수 있음을 이해해야 한다). 메인 컨트롤러(1401)는 전술한 컨트롤러(200)와 유사한 방식으로 구성될 수 있다. 이와 같이, 그것은 컨트롤러 로직(205), 글로벌 시스템 규칙(210), 시스템 상태(220) 및 템플릿(230)을 포함할 수 있다.

[0240] 시스템(1400a 및 1400b)은 각각 리소스(1420a, 1420b)에 각각 결합된 컨트롤러(1401a, 1401b)를 포함한다. 메인 컨트롤러(1401)는 서브시스템(1400a)의 컨트롤러(1401a) 및 서브시스템 (1400b)의 컨트롤러(1401b)와 같은 하나 이상의 다른 컨트롤러에 결합될 수 있다. 메인 컨트롤러(1400)의 글로벌 규칙(210)은 다른 컨트롤러를 관리 및 제어할 수 있는 규칙을 포함할 수 있다. 메인 컨트롤러(1401)는 컨트롤러 로직(205), 시스템 상태(220) 및 템플릿(230)과 함께 이러한 글로벌 규칙(210)을 사용하여, 본 명세서에서 도 1 내지 도 13e를 참조하여 설명

된 것과 유사한 방식으로 컨트롤러(1401a, 1401b)를 통해 서브시스템(1400a, 1400b)을 설정, 제공 및 배치할 수 있다.

[0241] 예를 들어, 메인 컨트롤러(1401)는 글로벌 규칙(210)(또는 그 서브세트)이 컨트롤러(1401a, 1401b) 및 그들의 서브시스템(1400a, 1400b)의 동작을 지시하는 방식으로 각각 글로벌 규칙(210)(또는 그 서브세트)을 규칙(1410a, 1410b)으로서 서브시스템(1400a, 1400b)에 로딩할 수 있다. 각각의 컨트롤러(1401a, 1401b)는 글로벌 규칙(210)의 동일한 또는 다른 서브세트일 수 있는 규칙(1410a, 1410b)을 가질 수 있다. 예를 들어, 글로벌 규칙(210)의 어느 서브세트가 제공된 서브시스템에 제공되는지는 배치되는 서브시스템의 유형에 의존할 수 있다. 컨트롤러(1401)는 또한 시스템 리소스(1420a, 1420b) 또는 컨트롤러(1401a, 1401b)에 로딩되도록 데이터를 로딩 또는 지시할 수 있다.

[0242] 메인 컨트롤러(1401)는 본 명세서에서 설명되는 방식으로; 예를 들어 도 13a 내지 도 13e에서 설명되는 리소스의 배치 및 관리를 참조하여, 다양한 배치 또는 관리의 단계에서 활성화 또는 비활성화될 수 있는 대역내 관리 연결(들)(270) 및/또는 대역외 관리 연결(들)(260) 또는 SAN 연결(280)을 통해 다른 컨트롤러(1401a, 1401b)에 연결될 수 있다. 대역내 관리 연결(270) 또는 대역외 관리 연결(260)의 선택적인 활성화 및 비활성화를 사용하여, 서브시스템(1400a, 1400b)은 다양한 시간에 서브시스템(1400a, 1400b)이 서로에 대하여 메인 시스템(100) 또는 컨트롤러(1401)의 지식(또는 제한된, 제어된 또는 한정된 지식)을 갖지 않을 수 있는 방식으로 배치될 수 있다

[0243] 예시적인 실시형태에서, 메인 컨트롤러(1401)는 메인 컨트롤러(1401)에 의해 배치되고 구성된 로컬 컨트롤러(1401a, 1401b)를 갖는 중앙 집중식 IT 시스템을 작동시켜 메인 컨트롤러(1401)가 복수의 IT 시스템을 배치 및/또는 실행할 수 있다. 이러한 IT 시스템은 서로 독립적이거나 독립적이지 않을 수 있다. 메인 컨트롤러(1401)는 생성된 IT 시스템으로부터 격리되거나 에어 갭(air-gapped)되는 별도의 애플리케이션으로서 모니터링을 설정할 수 있다. 모니터링을 위한 별도의 콘솔에는 메인 컨트롤러와 로컬 컨트롤러 사이의 연결 및/또는 선택적으로 활성화 또는 비활성화될 수 있는 환경들 사이의 연결이 제공될 수 있다. 컨트롤러(1401)는, 예를 들어 비즈니스를 포함하지만 이에 한정되지 않는 다양한 용도를 위한 격리된 시스템, 데이터 스토리지를 제조하기 위한 시스템, 데이터 센터, 및 다른 다양한 기능적 노드를 배치할 수 있으며, 각각이 정전 또는 침입의 경우에 다른 컨트롤러를 갖는다. 이러한 격리는 완전하거나 영구적일 수 있거나, 또는 예를 들어 일시적, 시간 또는 작업 의존, 통신 방향 의존 또는 다른 파라미터 의존과 같이 준격리될 수 있다. 예를 들어, 메인 컨트롤러(1401)는 특정의 미리 정의된 상황으로 제한될 수 있거나 제한되지 않을 수 있는 명령어를 시스템에 제공하도록 구성될 수 있는 반면, 서브시스템은 메인 컨트롤러와의 통신 능력이 제한되거나 없을 수 있다. 따라서, 이러한 서브시스템은 메인 컨트롤러(1401)를 침입하지 못할 수 있다. 메인 컨트롤러(1401)와 서브컨트롤러(1401a, 1401b)는, 예를 들어 본 명세서에서 설명되는 바와 같이(후술하는 특정예와 함께), 대역내 관리(270)를 비활성화하는 것에 의해, 일방향 기록하는 것에 의해 및/또는 대역외 관리(260)로의 통신을 제한하는 것에 의해 서로 분리될 수 있다. 예를 들어, 위반이 발생하면, 하나 이상의 컨트롤러는 하나 이상의 다른 컨트롤러에 대해 비활성화된 대역내 관리 연결(270)을 가질 수 있어 위반 또는 액세스의 확산을 방지할 수 있다. 시스템 섹션은 꺼지거나 격리될 수 있다.

[0244] 서브시스템(1400a, 1400b)은 또한 대역내 관리(270) 또는 대역외 관리(260)와 리소스를 공유하거나 이들을 통해 다른 환경 또는 시스템에 연결될 수 있다.

[0245] 도 14b 및 14c는 메인 컨트롤러를 갖는 컨트롤러를 제공하기 위한 가능한 단계를 도시하는 예시적인 흐름도이다.

[0246] 도 14b에서, 단계 1460에서, 메인 컨트롤러는 리소스(1420a 또는 1420b)와 같은 리소스를 제공 또는 설정한다. 단계 1461에서, 메인 컨트롤러는 서브컨트롤러를 제공 또는 설정한다. 메인 컨트롤러는 단계 1460 및 1461을 수행하기 위해 시스템 내에 리소스를 설정하기 위해 전술한 기술을 사용할 수 있다. 또한, 도 14b는 단계 1461 이전에 수행되는 단계 1460을 나타내지만, 반드시 그럴 필요는 없음을 이해해야 한다. 시스템 컨트롤러(210)를 사용하여, 메인 컨트롤러(1401)는 어떤 리소스가 필요한지를 결정하고 시스템 또는 네트워크에서 리소스를 찾을 수 있다. 메인 컨트롤러는 단계(1461)에서 시스템 규칙(210)을 시스템에 로딩하여 서브컨트롤러를 설정함으로써(또는 자신의 시스템 규칙을 설정하고 얻는 방법에 대한 명령어를 서브컨트롤러에 제공함으로써) 서브컨트롤러를 설정 또는 배치할 수 있다. 이들 명령어는 리소스의 구성, 애플리케이션의 구성, 서브컨트롤러가 실행하는 IT 시스템을 생성하기 위한 글로벌 시스템 규칙, 메인 컨트롤러에 재연결하여 새로운 또는 변경된 규칙을 수집하기 위한 명령어, 새로운 생산 환경을 위한 공간을 만들기 위해 애플리케이션 네트워크로부터 연결 해제하기 위한 명령어를 포함할 수 있지만 이에 한정되지 않는다. 리소스를 배치한 후, 단계 1463에서, 메인 컨트롤러는

시스템 규칙(210) 및/또는 시스템 상태(220)에 대한 업데이트를 통해 리소스를 서브컨트롤러에 할당할 수 있다.

[0247] 도 14C는 배치를 위한 대안적인 프로세스 흐름을 나타낸다. 도 14c의 예에서, 메인 컨트롤러는 단계 1470에서 서브컨트롤러를 배치한다(단계 1461과 관련하여 설명된 바와 같이 진행할 수 있음). 그 후, 단계 1475에서, 서브컨트롤러는 도 3c 및 도 7b에 나타낸 것과 같은 기술을 사용하여 리소스를 배치한다.

[0248] 도 15a는 시스템(100)을 위한 메인 컨트롤러(1501)가 환경(1502, 1503, 1504)을 생성하는 예시적인 시스템을 나타낸다. 환경(1502)은 리소스(1522)를 포함하고, 환경(1503)은 리소스(1523)를 포함하며, 환경(1504)은 리소스(1524)를 포함한다. 또한, 환경(1502, 1503, 1504)은 공유 리소스(1525)의 풀에 대한 액세스를 공유할 수 있다. 이러한 공유 리소스는, 예를 들어 공유 데이터 세트, API, 또는 서로 통신해야 하는 실행중인 애플리케이션을 포함할 수 있지만 이에 한정되지 않는다.

[0249] 도 15a의 예에서, 각 환경(1502, 1503, 1504)은 메인 컨트롤러(1501)를 공유한다. 메인 컨트롤러(1501)의 글로벌 시스템 규칙(210)은 환경을 배치하고 관리하는 규칙을 포함할 수 있다. 리소스(1522, 1523 및/또는 1524)는 하나 이상의 애플리케이션을 관리하기 위해 그들 각각의 환경(1501, 1502, 1503)을 필요로 한다. 이러한 애플리케이션을 위한 구성 규칙은, 이러한 각각의 환경이 다른 애플리케이션 환경과 상호 작용하고 운영하는 방법을 정의하기 위해, 메인 컨트롤러에 의해(또는 존재하는 경우에 환경 내의 로컬 컨트롤러에 의해) 구현된다. 메인 컨트롤러(1401)는 컨트롤러 로직(205), 시스템 상태(220) 및 템플릿(230)과 함께 글로벌 규칙(210)을 사용하여, 본 명세서에서 도 1 내지 도 14c를 참조하여 설명된 것과 유사한 방식으로 환경을 설정, 제공 및 배치할 수 있다. 환경이 로컬 컨트롤러를 포함하는 경우, 메인 컨트롤러(1501)는 글로벌 규칙(또는 그의 서브세트)이 해당 환경의 동작을 정의하는 방식으로 글로벌 규칙(210)(또는 그의 서브세트)을 로컬 컨트롤러 또는 관련 스토리지에 로딩할 수 있다.

[0250] 컨트롤러(1501)는 시스템 규칙(210)을 갖는 구성 규칙을 사용하여 환경(1502, 1503, 1504) 각각의 리소스(1522, 1523, 1524) 및/또는 공유 리소스(1525)을 배치 및 구성할 수 있다. 컨트롤러(1501)는 또한 각각의 환경(1502, 1503, 1504)의 모니터링을 허용하기 위해 환경을 모니터링하거나 리소스(1522, 1523, 1524)(또는 공유 리소스(1525))를 구성할 수 있다. 이러한 모니터링은 활성화 또는 비활성화될 수 있거나, 메인 컨트롤러를 통해 있을 수 있는 별도의 모니터링 콘솔에 대한 연결을 통해 이루어질 수 있다. 메인 컨트롤러(1501)는 도 13a 내지 도 13e 및 도 14a에서 리소스의 배치 및 관리를 참조하여 본 명세서에서 설명되는 방식으로 다양한 배치 또는 관리의 단계에서 활성화 또는 비활성화될 수 있는 대역내 관리 연결(들)(270) 및/또는 대역외 관리 연결(들)(260) 또는 SAN 연결(280)을 통해 다른 환경(1502, 1503, 1504) 중 하나 이상에 연결될 수 있다. 대역내 관리 연결(270) 또는 대역외 관리 연결(260) 또는 SAN 연결(280)의 활성화 및 비활성화를 사용하여, 환경(1502, 1503, 1504)은 다양한 시간에 서로에 대한 연결 또는 메인 시스템(100) 또는 컨트롤러(1501)에 대한 지식을 갖지 않거나, 제한된, 또는 제어된 지식을 가질 수 있는 방식으로 배치될 수 있다.

[0251] 환경은 다른 리소스와, 또는 외부의 외측 환경에 연결하는 외부 네트워크(1580)에 결합되고 상호 작용하는 리소스 또는 복수의 리소스를 포함할 수 있다. 환경은 물리적이거나 비물리적일 수 있다. 이 문맥에서 비물리적이란 환경이 동일한 물리 호스트(들)를 공유하지만 실제로 서로 분리되어 있음을 의미한다. 환경과 시스템은 동일하거나 유사하지만 다르거나, 동일하지 않은 하드웨어에 배치될 수 있다. 일부 예에서, 환경(1502, 1503, 1504)은 서로의 효과적인 사본일 수 있고; 그러나 다른 예에서는, 환경(1502, 1503, 1504)이 서로 상이한 기능을 제공할 수 있다. 일례로서, 환경의 리소스는 서버일 수 있다.

[0252] 본 명세서에서 설명되는 기술에 따라 별도의 환경 또는 서브시스템에 시스템 및 리소스를 배치하는 것은, 보안 및/또는 성능상의 이유로 애플리케이션을 격리시킬 수 있다. 환경을 분리하는 것은 또한 침입받은 리소스의 영향을 완화할 수도 있다. 예를 들어, 하나의 환경은 민감한 데이터를 포함할 수 있고 적은 인터넷 노출로 구성될 수 있지만, 다른 환경은 인터넷 대면 애플리케이션을 호스팅할 수 있다.

[0253] 도 15b는 컨트롤러가 도 15a에 나타낸 바와 같이 환경을 설정하는 예시적인 프로세스 흐름을 도시한다. 이러한 예에서, 시스템은 새로운 환경을 생성하고 설정하도록 작업될 수 있다. 이는 사용자 요청에 의해 또는 특정 작업 또는 일련의 작업에 참여할 때 수행되는 시스템 규칙에 의해 트리거될 수 있다. 후술하는 도 17a 내지 도 18b는 시스템이 새로운 환경을 생성하는 특정 변경 관리 작업 또는 일련의 작업의 예를 도시한다. 그러나, 컨트롤러가 새로운 환경을 생성하고 설정할 수 있는 많은 상황이 있을 수 있다.

[0254] 따라서, 도 15b를 참조하여, 새로운 환경을 설정함에 있어서, 컨트롤러는 환경 규칙을 선택한다(단계 1500.1). 환경 규칙에 따라, 글로벌 시스템 규칙(210) 및 템플릿(230)을 사용하여, 컨트롤러는 환경에 대한 리소스를 찾

는다(단계 1500.2). 규칙은 환경에 필요한 리소스를 찾을 때까지 진행하는 바람직한 리소스 선택의 계층을 가질 수 있다. 단계 1500.3에서, 컨트롤러는, 예를 들어 도 3c 또는 도 7b에 설명된 기술을 사용하여, 단계 1500.2에서 찾은 리소스를 환경에 할당한다. 그 후, 컨트롤러는 새로운 환경과 다른 시스템 구성요소 사이에 호환 가능하고 효율적인 연결을 보장하기 위해 새로운 환경에 대하여 시스템의 네트워킹 리소스를 구성한다(단계 1500.4). 각 리소스가 활성화되고 각 템플릿이 프로세싱됨에 따라 단계 1500.5에서 시스템 상태가 업데이트된다. 그 후, 컨트롤러는 환경의 리소스의 통합 및 상호 운용성을 설정하고 활성화하고 임의의 애플리케이션의 전원을 켜서 새로운 환경을 배치한다(단계 1500.6). 환경이 이용 가능해지면 단계 1500.7에서 시스템 상태가 다시 업데이트된다.

[0255] 도 15c는 컨트롤러가 도 15a에 나타낸 바와 같이 다수의 환경을 설정하는 예시적인 프로세스 흐름을 도시한다. 여러 환경을 설정할 때, 각 환경에 대해 도 15b에 설명된 기술을 사용하여 환경이 병렬로 설정될 수 있다. 그러나, 환경은 도 15c에 설명된 바와 같이 순차적 순서로 또는 직렬로 설정될 수 있음을 이해해야 한다. 도 15c를 참조하면, 단계 1500.10에서, 컨트롤러는 제1 새로운 환경을 설정하고 배치한다(도 15b의 단계 1500.1과 관련하여 설명된 바와 같이 수행될 수 있음). 다른 유형의 환경 및 다른 환경의 상호 운용 방식에 대해 다른 환경 규칙이 있을 수 있다. 단계 1500.11에서, 컨트롤러는 다음 환경에 대한 환경 규칙을 선택한다. 단계 1500.12에서, 컨트롤러는 시스템 규칙(210)에 의해 정의될 수 있는 우선 순위에 따라 리소스를 찾는다. 단계 1500.13에서, 컨트롤러는 단계 1500.12에서 찾은 리소스를 다음 환경에 할당한다. 환경은 리소스를 공유하거나 공유하지 않을 수 있다. 단계 1500.14에서, 컨트롤러는 다음 환경에 관하여 그리고 시스템 규칙(210)을 사용하여 종속성을 갖는 환경들 사이에서 시스템의 네트워킹 리소스를 구성한다. 각 리소스가 활성화되고 템플릿이 프로세싱되고 환경의 종속성을 포함하여 네트워킹 리소스가 구성됨에 따라 단계 1500.15에서 시스템 상태가 업데이트된다. 그 후, 컨트롤러는 다음 환경 및 환경들 간의 리소스의 통합 및 상호 운용성을 설정하고 활성화하고, 임의의 애플리케이션의 전원을 켜서 새로운 환경을 배치한다(단계 1500.16). 다음 환경을 사용 가능해짐에 따라 단계 1500.17에서 시스템 상태가 업데이트된다.

[0256] *모니터링을 지원하기 위한 일방향 통신:*

[0257] 도 16a는 제1 컨트롤러(1601)가 1601a, 1601b, 및/또는 1601c와 같은 하나 이상의 컨트롤러를 설정하기 위한 메인 컨트롤러로서 동작하는 예시적인 실시형태를 도시한다. 메인 컨트롤러(1601)는 복수의 클라우드 호스트, 시스템 및/또는 애플리케이션을 컨트롤러(200/1401/1501)와 같은 컨트롤러와 관련하여 전술한 기술을 사용하여 그들의 동작에서 서로에 의존하거나 그렇지 않을 수 있는 환경(1602, 1603, 1604)으로서 생성하는 데 사용될 수 있다. 도 16a에 도시된 바와 같이, IT 시스템, 환경, 클라우드 및/또는 이들의 임의의 조합(들)은 환경(1602, 1603, 1604)으로서 생성될 수 있다. 환경(1602)은 제2 컨트롤러(1601a)를 포함하고, 환경(1603)은 제3 컨트롤러(1601b)를 포함하고, 환경(1604)은 제4 컨트롤러(1601c)를 포함한다. 환경(1602, 1603, 1604)은 또한 각각 하나 이상의 리소스(1642, 1643, 1644)를 각각 포함할 수 있다. 리소스는 그들 상에서 실행중일 수 있는 하나 이상의 애플리케이션(1642, 1643, 1644)을 포함할 수 있다. 이들 애플리케이션은 공유 여부에 관계없이 할당된 리소스에 연결할 수 있다. 이들 또는 다른 애플리케이션은 인터넷 또는 공유 애플리케이션 또는 애플리케이션 네트워크를 또한 포함할 수 있는 풀(1660)의 하나 이상의 공유 리소스에서 실행할 수 있다. 애플리케이션은 사용자 또는 하나 이상의 환경 또는 클라우드에 서비스를 제공할 수 있다. 환경(1602, 1603, 1604)은 리소스 또는 데이터베이스를 공유할 수 있고 및/또는 특정 환경에 구체적으로 할당된 풀(1660) 내의 리소스를 포함하거나 사용할 수 있다. 메인 컨트롤러(1601) 및/또는 하나 이상의 환경을 포함하는 시스템의 다양한 구성요소는 또한 애플리케이션 네트워크 또는 인터넷과 같은 외부 네트워크(1615)에 연결 가능할 수 있다.

[0258] 임의의 리소스, 환경 또는 컨트롤러와, 다른 리소스, 환경, 컨트롤러 또는 외부 연결 사이에는, 본 명세서에서 도 13a 내지 도 13e와 관련하여 설명된 방식으로 선택적으로 활성화 및/또는 비활성화되도록 구성될 수 있는 연결이 있을 수 있다. 예를 들어, 임의의 리소스, 컨트롤러, 환경 또는 외부 연결은 대역내 관리 연결(270), 대역외 관리 연결(260), 또는 SAN 연결(280)을 통해 또는 물리 연결 해제에 의해 컨트롤러(1601), 환경(1602), 환경(1603) 및/또는 환경(1604), 리소스 또는 애플리케이션으로부터 비활성화되거나 연결 해제 가능할 수 있다. 일례로서, 컨트롤러(1601)와 임의의 환경(1602, 1603, 1604) 사이의 대역내 관리 연결(270)은 컨트롤러(1601)를 보호하기 위해 비활성화될 수 있다. 다른 예로서, 이러한 대역내 관리 연결(들)(270)은 환경(1602, 1603, 1604)의 작동 중에 선택적으로 비활성화 또는 활성화될 수 있다. 본 명세서에서 도 13a 내지 도 13e와 관련하여 설명된 보안 목적 이외에, 환경(1602, 1603, 1604)으로부터 메인 컨트롤러(1601)를 비활성화 또는 연결 해제하는 것은 메인 컨트롤러(1601)가 메인 컨트롤러(1601)로부터 또는 다른 클라우드 또는 환경으로부터 분리될 수 있는 환경(1602, 1603, 1604)을 클라우드로서 스핀하도록 허용할 수 있다. 이러한 의미에서, 컨트롤러(1601)는 다수

의 클라우드, 호스트 또는 시스템을 생성하도록 구성된다.

[0259] 본 명세서에서 설명된 비활성화 또는 연결 해제 요소를 사용하여, 사용자에게는 특정 용도를 위해 메인 컨트롤러(1601)를 통해 환경에 대한 제한된 액세스가 허용될 수 있다. 예를 들어, 개발자에게는 개발 환경에 대한 액세스가 제공될 수 있다. 다른 예로서, 애플리케이션의 관리자는 특정 애플리케이션 또는 애플리케이션 네트워크로 제한될 수 있다. 다른 예로서, 로그는 자신이 생성하는 환경 또는 컨트롤러에 의해 침입받지 않고 데이터를 수집하기 위해 메인 컨트롤러(1601)를 통해 보일 수 있다.

[0260] 메인 컨트롤러(1601)가 환경(1602)을 설정한 후, 환경(1602)은 메인 컨트롤러(1601)로부터 연결 해제될 수 있고, 그 결과 환경(1602)은 메인 컨트롤러(1601)와 독립적으로 작동할 수 있고 및/또는 메인 컨트롤러(1601) 또는 환경(1602)과 연관되고 환경(1602)에 의해 실행되는 다른 애플리케이션에 의해 선택적으로 모니터링 및 유지될 수 있다.

[0261] 환경(1602)과 같은 환경은 구매자 또는 사용자에 의해 환경(1602)에 대한 액세스를 허용하는 사용자 인터페이스 또는 콘솔(1640)에 결합될 수 있다. 환경(1602)은 사용자 콘솔을 애플리케이션으로서 호스팅할 수 있다. 환경(1602)은 사용자에 의해 원격으로 액세스될 수 있다. 각각의 환경(1602, 1603, 1604)은 공통 또는 별도의 사용자 인터페이스 또는 콘솔에 의해 액세스될 수 있다.

[0262] 도 16b는 환경(1602, 1603, 1604)이, 예를 들어 콘솔(환경(1641)과 직접 또는 간접적으로 연결할 수 있는 임의의 콘솔일 수 있음)을 사용하여 로그가 보일 수 있는 다른 환경(1641)에 기록하도록 구성될 수 있는 예시적인 시스템을 나타낸다. 이러한 방식으로, 환경(1641)은 하나 이상의 환경(1602, 1603, 1604)이 이벤트를 기록하는 로그 서버로서 기능할 수 있다. 그 후, 메인 컨트롤러(1601)는 로그 서버(1641)에 액세스하여 후술하는 바와 같이 이러한 환경(1602, 1603, 1604)과의 직접적인 연결을 유지하지 않고 환경(1602, 1603, 1604)에서 이벤트를 모니터링할 수 있다. 환경(1641)은 또한 메인 컨트롤러(1601)로부터 선택적으로 연결 해제될 수 있고 다른 환경(1602, 1603, 1604)으로부터만 판독되도록 구성될 수 있다.

[0263] 메인 컨트롤러(1601)는 도 16c에 의해 나타낸 바와 같이 메인 컨트롤러(1601)가 임의의 환경(1602, 1603, 1604)으로부터 연결 해제되더라도 그 환경(1602, 1603, 1604)의 일부 또는 전부를 모니터링하도록 구성될 수 있다. 도 16c는 메인 컨트롤러(1601)와 환경(1602, 1603, 1604) 사이의 대역내 관리 연결(270)이 연결 해제되어 있는 것을 나타내고, 이는 환경(1602, 1603, 1604)이 침입받은 경우에 메인 컨트롤러(1601)를 보호하는 것을 도울 수 있는 다. 도 16c에 나타낸 바와 같이, 메인 컨트롤러(1601)와 환경(1602) 사이의 대역내 연결(270)이 연결 해제되어 있는 경우에도, 대역외 연결(260)은 여전히 메인 컨트롤러(1601)와 1602와 같은 환경 사이에서 유지될 수 있다. 또한, 환경(1641)은 선택적으로 활성화 또는 비활성화될 수 있는 메인 컨트롤러(1601)에 대한 연결을 가질 수 있다. 메인 컨트롤러(1601)는 환경(1602, 1603, 1604)으로부터 격리되거나 에어갭되는 환경(1641) 내의 별도의 애플리케이션으로서 모니터링을 설정할 수 있다. 메인 컨트롤러(1601)는 모니터링을 위해 일방향 통신을 사용할 수 있다. 예를 들어, 로그는 환경(1602, 1603, 1604)으로부터 환경(1641)으로의 일방향 통신을 통해 제공될 수 있다. 이러한 일방향 기록 및 환경(1641)과 메인 컨트롤러(1601) 사이의 연결을 통해, 메인 컨트롤러(1601)는 메인 컨트롤러(1601)와 환경(1602, 1603, 1604) 사이에 대역내 연결(270)이 없더라도 환경(1641)을 통해 데이터를 수집하고 환경(1602, 1603, 1604)을 모니터링할 수 있어, 메인 컨트롤러(1601)를 침입하는 환경(1602, 1603, 1604)의 위험을 완화시킨다. 액세스는 필터링되거나 제어될 수 있고 및/또는 액세스는 인터넷과 독립적일 수 있다. 예를 들어, 도 16d에 나타낸 바와 같이, 메인 컨트롤러(1601)와 환경(1602) 사이의 대역내 연결(270)이 연결되면, 메인 컨트롤러(1601)는 인터넷과 같은 외부 네트워크(1615)로부터 환경(1602)을 연결 해제하도록 네트워크 스위치(1650)를 제어할 수 있다. 환경(1602)이 메인 컨트롤러(1601)와 대역내 연결(270)에 의해 연결될 때에 외부 네트워크(1615)로부터 환경(1602)의 연결 해제는 메인 컨트롤러(1601)에 대한 향상된 보안을 제공할 수 있다.

[0264] 따라서, 도 16b 내지 도 16d의 예시적인 실시형태는 메인 컨트롤러가 해당 환경(1602, 1603, 1604)에 대한 노출을 최소화하면서 환경(1602, 1603, 1604)을 어떻게 안전하게 모니터링할 수 있는지를 이해해야 한다. 따라서, 메인 컨트롤러(1601)는 환경(1602, 1603, 1604)이 일방향 기록 권한을 가질 수 있는 환경(1641)의 로그 서버를 통해 그들을 모니터링하는 메커니즘을 여전히 유지하면서 환경(1602, 1603, 1604)으로부터 자신을 연결해제(또는 적어도 대역내 링크로부터 자신을 연결 해제)할 수 있다. 따라서, 환경(1641)의 로그를 검토하는 과정에서, 메인 컨트롤러(1601)가 환경(1602)이 멀웨어에 의해 침입받을 수 있음을 발견하면, 메인 컨트롤러(1601)는 대역외 연결(260)만이 존재하도록 SDN 툴을 사용하여 해당 환경(1602)을 격리할 수 있다(예를 들어, 도 16c 참조). 또한, 컨트롤러(1601)는 가능한 문제에 대한 환경(1602)을 위해 관리자에게 통지를 보낼 수 있다. 컨트롤러는

또한 침입받은 환경과 임의의 다른 환경(1603, 1604) 사이의 임의의 연결(예를 들어, 대역내 관리 연결(270))을 선택적으로 비활성화함으로써 침입받은 환경(1602)을 격리할 수 있다. 다른 예에서, 메인 컨트롤러(1601)는 환경(1603) 내의 리소스가 매우 고온으로 실행중인 것을 로그를 통해 발견할 수 있다. 이는 메인 컨트롤러가 애플리케이션 또는 서비스를 환경(1603)으로부터 다른 환경(기존의 환경이든 새롭게 생성된 환경이든)에 개입시키고 마이그레이션할 수 있다.

[0265] 컨트롤러(1601)는 또한 구매자 또는 사용자 요청에 따라 유사한 시스템을 설정할 수 있다. 도 16e에 나타낸 바와 같이, 구매 애플리케이션(1650)은 예를 들어 콘솔 또는 다른 방식으로 제공될 수 있으며, 이는 구매자가 구매자를 위해 설정되는 클라우드, 호스트, 시스템 환경 또는 애플리케이션을 구매 또는 요청할 수 있게 한다. 구매 애플리케이션(1650)은 환경(1602)을 설정하도록 컨트롤러(1601)에 지시할 수 있다. 환경(1602)은, 예를 들어 리소스를 환경(1602)에 할당 또는 배정함으로써, IT 시스템을 배치 또는 구축할 컨트롤러(1601a)를 포함할 수 있다.

