(19)대한민국특허청(KR)
(12) 공개특허공보(A)

(51) 。Int. Cl.
G06F 9/06 (2006.01)
G06F 15/00 (2006.01)
G06F 9/00 (2006.01)

(11) 공개번호 10-2006-0100915
(43) 공개일자 2006년09월21일

(21) 출원번호 10-2005-0120757
(22) 출원일자 2005년12월09일

(30) 우선권주장 JP-P-2005-00070505 2005년03월14일 일본(JP)
JP-P-2005-00249530 2005년08월30일 일본(JP)

(71) 출원인 세이코 엡슨 가부시키가이샤
일본 도쿄도 신주쿠구 니시신주쿠 2초메 4-1

(72) 발명자 다니구치 신야
일본국 나가노켄 스와시 오와 3쵸메 3반 5고 세이코 엡슨가부시키가이샤 내
후카오 아키히토
일본국 나가노켄 스와시 오와 3쵸메 3반 5고 세이코 엡슨가부시키가이샤 내

(74) 대리인 한양특허법인

심사청구 : 있음

(54) 소프트웨어 인증 시스템, 소프트웨어 인증 방법 및소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한기록 매체

요약

본 발명은 소프트웨어의 동작을 그 실행 환경에 도입하기 전에 검증함으로써, 소프트웨어의 개발을 용이하게 하는 동시에 안정성이 높은 소프트웨어를 개발하는 데에 적합한 소프트웨어 인증 시스템을 제공한다.

호스트 단말은, 개별 기능 모듈이 호스트 단말에서 사용하는 리소스의 양을 측정하고, 측정한 리소스의 양을 네트워크 프린터에서 사용하는 리소스의 양으로 환산하며, 개별 기능 모듈로부터 상한값을 취득하고, 환산한 리소스의 양 및 취득한 상한값에 기초하여 개별 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성한다. 그리고, 로그 파일로부터 로그 정보를 판독하고, 판독한 로그 정보에 기초하여 개별 기능 모듈이 사용하는 리소스의 양이 상한에 도달했는지의 여부를 판정하며, 리소스의 양이 상한에 도달하고 있지 않다고 판정했을 때는 개별 기능 모듈의 실행 파일에 전자 서명 정보를 부가한다.

대표도

도 18

명세서

도면의 간단한 설명

도 1은 JAVA(등록상표) 소프트웨어의 구성을 나타내는 도면이다.

도 2는 호스트 단말(100)의 기능 개요를 나타내는 기능 블록도이다.

도 3은 호스트 단말(100)의 하드웨어 구성을 나타내는 블록도이다.

도 4는 리소스 제한 정보(400)의 데이터 구조를 나타내는 도면이다.

도 5는 모듈 정보(420)의 데이터 구조를 나타내는 도면이다.

도 6은 실행 환경 정보 등록 테이블(440)의 데이터 구조를 나타내는 도면이다.

도 7은 리소스 환산 테이블(22)의 데이터 구조를 나타내는 도면이다.

도 8은 리소스 관리 테이블(460)의 데이터 구조를 나타내는 도면이다.

도 9는 이벤트 리스너 테이블(480)의 데이터 구조를 나타내는 도면이다.

도 10은 개별 기능 모듈 제어 처리를 나타내는 플로우차트이다.

도 11은 실행 여부 판정 처리를 나타내는 플로우차트이다.

도 12는 모듈 기동 처리를 나타내는 플로우차트이다.

도 13은 이벤트 리스너 등록 처리를 나타내는 플로우차트이다.

도 14는 클래스 판독 처리를 나타내는 플로우차트이다.

도 15는 이벤트 리스너 제어 처리를 나타내는 플로우차트이다.

도 16은 이벤트 리스너 실행 처리를 나타내는 플로우차트이다.

도 17은 인스턴스 삭제 처리를 나타내는 플로우차트이다.

도 18은 모듈 인증 처리를 나타내는 플로우차트이다.

도 19는 에러가 발생한 경우의 로그 파일의 내용을 나타내는 도면이다.

도 20은 에러가 발생하지 않은 경우의 로그 파일의 내용을 나타내는 도면이다.

도 21은 리소스 관리 대상이 되는 개별 기능 모듈(b, c)을 병렬로 실행한 경우를 나타내는 타임차트이다.

도 22는 호스트 단말(100)의 기능 개요를 나타내는 기능 블록도이다.

도 23은 리소스 환산 테이블(22)의 데이터 구조를 나타내는 도면이다.

도 24는 리소스 관리 테이블(460)의 데이터 구조를 나타내는 도면이다.

도 25는 실행 여부 판정 처리를 나타내는 플로우차트이다.

도 26은 호스트 단말(100)의 기능 개요를 나타내는 기능 블록도이다.

도 27은 이벤트 리스너 실행 처리를 나타내는 플로우차트이다.