[0266] 도 16f는 환경(1602, 1603, 1604)이 각각 클라우드로서 동작중이고 컨트롤러를 포함하거나 포함하지 않을 수 있는 경우에 사용될 수 있는 사용자 인터페이스(1632, 1633, 1634)를 도시한다. 사용자 인터페이스(1632, 1633, 1634)(각각 환경(1602, 1603, 1604)에 대응함)는 각각 사용자 인터페이스와 환경의 연결을 관리하는 메인 컨트롤러(1601)를 통해 연결할 수 있다. 대안적으로 또는 추가적으로, 인터페이스(1640a)(콘솔 형태를 취할 수 있음)는 환경(1602)에 직접 결합될 수 있고, 인터페이스(1640b)(콘솔 형태를 취할 수 있음)는 환경(1603)에 직접 결합될 수 있고, 인터페이스(1640c)(콘솔 형태를 취할 수 있음)는 환경(1604)에 직접 결합될 수 있다. 메인 컨트롤러(1601)와의 연결이 분리, 연결 해제 또는 비활성화되는지의 여부에 관계없이, 사용자는 하나 이상의 인터페이스를 사용하여 환경 또는 클라우드를 사용할 수 있다.

[0267] *변경 관리 지원을 위한 시스템 클론 및 백업:*

[0268] 환경(1602, 1603, 1604) 중 일부는 개발자가 사용하는 전형적인 설정 소프트웨어의 클론일 수 있다. 이들은 또한 확장 방법으로서 현재 작업 환경의 클론일 수 있고; 예를 들어, 위치로 인한 대기 시간을 줄이기 위해 다른 위치의 다른 데이터센터에서 환경을 클로닝한다.

[0269] 따라서, 별도의 환경 또는 서브시스템에서 시스템 및 리소스를 설정하는 메인 컨트롤러는 IT 시스템의 일부를 클로닝 또는 백업하는 것을 가능하게 할 수 있다는 것을 이해해야 한다. 이는 본 명세서에 설명된 바와 같이 테스트 및 변경 관리에 사용될 수 있다. 이러한 변경은 코드, 구성 규칙, 보안 패치, 템플릿 및/또는 다른 변경에 대한 변경을 포함할 수 있지만 이에 한정되지 않는다.

[0270] 예시적인 실시형태에 따르면, 본 명세서에 설명된 IT 시스템 또는 컨트롤러는 하나 이상의 환경을 클로닝하도록 구성될 수 있다. 새로운 또는 클론 환경은 원래의 환경과 동일한 리소스를 포함하거나 포함하지 않을 수 있다. 예를 들어, 새로운 환경 또는 거의 클론 환경에서 물리 및/또는 가상의 완전히 다른 조합의 리소스를 사용하는 것이 바람직하거나 필요할 수 있다. 사용의 최적화가 관리될 수 있는 다른 위치 또는 시간대에 환경을 클로닝하는 것이 바람직할 수 있다. 환경을 가상 환경에 클로닝하는 것이 바람직할 수 있다. 환경의 클론에서, 컨트롤러 또는 메인 컨트롤러의 글로벌 시스템 규칙(210) 및 글로벌 템플릿(230)은 다양한 유형의 하드웨어를 구성 및/또는 실행하는 방법에 대한 정보를 포함할 수 있다. 시스템 규칙(210) 내의 구성 규칙은 리소스 및 애플리케이션이 특정의 이용 가능한 리소스가 제공되면 보다 최적이 되도록 리소스의 배치 및 사용을 지시할 수 있다.

[0271] 메인 컨트롤러 구조는 별도의 환경 또는 서브시스템에 시스템 및 리소스를 설정하는 능력을 제공하고, 환경을 클로닝하기 위한 구조를 제공하고, 개발 환경을 생성하기 위한 구조를 제공하고, 및/또는 표준화된 애플리케이션 및/또는 리소스 세트를 배치하기 위한 구조를 제공한다. 이러한 애플리케이션 또는 리소스는, 예를 들어 애플리케이션을 개발 및/또는 실행하거나, 일부를 백업하거나, IT 시스템 및 다른 재난 복구 애플리케이션(예를 들어, LAMP(apache, mysql, php) 스택, 웹 프론트엔드 및 리액트/리덕스를 실행하는 서버를 포함하는 시스템, 및 node.js를 실행하는 리소스, 및 몽고 데이터베이스 및 다른 표준화된 "스택")의 백업으로부터 복원하기 위해 사용될 수 있는 것을 포함할 수 있지만 이에 한정되지 않는다. 때로는 메인 컨트롤러가 다른 환경의 클론인 환경을 배치할 수 있으며, 원래의 환경을 생성하는 데 사용된 구성 규칙의 서브세트로부터 구성 규칙을 도출할 수 있다.

[0272] 예시적인 실시형태에 따르면, 시스템 또는 시스템의 서브세트의 변경 관리는 하나 이상의 환경 및 이러한 환경의 구성 규칙 또는 구성 규칙의 서브세트를 클로닝함으로써 달성될 수 있다. 예를 들어, 코드, 구성 규칙, 보안 패치, 템플릿에 대한 변경, 하드웨어 변경, 구성요소 및 종속 애플리케이션의 추가/제거 및 다른 변경을 행하기

위해 변경이 필요할 수 있다.

[0273] 예시적인 실시형태에 따르면, 시스템에 대한 이러한 변경은 변경의 직접적인 수동 입력의 에러를 피하기 위해 자동화될 수 있다. 실제 시스템에 대한 변경을 자동적으로 구현하기 전에 개발 환경에서 사용자에 의해 변경이 테스트될 수 있다. 예시적인 실시형태에 따르면, 실제 생산 환경은 생산 환경과 동일한 구성 규칙을 사용하여 구성된 환경을 자동적으로 켜고, 제공하고 및/또는 구성하기 위해 컨트롤러를 사용하여 클로닝될 수 있다. 클론 환경이 실행되고 작동될 수 있다(반면에, 백업 환경은 변경을 롤백할 필요가 있는 경우에 비상 사태로 유지되는 것이 바람직할 수 있다). 이는 시스템 규칙(210), 템플릿(230) 및/또는 시스템 상태(220)를 사용하여 상기 도 1 내지 도 16f를 참조하여 설명된 바와 같이 컨트롤러를 사용하여 새로운 시스템, 또는 환경을 생성, 구성 및/또는 제공하도록 수행될 수 있다. 새로운 환경은 나중에 생산 환경에서 구현될 변경을 테스트하기 위한 개발 환경으로서 사용될 수 있다. 컨트롤러는 소프트웨어 정의 구조로부터 개발 환경으로의 이러한 환경의 인프라구조를 생성할 수 있다.

[0274] 본 명세서에 정의된 생산 환경은 개발 및 테스트만을 위한 환경, 즉 개발 환경과 반대로 시스템을 작동시키기 위해 사용되는 환경을 의미한다.

[0275] 생산 환경이 클로닝되면, 인프라구조 또는 클론 개발 환경은 생산 환경과 마찬가지로 글로벌 시스템 규칙(210)에 따라 컨트롤러에 의해 구성 및 생성된다. 개발 환경에서의 변경은 코드, 템플릿(230)(기존 템플릿 변경 또는 새로운 템플릿 생성과 관련된 변경), 보안 및/또는 애플리케이션 또는 인프라구조 구성에 이루어질 수 있다. 개발 환경에서 구현된 새로운 변경이 개발 및/또는 테스트를 통해 원하는 대로 준비되면, 시스템이 자동적으로 개발 환경을 변경한 다음, 가동 준비를 하거나 생산 환경으로서 배치될 것이다. 그 후, 새로운 시스템 규칙(210)은 환경의 컨트롤러 및/또는 특정 환경에 대한 시스템 규칙 변경을 적용할 메인 컨트롤러에 업로드된다. 시스템 상태(220)는 컨트롤러에서 업데이트되고 추가 또는 수정된 템플릿(230)이 구현될 수 있다. 따라서, 인프라구조에 대한 전체 시스템 지식은 개발 환경 및/또는 메인 컨트롤러에 의해 인프라구조를 재생성하는 능력과 함께 유지될 수 있다. 본 명세서에 사용된 전체 시스템 지식은 리소스 상태, 리소스 이용 가능성, 및 시스템의 구성에 대한 시스템 지식을 포함할 수 있지만 이에 한정되지 않는다. 전체 시스템 지식은 리소스에 질의하기 위해 컨트롤러에 의해 시스템 규칙(210), 시스템 상태(220) 및/또는 대역내 관리 연결(들)(270), 대역외 관리 연결(들)(260) 및/또는 SAN 연결(280)로부터 수집될 수 있다. 리소스는 특히 리소스, 네트워크 또는 애플리케이션 이용, 구성 상태 또는 이용 가능성을 결정하기 위해 질의받을 수 있다.

[0276] 클론 인프라구조 또는 환경은 시스템 규칙(210)을 통해 정의된 소프트웨어일 수 있지만; 이것이 반드시 그럴 필요는 없다. 클론 인프라구조 또는 환경은 일반적으로 프론트 엔드 또는 사용자 인터페이스, 및 연산, 네트워킹, 스토리지 및/또는 애플리케이션 네트워킹 리소스를 포함하거나 포함하지 않을 수 있는 하나 이상의 할당된 리소스를 포함하거나 포함하지 않을 수 있다. 환경은 프론트 엔드, 미들웨어 및 데이터베이스로서 배열되거나 배열되지 않을 수 있다. 서비스 또는 개발 환경은 생산 환경의 시스템 규칙(210)으로 부팅될 수 있다. 컨트롤러에 의해 사용하기 위해 할당된 인프라구조 또는 환경은 특히 클론 목적으로 소프트웨어 정의될 수 있다. 따라서, 환경은 시스템 규칙(210)에 의해 배치될 수 있고 유사한 수단에 의해 클로닝 가능할 수 있다. 클론 또는 개발 환경은 변경이 요구되기 전 또는 변경될 때 시스템 규칙(210)을 사용하여 로컬 또는 메인 컨트롤러에 의해 자동적으로 설정될 수 있다.

[0277] 생산 환경의 데이터는 개발 환경이 생산 환경과 격리될 때까지 판독 전용 데이터 스토리지에 기록될 수 있으며, 그 결과 개발 환경에 의해 개발 및 테스트 프로세스에 사용될 것이다.

[0278] 생산 환경이 온라인 상태인 동안 사용자 또는 클라이언트는 개발 환경에서 변경을 수행하고 테스트할 수 있다. 개발 및 변경이 개발 환경에서 테스트되는 동안 데이터 스토리지의 데이터가 변경될 수 있다. 휘발성 또는 기록 가능한 시스템을 사용하면, 개발 환경이 설정되거나 배치된 후에 생산 환경의 데이터와 데이터의 핫 동기화가 또한 사용될 수 있다. 시스템, 애플리케이션, 및/또는 환경에 대한 원하는 변경이 개발 환경에서 이루어지고 테스트될 수 있다. 그 후, 환경 또는 전체 시스템 및 메인 컨트롤러를 위한 새로운 버전을 생성하기 위해 시스템 규칙(210)의 스크립트에 원하는 변경이 이루어진다.

[0279] 다른 예시적인 실시형태에 따르면, 새롭게 개발된 환경은 새로운 생산 환경으로서 자동적으로 구현될 수 있는 한편, 이전의 생산 환경은 유지되거나 완전히 기능적이서 상당한 양의 데이터의 손실없이 이전 상태 생산 환경으로의 복귀가 가능하다. 그 후, 개발 환경은 시스템 규칙(210) 내의 새로운 구성 규칙으로 부팅되고, 데이터베이스는 생산 데이터베이스와 동기화되고 기록 가능한 데이터베이스로 전환된다. 그 후, 원래 생산 데이터베이스는 판독 전용 데이터베이스로 전환될 수 있다. 이전 생산 환경으로 되돌리는 것이 바람직한 경우에 이전 생산

환경은 원하는 기간 동안 이전 생산 환경의 사본으로서 그대로 유지된다.

[0280] 환경은 물리 및/또는 가상 호스트, 네트워크, 및 다른 리소스를 포함하거나 수용할 수 있는 단일 서버 또는 인스턴스로서 구성될 수 있다. 다른 예시적인 실시형태에서, 환경은 물리 및/또는 가상 호스트, 네트워크, 및 다른 리소스를 포함하는 복수의 서버일 수 있다. 예를 들어, 로드 밸런스 인터넷 대면 애플리케이션(load-balanced internet-facing application)을 형성하는 복수의 서버가 있을 수 있으며; 이들 서버는 복수의 API/미들웨어 애플리케이션(하나 또는 복수의 서버에서 호스팅될 수 있음)에 연결될 수 있다. 환경의 데이터베이스는 API가 환경에서 쿼리를 전달하는 하나 이상의 데이터베이스를 포함할 수 있다. 환경은 시스템 규칙(210)으로부터 정적 또는 휘발성인 형태로 구축될 수 있다. 환경 또는 인스턴스는 가상 또는 물리적이거나 각각의 조합일 수 있다.

[0281] 시스템 규칙(210) 내의 애플리케이션의 구성 규칙 또는 시스템의 구성 규칙은 다양한 연산 백엔드(예를 들어, 베어메탈, AMD epyc 서버, qemu/kvm의 Intel Haswell)를 지정할 수 있으며, 새로운 연산 백엔드에서 애플리케이션 또는 서비스를 실행하는 방법에 대한 규칙을 포함할 수 있다. 따라서, 예를 들어, 테스트를 위해 리소스의 가용성이 감소된 상황이 있는 경우에 애플리케이션이 가상화될 수 있다.

[0282] 본 명세서에 설명된 예를 사용하고 이에 따라, 테스트 환경은 원래 환경이 물리 리소스를 사용하는 가상 리소스 상에 배치될 수 있다. 도 1 내지 도 18b를 참조하여 본 명세서에 설명된 바와 같이 컨트롤러를 사용하여, 그리고 본명세서에 더욱 설명되는 바와 같이, 시스템 또는 환경은 물리 환경으로부터 가상 리소스를 전체적으로 또는 부분적으로 포함하거나 포함하지 않을 수 있는 환경으로 클로닝될 수 있다.

[0283] 도 17a는 시스템(100)이 컨트롤러(1701)와 하나 이상의 환경, 예를 들어 1702, 1703, 1704를 포함하는 예시적인 실시형태를 도시한다. 시스템(100)은 정적 시스템, 즉 액티브 사용자 데이터가 지속적으로 시스템의 상태를 변경하지 않거나 또는 빈번하게 데이터를 조작하지 않는 것; 예를 들어 정적 웹 페이지만을 호스팅하는 시스템일 수 있다. 시스템은 사용자(또는 애플리케이션) 인터페이스(110)와 결합될 수 있다.

[0284] 컨트롤러(1701)는 본 명세서에 설명된 컨트롤러(200/1401/1501/1601)와 유사한 방식으로 구성될 수 있고, 글로벌 시스템 규칙(210), 컨트롤러 로직(205), 템플릿(230) 및 시스템 상태 요소(220)를 마찬가지로 포함할 수 있다. 컨트롤러(1701)는, 본 명세서에서 도 14a 내지 도 16f를 참조하여 설명된 방식으로 하나 이상의 다른 컨트롤러 또는 환경에 결합될 수 있다. 컨트롤러(1701)의 글로벌 규칙(210)은 다른 컨트롤러 및/또는 환경을 관리 및 제어할 수 있는 규칙을 포함할 수 있다. 이러한 글로벌 규칙(210), 컨트롤러 로직(205), 시스템 상태(220) 및 템플릿(230)은 본 명세에서 도 1 내지 도 16f를 참조하여 설명된 것과 유사한 방식으로 컨트롤러(1701)를 통해 시스템 또는 환경을 설정, 제공 및 배치하는 데 사용될 수 있다. 각 환경은 다른 환경과 관련하여 포함하는 환경의 동작을 정의하는 글로벌 시스템 규칙(210)의 서브세트를 사용하여 구성될 수 있다.

[0285] 글로벌 시스템 규칙(210)은 또한 변경 관리 규칙(1711)을 포함할 수 있다. 변경 관리 규칙(1711)은 시스템(100), 글로벌 시스템 규칙(210), 및/또는 컨트롤러 로직 (205)에 대한 변경이 필요할 수 있을 때에 사용될 수 있는 규칙 및/또는 명령어 세트를 포함한다. 변경 관리 규칙(1711)은 사용자 또는 개발자가 변경을 개발하고, 테스트 환경에서 변경을 테스트한 다음, 변경을 시스템 규칙(210) 내에서 새로운 구성 규칙 세트로 자동 변환함으로써 변경을 구현할 수 있도록 구성될 수 있다. 변경 관리 규칙(1711)은 (도 17a에 나타낸 바와 같이) 글로벌 시스템 규칙(210)의 서브세트일 수 있거나, 또는 글로벌 시스템 규칙(210)과 별개일 수 있다. 변경 관리 규칙은 글로벌 시스템 규칙(210)의 서브세트를 사용할 수 있다. 예를 들어, 글로벌 시스템 규칙(210)은 새로운 환경을 생성하도록 구성된 환경 생성 규칙의 서브세트를 포함할 수 있다. 변경 관리 규칙(1711)은 시스템(100)의 일부 또는 모든 양태를 복사 및 클로닝하기 위해 컨트롤러(1701)에 의해 구성되고 설정된 시스템 또는 환경을 설정하고 사용하도록 구성될 수 있다. 변경 관리 규칙(1711)은 테스트 및 구현을 위한 시스템의 클론을 사용함으로써 구현 전에 시스템에 제안된 새로운 변경의 테스트를 허용하도록 구성될 수 있다.

[0286] 도 17a에 의해 나타낸 클론(1705)은 특정 환경 또는 시스템(100)의 일부의 규칙, 로직, 애플리케이션 및/또는 리소스를 포함할 수 있다. 클론(1705)은 시스템(100)과 유사하거나 유사하지 않은 하드웨어를 포함할 수 있고, 가상 리소스를 사용하거나 사용하지 않을 수 있다. 클론(1705)은 애플리케이션으로서 설정될 수 있다. 클론(1705)은 시스템(100) 또는 컨트롤러(1701)의 시스템 규칙(210) 내의 구성 규칙을 사용하여 설정 및 구성될 수 있다. 클론(1705)은 컨트롤러를 포함하거나 포함하지 않을 수 있다. 클론(1705)은 상기에서 더욱 상세하게 설명된 바와 같이 할당된 네트워킹, 연산 리소스, 애플리케이션 네트워크 및/또는 데이터 스토리지 리소스를 포함할 수 있다. 이러한 리소스는 컨트롤러(1701)에 의해 제어되는 변경 관리 규칙(1711)을 사용하여 할당될 수 있다. 클론(1705)은 사용자에 의해 클론(1705)에 변경이 이루어질 수 있게 하는 사용자 인터페이스에 결합될 수 있다.

사용자 인터페이스는 시스템(100)의 사용자 인터페이스(110)와 동일하거나 상이할 수 있다. 클론(1705)은 전체 시스템(100) 또는 하나 이상의 환경 및/또는 컨트롤러와 같은 시스템(100)의 일부에 사용될 수 있다. 클론(1705)은 시스템(100)의 완전한 사본일 수도 아닐 수도 있다. 클론(1705)은, 선택적으로 완전히 활성화 및/또는 비활성화될 수 있고, 및/또는 단방향 판독 및/또는 기록 연결로 변환될 수 있는 대역내 관리 연결(270), 대역외 관리 연결(260) 및/또는 SAN 연결(280)을 통해 시스템(100)에 결합될 수 있다. 따라서, 클론 환경(1705)의 데이터에 대한 연결은 클론 환경(1705)이 테스트 동안 생산 환경으로부터 격리될 때 또는 클론 환경(1705)이 새로운 생산 환경으로서 온라인으로 준비될 때까지 클론 데이터를 판독 전용으로 하도록 변경될 수 있다. 예를 들어, 클론(1705)이 환경(1702)에 대한 데이터 연결을 갖는 경우, 이 데이터 연결은 격리 목적으로 판독 전용이 될 수 있다.

[0287] 선택적인 백업(1706)은 전체 시스템 또는 하나 이상의 환경 및/또는 컨트롤러와 같은 시스템의 일부에 사용될 수 있다. 백업(1706)은 상기에서 더욱 상세하게 설명된 바와 같이 네트워킹, 연산, 애플리케이션 네트워크 및/또는 데이터 스토리지 리소스를 포함할 수 있다. 백업(1706)은 컨트롤러를 포함하거나 포함하지 않을 수 있다. 백업(1706)은 시스템(100)의 완전한 사본일 수 있다. 백업(1706)은 애플리케이션으로서 또는 시스템(100)과 유사하거나 유사하지 않은 하드웨어를 사용하여 설정될 수 있다. 백업(1706)은, 선택적으로 완전히 활성화 및/또는 비활성화될 수 있고, 및/또는 단방향 판독 및/또는 기록 연결로 변환될 수 있는 대역내 관리 연결(270), 대역외 관리 연결(260) 및/또는 SAN 연결(280)을 통해 시스템(100)에 결합될 수 있다.

[0288] 도 17b는 시스템 변경 관리에서 도 17a의 클론 및 백업 시스템을 사용하기 위한 예시적인 프로세스 흐름을 도시한다. 단계 1785에서, 사용자 또는 관리 애플리케이션은 시스템에 대한 변경을 개시한다. 이러한 변경은 코드, 구성 규칙, 보안 패치, 템플릿에 대한 변경, 하드웨어 변경, 구성요소 및/또는 종속 애플리케이션의 추가/제거 및 다른 변경을 포함할 수 있지만 이에 한정되지 않는다. 단계 1786에서, 컨트롤러(1701)는 도 14a 내지 도 16f와 관련하여 설명된 방식으로 환경을 설정하여 클론 환경(1705)이 된다(여기서 클론 환경은 자신의 새로운 컨트롤러를 가질 수 있거나 원래 환경에 대해 동일한 컨트롤러를 사용할 수 있다).

[0289] 단계 1787에서, 컨트롤러(1701)는 변경 관리 규칙(1711)을 포함하는 글로벌 규칙(210)을 사용하여 시스템의 환경 또는 환경들의 전부 또는 일부(예를 들어, "생산 환경")를 클론 환경(1705)에 클로닝할 수 있다(예를 들어, 여기서 클론 환경(1705)은 "개발 환경"으로서 기능할 수 있다). 이와 같이, 컨트롤러(1701)는 리소스를 식별 및 할당하고, 시스템 규칙(210)을 사용하여 클론 리소스를 설정 및 할당하고 다음 중 임의의 것을 환경으로부터 클론에 복사한다: 데이터, 구성, 코드, 실행 파일 및 애플리케이션에 전력을 공급하는 데 필요한 기타 정보. 단계 1788에서, 컨트롤러(1701)는 시스템 규칙(210) 내의 구성 규칙을 사용하여 (컨트롤러의 유무에 관계없이) 백업(1706)으로서 기능하도록 다른 환경을 설정함으로써 시스템을 선택적으로 백업하고, 템플릿 (230), 컨트롤러 로직(205) 및 글로벌 규칙(210)을 복사한다.

[0290] 클론(1705)이 생산 환경으로 만들어진 후, 클론(1705)은 클론의 코드, 구성 규칙, 보안 패치, 템플릿 및 다른 변경에 대해 변경이 이루어질 수 있는 개발 환경으로서 사용될 수 있다. 단계 1789에서, 개발 환경에 대한 변경은 구현 전에 테스트될 수 있다. 테스트 동안, 클론(1706)은 생산 환경(시스템(100)) 또는 시스템의 다른 구성요소로부터 격리될 수 있다. 이는 컨트롤러(1701)가 시스템(100)과 클론(1706) 사이의 하나 이상의 연결을 선택적으로 비활성화함으로써(예를 들어, 대역내 관리 연결(270)을 비활성화하고 및/또는 애플리케이션 네트워크 연결을 비활성화함으로써) 달성될 수 있다. 단계 1790에서, 변경된 개발 환경이 준비되어 있는지에 대한 결정이 이루어진다. 단계 1709가, 개발 환경이 아직 준비되지 않았다고 결정하는 경우(전형적으로 개발자에 의해 이루어지는 결정임), 프로세스 흐름은 클론 환경(1705)에 대한 추가 변경을 위해 단계 1789로 되돌아간다. 단계 1790이, 개발 환경이 준비되었다고 결정하는 경우, 단계 1791에서 개발 및 생산 환경이 전환될 수 있다. 즉, 컨트롤러는 개발 환경(1705)을 새로운 생산 환경으로 전환하고, 이전 생산 환경은 개발/새로운 생산 환경으로의 전이가 완료되고 만족될 때까지 유지될 수 있다.

[0291] 도 18a는 시스템의 변경 관리에 설정되고 사용될 수 있는 시스템(100)의 다른 예시적인 실시형태를 도시한다. 도 18a의 예에서, 시스템(100)은 컨트롤러(1801) 및 하나 이상의 환경(1802, 1803, 1804, 1805)을 포함한다. 시스템은 클론 환경(1807) 및 백업 시스템(1808)과 함께 도시되어 있다.

[0292] 컨트롤러(1801)는 본 명세서에 설명된 컨트롤러(200/1401/1501/1601/1701)와 유사한 방식으로 구성될 수 있고, 글로벌 시스템 규칙(210), 컨트롤러 로직(205), 템플릿(230) 및 시스템 상태(220) 요소를 포함할 수 있다.]컨트롤러(1801)는, 본 명세서에서 도 14a 내지 도 16f를 참조하여 설명된 방식으로 하나 이상의 다른 컨트롤러 또는 환경에 결합될 수 있다.]컨트롤러(1801)의 글로벌 규칙(210)은 다른 컨트롤러 및/또는 환경을 관리 및 제어

할 수 있는 규칙을 포함할 수 있다. 이러한 글로벌 규칙(210), 컨트롤러 로직(205), 시스템 상태(220) 및 템플릿(230)은 본 명세에서 도 1 내지 도 17b를 참조하여 설명된 것과 유사한 방식으로 컨트롤러(1801)를 통해 시스템 또는 환경을 설정, 제공 및 배치하는 데 사용될 수 있다. 각 환경은 다른 환경과 관련하여 포함하는 환경의 동작을 정의하는 글로벌 규칙(210)의 서브세트를 사용하여 구성될 수 있다.

[0293] 글로벌 시스템 규칙(210)은 또한 변경 관리 규칙(1811)을 포함할 수 있다. 변경 관리 규칙(1811)은 시스템, 글로벌 규칙, 및/또는 로직에 대한 변경이 필요할 수 있을 때에 사용될 수 있는 규칙 및/또는 명령어 세트를 포함한다. 변경 관리 규칙은 사용자 또는 개발자가 변경을 개발하고, 테스트 환경에서 변경을 테스트한 다음, 변경을 시스템 규칙(210) 내에서 새로운 구성 규칙 세트로 자동 변환함으로써 변경을 구현할 수 있도록 구성될 수 있다. 변경 관리 규칙(1711)은 (도 18a에 나타낸 바와 같이) 글로벌 시스템 규칙(210)의 서브세트일 수 있거나, 또는 글로벌 시스템 규칙(210)과 별개일 수 있다. 변경 관리 규칙(1711)은 글로벌 시스템 규칙(210)의 서브세트를 사용할 수 있다. 예를 들어, 글로벌 시스템 규칙(210)은 새로운 환경을 생성하도록 구성된 환경 생성 규칙의 서브세트를 포함할 수 있다. 변경 관리 규칙(1811)은 시스템(100)의 일부 또는 모든 양태를 복사 및 클로닝하기 위해 컨트롤러(1801)에 의해 설정되고 배치되는 시스템 또는 환경을 설정하고 사용하도록 구성될 수 있다. 변경 관리 규칙(1811)은 테스트 및 구현을 위해 시스템의 클론을 사용함으로써 구현 전에 시스템에 제안된 새로운 변경의 테스트를 허용하도록 구성될 수 있다.

[0294] 도 18a에 의해 나타낸 바와 같이 클론 환경(1807)은 하나 이상의 환경에 할당되고 컨트롤러(1801)의 글로벌 시스템 규칙(210) 및 변경 관리 규칙(1811)에 따라 설정될 수 있는 규칙, 컨트롤러 로직, 템플릿, 시스템 상태 데이터, 및 할당된 리소스(1820)를 갖는 컨트롤러(1807a)를 포함할 수 있다. 백업 시스템(1808)은 또한 하나 이상의 환경에 할당되고 컨트롤러(1801)의 글로벌 시스템 규칙(210) 및 변경 관리 규칙(1811)에 따라 설정될 수 있는 규칙, 컨트롤러 로직, 템플릿, 시스템 상태 데이터, 및 할당된 리소스(1821)를 갖는 컨트롤러(1808a)를 포함할 수 있다. 시스템은 사용자(또는 애플리케이션) 인터페이스(110) 또는 다른 사용자 인터페이스에 결합될 수 있다.

[0295] 클론 환경(1807)은 특정 환경 또는 시스템의 일부의 규칙, 로직, 템플릿, 시스템 상태, 애플리케이션 및/또는 리소스를 포함할 수 있다. 클론(1807)은 시스템(100)과 유사하거나 유사하지 않은 하드웨어를 포함할 수 있고, 클론(1807)은 가상 리소스를 사용하거나 사용하지 않을 수 있다. 클론(1807)은 애플리케이션으로서 설정될 수 있다. 클론(1807)은 환경에 대한 시스템(1801) 또는 컨트롤러(210)의 시스템 규칙(100) 내의 구성 규칙을 사용하여 설정 및 구성될 수 있다. 클론(1807)은 컨트롤러를 포함하거나 포함하지 않을 수 있고, 생산 환경과 컨트롤러를 공유할 수 있다. 클론(1807)은 상기에서 더욱 상세하게 설명된 바와 같이 할당된 네트워킹, 연산 리소스, 애플리케이션 네트워크 및/또는 데이터 스토리지 리소스를 포함할 수 있다. 이러한 리소스는 컨트롤러(1801)에 의해 제어되는 변경 관리 규칙(1811)을 사용하여 할당될 수 있다. 클론(1807)은 사용자에 의해 클론(1807)에 변경이 이루어질 수 있게 하는 사용자 인터페이스에 결합될 수 있다. 사용자 인터페이스는 시스템(100)의 사용자 인터페이스(110)와 동일하거나 상이할 수 있다.