도 28은 모듈 인증 처리를 나타내는 플로우차트이다.

도 29는 파일 조작이 행해진 경우의 로그 파일의 내용을 나타내는 도면이다.

도 30은 파일 조작이 행해진 경우의 로그 파일의 내용을 나타내는 도면이다.

도 31은 모듈 인증 처리를 나타내는 플로우차트이다.

도 32는 클래스 검증 처리를 나타내는 플로우차트이다.

<도면의 주요 부분에 대한 부호의 설명>

100 : 호스트 단말 110 : OS

120 : 공통 기능 모듈 130 : 개별 기능 모듈

10, 16 : 리소스 측정부 12, 20 : 리소스 제한부

14 : 개별 기능 모듈 관리부 18 : 상한값 취득부

22 : 리소스 환산 테이블 24, 34 : 리소스 환산부

26 : 로그 정보 생성부 28 : 로그 정보 취득부

30 : 동작 판정부 32 : 모듈 인증부

36 : 리소스 감시부 50 : CPU

52 : ROM 54 : RAM

58 : I/F 60 : 입력 장치

62 : 기억 장치 64 : 표시 장치

400 : 리소스 제한 정보 420 : 모듈 정보

440 : 실행 환경 정보 등록 테이블 460 : 리소스 관리 테이블

480 : 이벤트 리스너 테이블 520 : 실행 결과 등록 테이블

199 : 네트워크

발명의 상세한 설명

발명의 목적

발명이 속하는 기술 및 그 분야의 종래기술

본 발명은 소프트웨어를 인증하는 시스템, 프로그램 및 방법에 관한 것으로, 특히 소프트웨어의 동작을 그 실행 환경에 도입하기 전에 검증함으로써, 소프트웨어의 개발을 용이하게 하는 동시에 안정성이 높은 소프트웨어를 개발하는 데에 적합한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법에 관한 것이다.

프린터 등의 내장 기기에는, 내장용 애플리케이션이라고 하는 소프트웨어를 내장하여 그 동작을 제어하고 있다. 그러나, 내장용 애플리케이션을 작성하는 데는, 일반적으로 전용의 개발 환경과 전용의 하드웨어를 필요로 하기 때문에, 누구든지 간단히 작성할 수 있는 것은 아니었다. 이 문제를 해결하기 위해, 일본 특허공개 2004-185595호 공보에 기재된 정보 처리 장치가 제안되어 있다

일본 특허공개 2004-185595호 공보에 기재된 발명은, 화상 형성 장치 상에서 실행되는 애플리케이션을 PC 상에서 실행할 수 있는 에뮬레이터를 갖고 구성되어 있다. 이에 의해, 내장 기기를 사용하지 않고도 내장용 애플리케이션을 개발할 수 있다

또한, 프로그래밍 기술이 미숙한 사람이 작성하는 내장용 애플리케이션은, 예고없이 동작을 일으켜, 내장 기기 자체의 동작이 속행할 수 없게 된다는 문제가 있다. 이 문제를 해결하기 위해, 일본 특허공개 2004-94782호 공보에 기재된 리소스 관리 시스템이 제안되어 있다.

일본 특허공개 2004-94782호 공보에 기재된 발명은, 소프트웨어가 정보 기기에서 실행할 때에 이용하는 리소스에 대해 동작 가능한 범위를 설정하는 제한 설정부와,제한 설정부에서 설정한 동작 가능한 범위 내에서 동작하고 있음을 검증하는 동작 범위 검증부를 갖고 구성되어 있다. 동작 범위 검증부는, 소프트웨어로부터 리소스 이용 요구가 있었을 때, 설정되어 있는 동작 가능한 범위와 요구된 리소스의 양을 비교하여, 동작 가능한 범위 외의 경우에는 그 소프트웨어의 실행을 중지시킨다.

발명이 이루고자 하는 기술적 과제

일본 특허공개 2004-185595호 공보에 기재된 발명에서는, 내장용 애플리케이션을 PC 상에서 가상적으로 실행할 수 있기 때문에, 내장용 애플리케이션의 동작을 PC 상에서 어느 정도 검증할 수 있다.