[0296] 클론(1807)은 전체 시스템(100) 또는 하나 이상의 환경 및/또는 컨트롤러와 같은 시스템(100)의 일부에 사용될 수 있다. 예시적인 실시형태에서, 클론(1807)은 환경(1802)의 데이터 리소스(1820)에 결합되는 핫 스탠바이 데이터 리소스(1820a)를 포함할 수 있다. 핫 스탠바이 데이터 리소스(1820a)는 클론(1807)을 설정하고 변경을 테스트할 때 사용될 수 있다. 핫 스탠바이 데이터 리소스(1820a)는, 예를 들어 도 18b와 관련하여 본 명세서에 설명된 바와 같이, 변경 관리 동안 스토리지 리소스(1820)로부터 선택적으로 연결 해제 가능하거나 격리될 수 있다. 클론(1807)은 시스템(100)의 완전한 사본일 수도 아닐 수도 있다. 클론(1807)은, 선택적으로 완전히 활성화 및/또는 비활성화될 수 있고, 및/또는 단방향 판독 및/또는 기록 연결로 변환될 수 있는 대역내 관리 연결(270), 대역외 관리 연결(260) 및/또는 SAN 연결(280)을 통해 시스템(100)에 결합될 수 있다. 따라서, 클론 환경(1807)의 휘발성 데이터에 대한 연결은 클론 환경(1807)이 테스트 동안 생산 환경으로부터 격리될 때 또는 새로운 생산 환경으로서 온라인으로 준비될 때까지 클론 데이터를 판독 전용으로 하도록 변경될 수 있다.

[0297] 이전 생산 환경을 새로운 생산 환경으로 전환할 때, 컨트롤러(1801)는 프론트 엔드, 로드 밸런서 또는 다른 애플리케이션 또는 리소스가 새로운 생산 환경으로 향하도록 지시할 수 있다. 따라서, 변경이 발생하면 사용자, 애플리케이션 리소스 및/또는 다른 연결이 재지향(redirect)될 수 있다. 이는, 예를 들어, ip/ipoib 주소의 리스트, 인피니밴드 GUID, dns 서버, 인피니밴드 파티션/opensm 구성의 변경, 또는 네트워킹 리소스에 명령어를 전송함으로써 달성될 수 있는 소프트웨어 정의 네트워킹(SDN) 구성의 변경을 포함하지만 이에 한정되지 않는 방법으로 달성될 수 있다. 프론트 엔드, 로드 밸런서 또는 다른 애플리케이션 및/또는 리소스는 시스템, 환경, 및/또는 데이터베이스, 미들웨어 및/또는 다른 백엔드를 포함하지만 이에 한정되지 않는 다른 애플리케이션으로

향할 수 있다. 이와 같이 로드 밸런서는 변경 관리에 사용되어 이전 생산 환경으로부터 새로운 환경으로 전환할 수 있다.

[0298] 클론(1807) 및 백업(1808)은 시스템에 대한 변경의 양태를 관리하는 데 설정되고 사용될 수 있다. 이러한 변경은 코드, 구성 규칙, 보안 패치, 템플릿에 대한 변경, 하드웨어 변경, 구성요소 및/또는 종속 애플리케이션의 추가/제거 및 다른 변경을 포함할 수 있지만 이에 한정되지 않는다. 백업(1808)은 전체 시스템 또는 하나 이상의 환경 및/또는 컨트롤러(1801)와 같은 시스템의 일부에 사용될 수 있다. 백업(1808)은 상기에서 더욱 상세하게 설명된 바와 같이 네트워킹, 연산 리소스, 애플리케이션 네트워크 및/또는 데이터 스토리지 리소스를 포함할 수 있다. 백업(1808)은 컨트롤러를 포함하거나 포함하지 않을 수 있다. 백업(1808)은 시스템(100)의 완전한 사본일 수 있다. 백업(1808)은 백업에 포함된 구성 규칙으로부터 시스템/환경/애플리케이션을 재구축하는데 필요한 데이터를 포함할 수 있고 모든 애플리케이션 데이터를 포함할 수 있다. 백업(1808)은 애플리케이션으로서 또는 시스템(100)과 유사하거나 유사하지 않은 하드웨어를 사용하여 설정될 수 있다. 백업(1808)은, 선택적으로 활성화 및/또는 비활성화될 수 있고, 및/또는 일방향 판독 및/또는 기록 연결로 변환될 수 있는 대역내 관리 연결(270), 대역외 관리 연결(260) 및/또는 SAN 연결(280)을 통해 시스템(100)에 결합될 수 있다.

[0299] 도 18b는 특히 도 18a 시스템이 휘발성 데이터를 포함하거나 데이터베이스가 기록 가능한 경우, 변경 관리에서 도 18a 시스템의 사용을 도시하는 예시적인 프로세스 흐름이다. 이러한 데이터베이스는 시스템의 환경에 의해 사용되는 스토리지 리소스의 일부일 수 있다. 단계 1870에서, 시스템은 글로벌 시스템 규칙을 사용하여 (생산 환경을 포함하여) 배치된다.

[0300] 단계 1871에서, 생산 환경은 그 후에 변경 관리 규칙(1811)을 포함하는 글로벌 시스템 규칙(210), 및 클론 환경이 시스템에 기록하는 것이 비활성화된 판독 전용 환경을 생성하기 위해 클론 환경의 메인 컨트롤러(1801) 또는 컨트롤러에 의한 리소스 할당을 사용하여 클로닝될 수 있다. 그 후, 클론 환경은 개발 환경으로서 사용될 수 있다.

[0301] 단계 1872에서, 핫 스탠바이(1820a)가 활성화되고 시스템 100)에서 변경되는 임의의 휘발성 데이터를 저장하기 위해 클론 환경(1807)에 할당된다. 클론 데이터는 개발 환경의 새 버전이 업데이트된 데이터로 테스트될 수 있도록 업데이트된다. 핫 싱크 데이터(hot synched data)는 언제든지 끌 수 있다. 예를 들어, 핫 싱크 데이터는 기록이 이전 환경 또는 생산으로부터 개발 환경으로 테스트될 때 꺼질 수 있다.

[0302] 단계 1873에서, 사용자는 개발 환경으로서 클론 환경(1807)을 사용하여 변경을 수행할 수 있다. 그 후, 개발 환경에 대한 변경은 단계 1874에서 테스트된다. 단계 1875에서, 변경된 개발 환경이 준비되어 있는지에 대한 결정이 이루어진다(전형적으로 이러한 결정은 개발자에 의해 이루어진다). 단계(1875)가 변경이 준비되지 않았다고 결정하는 경우, 프로세스 흐름은 단계 1873으로 되돌아가서 사용자가 되돌아가서 개발 환경에 다른 변경을 행할 수 있다. 단계 1875가 변경이 가동할 준비가 되었다고 결정하는 경우, 프로세스 흐름은 단계 1876로 진행하고, 여기서 구성 규칙이 특정 환경에 대하여 시스템 또는 컨트롤러에서 업데이트되고 새로운 업데이트된 환경을 배치하는 데 사용될 것이다.

[0303] 그 후, 단계 1877에서, 개발 환경(또는 새로운 환경)은 가동 전에 원하는 리소스 및 하드웨어 할당에 의해 원하는 최종 구성의 변화와 함께 재배치될 수 있다. 다음 단계 1878에서, 원래 생산 환경의 기록 능력이 비활성화되고, 원래 생산 환경은 판독 전용이 된다. 원래 생산 환경은 판독 전용이지만 원래 프로덕션 환경(또는 아마도 새로운 프로덕션 환경)으로부터의 임의의 새로운 데이터가 1878의 일부로서 캐시되어 전이 데이터로 식별될 수 있다. 일례로서, 데이터는 데이터베이스 서버 또는 다른 적절한 위치(예를 들어, 공유 환경)에 캐시될 수 있다. 그 후, 개발 환경(또는 새로운 환경) 및 이전 프로덕션 환경은 단계 1879에서 전환되어 개발 환경(또는 새로운 환경)이 생산 환경이 된다.

[0304] 이 전환 후에, 새로운 생산 환경이 단계 1880에서 기록 가능하게 된다. 새로운 생산 환경이 단계 1881에서 개발자에 의해 결정된 대로 작동한다고 간주하게 되면, (이러한 데이터가 단계 1878에서 캐시되어 있는 경우) 전환하는 프로세스 중의 임의의 데이터 손실이 새로운 환경에 기록된 데이터로 단계 1884에서 조정될 수 있다. 이러한 조정 후에, 변경은 종료된다(단계 1885).

[0305] 단계 1881이 새로운 생산 환경이 작동하지 않는다고 결정하는 경우(예를 들어, 시스템이 이전 시스템으로 복귀해야 할 필요가 있는 문제가 식별되는 경우), 환경은 단계 1882에서 다시 전환되어 이전 생산 환경이 다시 생산 환경이 된다. 단계 182의 일부로서, 컨트롤러(1801) 상의 주요 환경에 대한 구성 규칙은 이제 복귀된 생산 환경에 사용되었던 이전 버전으로 되돌아간다.

[0306] 단계 1883에서, 데이터베이스의 변경은, 예를 들어 캐시된 데이터를 사용하여 결정될 수 있고; 이전 구성 규칙을 사용하여 이전 생산 환경에 데이터가 복원된다. 단계 1883을 지원하기 위해, 데이터베이스는 그에 대해 이루어진 변경의 로그를 유지하고, 이에 따라 단계 1883이 복귀될 필요가 있을 수 있는 변경을 결정할 수 있게 한다. 백업 데이터베이스는 캐시된 데이터가 추적 및 클럭킹되는 전술한 바와 같이 데이터를 캐시하는 데 사용될 수 있고, 클럭은 어떤 변경이 이루어졌는지를 결정하기 위해 복귀될 수 있다. 이를 위해 스냅샷 및 로그가 사용될 수 있다.

[0307] 캐싱된 데이터가 1883에서 복원된 후, 프로세스는 다시 시작하기 원한다면 단계 1871로 되돌아갈 수 있다.

[0308] 본 명세서에 설명된 예시적인 변경 관리 시스템은, 예를 들어 하드웨어 또는 소프트웨어를 업그레이드, 추가 또는 제거할 때, 소프트웨어를 패치할 때, 시스템 장애가 검출될 때, 하드웨어 장애 또는 검출 중에 호스트를 마이그레이션할 때, 동적 리소스 마이그레이션를 위해, 구성 규칙 또는 템플릿의 변경을 위해, 및/또는 임의의 다른 시스템 관련 변경을 행할 때 사용될 수 있다. 컨트롤러(1801) 또는 시스템(100)은 장애를 검출하도록 구성될 수 있고, 장애의 검출시 컨트롤러에 대해 시스템에 이용 가능한 다른 하드웨어에 대한 변경 관리 규칙 또는 기존 구성 규칙을 자동적으로 구현할 수 있다. 사용될 수 있는 장애 검출 방법의 예는 호스트의 핑, 애플리케이션의 쿼리 및 다양한 테스트 또는 테스트 스위트의 실행을 포함하지만 이에 한정되지 않는다. 본 명세서에 설명된 변경 관리 구성 규칙은 장애가 검출될 때에 구현될 수 있다. 이러한 규칙은 백업 환경의 자동 생성, 장애가 검출될 때에 컨트롤러에 의해 구현되는 데이터 또는 리소스의 자동 마이그레이션을 트리거할 수 있다. 백업 리소스의 선택은 리소스 파라미터에 기초할 수 있다. 이러한 리소스 파라미터는 사용 정보, 속도, 구성 규칙, 및 데이터 용량 및 사용을 포함할 수 있지만 이에 한정되지 않는다.

[0309] 본 명세서에 설명된 바와 같이, 변경이 발생할 때마다, 컨트롤러는 그의 로그 및 실제로 실행된 것의 로그를 생성할 것이다. 보안 또는 시스템 업데이트를 위해, 본 명세서에 설명된 컨트롤러는 구성 규칙에 따라 자동적으로 켜지고 꺼지고 IT 시스템 상태를 업데이트하도록 구성될 수 있다. 전력을 절약하기 위해 리소스를 끌 수 있다. 다른 시간에 다른 효율성을 위해 리소스를 켜거나 마이그레이션할 수 있다. 마이그레이션시 구성 규칙을 따르고 환경 또는 시스템의 백업 또는 복사가 이루어질 수 있다. 보안 위반이 있는 경우 컨트롤러는 공격받은 영역을 분리하고 차단할 수 있다.

[0310] 본 발명은 예시적인 실시형태와 관련하여 위에서 설명되었지만, 여전히 본 발명의 범위 내에 있는 다양한 변형이 이루어질 수 있다. 본 발명의 이러한 변형은 본 명세서의 교시를 검토하면 인식할 수 있을 것이다.

[0311] 부록 A: 예시적인 스토리지 연결 프로세스

[0312] 본 부록은 다수의 시스템들 사이의 스토리지 리소스 공유와 연관되는 예시적인 프로세스 및 예시적인 규칙들을 설명한다. 본 부록은 스토리지 연결 프로세스의 일례일 뿐이며 연산 리소스를 스토리지 리소스에 연결하기 위한 다른 기술들이 사용될 수 있다는 점에 유의해야 한다. 달리 주지하지 않은 한, 이들 규칙은 스토리지 연결을 개시하려고 하는 모든 시스템에 적용된다.

[0313] 이 부록 A에 대한 정의:

[0314] 스토리지 리소스: 스토리지 전송을 통해 공유될 수 있는 블록, 파일, 또는 파일 시스템.

[0315] 스토리지 전송: 스토리지 리소스를 로컬로 또는 원격으로 공유하는 방법. 예시들은 iSCSI/iSER, NVMEoF, NFS, Samba 파일 공유(File Share)이다.

[0316] 시스템: 지정된 스토리지 전송을 통해 스토리지 리소스에 연결하려고 시도할 수 있는 모든 것. 시스템: 지정된 스토리지 전송을 통해 스토리지 리소스에 연결하려고 시도할 수 있는 모든 것.

[0317] 시스템은 임의의 수의 스토리지 전송을 지원할 수 있고, 어느 전송을 사용할지를 스스로 결정할 수 있다. 판독-전용: 판독-전용 스토리지 리소스는 내포된 데이터의 수정을 허용하지 않는다. 추가적인 보호수단으로서, 일부 데이터스토어가 데이터를 지원하는 스토리지 리소스를 판독 전용으로 설정할 수 있다(예컨대, LVM LV를 판독 전용으로 설정).

[0318] 판독-기록(또는 휘발성): 판독-기록(휘발성) 스토리지 리소스는 스토리지 리소스에 연결되는 시스템에 의해 내용이 수정될 수 있는 스토리지 리소스이다.

[0319] 규칙: 시스템이 정해진 스토리지 리소스에 연결될 수 있는지의 여부를 컨트롤러가 결정할 때 준수해야 하는 일련의 규칙이 존재한다.

[0320] 1. 판독-기록(또는 휘발성): 판독-기록(휘발성) 스토리지 리소스는 스토리지 리소스에 연결되는 시스템에 의해 내용이 수정될 수 있는 스토리지 리소스이다.

[0321] 2. 판독-기록 스토리지 리소스는 단일의 시스템에 의해서만 연결되어야 한다.

[0322] 3. 판독-기록 스토리지 리소스는 판독-전용으로서 연결되지 않아야 한다.

[0323] 4. 판독 전용 스토리지 리소스는 다수의 스토리지 전송에서 내보내질 수 있다.

[0324] 5. 판독 전용 스토리지 리소스는 다수의 시스템으로부터 연결될 수 있다.

[0325] 6. 판독 전용 스토리지 리소스는 판독-기록으로서 연결되지 않아야 한다.

[0326] 프로세스

[0327] 연결 프로세스를 함수로 간주하려면, 다음과 같은 두 가지 인수가 필요하다:

[0328] 1. 스토리지 리소스 ID

[0329] 2. 지원되는 스토리지 전송 리스트(명령에 따라 우선처리됨)

[0330] 먼저, 요청된 스토리지 리소스가 판독 전용인지 판독-기록인지를 결정한다.

[0331] 판독-기록이면, 판독-기록 스토리지 리소스를 단일의 연결로 제한하고 있기 때문에, 스토리지 리소스가 이미 연결되어 있는지를 확인하기 위해 검사해야 한다. 이미 연결되어 있으면, 스토리지 리소스를 요청하는 시스템이 현재 연결되어 있는 시스템인지를 확인한다(이는, 예를 들어, 다시 연결하는 경우에 발생함). 그렇지 않으면, 다수의 시스템이 동일한 판독-기록 스토리지 리소스에 연결될 수 없기 때문에, 에러가 발생한다. 요청 시스템이 이 스토리지 리소스에 연결된 시스템이면, 가용 스토리지 전송들 중 하나가 이 스토리지 리소스에 대한 현재 내보내기와 일치하는지를 확인한다. 그렇다면, 연결 정보를 요청 시스템에 전달한다. 그렇지 않다면, 판독-기록 스토리지 리소스를 다수의 스토리지 전송에 대하여 기능시킬 수 없으므로, 에러가 발생한다.

[0332] 판독 전용 및 연결되지 않은 판독-기록 스토리지 리소스의 경우, 공급된 스토리지 전송 리스트를 반복하고 해당 전송을 사용하여 스토리지 리소스를 내보내려고 시도한다. 내보내기에 실패하면, 스토리지 전송에 성공하거나 소진할 때까지 해당 리스트를 계속한다. 소진되면, 요청 시스템에게 스토리지 리소스에 연결할 수 없음을 통지한다. 내보내기에 성공하면, 연결 정보 및 신규 (리소스, 전송) => (시스템) 관계를 데이터베이스에 저장한다. 이후, 요청 시스템에 스토리지 전송 연결 정보가 전달된다.

[0333] 시스템: 스토로지 연결은 현재 정상 작동 중에 컨트롤러 및 연산 데몬에 의해 수행된다. 그러나, 향후의 반복들은 서비스를 스토리지 리소스에 직접 연결하고 연산 데몬을 우회할 수도 있다. 이는 예시적인 서비스 물리적 배치에 대한 요구 사항일 수 있으며, 가상 머신 배치에 대해서도 동일한 프로세스를 사용할 수 있음은 물론이다.

[0334] 부록 B: 예시적인 OverlayFS 연결

[0335] 서비스는 OverlayFS를 이용해서 공통 파일 시스템 객체를 재사용하고, 서비스 패키지 사이즈를 줄인다.

[0336] 이 예에서의 서비스는 3개 이상의 스토리지 리소스를 포함한다:

[0337] 1. 플랫폼. 이는 기본 리눅스 파일시스템을 포함하고 판독 전용으로 액세스된다.

[0338] 2. 서비스. 이는 서비스의 운용과 직접적으로 관련되는 모든 소프트웨어(NetThunder ServiceDaemon, OpenRC 스크립트, 바이너리 등)를 포함한다. 이 스토리지 리소스는 판독 전용으로 액세스된다.

[0339] 3. 휘발성. 이들 스토리지 리소스는 시스템에 대한 모든 변화를 포함하고, 서비스 내에서 FVM에 의해 관리된다(물리적, 컨테이너, 및 가상 머신 배치의 경우).

[0340] 가상 머신에서의 실행시에, 서비스는 다음을 수행하는 로직을 포함하는 initramfs와 함께 커스텀 Finux 커널을 사용해서 Qemu에서 직접 커널 부팅된다.

[0341] 1. 가용 판독-기록 디스크로부터 LVM 볼륨 그룹(VG)을 어셈블한다.

[0342] * 이 VG는 서비스에 대한 휘발성 스토리지 데이터를 모두 포함하는 하나의 Fogical 볼륨(FV)을 포함한다.

[0343] 2. 플랫폼, 서비스, 및 FV를 마운트

[0344] 3. 통합 파일시스템(이 경우에는, OverlayFS)을 사용해서 3개의 파일시스템을 결합한다.

[0345] 동일한 프로세스가 물리적 배치에 사용될 수 있다. 한 가지 옵션은 커널을 PXEBoot 또는 IPMI ISO Boot를 통해 부팅되는 경량 OS에 원격으로 제공하고 나서, 신규 리얼 커널에 kexec하는 것이다. 또는 경량 OS를 스킵하고, PXE를 커널에 직접 부팅하는 것이다. 이러한 시스템은 스토리지 리소스에 연결하기 위해 커널 initramfs에 추가적인 로직을 필요로 할 수 있다.

[0346] OverlayFS 구성은 다음과 같다:

[0347] /----------\

[0348] | Volatile Fayer (LV) (RW) |

[0349] +----------+

[0350] | Service Layer (RO) |

[0351] +----------+

[0352] | Platform Layer (RO) |

[0353] \----------/

[0354] OverlayFS에 의한 일부 제한으로 인해, 특수한 디렉토리 '/data'를 "out of tree"로 표시될 수 있게 한다. 이 디렉토리는 서비스 패키지가 생성될 때 '/data'를 생성하면 서비스에 이용 가능하다. 이 특수한 디렉토리는 'mount-rbind'를 통해 마운트되어 OverlayFS 내부에 없는 휘발성 레이어의 서브세트에의 액세스를 허용한다. 이는 OverlayFS의 일부인 공유 디렉토리를 지원하지 않는 NFS(Network File System)와 같은 애플리케이션에 필요한다.

[0355] 커널 파일시스템 레이아웃:

[0356] /

[0357] +-- platform/

[0358] +-- bin/

[0359] +-- .../

[0360] +-- service/

[0361] +-- data/ [optional]

[0362] +-- bin/

[0363] +-- ...

[0364] +-- volatile

[0365] +-- work/

[0366] +-- root/

[0367] +-- bin/

[0368] +-- data/ [if present in /service/]

[0369] +-- ...

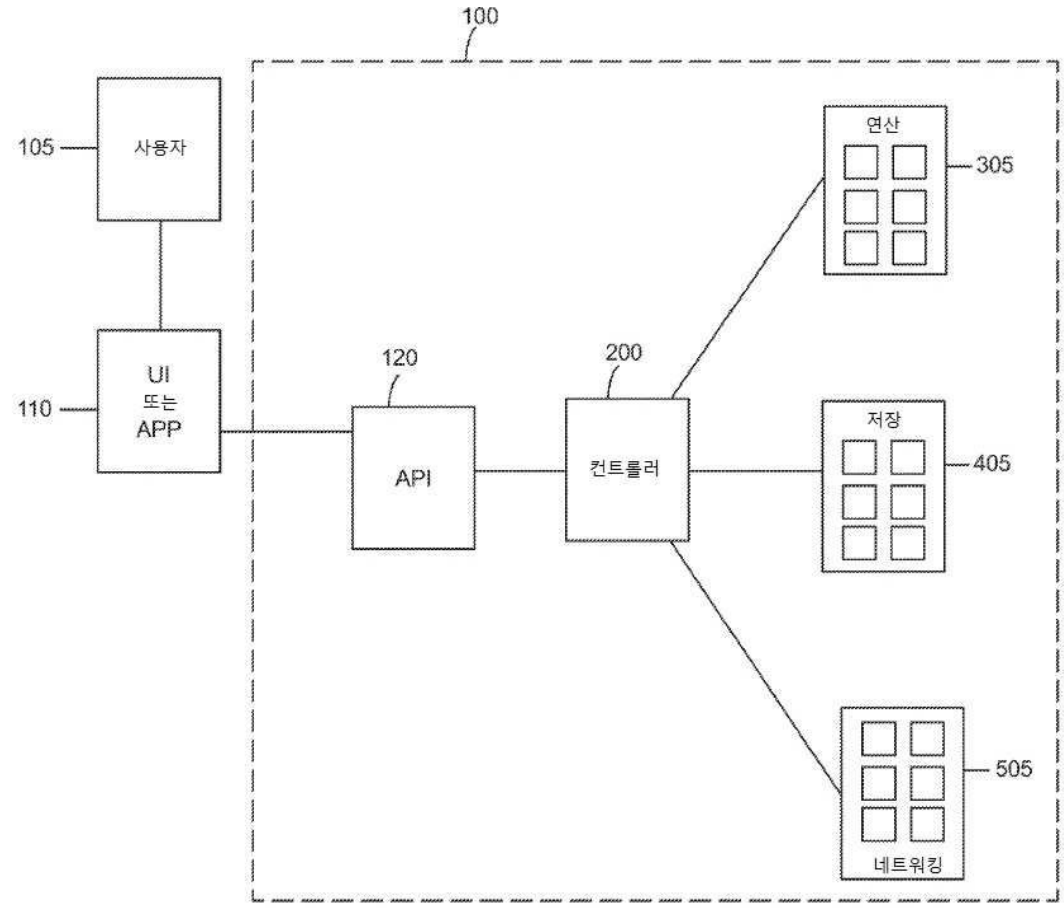
[0370] +-- new_root/

[0371] +-- ...

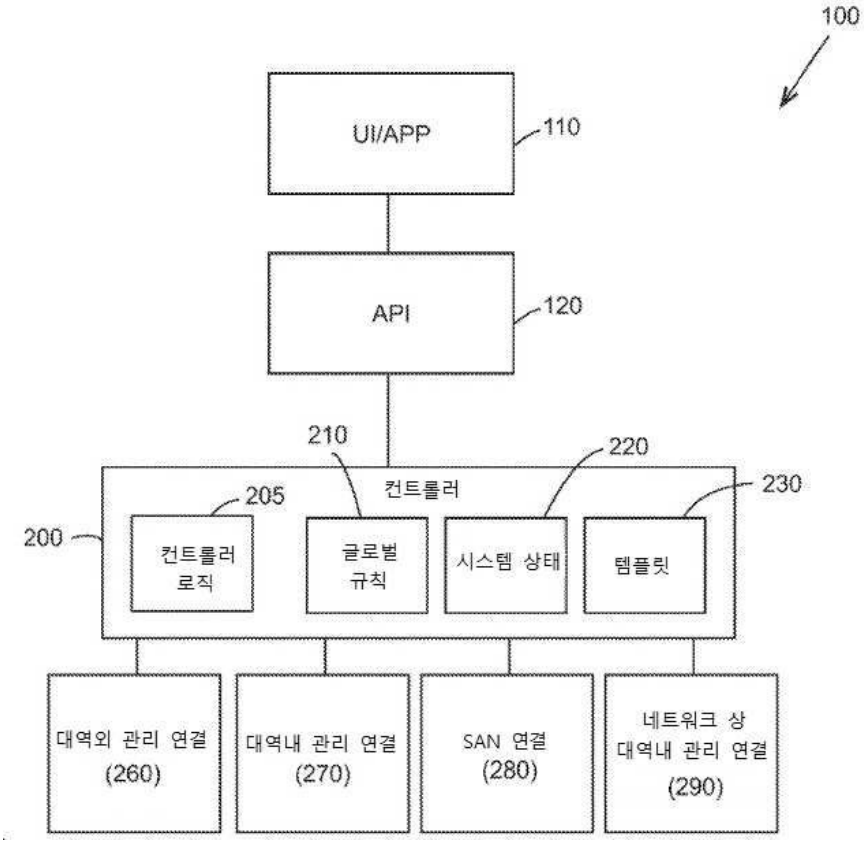
[0372] /new_root 디렉토리를 생성하고, 이를 OverlayFS를 구성하기 위한 타겟으로서 사용한다. OverlayFS가 구성되면, /new_directory에 exec_root하고 시스템이 모든 가용 리소스로 정상적으로 시작한다.