그러나, 내장용 애플리케이션을 PC 상에서 실행한 경우와, 실제로 그 내장용 애플리케이션을 내장 기기 상에서 실행한 경우에, 그 내장용 애플리케이션이 각각의 실행 환경에서 사용하는 리소스(예를 들어, 메모리)의 양이 완전히 일치하지는 않는다. 이는, 내장용 애플리케이션을 실행하는 환경이 다른 것에 기인한다. 예를 들어, 내장용 애플리케이션을 실행하기 위해 라이브러리를 사용하는데, 이 라이브러리는, 하드웨어 구성의 차이로부터 PC용과 내장 기기용에서 각각 프로그램 구조가 다르다. 그 때문에, 같은 함수를 사용하는 경우에도, PC 상과 내장 기기 상에서는, 기능은 같지만 프로그램 구조가 다른 라이브러리가 링크되어 오브젝트가 생성된다. 그들 오브젝트는 프로그램 구조가 다르므로, 당연히 사용하는 리소스의 양도 완전히 일치하지는 않는다. 따라서, PC 상에서는 적절하게 동작한 내장용 애플리케이션에서도, 실제로 내장 기기에 내장할 때 사용하는 리소스의 양이 너무 많기 때문에, 복수의 내장용 애플리케이션이 내장 기기 상에서 기동한 경우에, 다른 내장용 애플리케이션과 경합하여 동작이 불안정하게 될 가능성이 있었다. 특히, 프린터 등의 내장 기기는, PC와 달리 사용 가능한 리소스의 양이 극단적으로 적기 때문에, 개개의 내장용 애플리케이션이 사용하는 리소스의 양을 상세하게 관리하는 것은 안정동작을 실현하는 데 있어서 매우 중요하다.

또한, 내장용 애플리케이션 자체 또는 그것이 취급하는 파일에 대해서는, 파일명이나 패스명에 일정한 제한이 있다. PC에서는 사용할 수 있는 길이나 문자종류도 내장 기기에서는 사용할 수 없는 경우가 있다. 따라서, PC 상에서는 적절하게 동작한 내장용 애플리케이션에서도, 실제로 내장 기기에 내장하였을 때 파일명이나 패스명이 적절하지 않기 때문에, 동작에 결함이 발생할 가능성이 있었다.

한편, 일본 특허공개 2004-94782호 공보에 기재된 발명에 있어서는, 어디까지나 내장용 애플리케이션을 내장 기기 상에서 실행한 경우에 동작이 불안정해지는 것을 방지하는 것이고, 내장용 애플리케이션이 사용하는 리소스의 양, 파일명이나 패스명 등을 내장 기기에 도입하기 전에 검증할 수는 없다.

이러한 문제는, 내장용 애플리케이션을 내장 기기 상에서 실행하는 경우에 한정되지 않고, 특정의 실행 환경에서 실행시키는 소프트웨어를 다른 실행 환경에서 개발하는 모든 경우에 대해서도 동일하게 상정된다.

그래서, 본 발명은, 이러한 종래의 기술이 갖는 미해결의 과제에 착안하여 이루어진 것으로, 소프트웨어의 동작을 그 실행 환경에 도입하기 전에 검증함으로써, 소프트웨어의 개발을 용이하게 하는 동시에 안정성이 높은 소프트웨어를 개발하는 데에 매우 적합한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 제공하는 것을 목적으로 하고 있다.

발명의 구성 및 작용

(형태 1) 상기 목적을 달성하기 위해, 형태 1의 소프트웨어 인증 시스템은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 시스템으로서, 상기 로그 정보를 취득하는 로그 정보 취득 수단과, 상기 로그 정보 취득 수단으로 취득한 로그 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 수단과, 상기 동작 판정 수단이 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 수단을 구비하는 것을 특징으로 한다.

이러한 구성이면, 리소스 관리 시스템에서는, 리소스 측정 수단에 의해, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양이 측정되고, 리소스 환산 수단에 의해, 측정된 리소스의 양이 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산된다. 또한, 리소스 제한 정보 취득 수단에 의해, 리소스 제한 정보가 취득된다. 그리고, 로그 정보 생성 수단에 의해, 환산된 리소스의 양 및 취득된 리소스 제한 정보에 기초하여 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보가 생성된다.

본 시스템에서는, 로그 정보 취득 수단에 의해, 이와 같이 생성된 로그 정보가 취득되고, 동작 판정 수단에 의해, 취득된 로그 정보에 기초하여 기능 모듈이 사용하는 리소스의 양이 상한에 도달했는지의 여부가 판정된다. 그 결과, 상한에 도달하고 있지 않다고 판정되면, 소프트웨어 인증 수단에 의해 소프트웨어에 인증 정보가 부가된다.

이에 의해, 기능 모듈이 사용하는 리소스의 양이 상한에 도달하지 않은 소프트웨어에 대해서만 인증 정보가 부가되므로, 소프트웨어의 동작을 비교적 확실하게 보증할 수 있다. 따라서, 종래에 비해, 소프트웨어의 개발을 용이하게 할 수 있는 동시에 안정성이 높은 소프트웨어를 개발할 수 있다는 효과가 얻어진다.

여기서, 리소스란 기능 모듈이 사용 가능한 자원을 말하고, 하드웨어 자원에 한정되지 않으며, 소프트웨어 자원의 그 밖의 자원이 포함된다. 이하, 형태 2 및 3의 소프트웨어 인증 시스템, 형태 5 내지 7의 소프트웨어 인증 프로그램, 및 형태 9 내지 14의 소프트웨어 인증 방법에 있어서 동일하다.

또한, 리소스의 양으로서는, 예를 들어 기능 모듈이 사용하는 메모리량 또는 기동 가능한 기능 모듈의 수가 포함된다. 또, 예를 들어 기능 모듈을 이용하는 애플리케이션이 확보하는 리소스의 양(메모리량, 기능 모듈 수)이 포함된다. 이하, 형태 2 및 3의 소프트웨어 인증 시스템, 형태 5 내지 7의 소프트웨어 인증 프로그램, 및 형태 9 내지 14의 소프트웨어 인증 방법에 있어서 동일하다.

또한, 로그 정보 취득 수단은, 로그 정보를 취득하도록 되어 있으면 어떠한 구성이어도 되고, 예를 들어 입력 장치 등으로부터 로그 정보를 입력하도록 되어 있어도 되며, 외부의 장치 등으로부터 로그 정보를 획득 또는 수신하도록 되어 있어도 되고, 기억 장치나 기억 매체 등으로부터 로그 정보를 독출하도록 되어 있어도 되며, 기능 모듈 그 밖의 데이터로부터 로그 정보를 추출하도록 되어 있어도 된다. 따라서,취득에는 적어도 입력, 획득, 수신, 독출 및 추출이 포함된다. 이하, 형태 2 및 3의 소프트웨어 인증 시스템에 있어서 동일하다.

또, 리소스의 환산 방법으로서는, 예를 들어 다음의 방법이 생각된다. 제1 방법(일정 비율 환산 방법)은, 측정한 리소스량에 대해 일정한 비율의 리소스량을 증가 또는 감소하여 환산 후의 리소스량을 구한다. 제2 방법(일정량 환산 방법)은, 측정한 리소스량에 관하지 않고, 그 리소스량으로부터 일정량을 증가 또는 감소하여 환산 후의 리소스량을 구한다. 제3 방법(정수 환산 방법)은, 측정한 리소스량에 관하지 않고, 그 리소스량을 정수로 치환하여 환산 후의 리소스량을 구한다. 제4 방법(혼합형 환산 방법)은, 제1～제3 방법 중에서 리소스의 종류 또는 측정한 리소스량에 기초하여 선택하고, 선택한 방법에 의해 환산 후의 리소스량을 구한다. 환산 방법의 선택은, 예를 들어 다음과 같이 임계값 A, B를 설정하여 행할 수 있다.

리소스량<A일 때는 정수 환산 방법을 선택하고, A<리소스량<B일 때는 일정량 환산 방법을 선택하며, B<리소스량일 때는 일정 비율 환산 방법을 선택한다. 단, A<B이다. 이하, 형태 2의 소프트웨어 인증 시스템, 형태 5 및 6의 소프트웨어 인증 프로그램, 및 형태 9 내지 12의 소프트웨어 인증 방법에 있어서 동일하다.

또한, 리소스의 제한 조건으로서는, 예를 들어 제2 실행 환경에서의 리소스의 상한값을 설정할 수 있다. 이하, 형태 5의 소프트웨어 인증 프로그램 및 형태 9 및 10의 소프트웨어 인증 방법에 있어서 동일하다.

또, 인증 정보란, 제2 실행 환경에서 소프트웨어를 실행할 때에 실행의 여부를 판정하기 위해 이용되는 정보로서, 안전성의 관점에서는 타인이 복제할 수 없고, 실행의 여부를 판정할 수 있는 정보인 것이 바람직하다. 또한, 정보의 형식은 임의의 것을 채용할 수 있다. 이하, 형태 2 및 3의 소프트웨어 인증 시스템, 형태 5 내지 7 의 소프트웨어 인증 프로그램, 및 형태 9 내지 14의 소프트웨어 인증 방법에 있어서 동일하다.

또한, 본 시스템은, 단일의 장치, 단말의 그 밖의 기기로서 실현하도록 해도 되고, 복수의 장치, 단말의 그 밖의 기기를 통신 가능하게 접속한 네트워크 시스템으로서 실현하도록 해도 된다. 후자의 경우, 각 구성요소는 각각 통신 가능하게 접속되어 있으면, 복수의 기기 중 어느 하나에 속해 있어도 된다. 이하, 형태 2 및 3의 소프트웨어 인증 시스템에 있어서 동일하다.

또, 리소스 관리 시스템의 보다 구체적인 구성으로서는, 다음의 2개의 구성을 제안할 수 있다.

제1 구성은, 제1 기능 모듈과, 그 실행에 상기 제1 기능 모듈을 필요로 하는 복수의 제2 기능 모듈을 포함하는 소프트웨어가 사용하는 리소스를 관리하는 리소스 관리 시스템으로서, 상기 제2 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 제2 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 상기 제2 실행 환경에서의 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비한다.