도면

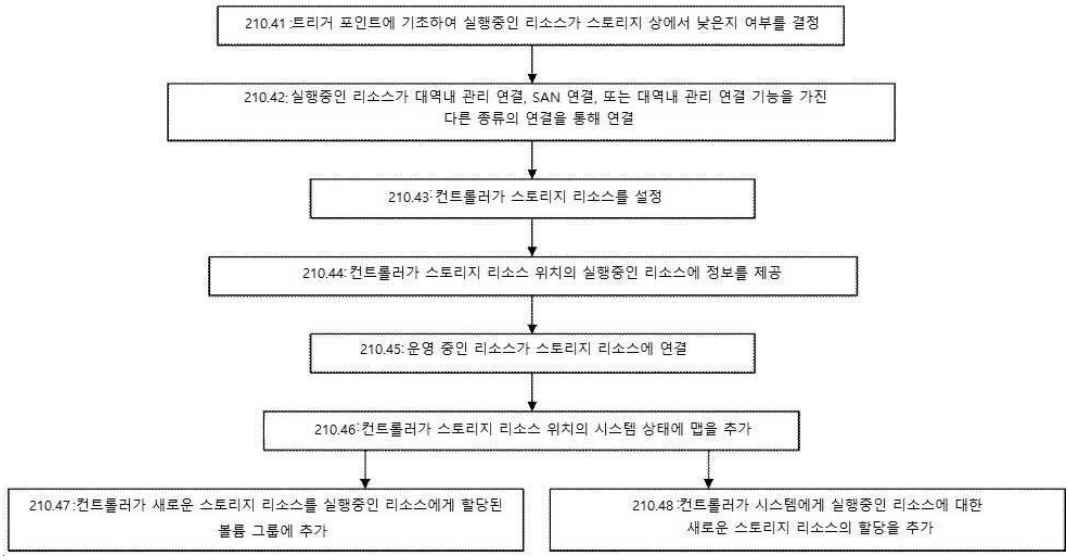
도면1



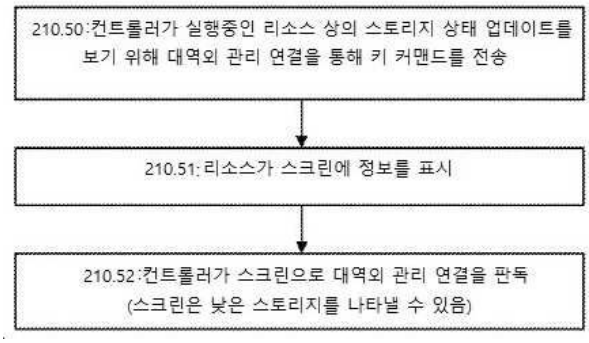
도면2a


100
UI/APP
110
API
120
210
220
230
205
컨트롤러
200
컨트롤러 로직
글로벌 규칙
시스템 상태
템플릿
대역외 관리 연결 (260)
대역내 관리 연결 (270)
SAN 연결 (280)
네트워크 상 대역내 관리 연결 (290)

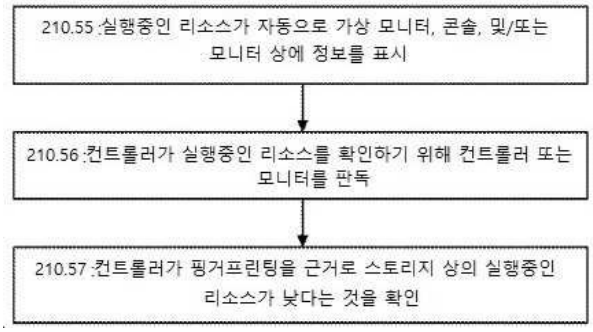
도면2b


210.41 :트리거 포인트에 기초하여 실행중인 리소스가 스토리지 상에서 낮은지 여부를 결정
210.42: 실행중인 리소스가 대역내 관리 연결, SAN 연결, 또는 대역내 관리 연결 기능을 가진 다른 종류의 연결을 통해 연결
210.43: 컨트롤러가 스토리지 리소스를 설정
210.44: 컨트롤러가 스토리지 리소스 위치의 실행중인 리소스에 정보를 제공
210.45: 운영 중인 리소스가 스토리지 리소스에 연결
210.46: 컨트롤러가 스토리지 리소스 위치의 시스템 상태에 맵을 추가
210.47 :컨트롤러가 새로운 스토리지 리소스를 실행중인 리소스에게 할당된 볼륨 그룹에 추가
210.48 :컨트롤러가 시스템에게 실행중인 리소스에 대한 새로운 스토리지 리소스의 할당을 추가

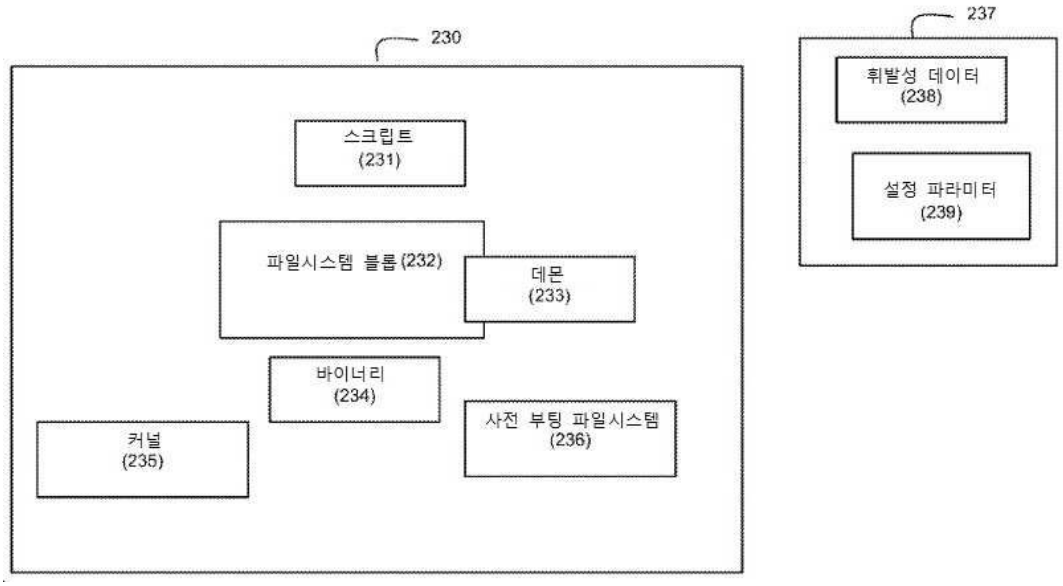
도면2c



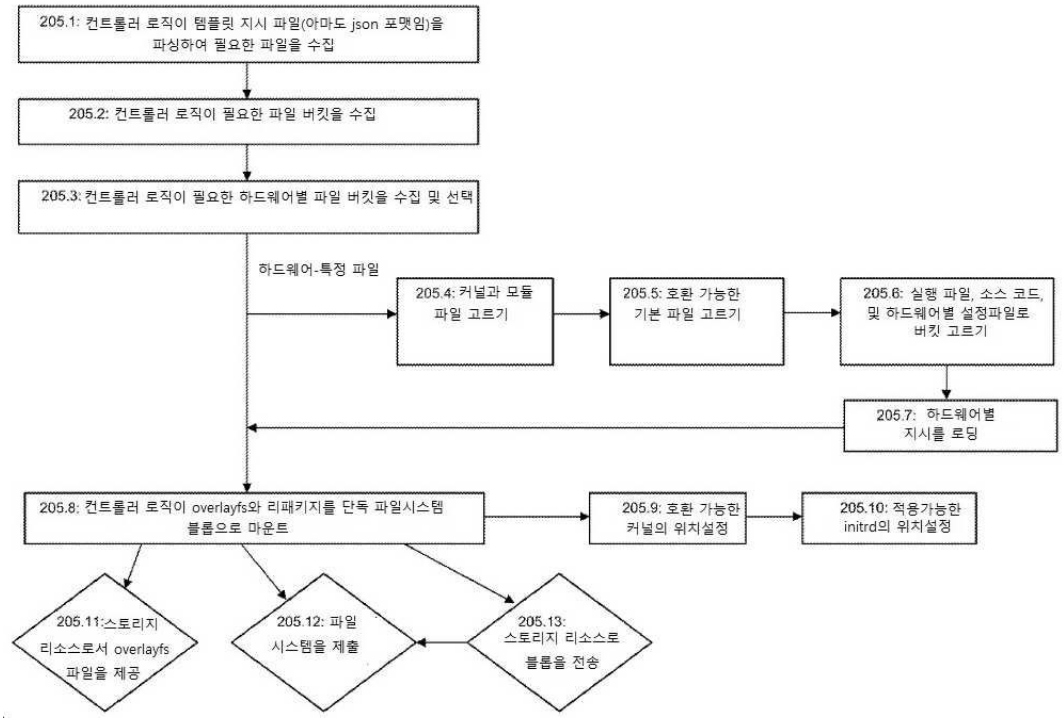
도면2d



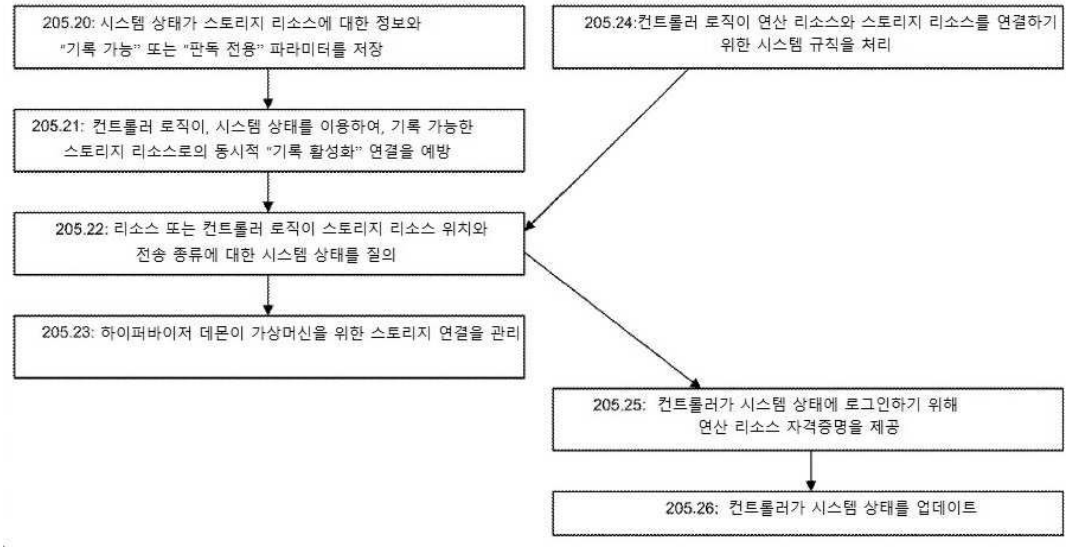
도면2e



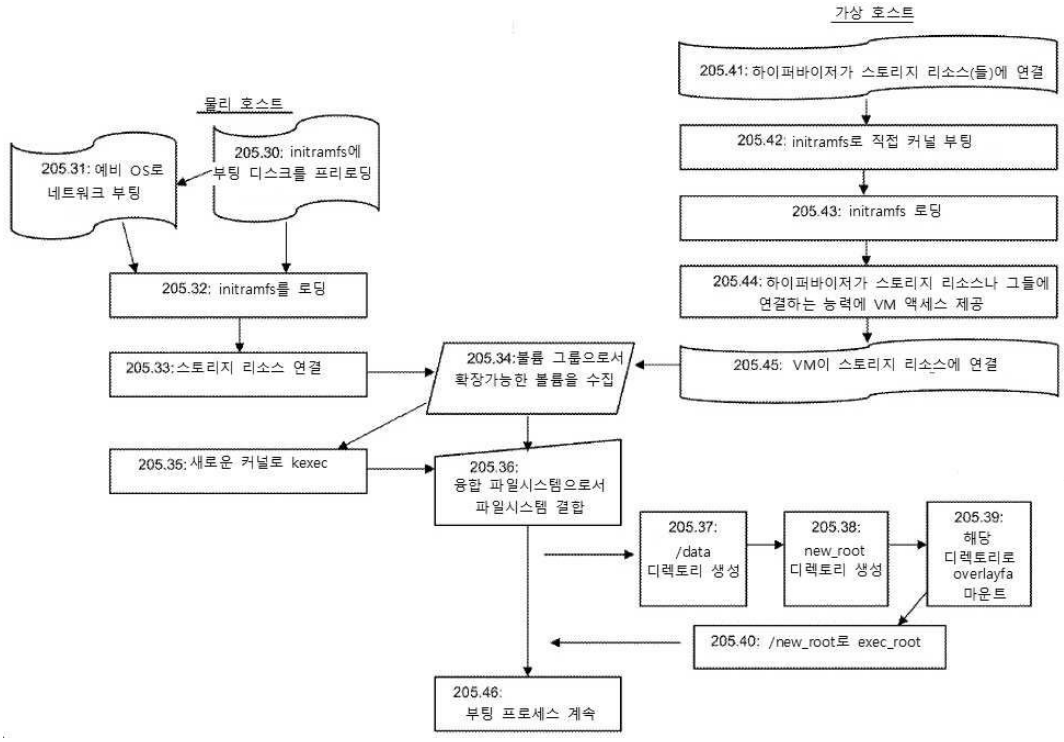
도면2f


205.1: 컨트롤러 로직이 템플릿 지시 파일(아마도 json 포맷임)을 파싱하여 필요한 파일을 수집
205.2: 컨트롤러 로직이 필요한 파일 버킷을 수집
205.3: 컨트롤러 로직이 필요한 하드웨어별 파일 버킷을 수집 및 선택
하드웨어-특정 파일
205.4: 커널과 모듈 파일 고르기
205.5: 호환 가능한 기본 파일 고르기
205.6: 실행 파일, 소스 코드, 및 하드웨어별 설정파일로 버킷 고르기
205.7: 하드웨어별 지시를 로딩
205.8: 컨트롤러 로직이 overlayfs와 리패키지를 단독 파일시스템 블롭으로 마운트
205.9: 호환 가능한 커널의 위치설정
205.10: 적용가능한 initrd의 위치설정
205.11:스토리지 리소스로서 overlayfs 파일을 제공
205.12: 파일 시스템을 제출
205.13: 스토리지 리소스로 블롭을 전송

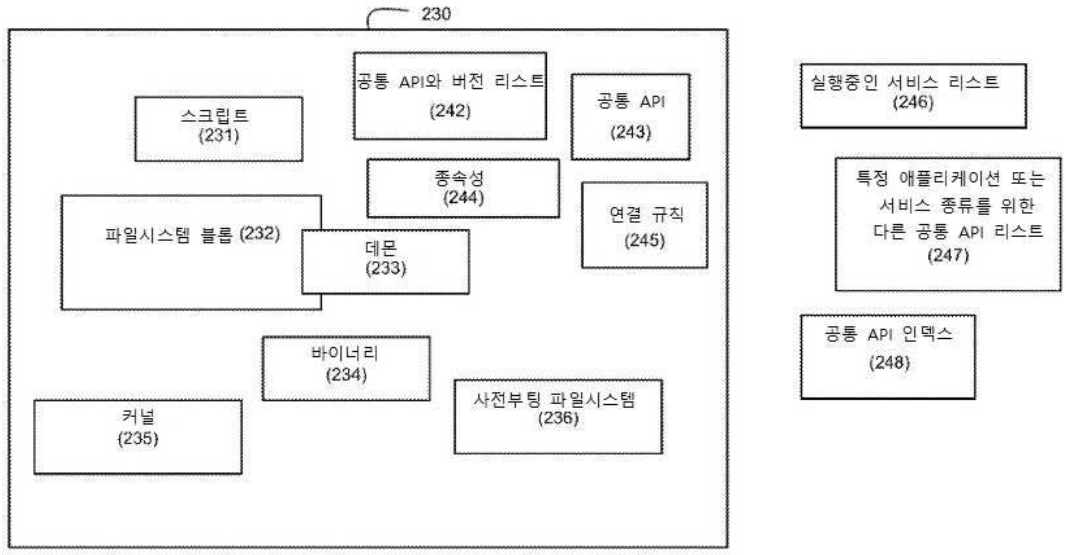
도면2g


205.20: 시스템 상태가 스토리지 리소스에 대한 정보와 "기록 가능" 또는 "판독 전용" 파라미터를 저장
205.24:컨트롤러 로직이 연산 리소스와 스토리지 리소스를 연결하기 위한 시스템 규칙을 처리
205.21: 컨트롤러 로직이, 시스템 상태를 이용하여, 기록 가능한 스토리지 리소스로의 동시적 "기록 활성화" 연결을 예방
205.22: 리소스 또는 컨트롤러 로직이 스토리지 리소스 위치와 전송 종류에 대한 시스템 상태를 질의
205.23: 하이퍼바이저 데몬이 가상머신을 위한 스토리지 연결을 관리
205.25: 컨트롤러가 시스템 상태에 로그인하기 위해 연산 리소스 자격증명을 제공
205.26: 컨트롤러가 시스템 상태를 업데이트

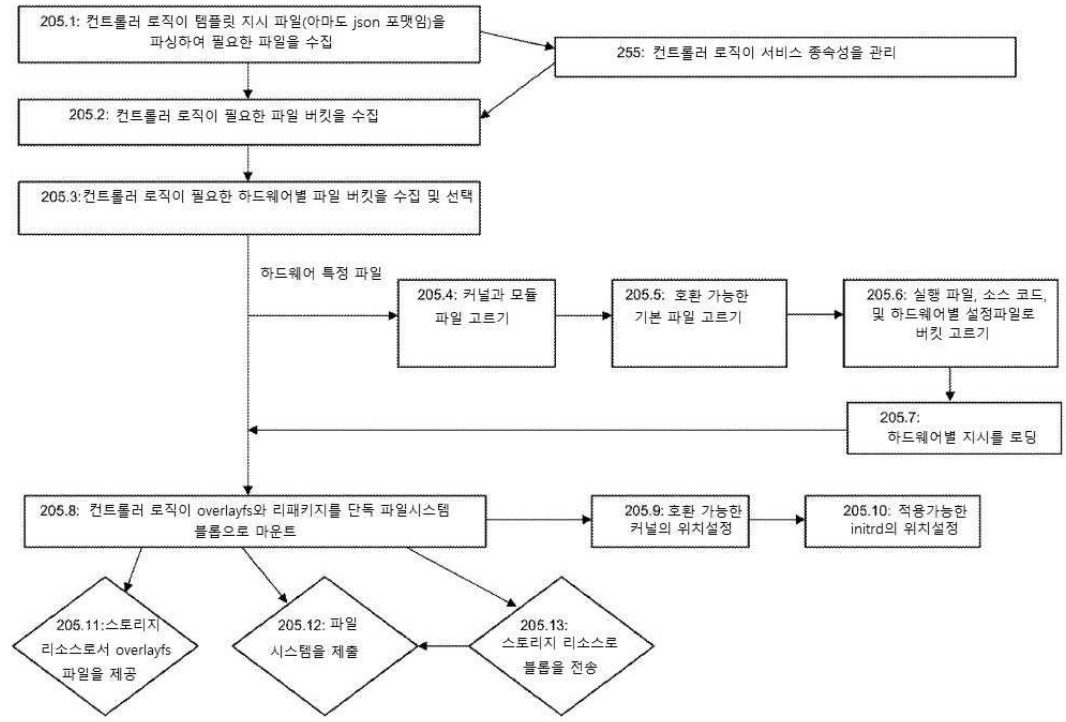
도면2h


가상 호스트
205.41: 하이퍼바이저가 스토리지 리소스(들)에 연결
205.42: initramfs로 직접 커널 부팅
205.43: initramfs 로딩
205.44: 하이퍼바이저가 스토리지 리소스나 그들에 연결하는 능력에 VM 액세스 제공
205.45: VM이 스토리지 리소스에 연결
물리 호스트
205.31: 예비 OS로 네트워크 부팅
205.30: initramfs에 부팅 디스크를 프리로딩
205.32: initramfs를 로딩
205.33: 스토리지 리소스 연결
205.34: 볼륨 그룹으로서 확장가능한 볼륨을 수집
205.35: 새로운 커널로 kexec
205.36: 융합 파일시스템으로서 파일시스템 결합
205.37: /data 디렉토리 생성
205.38: new_root 디렉토리 생성
205.39: 해당 디렉토리로 overlayfa 마운트
205.40: /new_root로 exec_root
205.46: 부팅 프로세스 계속

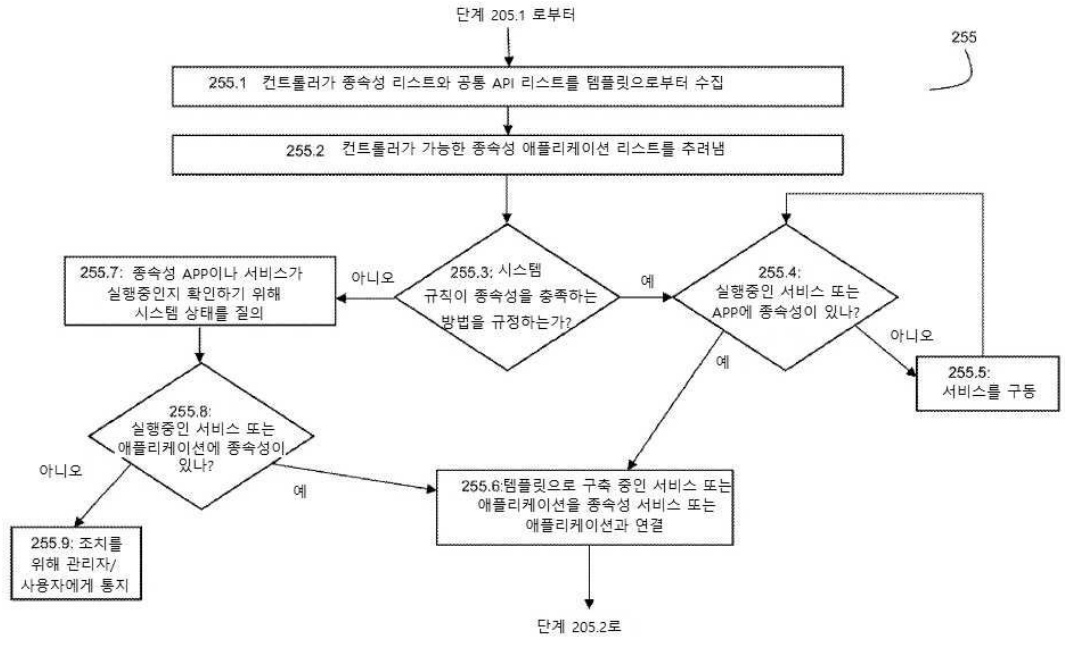
도면2i


230
스크립트 (231)
공통 API와 버전 리스트 (242)
공통 API (243)
종속성 (244)
연결 규칙 (245)
파일시스템 볼륨 (232)
데몬 (233)
바이너리 (234)
커널 (235)
사전부팅 파일시스템 (236)
실행중인 서비스 리스트 (246)
특정 애플리케이션 또는 서비스 종류를 위한 다른 공통 API 리스트 (247)
공통 API 인덱스 (248)

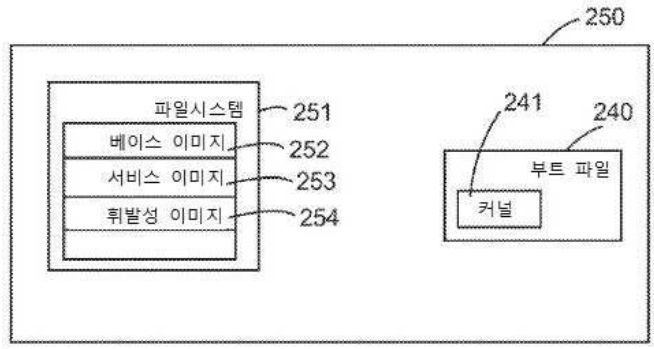
도면2j


205.1: 컨트롤러 로직이 템플릿 지시 파일(아마도 json 포맷임)을 파싱하여 필요한 파일을 수집
255: 컨트롤러 로직이 서비스 종속성을 관리
205.2: 컨트롤러 로직이 필요한 파일 버킷을 수집
205.3:컨트롤러 로직이 필요한 하드웨어별 파일 버킷을 수집 및 선택
하드웨어 특정 파일
205.4: 커널과 모듈 파일 고르기
205.5: 호환 가능한 기본 파일 고르기
205.6: 실행 파일, 소스 코드, 및 하드웨어별 설정파일로 버킷 고르기
205.7: 하드웨어별 지시를 로딩
205.8: 컨트롤러 로직이 overlayfs와 리패키지를 단독 파일시스템 블롭으로 마운트
205.9: 호환 가능한 커널의 위치설정
205.10: 적용가능한 initrd의 위치설정
205.11:스토리지 리소스로서 overlayfs 파일을 제공
205.12: 파일 시스템을 제출
205.13: 스토리지 리소스로 블롭을 전송

도면2k


단계 205.1 로부터
255
255.1 컨트롤러가 종속성 리스트와 공통 API 리스트를 템플릿으로부터 수집
255.2 컨트롤러가 가능한 종속성 애플리케이션 리스트를 추려냄
255.3: 시스템 규칙이 종속성을 충족하는 방법을 규정하는가?
아니오
예
255.4: 실행중인 서비스 또는 APP에 종속성이 있나?
아니오
예
255.5: 서비스를 구동
255.7: 종속성 APP이나 서비스가 실행중인지 확인하기 위해 시스템 상태를 질의
255.8: 실행중인 서비스 또는 애플리케이션에 종속성이 있나?
아니오
예
255.9: 조치를 위해 관리자/사용자에게 통지
255.6:템플릿으로 구축 중인 서비스 또는 애플리케이션을 종속성 서비스 또는 애플리케이션과 연결
단계 205.2로

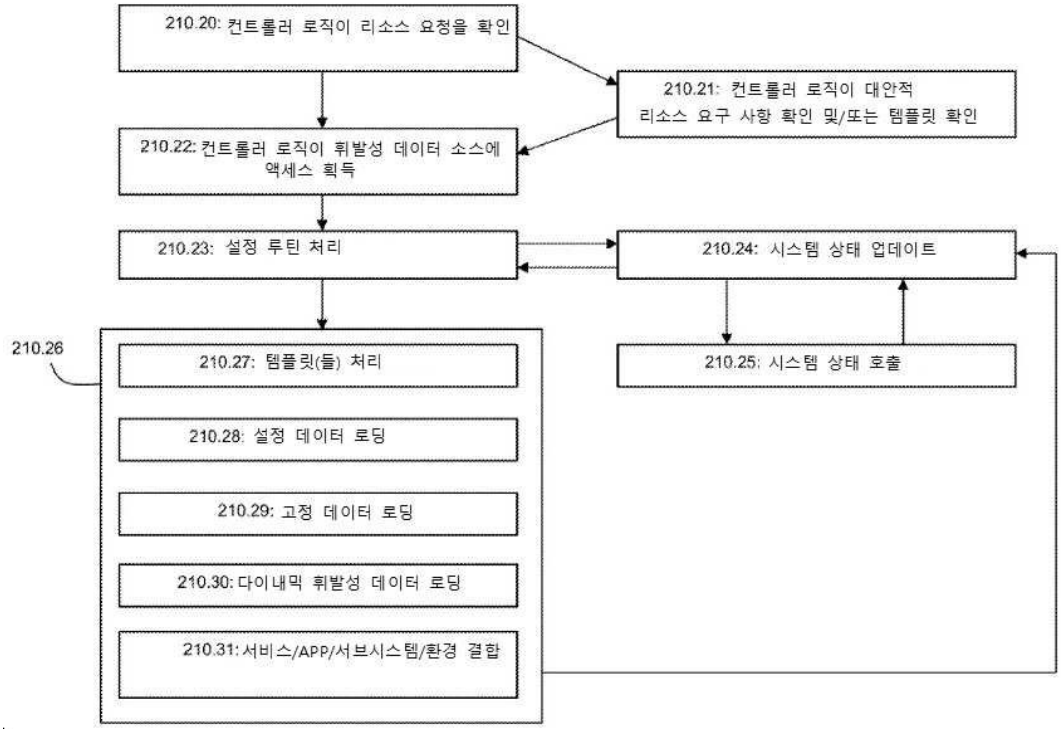
도면21


250
파일시스템
251
베이스 이미지
252
서비스 이미지
253
휘발성 이미지
254
241
240
부트 파일
커널

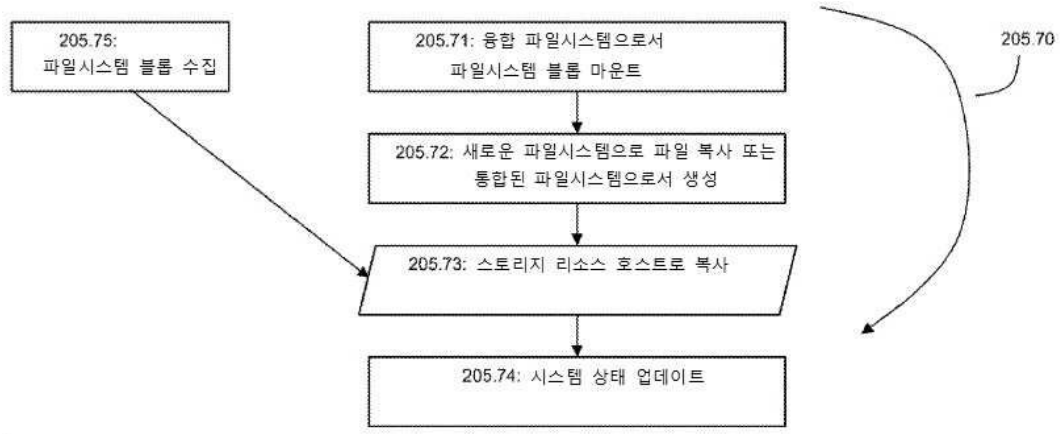
도면2m


데이터 (210.3)
휘발성 데이터에 대한 포인터 (210.4)
설정 파라미터 (210.5)
휘발성 데이터 (210.6)
요청된 템플릿 리스트 (210.7)
시스템 상태 포인터 (210.8)
서브시스템 규칙 (210.15)
시스템 규칙 실행 루틴 (210.16)
데이터 백업/대기 루틴 (210.17)
리소스 요청 리스트 (210.18)
대안적 리소스 옵션 (210.19)
생성을 위한 시스템 상태 질의 (210.1)
설정 루틴 (210.2 H)
템플릿 처리 순서 (210.9)
API 호출 (210.10)
요청된 서비스 (210.11)
배치 처리(종속성 포함) (210.12)
데이터 로딩 루틴 (210.13)
시스템 상태 질의 (210.14)
210

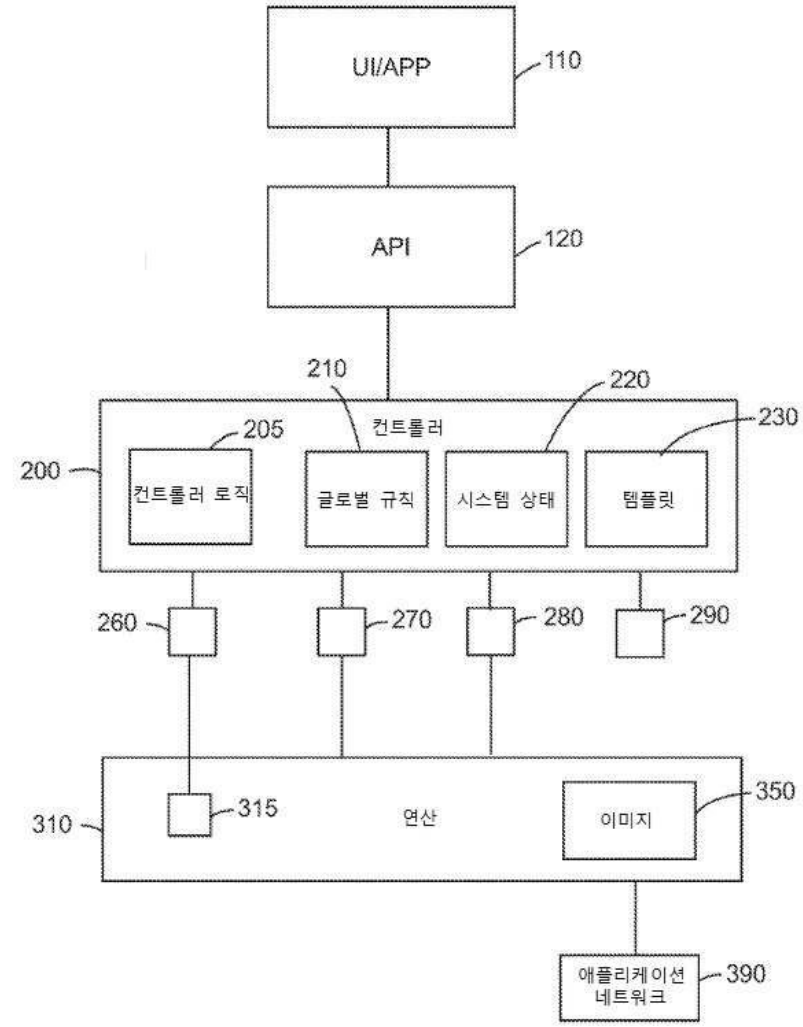
도면2n


210.20: 컨트롤러 로직이 리소스 요청을 확인
210.21: 컨트롤러 로직이 대안적 리소스 요구 사항 확인 및/또는 템플릿 확인
210.22: 컨트롤러 로직이 휘발성 데이터 소스에 액세스 획득
210.23: 설정 루틴 처리
210.24: 시스템 상태 업데이트
210.26
210.27: 템플릿(들) 처리
210.25: 시스템 상태 호출
210.28: 설정 데이터 로딩
210.29: 고정 데이터 로딩
210.30: 다이내믹 휘발성 데이터 로딩
210.31: 서비스/APP/서브시스템/환경 결합