제2 구성은, 제1 기능 모듈과, 그 실행에 상기 제1 기능 모듈을 필요로 하는 복수의 제2 기능 모듈을 포함하는 소프트웨어가 사용하는 리소스를 관리하는 리소스 관리 시스템으로서, 상기 제1 기능 모듈이 상기 제1 실행 환경에서 상기 제2 기능 모듈의 실행에 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 제2 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 상기 제2 실행 환경에서의 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비한다. 이하, 형태 5의 소프트웨어 인증 프로그램 및 형태 9 및 10의 소프트웨어 인증 방법에 있어서 동일하다.

(형태 2) 또, 형태 2의 소프트웨어 인증 시스템은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 시스템으로서, 상기 로그 정보를 취득하는 로그 정보 취득 수단과, 상기 로그 정보 취득 수단으로 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 수단과, 상기 동작 판정 수단이 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 수단을 구비하는 것을 특징으로 한다.

이러한 구성이면, 리소스 관리 시스템에서는, 리소스 측정 수단에 의해, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양이 측정된다. 또한, 리소스 상한값 취득 수단에 의해, 제2 실행 환경에서의 리소스의 상한값이 취득되고, 리소스 환산 수단에 의해, 취득된 리소스의 상한값이 제1 실행 환경에서의 리소스의 상한값으로 환산된다. 그리고, 로그 정보 생성 수단에 의해, 측정된 리소스의 양 및 환산된 리소스의 상한값에 기초하여 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보가 생성된다.

본 시스템에서는, 로그 정보 취득 수단에 의해, 이와 같이 생성된 로그 정보가 취득되고, 동작 판정 수단에 의해, 취득된 로그 정보에 기초하여 기능 모듈이 사용하는 리소스의 양이 상한에 도달했는지의 여부가 판정된다. 그 결과, 상한에 도달하고 있지 않다고 판정되면, 소프트웨어 인증 수단에 의해 소프트웨어에 인증 정보가 부가된다.

이에 의해, 기능 모듈이 사용하는 리소스의 양이 상한에 도달하지 않은 소프트웨어에 대해서만 인증 정보가 부가되므로, 소프트웨어의 동작을 비교적 확실하게 보증할 수 있다. 따라서, 종래에 비해, 소프트웨어의 개발을 용이하게 할 수 있는 동시에 안정성이 높은 소프트웨어를 개발할 수 있다는 효과가 얻어진다.

여기서, 리소스 관리 시스템의 보다 구체적인 구성으로서는, 다음의 2개의 구성을 제안할 수 있다.

제1 구성은, 제1 기능 모듈과, 그 실행에 상기 제1 기능 모듈을 필요로 하는 복수의 제2 기능 모듈을 포함하는 소프트웨어가 사용하는 리소스를 관리하는 리소스 관리 시스템으로서, 상기 제2 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비한다.

제2 구성은, 제1 기능 모듈과, 그 실행에 상기 제1 기능 모듈을 필요로 하는 복수의 제2 기능 모듈을 포함하는 소프트웨어가 사용하는 리소스를 관리하는 리소스 관리 시스템으로서, 상기 제1 기능 모듈이 상기 제1 실행 환경에서 상기 제2 기능 모듈의 실행에 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비한다. 이하, 형태 6의 소프트웨어 인증 프로그램 및 형태 11 및 12의 소프트웨어 인증 방법에 있어서 동일하다.

(형태 3) 또한, 형태 3의 소프트웨어 인증 시스템은, 제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 시스템으로서, 상기 로그 정보를 취득하는 로그 정보 취득 수단과, 상기 로그 정보 취득 수단으로 취득한 로그 정보에 기초하여 상기 리소스 사용 상황이 상기 제1 실행 환경에 적합한지를 판정하는 동작 판정 수단과, 상기 동작 판정 수단이 적합하다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 수단을 구비하는 것을 특징으로 한다.

이러한 구성이면, 리소스 관리 시스템에서는, 리소스 감시 수단에 의해, 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 사용 상황이 감시되고, 로그 정보 생성 수단에 의해, 그 감시 결과에 기초하여 리소스 사용 상황을 나타내는 로그 정보가 생성된다.

본 시스템에서는, 로그 정보 취득 수단에 의해, 이와 같이 생성된 로그 정보가 취득되고, 동작 판정 수단에 의해, 취득된 로그 정보에 기초하여 리소스 사용 상황이 제1 실행 환경에 적합한지의 여부가 판정된다. 그 결과, 리소스 사용 상황이 적합하다고 판정되면, 소프트웨어 인증 수단에 의해 소프트웨어에 인증 정보가 부가된다.

이에 의해, 기능 모듈이 사용하는 리소스의 사용 상황이 제1 실행 환경에 적합한 소프트웨어에 대해서만 인증 정보가 부가되므로, 소프트웨어의 동작을 비교적 확실하게 보증할 수 있다. 따라서, 종래에 비해, 소프트웨어의 개발을 용이하게 할 수 있는 동시에 안정성이 높은 소프트웨어를 개발할 수 있다는 효과가 얻어진다.

여기서, 리소스 사용 상황으로서는, 예를 들어 기능 모듈 자체 또는 그것이 취급하는 파일에 대한 파일명 및 패스명의 길이, 파일명 및 패스명에 사용되는 문자 종류가 포함된다. 이하, 형태 7의 소프트웨어 인증 프로그램 및 형태 13 및 14의 소프트웨어 인증 방법에 있어서 동일하다.

(형태 4) 또, 형태 4의 소프트웨어 인증 시스템은, 형태 1 내지 3 중 어느 하나의 소프트웨어 인증 시스템에 있어서, 상기 기능 모듈의 실행에 필요한 실행 파일을 취득하는 실행 파일 취득 수단과, 상기 실행 파일 취득 수단으로 취득한 실행 파일

에 기초하여, 상기 기능 모듈을 구성하는 명령 또는 명령군이 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어지는지를 판정하는 제2 동작 판정 수단을 구비하고, 상기 소프트웨어 인증 수단은, 상기 제2 동작 판정 수단이 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어진다고 판정했을 때는, 상기 소프트웨어에 상기 인증 정보를 부가하도록 되어 있는 것을 특징으로 한다.

이러한 구성이면, 실행 파일 취득 수단에 의해, 기능 모듈의 실행에 필요한 실행 파일이 취득되고, 제2 동작 판정 수단에 의해, 취득된 실행 파일에 기초하여 기능 모듈을 구성하는 명령 또는 명령군이 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어지는지의 여부가 판정된다. 그 결과, 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어진다고 판정되면, 소프트웨어 인증 수단에 의해 소프트웨어에 인증 정보가 부가된다.

이에 의해, 기능 모듈을 구성하는 명령 또는 명령군이 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어지는 소프트웨어에 대해서만 인증 정보가 부가되므로, 소프트웨어의 동작을 더 확실하게 보증할 수 있다.

(형태 5) 한편, 상기 목적을 달성하기 위해, 형태 5의 소프트웨어 인증 프로그램은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로서 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 프로그램으로서, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계로 이루어지는 처리를 컴퓨터에 실행시키기 위한 프로그램을 포함하는 것을 특징으로 한다.

이러한 구성이면, 컴퓨터에 의해 프로그램이 판독되고, 판독된 프로그램에 따라 컴퓨터가 처리를 실행하면, 형태 1의 소프트웨어 인증 시스템과 동등한 작용 및 효과가 얻어진다.

여기서, 로그 정보 취득 단계는, 로그 정보를 취득하면 어떠한 형태이어도 되고, 예를 들어 입력 장치 등으로부터 로그 정보를 입력해도 되며, 외부의 장치 등으로부터 로그 정보를 획득 또는 수신해도 되고, 기억 장치나 기억 매체 등으로부터 로그 정보를 독출해도 되며, 기능 모듈의 그 밖의 데이터로부터 로그 정보를 추출해도 된다. 따라서, 취득에는 적어도 입력, 획득, 수신, 독출 및 추출이 포함된다. 이하, 형태 6 및 7의 소프트웨어 인증 프로그램 및 형태 9 내지 14의 소프트웨어 인증 방법에 있어서 동일하다.

(형태 6) 또, 형태 6의 소프트웨어 인증 프로그램은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 프로그램으로서, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계로 이루어지는 처리를 컴퓨터로 실행시키기 위한 프로그램을 포함하는 것을 특징으로 한다.

이러한 구성이면, 컴퓨터에 의해 프로그램이 판독되고, 판독된 프로그램에 따라 컴퓨터가 처리를 실행하면, 형태 2의 소프트웨어 인증 시스템과 동등한 작용 및 효과가 얻어진다.

(형태 7) 또한, 형태 7의 소프트웨어 인증 프로그램은, 제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 프로그램으로서, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 리소스 사용 상황이 상기

제1 실행 환경에 적합한지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 적합하다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계로 이루어지는 처리를 컴퓨터로 실행시키기 위한 프로그램을 포함하는 것을 특징으로 한다.

이러한 구성이면, 컴퓨터에 의해 프로그램이 판독되고, 판독된 프로그램에 따라 컴퓨터가 처리를 실행하면, 형태 3의 소프트웨어 인증 시스템과 동등한 작용 및 효과가 얻어진다.

(형태 8) 또, 형태 8의 소프트웨어 인증 프로그램은, 형태 5 내지 7 중 어느 하나의 소프트웨어 인증 프로그램에 있어서, 상기 기능 모듈의 실행에 필요한 실행 파일을 취득하는 실행 파일 취득 단계와, 상기 실행 파일 취득 단계에서 취득한 실행 파일에 기초하여, 상기 기능 모듈을 구성하는 명령 또는 명령군이 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어지는지를 판정하는 제2 동작 판정 단계로 이루어지는 처리를 컴퓨터로 실행시키기 위한 프로그램을 포함하고, 상기 소프트웨어 인증 단계는, 상기 제2 동작 판정 단계에서 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어진다고 판정했을 때는, 상기 소프트웨어에 상기 인증 정보를 부가하는 것을 특징으로 한다.