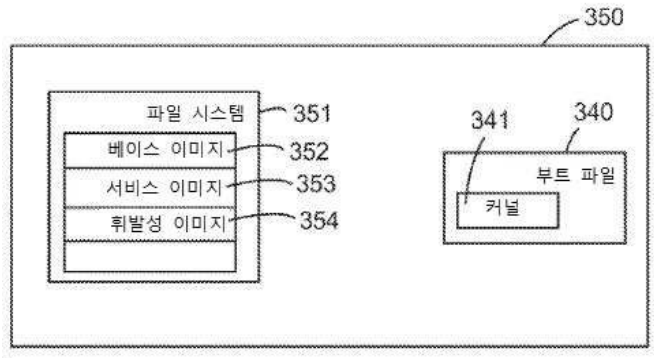
도면2o


205.75: 파일시스템 블롭 수집
205.71: 융합 파일시스템으로서 파일시스템 블롭 마운트
205.70
205.72: 새로운 파일시스템으로 파일 복사 또는 통합된 파일시스템으로서 생성
205.73: 스토리지 리소스 호스트로 복사
205.74: 시스템 상태 업데이트

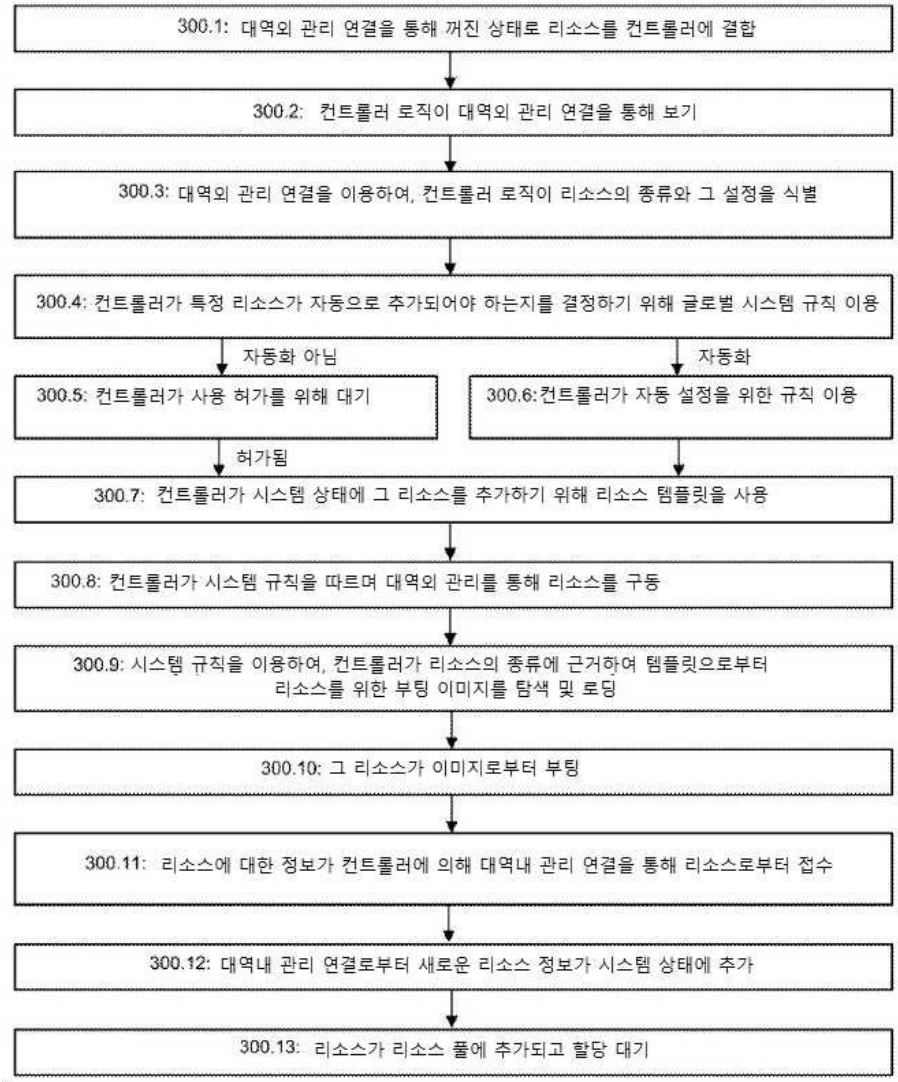
도면3a


UI/APP
110
API
120
210
220
230
205
컨트롤러
200
컨트롤러 로직
글로벌 규칙
시스템 상태
템플릿
260
270
280
290
310
315
연산
이미지
350
애플리케이션
네트워크
390

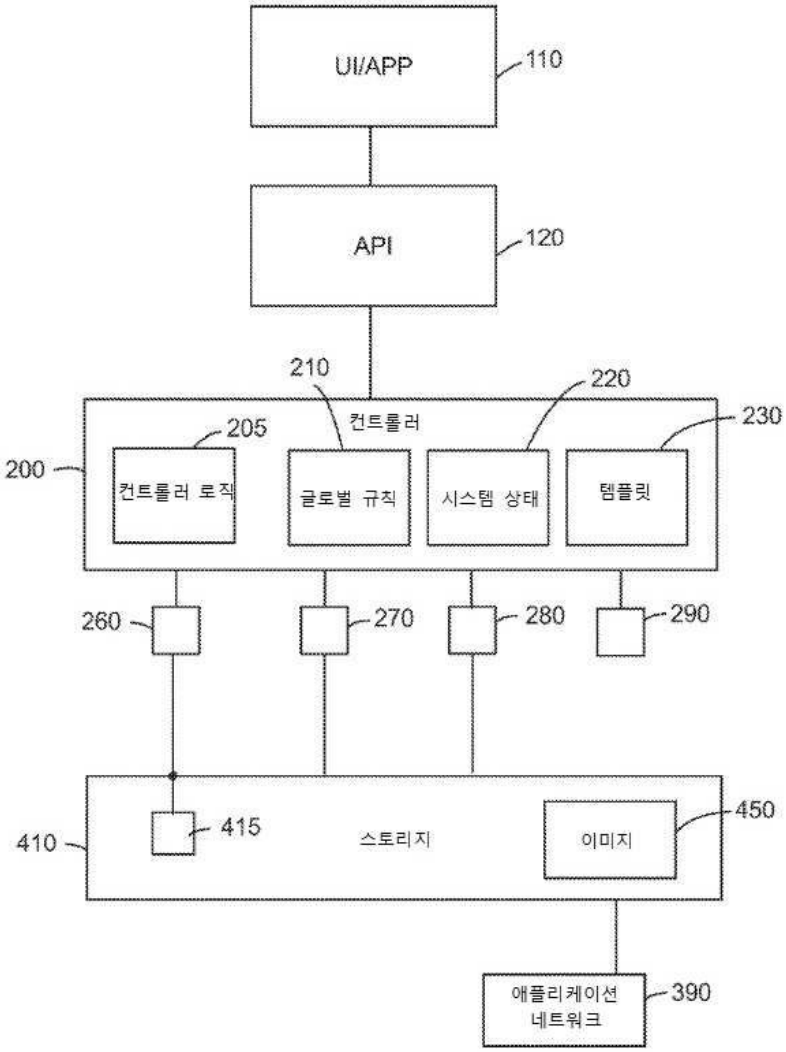
도면3b


350
파일 시스템
351
베이스 이미지
352
서비스 이미지
353
휘발성 이미지
354
341
340
부트 파일
커널

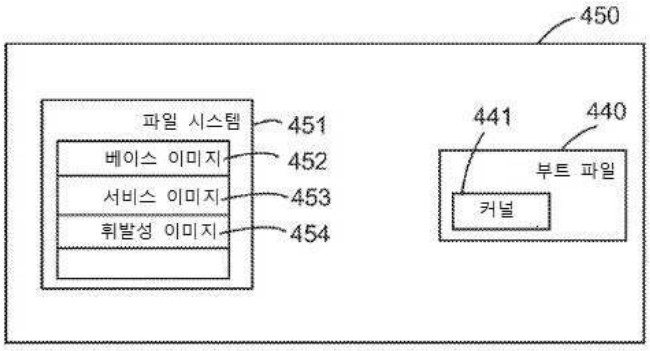
도면3c


300.1: 대역외 관리 연결을 통해 꺼진 상태로 리소스를 컨트롤러에 결합
300.2: 컨트롤러 로직이 대역외 관리 연결을 통해 보기
300.3: 대역외 관리 연결을 이용하여, 컨트롤러 로직이 리소스의 종류와 그 설정을 식별
300.4: 컨트롤러가 특정 리소스가 자동으로 추가되어야 하는지를 결정하기 위해 글로벌 시스템 규칙 이용
자동화 아님
자동화
300.5: 컨트롤러가 사용 허가를 위해 대기
300.6:컨트롤러가 자동 설정을 위한 규칙 이용
허가됨
300.7: 컨트롤러가 시스템 상태에 그 리소스를 추가하기 위해 리소스 템플릿을 사용
300.8: 컨트롤러가 시스템 규칙을 따르며 대역외 관리를 통해 리소스를 구동
300.9: 시스템 규칙을 이용하여, 컨트롤러가 리소스의 종류에 근거하여 템플릿으로부터
리소스를 위한 부팅 이미지를 탐색 및 로딩
300.10: 그 리소스가 이미지로부터 부팅
300.11: 리소스에 대한 정보가 컨트롤러에 의해 대역내 관리 연결을 통해 리소스로부터 접수
300.12: 대역내 관리 연결로부터 새로운 리소스 정보가 시스템 상태에 추가
300.13: 리소스가 리소스 풀에 추가되고 할당 대기

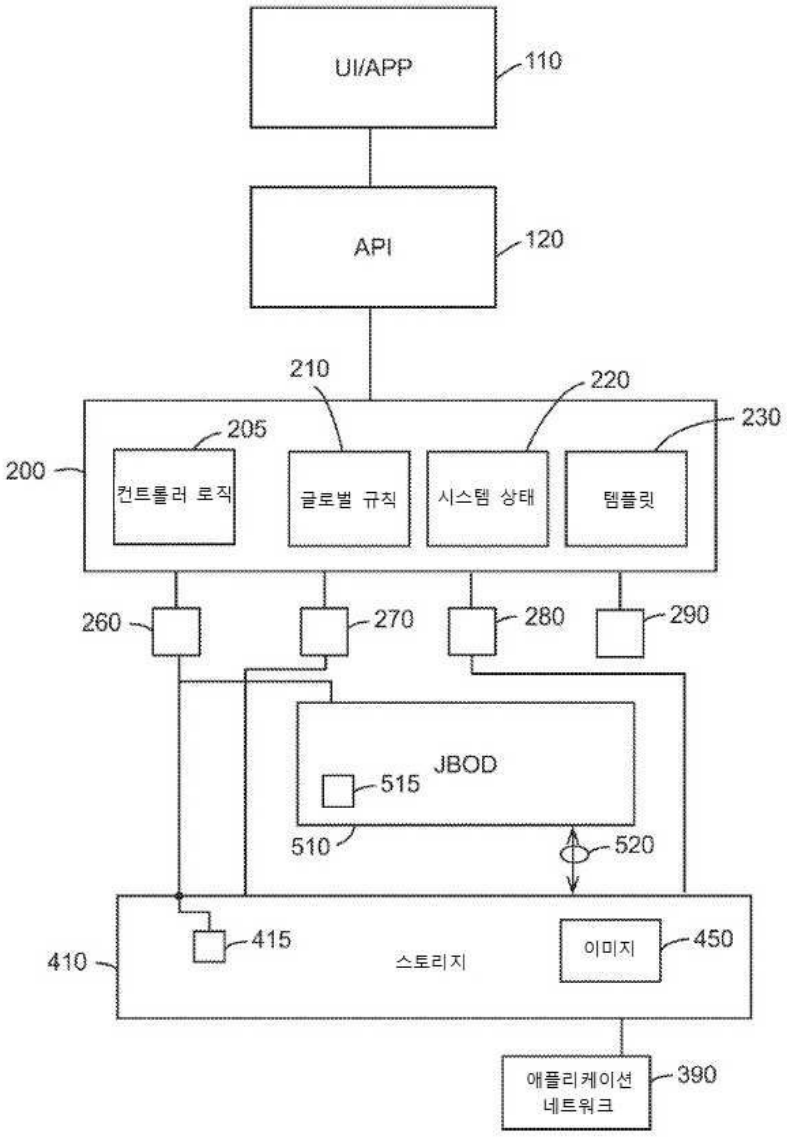
도면4a


UI/APP
110
API
120
210
220
230
205
컨트롤러
200
컨트롤러 로직
글로벌 규칙
시스템 상태
템플릿
260
270
280
290
410
415
스토리지
450
이미지
애플리케이션
네트워크
390

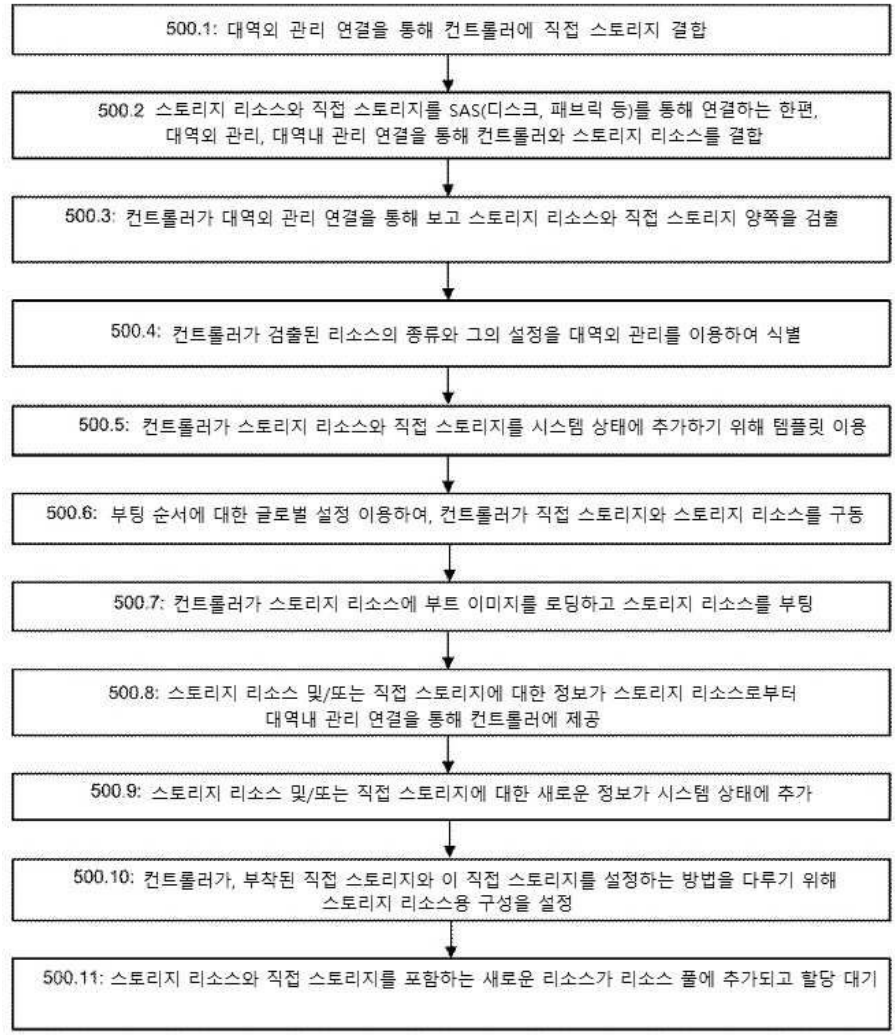
도면4b


450
파일 시스템
451
베이스 이미지
452
서비스 이미지
453
휘발성 이미지
454
441
440
부트 파일
커널

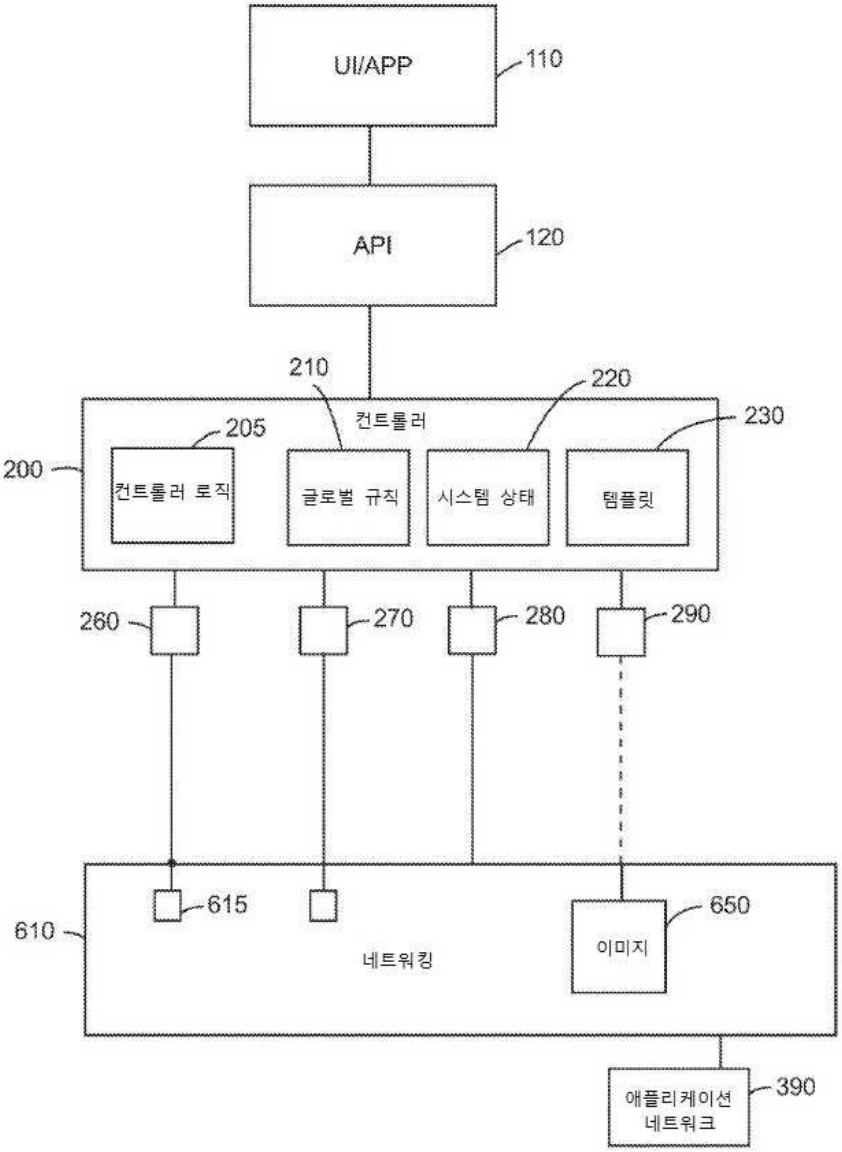
도면5a


UI/APP
110
API
120
210
220
230
205
200
컨트롤러 로직
글로벌 규칙
시스템 상태
템플릿
260
270
280
290
JBOD
515
510
520
410
415
스토리지
이미지
450
애플리케이션
네트워크
390

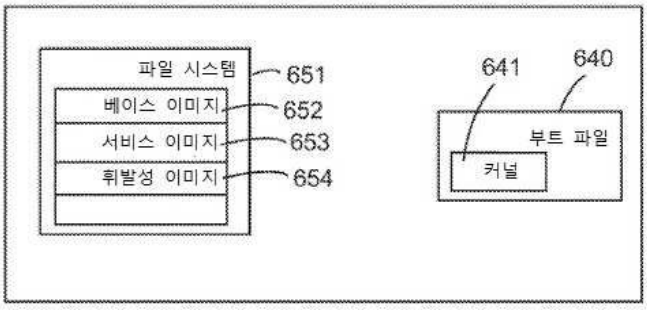
도면5b


500.1: 대역외 관리 연결을 통해 컨트롤러에 직접 스토리지 결합
500.2 스토리지 리소스와 직접 스토리지를 SAS(디스크, 패브릭 등)를 통해 연결하는 한편, 대역외 관리, 대역내 관리 연결을 통해 컨트롤러와 스토리지 리소스를 결합
500.3: 컨트롤러가 대역외 관리 연결을 통해 보고 스토리지 리소스와 직접 스토리지 양쪽을 검출
500.4: 컨트롤러가 검출된 리소스의 종류와 그의 설정을 대역외 관리를 이용하여 식별
500.5: 컨트롤러가 스토리지 리소스와 직접 스토리지를 시스템 상태에 추가하기 위해 템플릿 이용
500.6: 부팅 순서에 대한 글로벌 설정 이용하여, 컨트롤러가 직접 스토리지와 스토리지 리소스를 구동
500.7: 컨트롤러가 스토리지 리소스에 부트 이미지를 로딩하고 스토리지 리소스를 부팅
500.8: 스토리지 리소스 및/또는 직접 스토리지에 대한 정보가 스토리지 리소스로부터 대역내 관리 연결을 통해 컨트롤러에 제공
500.9: 스토리지 리소스 및/또는 직접 스토리지에 대한 새로운 정보가 시스템 상태에 추가
500.10: 컨트롤러가, 부착된 직접 스토리지와 이 직접 스토리지를 설정하는 방법을 다루기 위해 스토리지 리소스용 구성을 설정
500.11: 스토리지 리소스와 직접 스토리지를 포함하는 새로운 리소스가 리소스 풀에 추가되고 할당 대기

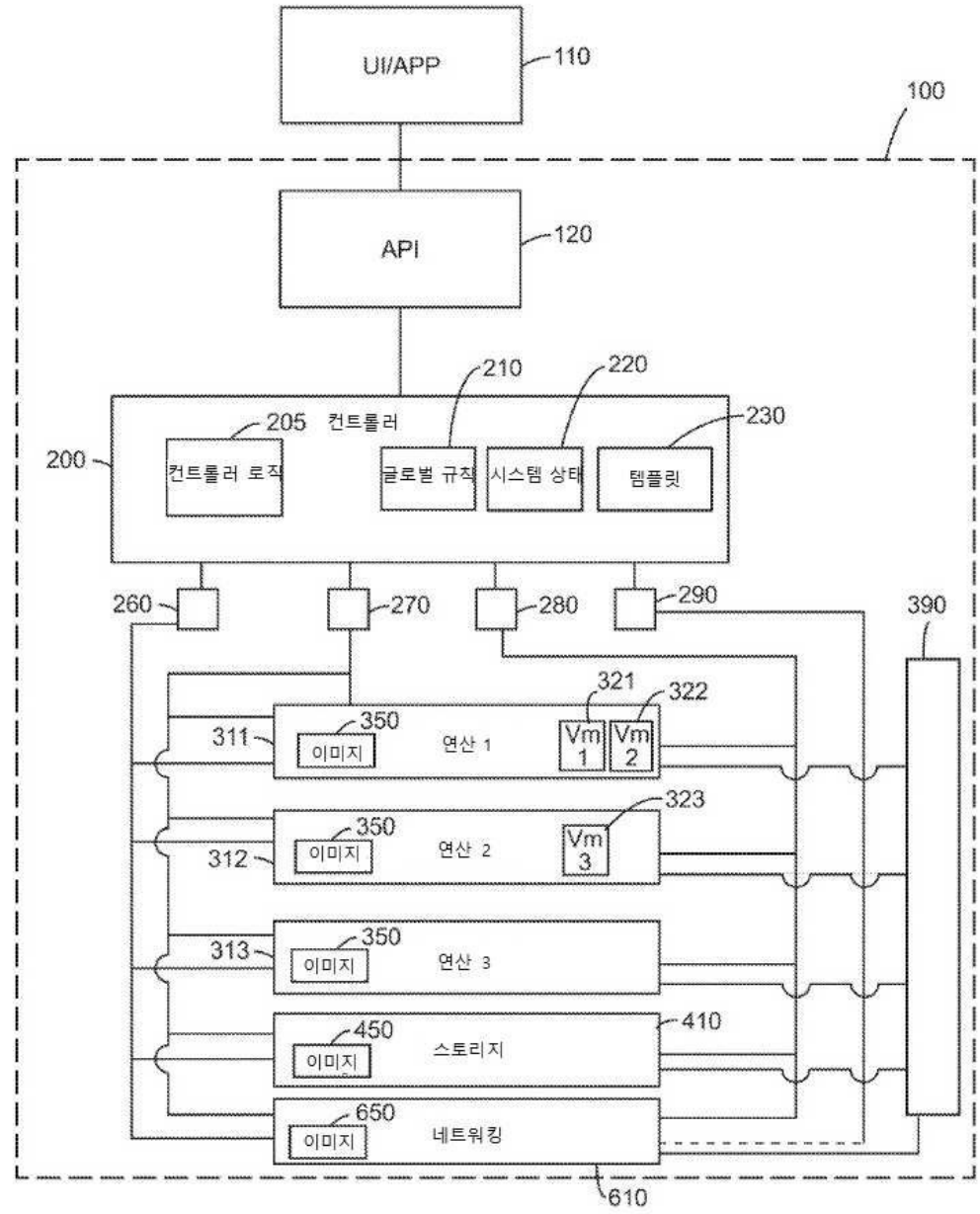
도면6a


UI/APP
110
API
120
210
220
230
205
200
컨트롤러
컨트롤러 로직
글로벌 규칙
시스템 상태
템플릿
260
270
280
290
610
615
650
이미지
네트워킹
390
애플리케이션
네트워크

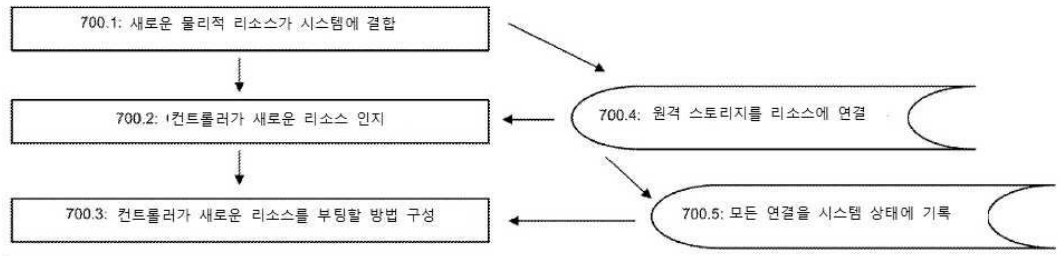
도면6b


파일 시스템
651
베이스 이미지
652
서비스 이미지
653
휘발성 이미지
654
641
640
부트 파일
커널

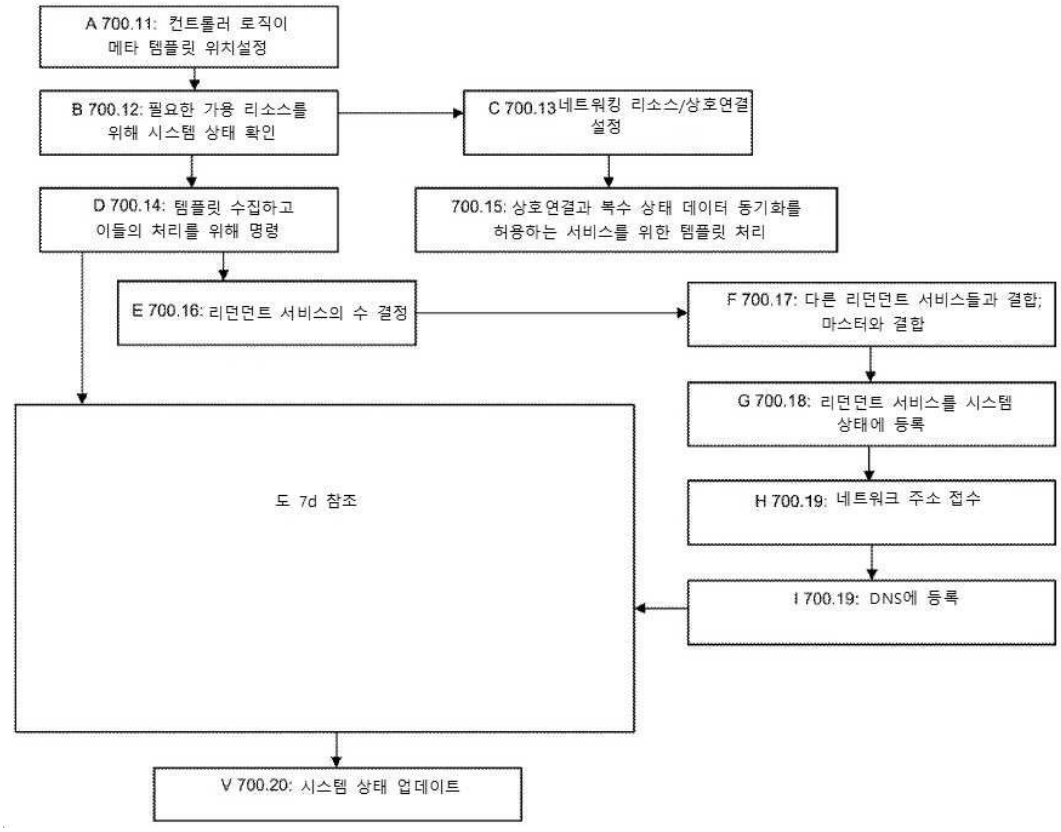
도면7a


UI/APP
110
100
API
120
210
220
205
컨트롤러
230
200
컨트롤러 로직
글로벌 규칙
시스템 상태
템플릿
260
270
280
290
390
321
322
311
350
이미지
연산 1
Vm 1
Vm 2
323
312
연산 2
Vm 3
313
연산 3
410
450
스토리지
650
네트워킹
610

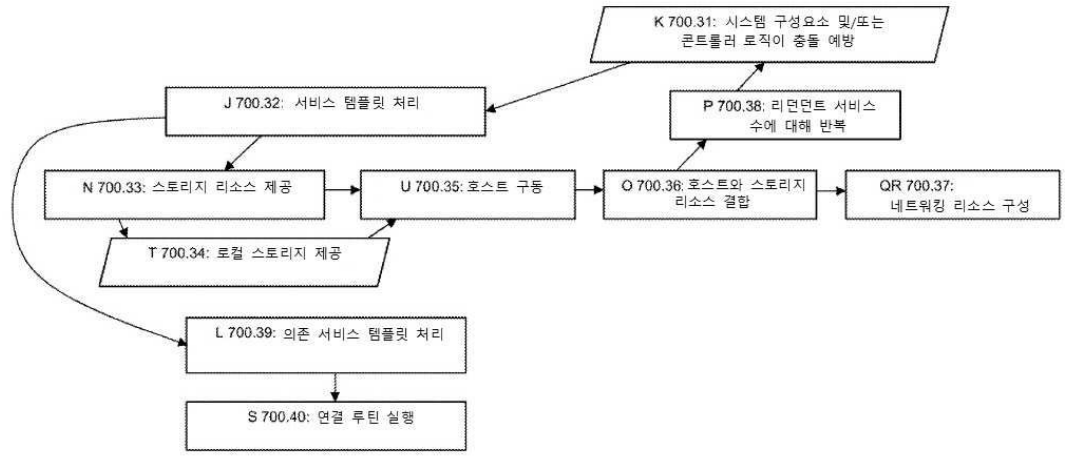
도면7b


700.1: 새로운 물리적 리소스가 시스템에 결합
700.2: 컨트롤러가 새로운 리소스 인지
700.3: 컨트롤러가 새로운 리소스를 부팅할 방법 구성
700.4: 원격 스토리지를 리소스에 연결
700.5: 모든 연결을 시스템 상태에 기록

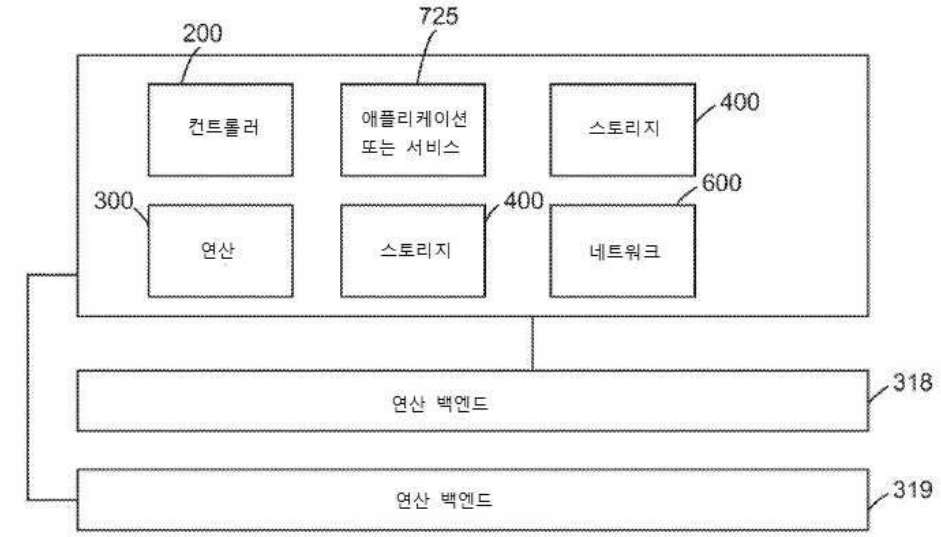
도면7c


A 700.11: 컨트롤러 로직이 메타 템플릿 위치설정
B 700.12: 필요한 가용 리소스를 위해 시스템 상태 확인
C 700.13 네트워킹 리소스/상호연결 설정
D 700.14: 템플릿 수집하고 이들의 처리를 위해 명령
700.15: 상호연결과 복수 상태 데이터 동기화를 허용하는 서비스를 위한 템플릿 처리
E 700.16: 리던던트 서비스의 수 결정
F 700.17: 다른 리던던트 서비스들과 결합; 마스터와 결합
G 700.18: 리던던트 서비스를 시스템 상태에 등록
도 7d 참조
H 700.19: 네트워크 주소 접수
I 700.19: DNS에 등록
V 700.20: 시스템 상태 업데이트

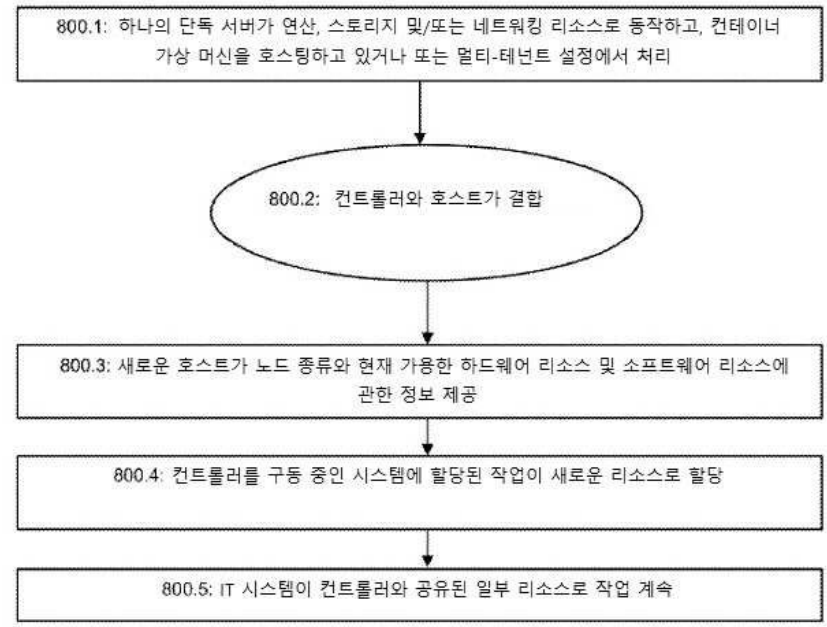
도면7d


K 700.31: 시스템 구성요소 및/또는 콘트롤러 로직이 충돌 예방
J 700.32: 서비스 템플릿 처리
P 700.38: 리던던트 서비스 수에 대해 반복
N 700.33: 스토리지 리소스 제공
U 700.35: 호스트 구동
O 700.36: 호스트와 스토리지 리소스 결합
QR 700.37: 네트워킹 리소스 구성
T 700.34: 로컬 스토리지 제공
L 700.39: 의존 서비스 템플릿 처리
S 700.40: 연결 루틴 실행

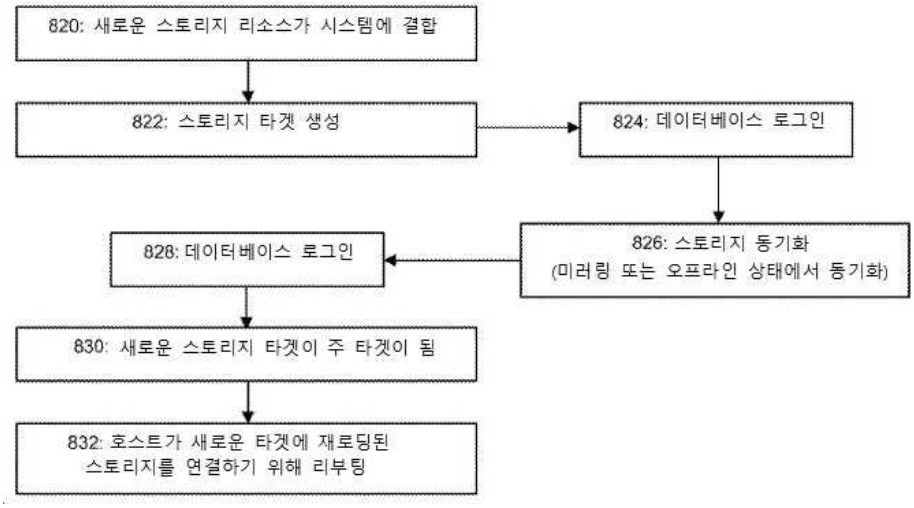
도면8a


200
725
컨트롤러
애플리케이션
또는 서비스
스토리지
400
300
연산
스토리지
400
600
네트워크
연산 백엔드
318
연산 백엔드
319

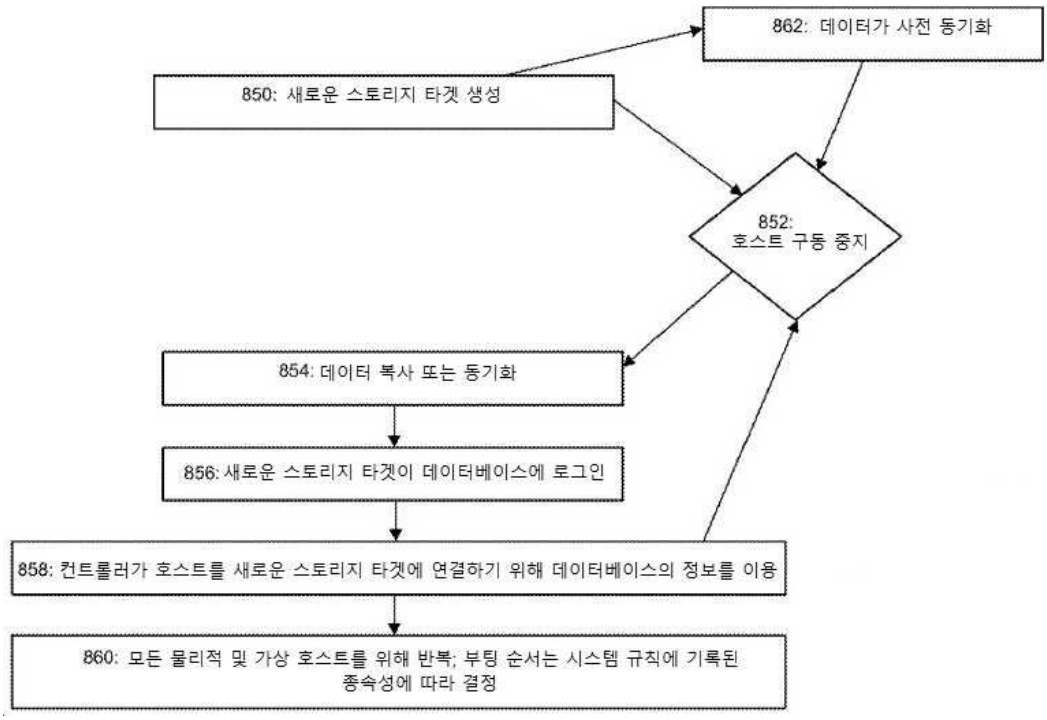
도면8b


800.1: 하나의 단독 서버가 연산, 스토리지 및/또는 네트워킹 리소스로 동작하고, 컨테이너 가상 머신을 호스팅하고 있거나 또는 멀티-테넌트 설정에서 처리
800.2: 컨트롤러와 호스트가 결합
800.3: 새로운 호스트가 노드 종류와 현재 가용한 하드웨어 리소스 및 소프트웨어 리소스에 관한 정보 제공
800.4: 컨트롤러를 구동 중인 시스템에 할당된 작업이 새로운 리소스로 할당
800.5: IT 시스템이 컨트롤러와 공유된 일부 리소스로 작업 계속

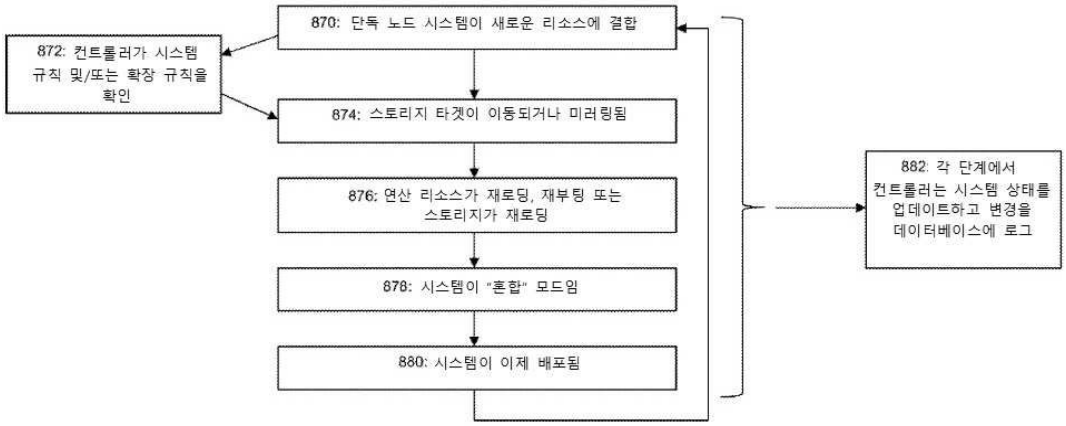
도면8c


820: 새로운 스토리지 리소스가 시스템에 결합
822: 스토리지 타겟 생성
824: 데이터베이스 로그인
826: 스토리지 동기화
(미러링 또는 오프라인 상태에서 동기화)
828: 데이터베이스 로그인
830: 새로운 스토리지 타겟이 주 타겟이 됨
832: 호스트가 새로운 타겟에 재로딩된
스토리지를 연결하기 위해 리부팅

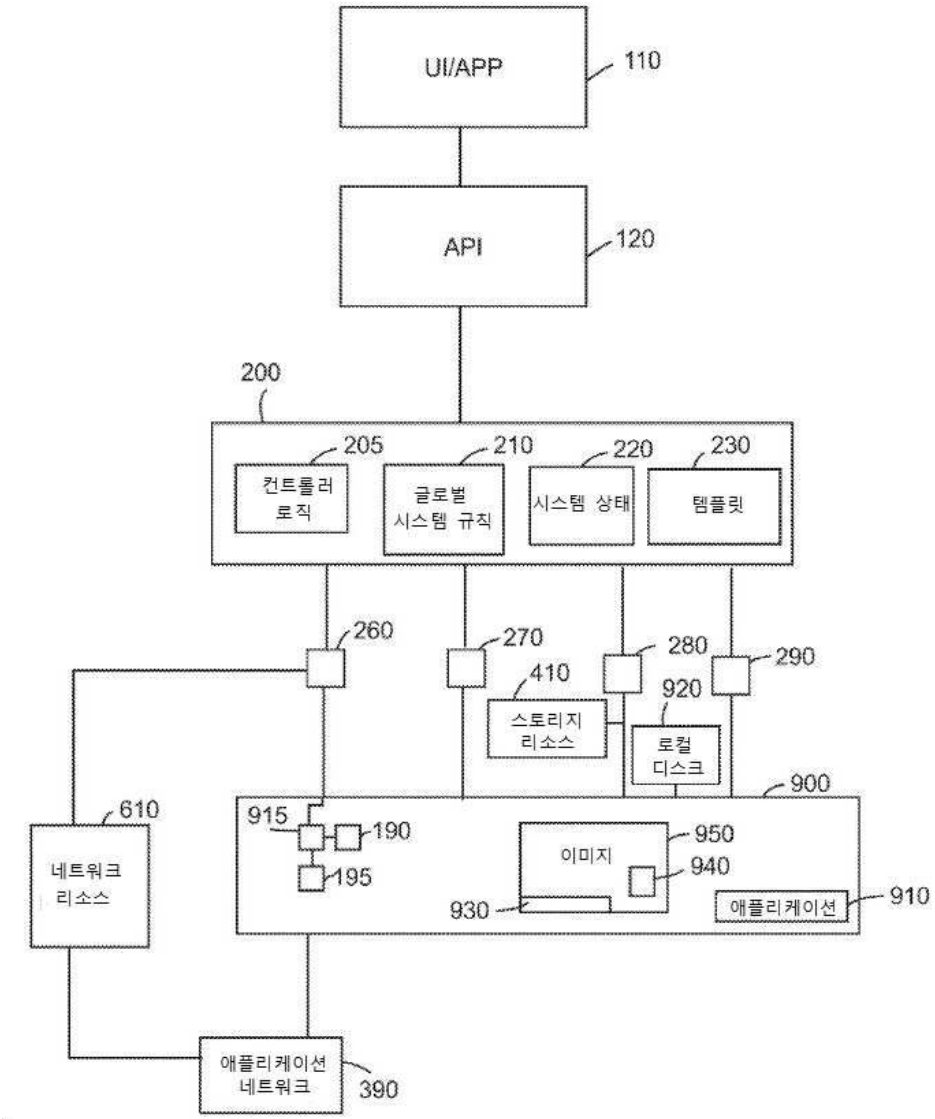
도면8d


862: 데이터가 사전 동기화
850: 새로운 스토리지 타겟 생성
852:
호스트 구동 중지
854: 데이터 복사 또는 동기화
856: 새로운 스토리지 타겟이 데이터베이스에 로그인
858: 컨트롤러가 호스트를 새로운 스토리지 타겟에 연결하기 위해 데이터베이스의 정보를 이용
860: 모든 물리적 및 가상 호스트를 위해 반복; 부팅 순서는 시스템 규칙에 기록된
종속성에 따라 결정

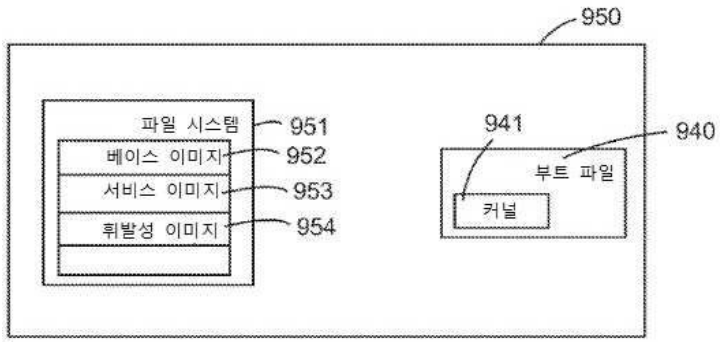
도면8e


870: 단독 노드 시스템이 새로운 리소스에 결합
872: 컨트롤러가 시스템 규칙 및/또는 확장 규칙을 확인
874: 스토리지 타겟이 이동되거나 미러링됨
876: 연산 리소스가 재로딩, 재부팅 또는 스토리지가 재로딩
878: 시스템이 "혼합" 모드임
880: 시스템이 이제 배포됨
882: 각 단계에서 컨트롤러는 시스템 상태를 업데이트하고 변경을 데이터베이스에 로그

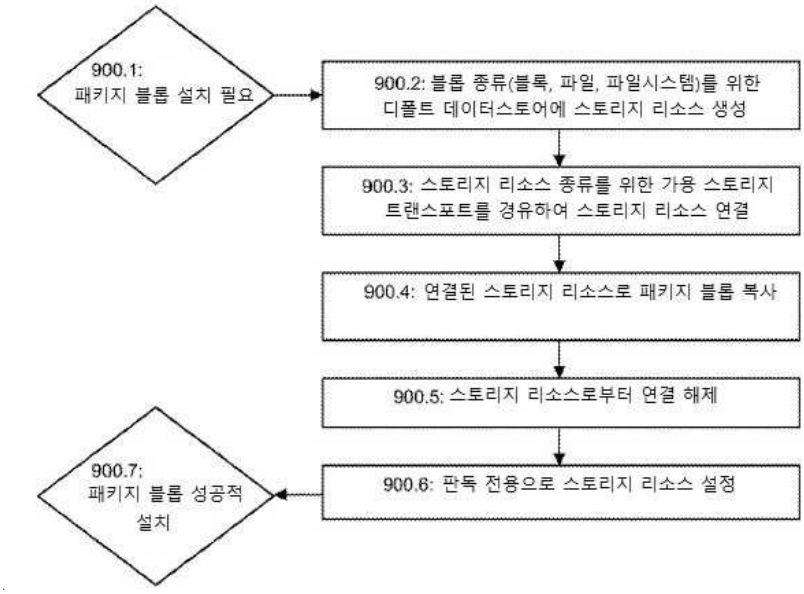
도면9a


UI/APP
110
API
120
200
205
컨트롤러 로직
210
글로벌 시스템 규칙
220
시스템 상태
230
템플릿
260
270
410
스토리지 리소스
280
920
로컬 디스크
290
900
610
네트워크 리소스
915
190
195
이미지
950
940
930
애플리케이션
910
애플리케이션 네트워크
390

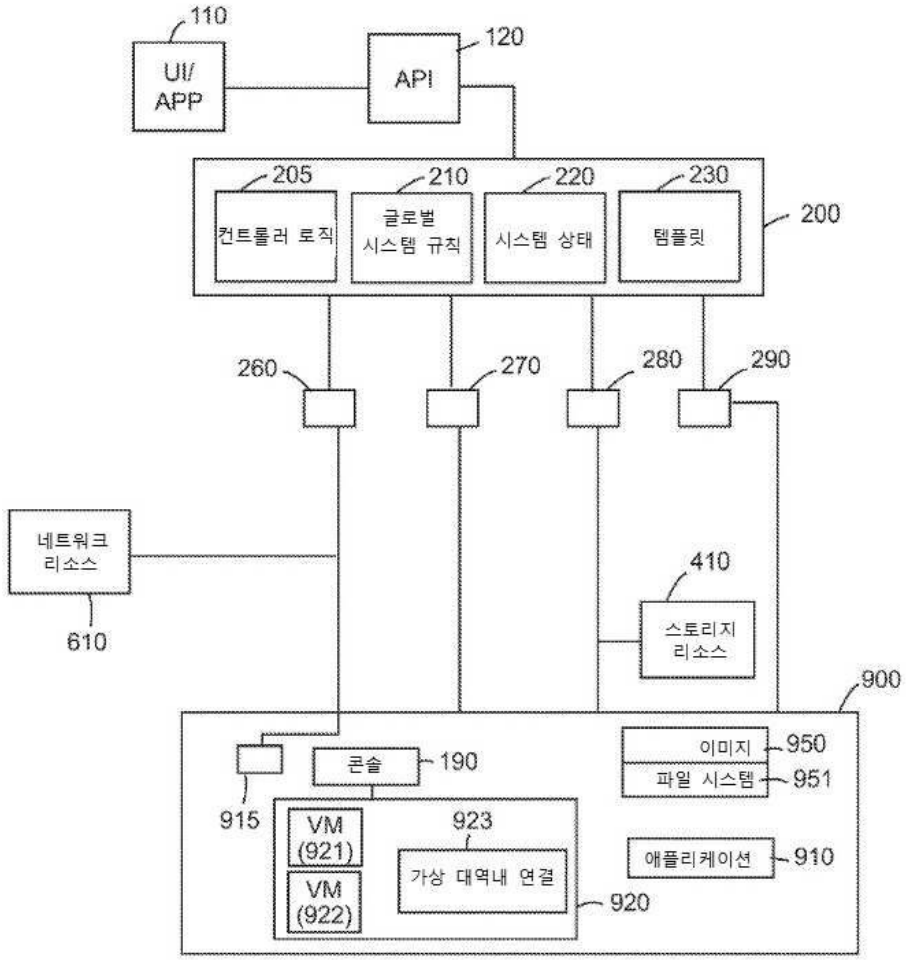
도면9b


950
파일 시스템
951
베이스 이미지
952
서비스 이미지
953
휘발성 이미지
954
941
940
부트 파일
커널

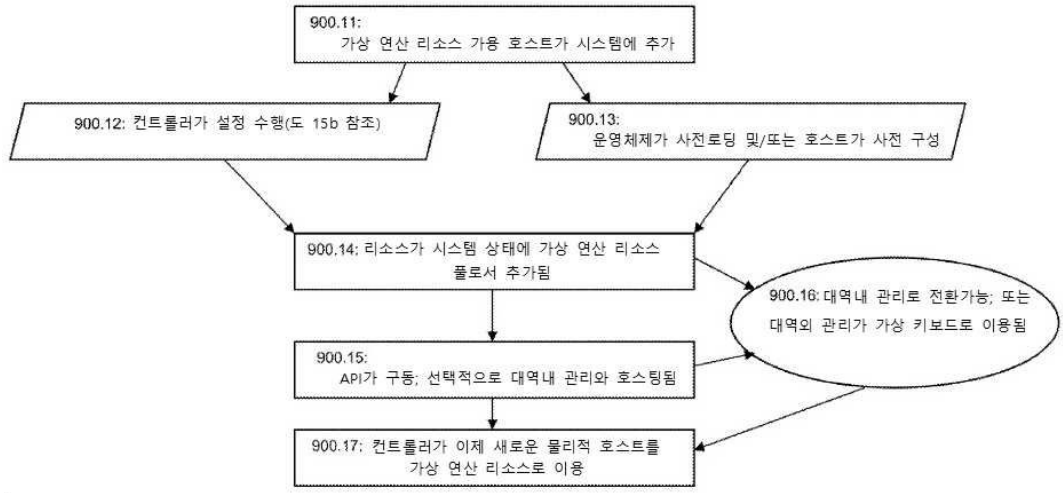
도면9c


900.1:
패키지 블롭 설치 필요
900.2: 블롭 종류(블록, 파일, 파일시스템)를 위한 디폴트 데이터스토어에 스토리지 리소스 생성
900.3: 스토리지 리소스 종류를 위한 가용 스토리지 트랜스포트를 경유하여 스토리지 리소스 연결
900.4: 연결된 스토리지 리소스로 패키지 블롭 복사
900.5: 스토리지 리소스로부터 연결 해제
900.6: 판독 전용으로 스토리지 리소스 설정
900.7:
패키지 블롭 성공적 설치

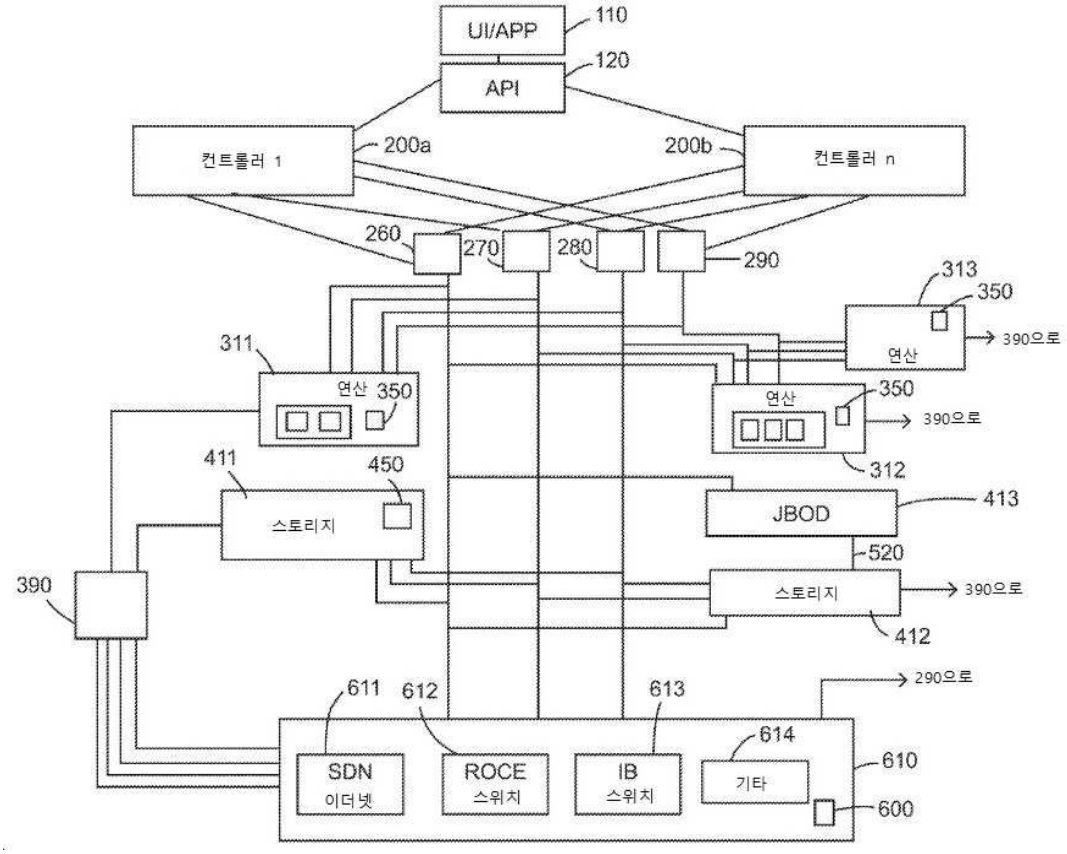
도면9d


110
UI/
APP
120
API
205
컨트롤러 로직
210
글로벌
시스템 규칙
220
시스템 상태
230
템플릿
200
260
270
280
290
네트워크
리소스
610
410
스토리지
리소스
900
915
콘솔
190
VM
(921)
VM
(922)
923
가상 대역내 연결
920
이미지
950
파일 시스템
951
애플리케이션
910

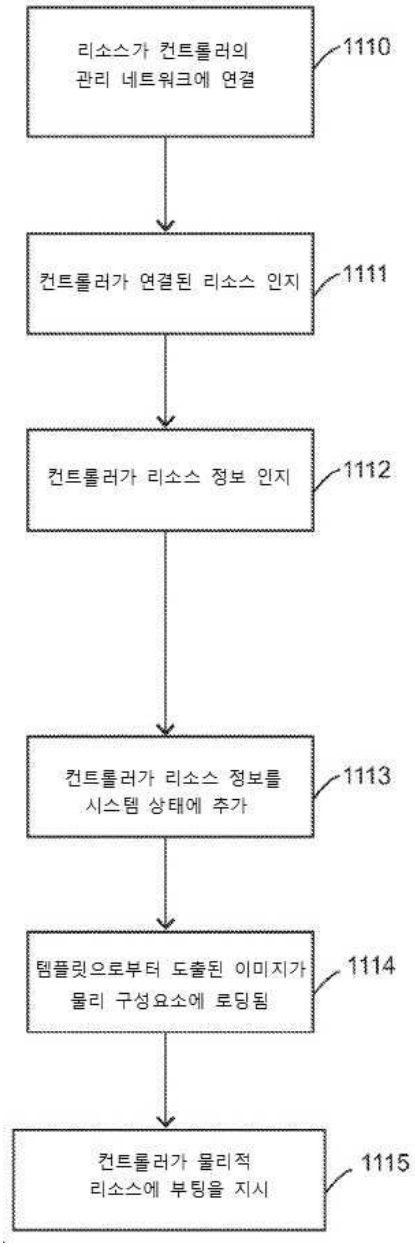
도면9e


900.11:
가상 연산 리소스 가용 호스트가 시스템에 추가
900.12: 컨트롤러가 설정 수행(도 15b 참조)
900.13:
운영체제가 사전로딩 및/또는 호스트가 사전 구성
900.14: 리소스가 시스템 상태에 가상 연산 리소스
풀로서 추가됨
900.16: 대역내 관리로 전환가능; 또는
대역외 관리가 가상 키보드로 이용됨
900.15:
API가 구동; 선택적으로 대역내 관리와 호스팅됨
900.17: 컨트롤러가 이제 새로운 물리적 호스트를
가상 연산 리소스로 이용