이러한 구성이면, 컴퓨터에 의해 프로그램이 판독되고, 판독된 프로그램에 따라 컴퓨터가 처리를 실행하면, 형태 4의 소프트웨어 인증 시스템과 동등한 작용 및 효과가 얻어진다.

(형태 9) 한편, 상기 목적을 달성하기 위해, 형태 9의 소프트웨어 인증 방법은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 한다.

이에 의해, 형태 1의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(형태 10) 또한, 형태 10의 소프트웨어 인증 방법은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서, 연산 수단이, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 연산 수단이, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 연산 수단이, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 한다.

이에 의해, 형태 1의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(형태 11) 또, 형태 11의 소프트웨어 인증 방법은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 한다.

이에 의해, 형태 2의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(형태 12) 또한, 형태 12의 소프트웨어 인증 방법은, 기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서, 연산 수단이, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 연산 수단이, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 연산 수단이, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 한다.

이에 의해, 형태 2의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(형태 13) 또, 형태 13의 소프트웨어 인증 방법은, 제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 리소스 사용 상황이 상기 제1 실행 환경에 적합한지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 적합하다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 한다.

이에 의해, 형태 3의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(형태 14) 또한, 형태 14의 소프트웨어 인증 방법은, 제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서, 연산 수단이, 상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 연산 수단이, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여 상기 리소스 사용 상황이 상기 제1 실행 환경에 적합한지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 적합하다고 판정했을 때는, 상기 연산 수단이, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 한다.

이에 의해, 형태 3의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(형태 15) 또, 형태 15의 소프트웨어 인증 방법은, 형태 9 내지 14 중 어느 하나의 소프트웨어 인증 방법에 있어서, 상기 기능 모듈의 실행에 필요한 실행 파일을 취득하는 실행 파일 취득 단계와, 상기 실행 파일 취득 단계에서 취득한 실행 파일에 기초하여, 상기 기능 모듈을 구성하는 명령 또는 명령군이 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어지는지를 판정하는 제2 동작 판정 단계를 포함하고, 상기 소프트웨어 인증 단계는, 상기 제2 동작 판정 단계에서 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어진다고 판정했을 때는, 상기 소프트웨어에 상기 인증 정보를 부가하는 것을 특징으로 한다.

이에 의해, 형태 4의 소프트웨어 인증 시스템과 동등한 효과가 얻어진다.

(실시 형태)

이하, 본 발명의 제1 실시 형태를 도면을 참조하면서 설명한다. 도 1 내지 도 21은 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법의 제1 실시 형태를 나타내는 도면이다.

본 실시 형태는, 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 도 2에 나타내는 바와 같이, 호스트 단말(100) 상의 JAVA(등록상표) 애플리케이션의 실행 환경에 있어서, 네트워크 프린터의 동작을 제어하기 위한 JAVA(등록상표) 클래스 세트를 에뮬레이션하는 경우에 대해서 적용한 것이다.

우선, 본 발명을 적용하는 호스트 단말(100)의 기능개요를 설명한다.

도 1은 JAVA(등록상표) 소프트웨어의 구성을 나타내는 도면이다.

JAVA(등록상표) 애플리케이션의 실행 환경에서는, JAVA(등록상표) 클래스 세트의 실행을 제어하는 JAVA(등록상표) 클래스 및 JVM(Java(등록상표) Virtual Machine)으로 이루어지는 공통 기능 모듈을 OS(Operating System)상에서 실행하고, JAVA(등록상표) 클래스 세트인 개별 기능 모듈을 공통 기능 모듈 상에서 실행한다. 여기서, JAVA(등록상표) 소프트웨어는 공통 기능 모듈 및 개별 기능 모듈로 구성된다.

공통 기능 모듈은, 도 1에 나타내는 바와 같이, 복수의 개별 기능 모듈을 실행 가능하게 되어 있다. 도 1의 예에서는, 공통 기능 모듈(a) 상에서 2개의 개별 기능 모듈(b, c)이 실행되어 있는 경우를 나타낸다. 여기서, 개별 기능 모듈(b)이 사용하는 리소스의 양을 x1, 공통 기능 모듈(a)이 개별 기능 모듈(b)의 실행에 사용하는 리소스의 양을 x2, 개별 기능 모듈(b)이 사용 가능한 리소스의 상한값을 X_{max}로 한 경우, 본 실시 형태에서는 $x1+x2 \leq X_{max}$가 되도록 리소스의 양을 제한한다.

도 2는 호스트 단말(100)의 기능 개요를 나타내는 기능 블록도이다.