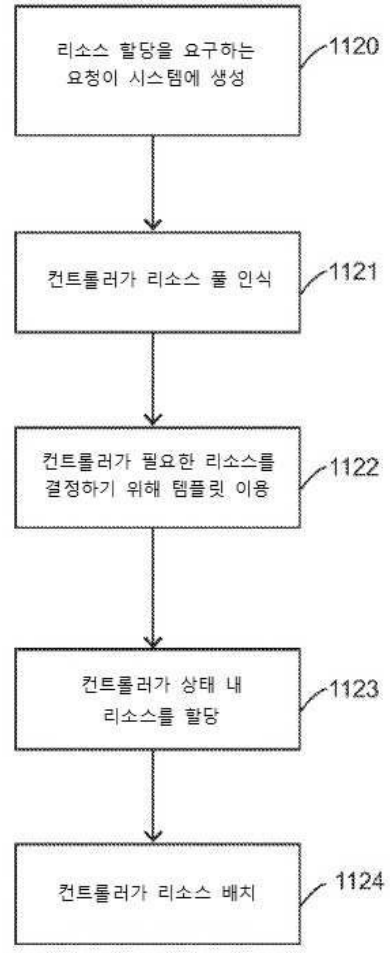
도면10


UI/APP
110
API
120
컨트롤러 1
200a
200b
컨트롤러 n
260
270
280
290
313
350
연산
390으로
311
연산
350
350
연산
390으로
411
450
312
스토리지
JBOD
413
520
390
스토리지
390으로
412
290으로
611
612
613
614
610
SDN
이더넷
ROCE
스위치
IB
스위치
기타
600

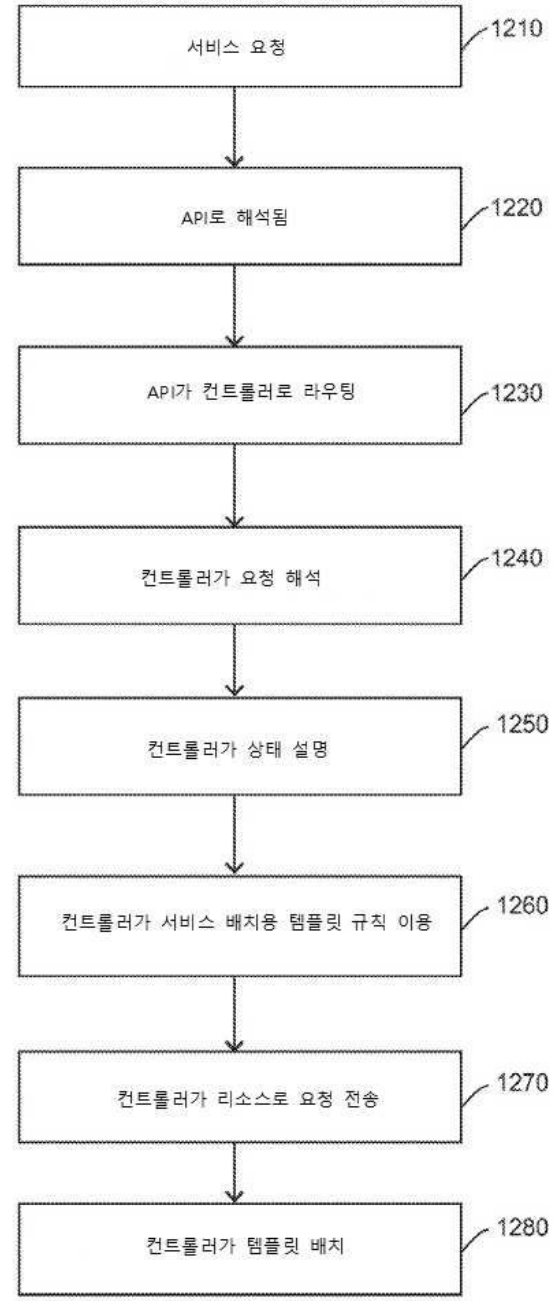
도면11a


리소스가 컨트롤러의 관리 네트워크에 연결
1110
컨트롤러가 연결된 리소스 인지
1111
컨트롤러가 리소스 정보 인지
1112
컨트롤러가 리소스 정보를 시스템 상태에 추가
1113
템플릿으로부터 도출된 이미지가 물리 구성요소에 로딩됨
1114
컨트롤러가 물리적 리소스에 부팅을 지시
1115

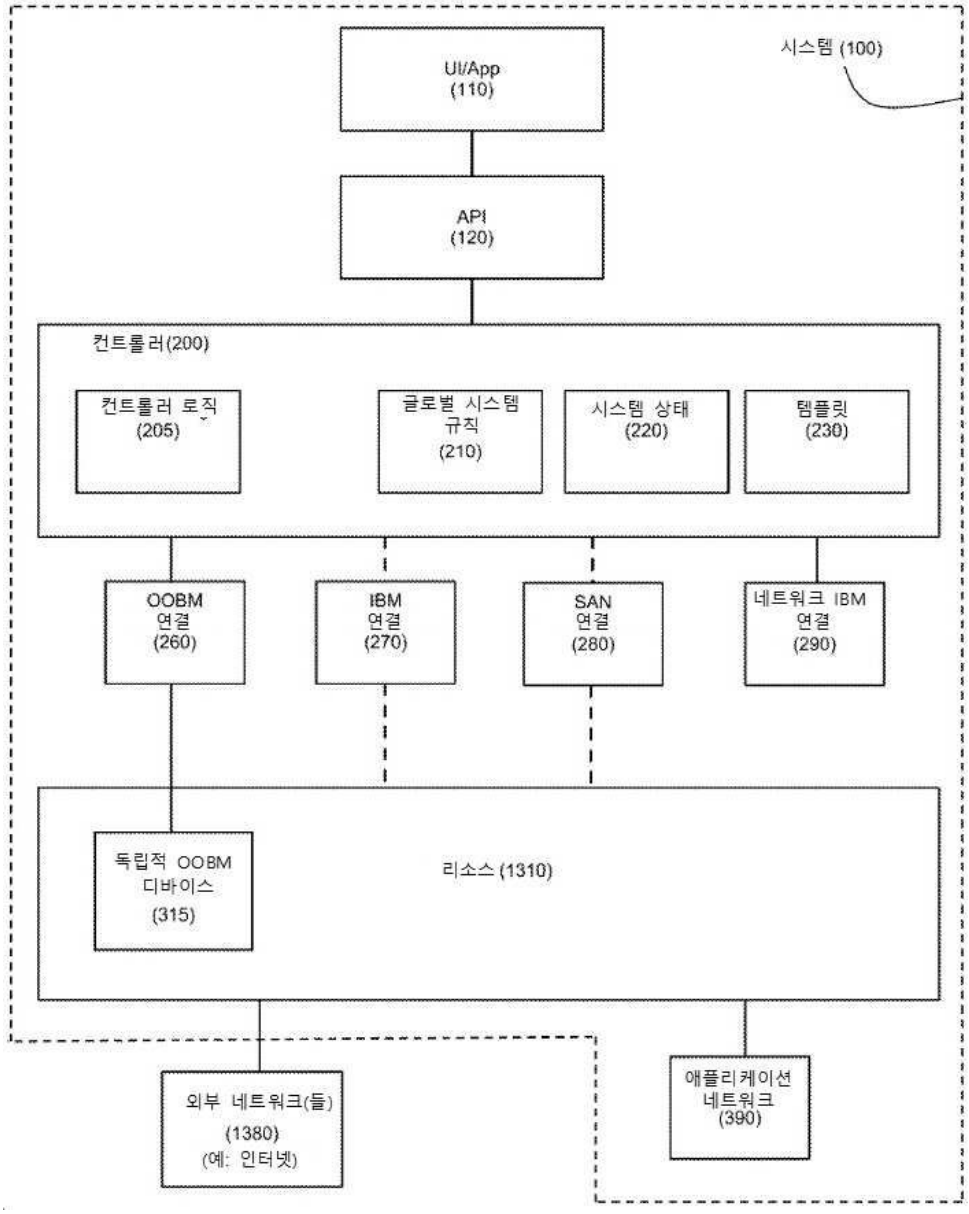
도면11b


리소스 할당을 요구하는 요청이 시스템에 생성
1120
컨트롤러가 리소스 풀 인식
1121
컨트롤러가 필요한 리소스를 결정하기 위해 템플릿 이용
1122
컨트롤러가 상태 내 리소스를 할당
1123
컨트롤러가 리소스 배치
1124

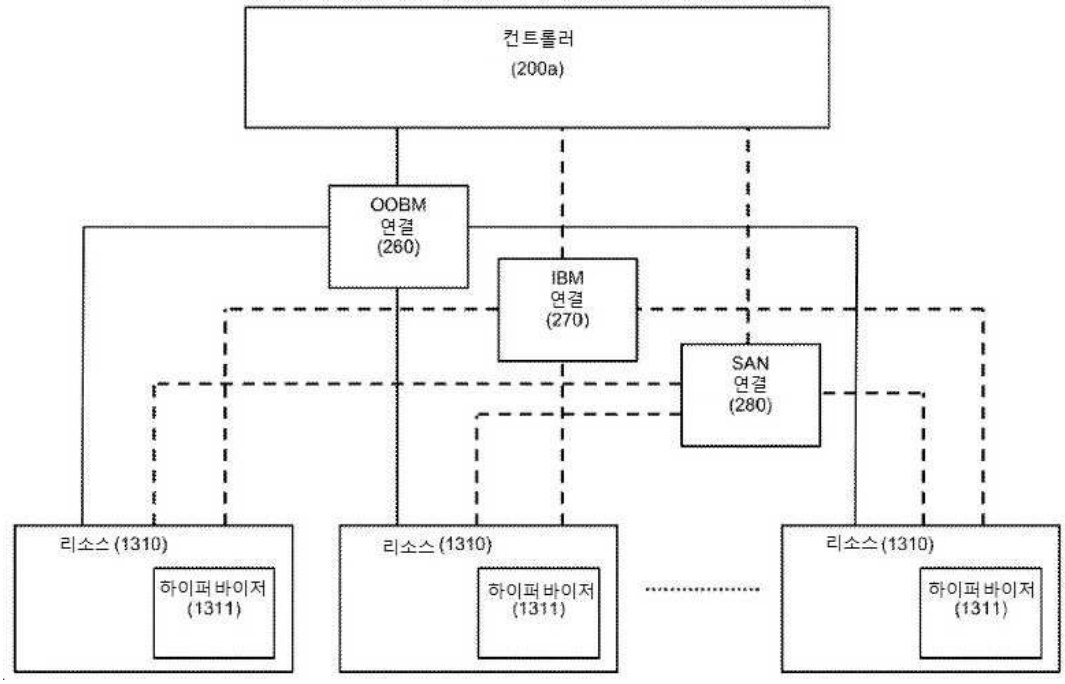
도면12


서비스 요청
1210
API로 해석됨
1220
API가 컨트롤러로 라우팅
1230
컨트롤러가 요청 해석
1240
컨트롤러가 상태 설명
1250
컨트롤러가 서비스 배치용 템플릿 규칙 이용
1260
컨트롤러가 리소스로 요청 전송
1270
컨트롤러가 템플릿 배치
1280

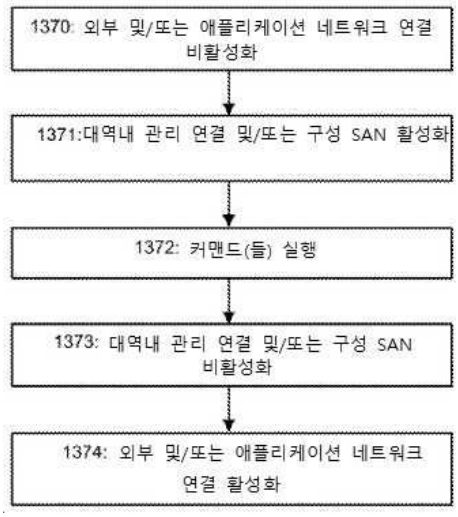
도면13a


시스템 (100)
UI/App
(110)
API
(120)
컨트롤러(200)
컨트롤러 로직
(205)
글로벌 시스템
규칙
(210)
시스템 상태
(220)
템플릿
(230)
OOBM
연결
(260)
IBM
연결
(270)
SAN
연결
(280)
네트워크 IBM
연결
(290)
독립적 OOBM
디바이스
(315)
리소스 (1310)
외부 네트워크(들)
(1380)
(예: 인터넷)
애플리케이션
네트워크
(390)

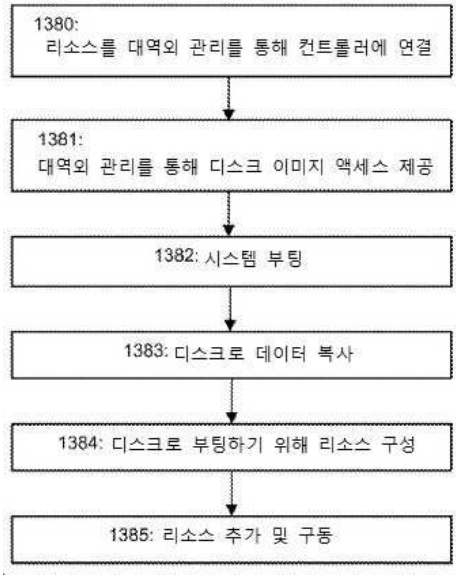
도면13b


컨트롤러
(200a)
OOBM
연결
(260)
IBM
연결
(270)
SAN
연결
(280)
리소스 (1310)
하이퍼바이저
(1311)
리소스 (1310)
하이퍼바이저
(1311)
리소스 (1310)
하이퍼바이저
(1311)

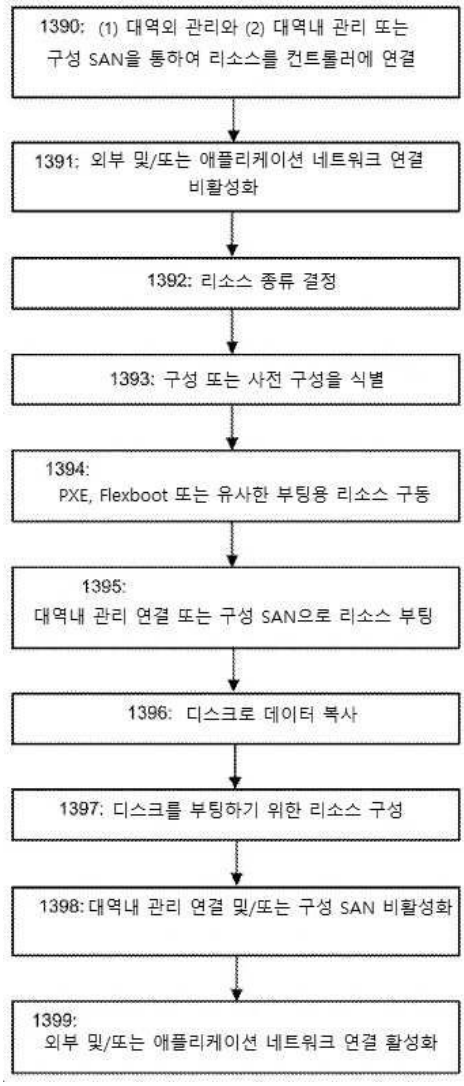
도면13c


1370: 외부 및/또는 애플리케이션 네트워크 연결 비활성화
1371:대역내 관리 연결 및/또는 구성 SAN 활성화
1372: 커맨드(들) 실행
1373: 대역내 관리 연결 및/또는 구성 SAN 비활성화
1374: 외부 및/또는 애플리케이션 네트워크 연결 활성화

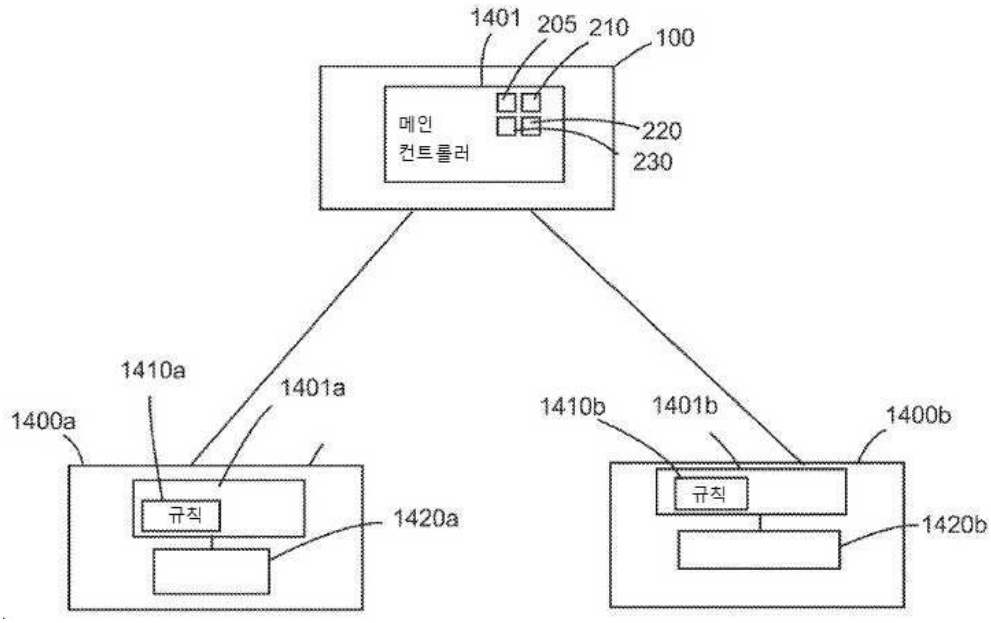
도면13d


1380:
리소스를 대역외 관리를 통해 컨트롤러에 연결
1381:
대역외 관리를 통해 디스크 이미지 액세스 제공
1382: 시스템 부팅
1383: 디스크로 데이터 복사
1384: 디스크로 부팅하기 위해 리소스 구성
1385: 리소스 추가 및 구동

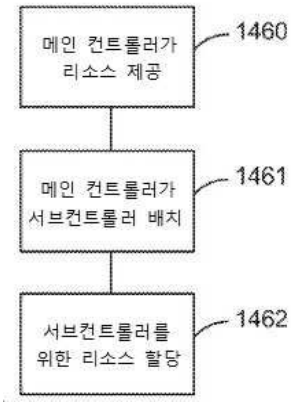
도면13e


1390: (1) 대역외 관리와 (2) 대역내 관리 또는
구성 SAN을 통하여 리소스를 컨트롤러에 연결
1391: 외부 및/또는 애플리케이션 네트워크 연결
비활성화
1392: 리소스 종류 결정
1393: 구성 또는 사전 구성을 식별
1394:
PXE, Flexboot 또는 유사한 부팅용 리소스 구동
1395:
대역내 관리 연결 또는 구성 SAN으로 리소스 부팅
1396: 디스크로 데이터 복사
1397: 디스크를 부팅하기 위한 리소스 구성
1398: 대역내 관리 연결 및/또는 구성 SAN 비활성화
1399:
외부 및/또는 애플리케이션 네트워크 연결 활성화

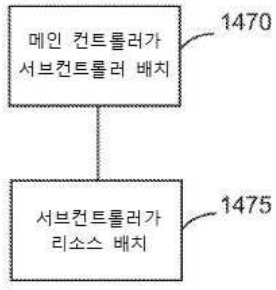
도면14a



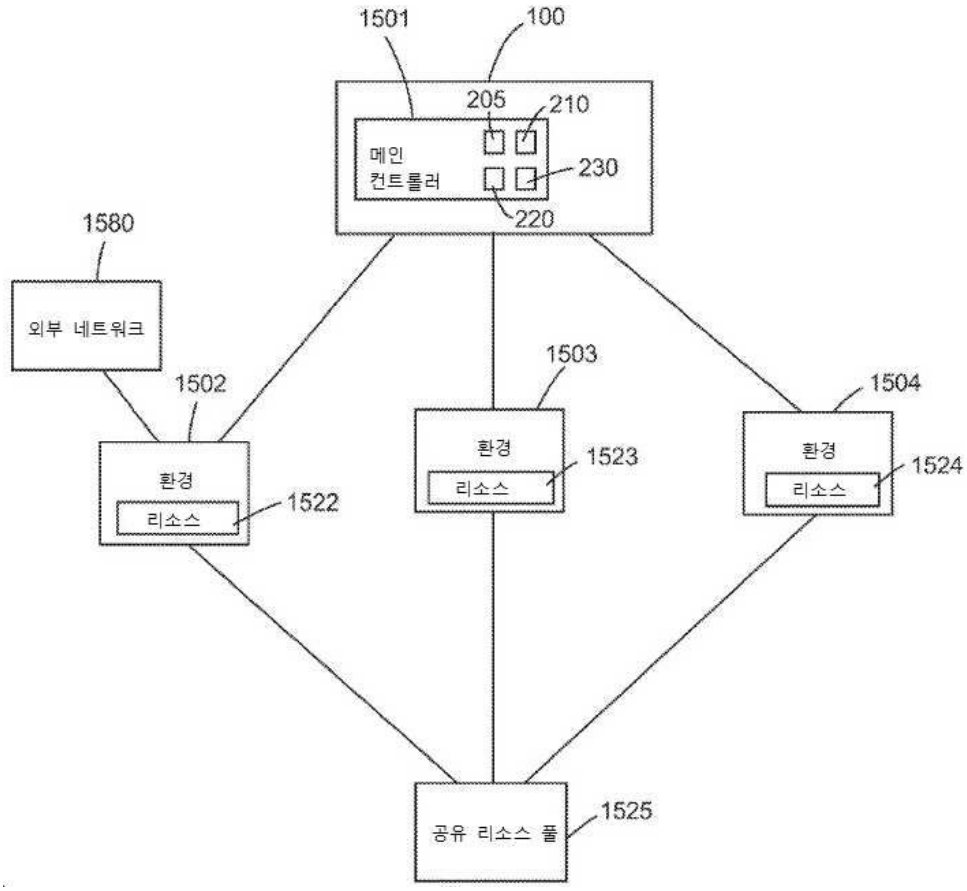
도면14b



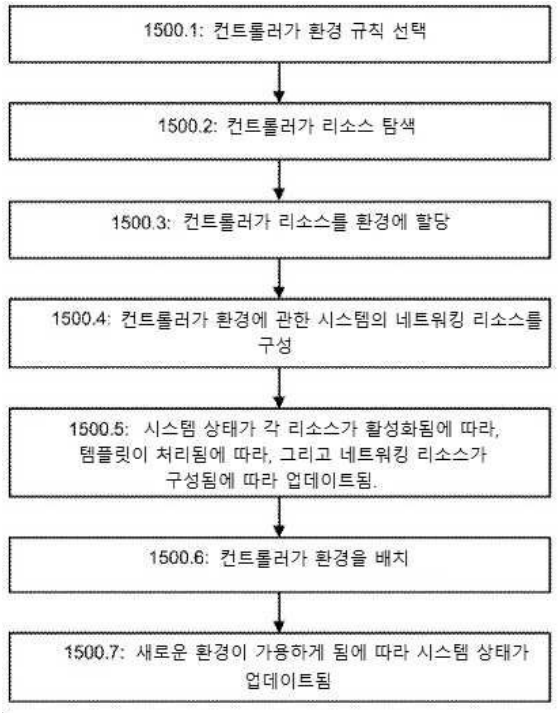
도면14c



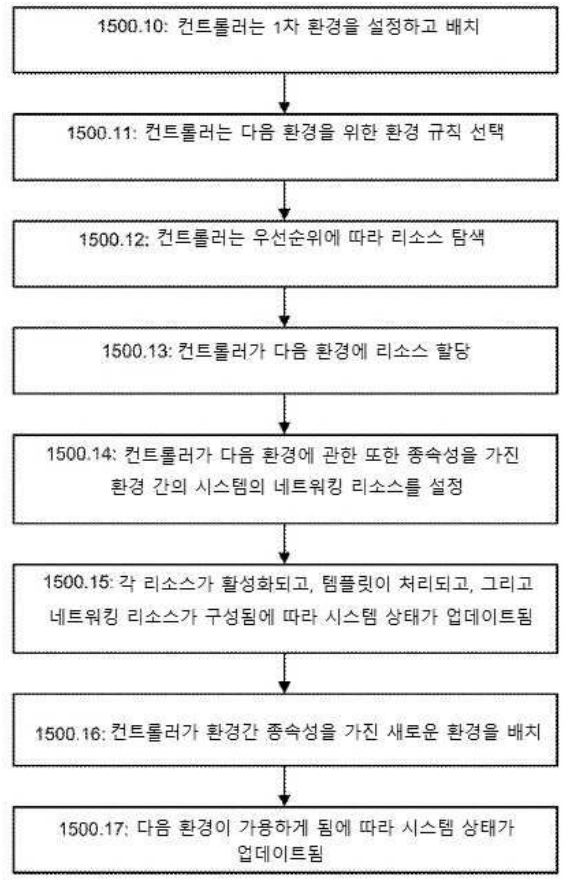
도면15a


1501
100
205
210
메인
컨트롤러
230
220
1580
외부 네트워크
1502
1503
1504
환경
리소스
1522
환경
리소스
1523
환경
리소스
1524
공유 리소스 풀
1525

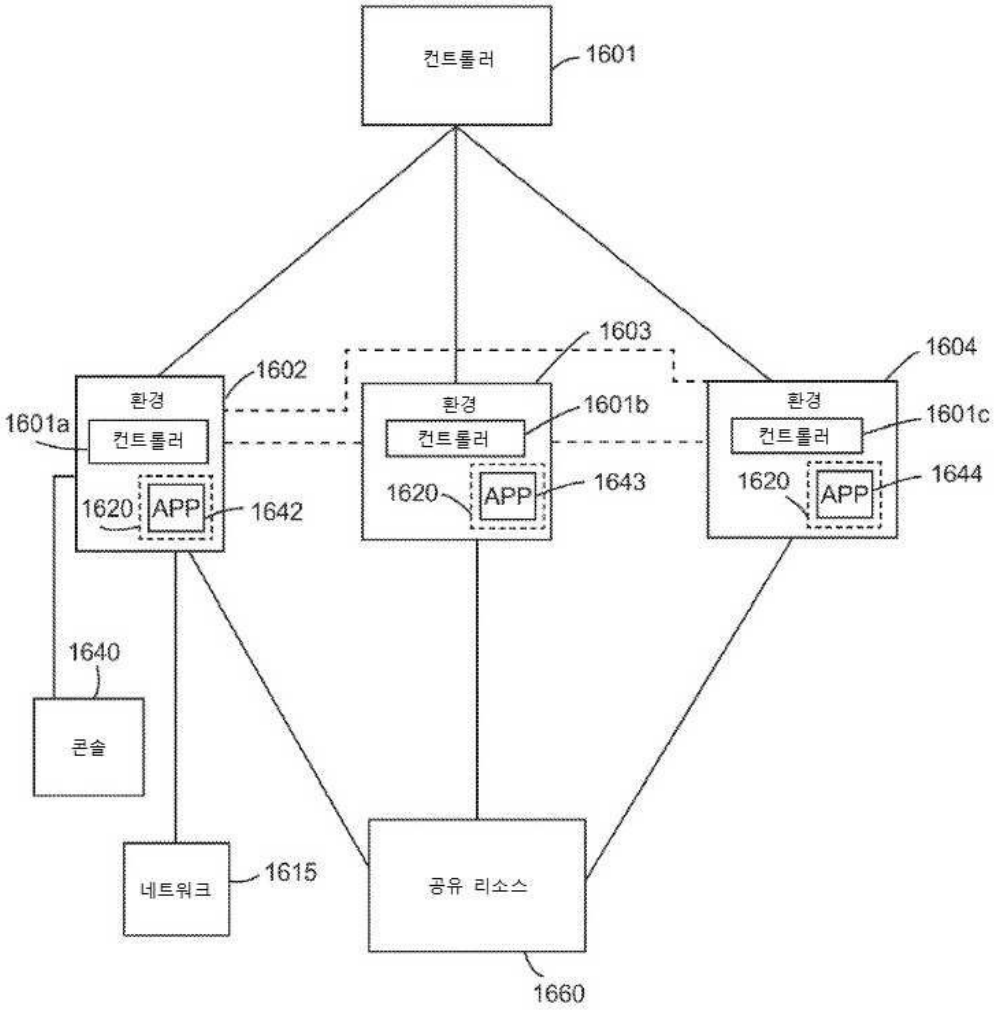
도면15b


1500.1: 컨트롤러가 환경 규칙 선택
1500.2: 컨트롤러가 리소스 탐색
1500.3: 컨트롤러가 리소스를 환경에 할당
1500.4: 컨트롤러가 환경에 관한 시스템의 네트워킹 리소스를 구성
1500.5: 시스템 상태가 각 리소스가 활성화됨에 따라, 템플릿이 처리됨에 따라, 그리고 네트워킹 리소스가 구성됨에 따라 업데이트됨.
1500.6: 컨트롤러가 환경을 배치
1500.7: 새로운 환경이 가용하게 됨에 따라 시스템 상태가 업데이트됨

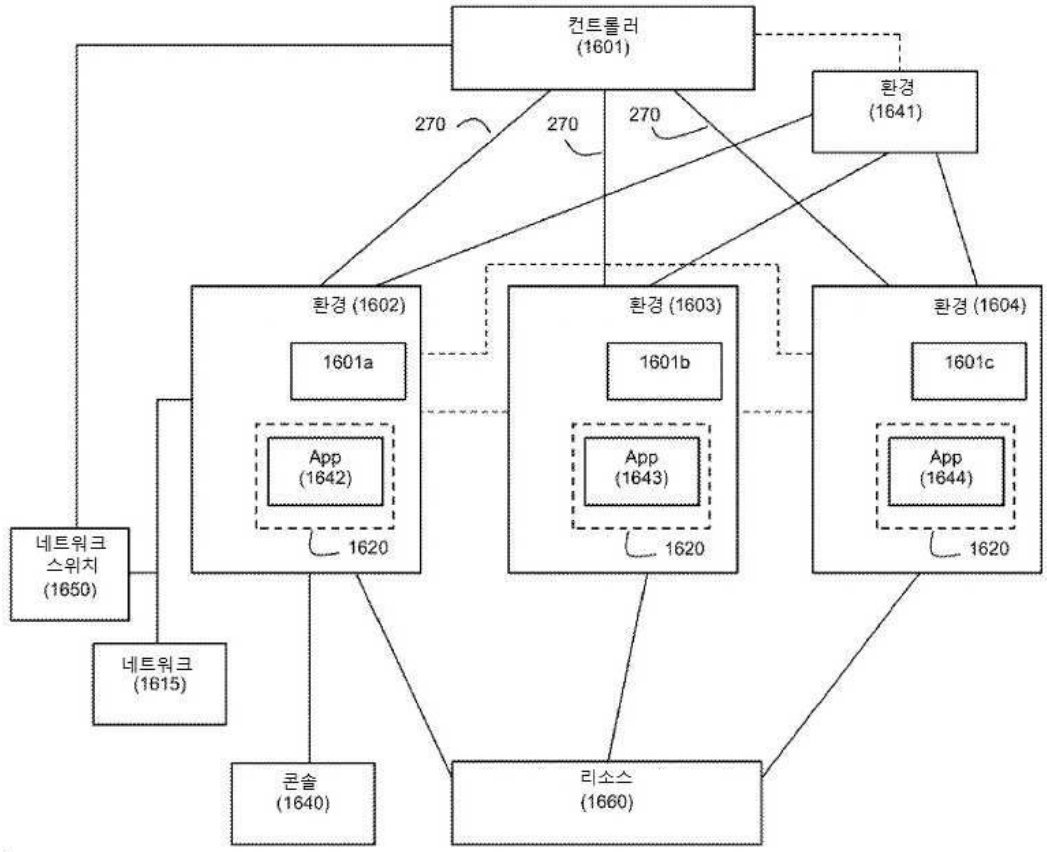
도면15c


1500.10: 컨트롤러는 1차 환경을 설정하고 배치
1500.11: 컨트롤러는 다음 환경을 위한 환경 규칙 선택
1500.12: 컨트롤러는 우선순위에 따라 리소스 탐색
1500.13: 컨트롤러가 다음 환경에 리소스 할당
1500.14: 컨트롤러가 다음 환경에 관한 또한 종속성을 가진 환경 간의 시스템의 네트워킹 리소스를 설정
1500.15: 각 리소스가 활성화되고, 템플릿이 처리되고, 그리고 네트워킹 리소스가 구성됨에 따라 시스템 상태가 업데이트됨
1500.16: 컨트롤러가 환경간 종속성을 가진 새로운 환경을 배치
1500.17: 다음 환경이 가용하게 됨에 따라 시스템 상태가 업데이트됨

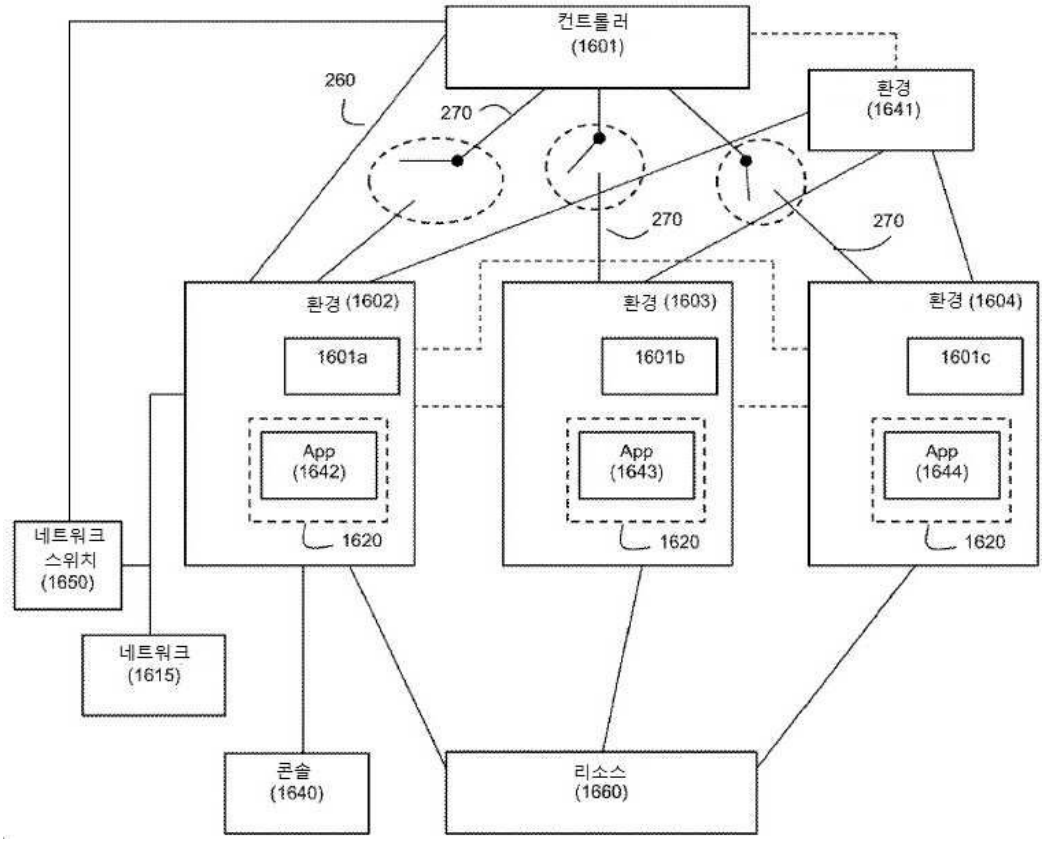
도면16a


컨트롤러
1601
1603
1604
환경
컨트롤러
1620
APP
1602
1642
1601a
환경
컨트롤러
1620
APP
1601b
1643
환경
컨트롤러
1620
APP
1601c
1644
1640
콘솔
네트워크
1615
공유 리소스
1660

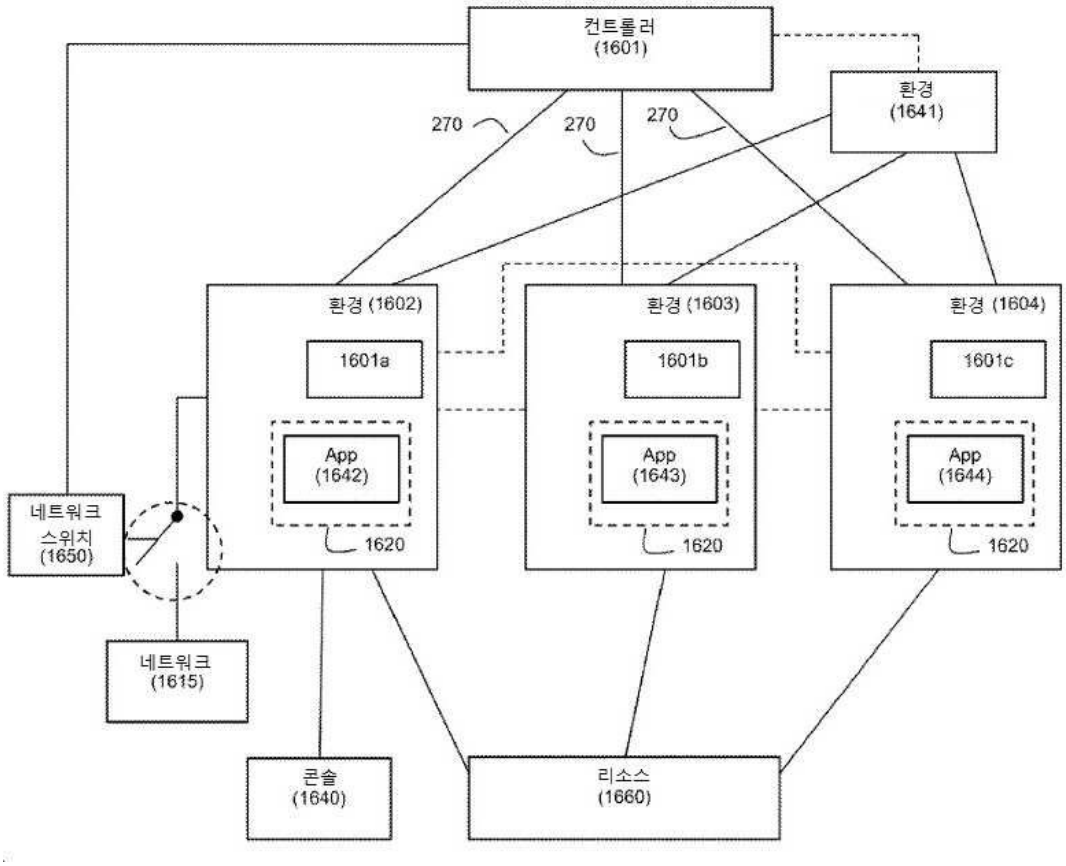
도면16b


컨트롤러
(1601)
환경
(1641)
270
270
270
환경 (1602)
환경 (1603)
환경 (1604)
1601a
1601b
1601c
App
(1642)
App
(1643)
App
(1644)
1620
1620
1620
네트워크
스위치
(1650)
네트워크
(1615)
콘솔
(1640)
리소스
(1660)

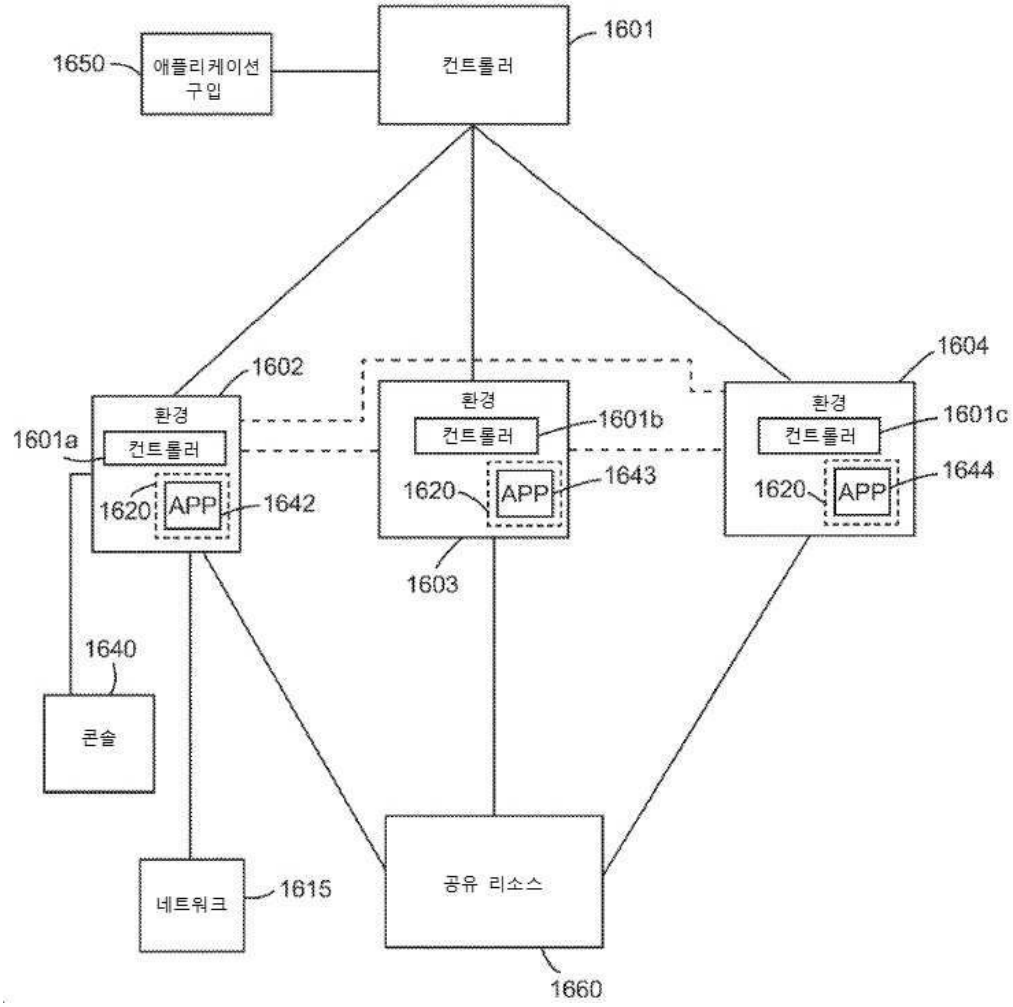
도면16c


컨트롤러
(1601)
260
270
환경
(1641)
270
270
환경 (1602)
환경 (1603)
환경 (1604)
1601a
1601b
1601c
App
(1642)
App
(1643)
App
(1644)
1620
1620
1620
네트워크
스위치
(1650)
네트워크
(1615)
콘솔
(1640)
리소스
(1660)

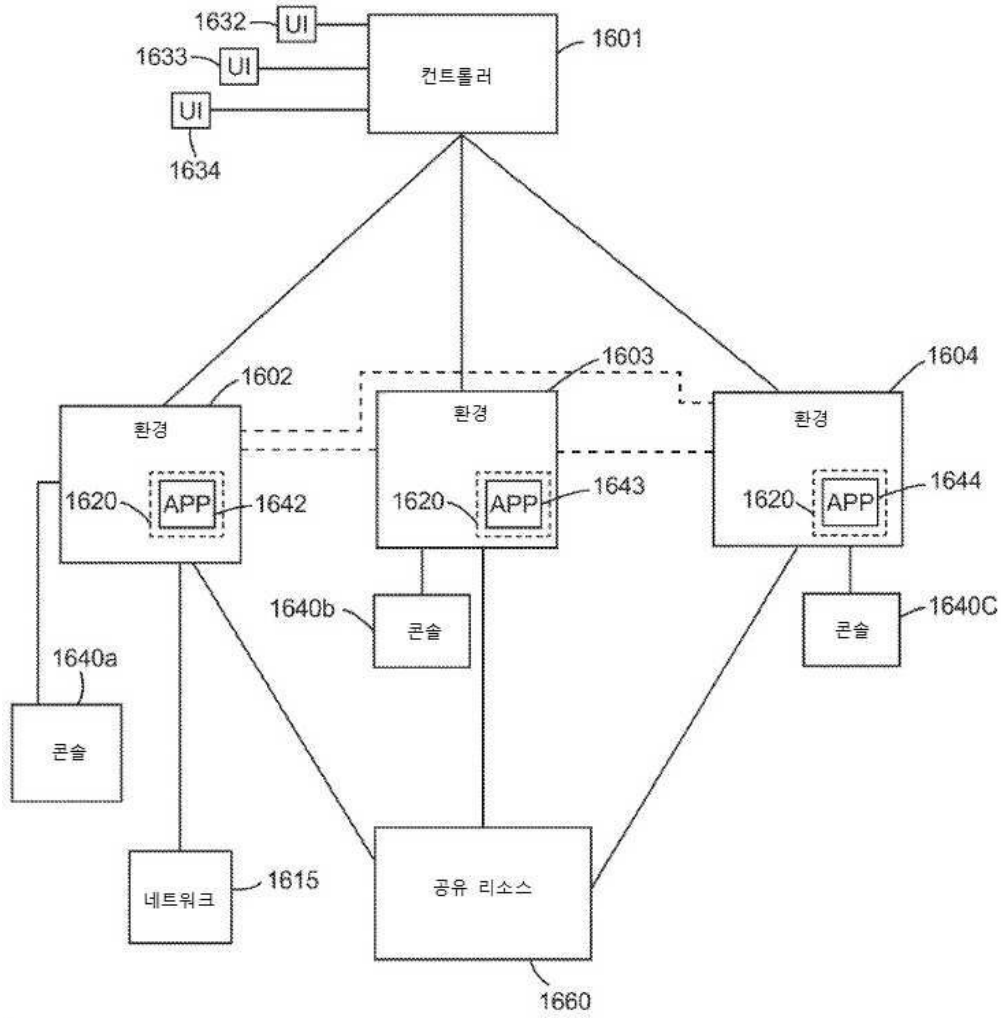
도면16d


컨트롤러
(1601)
환경
(1641)
270
270
270
환경 (1602)
환경 (1603)
환경 (1604)
1601a
1601b
1601c
App
(1642)
App
(1643)
App
(1644)
1620
1620
1620
네트워크
스위치
(1650)
네트워크
(1615)
콘솔
(1640)
리소스
(1660)

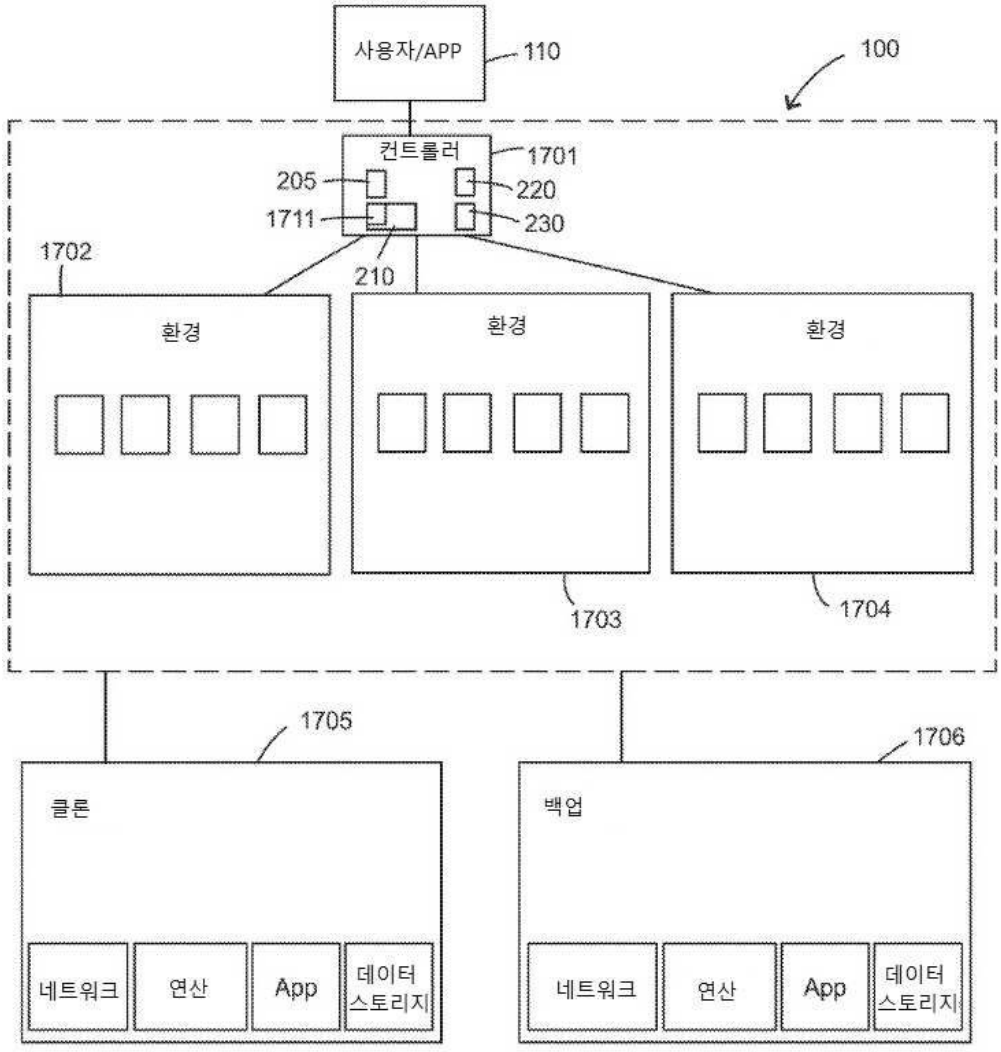
도면16e


1650
애플리케이션 구입
컨트롤러
1601
1602
환경
1601a
컨트롤러
1620
APP
1642
환경
컨트롤러
1601b
1620
APP
1643
1603
1604
환경
컨트롤러
1601c
1620
APP
1644
1640
콘솔
네트워크
1615
공유 리소스
1660

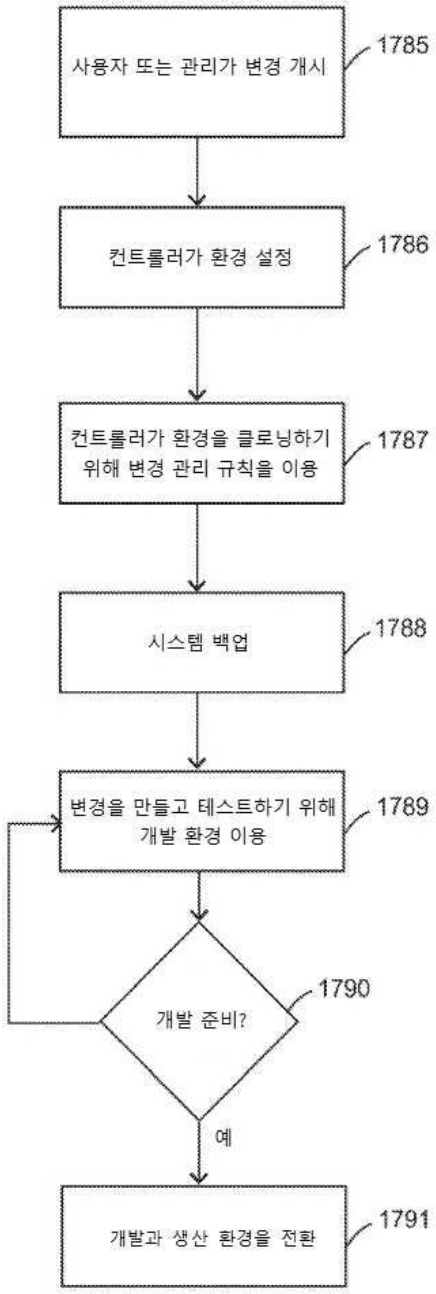
도면16f


1632
UI
1633
UI
UI
1634
컨트롤러
1601
1602
1603
1604
환경
환경
환경
1620
APP
1642
1620
APP
1643
1620
APP
1644
1640b
콘솔
1640C
콘솔
1640a
콘솔
네트워크
1615
공유 리소스
1660

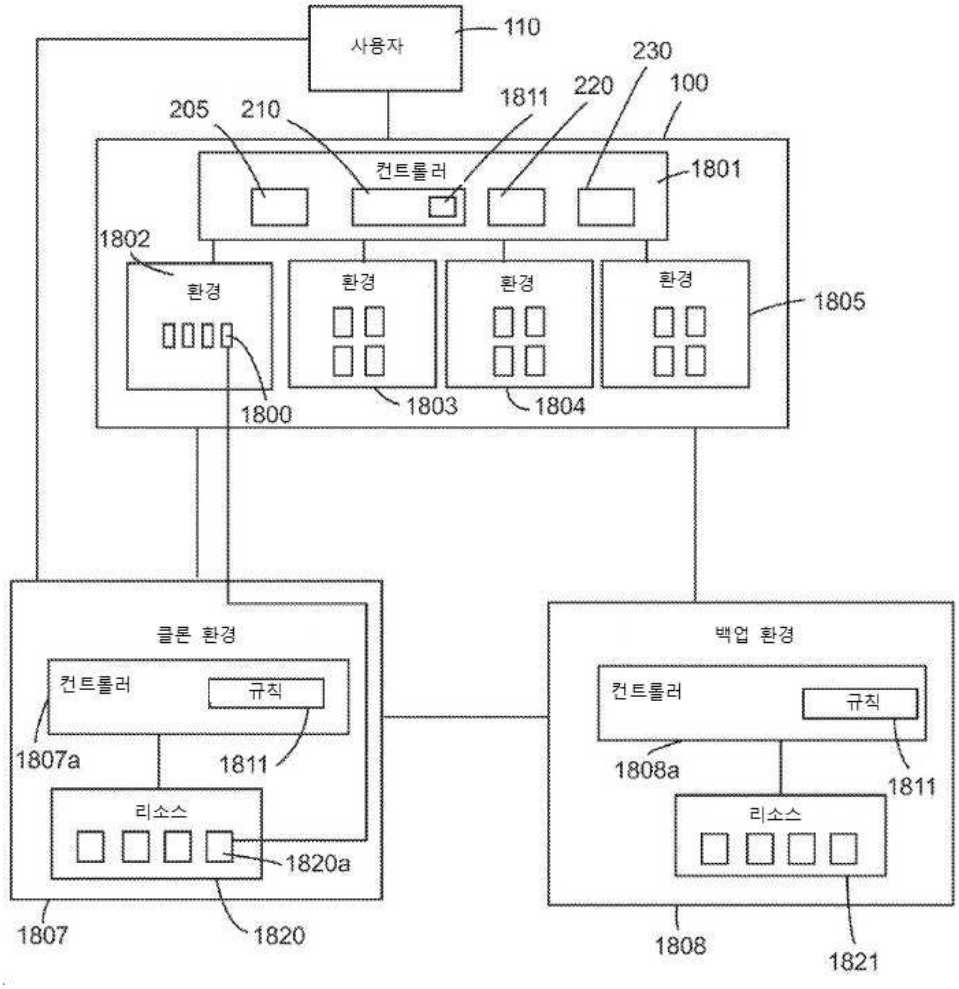
도면17a


사용자/APP
110
100
컨트롤러
1701
205
220
1711
230
210
1702
환경
환경
환경
1703
1704
1705
1706
클론
백업
네트워크
연산
App
데이터 스토리지
네트워크
연산
App
데이터 스토리지

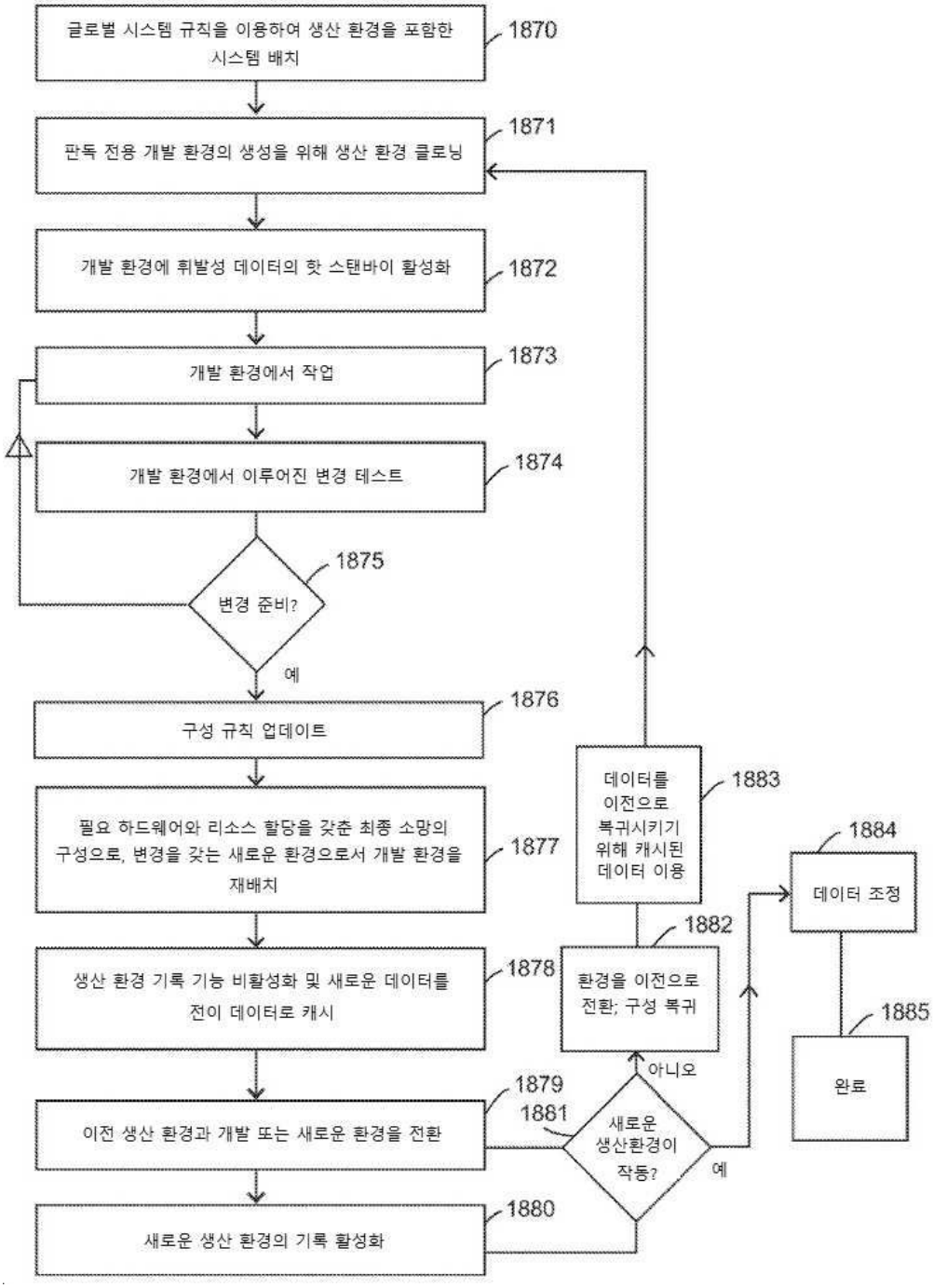
도면17b


사용자 또는 관리가 변경 개시
1785
컨트롤러가 환경 설정
1786
컨트롤러가 환경을 클로닝하기 위해 변경 관리 규칙을 이용
1787
시스템 백업
1788
변경을 만들고 테스트하기 위해 개발 환경 이용
1789
개발 준비?
1790
예
개발과 생산 환경을 전환
1791

도면18a


사용자
110
230
100
220
1811
205
210
컨트롤러
1801
1802
환경
환경
환경
환경
1805
1800
1803
1804
클론 환경
컨트롤러
규직
1807a
1811
리소스
1820a
1807
1820
백업 환경
컨트롤러
규직
1808a
1811
리소스
1808
1821

도면18b


글로벌 시스템 규칙을 이용하여 생산 환경을 포함한 시스템 배치
1870
판독 전용 개발 환경의 생성을 위해 생산 환경 클로닝
1871
개발 환경에 휘발성 데이터의 핫 스탠바이 활성화
1872
개발 환경에서 작업
1873
개발 환경에서 이루어진 변경 테스트
1874
변경 준비?
1875
예
구성 규칙 업데이트
1876
필요 하드웨어와 리소스 할당을 갖춘 최종 소망의 구성으로, 변경을 갖는 새로운 환경으로서 개발 환경을 재배치
1877
생산 환경 기록 기능 비활성화 및 새로운 데이터를 전이 데이터로 캐시
1878
이전 생산 환경과 개발 또는 새로운 환경을 전환
1879
1881
새로운 생산환경이 작동?
아니오
예
환경을 이전으로 전환; 구성 복귀
1882
데이터를 이전으로 복귀시키기 위해 캐시된 데이터 이용
1883
데이터 조정
1884
완료
1885
새로운 생산 환경의 기록 활성화
1880