호스트 단말(100)은, 도 2에 나타내는 바와 같이, OS(110)와 공통 기능 모듈(120)과 복수의 개별 기능 모듈(130)과 애플리케이션 인증부(140)를 갖고 구성되어 있다.

OS(110)는, JAVA(등록상표) 소프트웨어가 사용하는 리소스의 양을 측정하는 리소스 측정부(10)와, JAVA(등록상표) 소프트웨어 전체가 사용하는 리소스의 양을 제한하는 리소스 제한부(12)를 갖고 구성되어 있다.

리소스 제한부(12)는, 리소스 측정부(10)에서 측정한 리소스의 양이 JAVA(등록상표) 소프트웨어에 할당된 소정의 상한값 미만이 되도록, JAVA(등록상표) 소프트웨어가 사용하는 리소스를 제한한다.

공통 기능 모듈(120)은, 개별 기능 모듈(130)의 실행을 관리하는 개별 기능 모듈 관리부(14)와, 개별 기능 모듈 관리부(14) 및 개별 기능 모듈(130)이 사용하는 리소스의 양을 측정하는 리소스 측정부(16)와, 호스트 단말(100)에서 소정 조건으로 사용되는 리소스의 양 및 네트워크 프린터에서 같은 소정 조건으로 사용되는 리소스의 양에 기초하여 결정되는 환산율을 등록한 리소스 환산 테이블(22)과, 리소스의 양을 환산하는 리소스 환산부(24)를 갖고 구성되어 있다.

리소스 측정부(16)는, 각 개별 기능 모듈(130)마다 그 개별 기능 모듈(130)이 사용하는 리소스의 양 및 개별 기능 모듈 관리부(14)가 그 개별 기능 모듈(130)의 실행에 사용하는 리소스의 양을 측정한다.

리소스 환산부(24)는, 리소스 환산 테이블(22)에 기초하여, 리소스 측정부(16)에서 측정한 리소스의 양을 네트워크 프린터에서 사용하는 리소스의 양으로 환산한다.

공통 기능 모듈(120)은, 또, 개별 기능 모듈(130)이 네트워크 프린터에서의 리소스의 상한값을 취득하는 상한값 취득부(18)와, 개별 기능 모듈 관리부(14) 및 개별 기능 모듈(130)이 사용하는 리소스의 양을 제한하는 리소스 제한부(20)와, 로그 정보를 생성하는 로그 정보 생성부(26)를 갖고 구성되어 있다.

리소스 제한부(20)는, 리소스 환산부(24)로 환산한 리소스의 양이 상한값 취득부(18)로 취득한 상한값 미만이 되도록, 개별 기능 모듈(130)이 사용하는 리소스의 양 및 개별 기능 모듈 관리부(14)가 그 개별 기능 모듈(130)의 실행에 사용하는 리소스의 양을 제한한다.

로그 정보 생성부(26)는, 리소스 환산부(24)로 환산한 리소스의 양이 상한값 취득부(18)로 취득한 상한값 이상이라고 판정했을 때는, 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성한다.

애플리케이션 인증부(140)는, 로그 정보 생성부(26)가 생성한 로그 정보를 취득하는 로그 정보 취득부(28)와, 로그 정보 취득부(28)로 취득한 로그 정보에 기초하여 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달했는지의 여부를 판정하는 동작 판정부(30)와, 동작 판정부(30)가 상한에 도달하고 있지 않다고 판정했을 때는 개별 기능 모듈(130)에 전자서명 정보를 부가하는 모듈 인증부(32)를 갖고 구성되어 있다.

다음에, 호스트 단말(100)의 구성을 설명한다.

도 3은 호스트 단말(100)의 하드웨어 구성을 나타내는 블록도이다.

호스트 단말(100)은, 도 3에 나타내는 바와 같이, 제어 프로그램에 기초하여 연산 및 시스템 전체를 제어하는 CPU(50)와, 소정 영역에 미리 CPU(50)의 제어 프로그램 등을 저장하고 있는 ROM(52)과, ROM(52) 등으로부터 독출한 데이터나 CPU(50)의 연산 과정에서 필요한 연산 결과를 저장하기 위한 RAM(54)과, 외부 장치에 대해 데이터의 입출력을 매개하는 I/F(58)로 구성되어 있고, 이들은 데이터를 전송하기 위한 신호선인 버스(39)에서 서로 데이터 수수(授受) 가능하게 접속되어 있다.

I/F(58)에는, 외부 장치로서, 휴먼 인터페이스로서 데이터의 입력이 가능한 키보드나 마우스 등으로 이루어지는 입력 장치(60)와, 데이터나 테이블 등을 파일로서 저장하는 기억 장치(62)와, 화상 신호에 기초하여 화면을 표시하는 표시 장치(64)와, 네트워크(199)에 접속하기 위한 신호선이 접속되어 있다.

다음에, 기억 장치(62)의 데이터 구조를 설명한다.

기억 장치(62)는, 공통 기능 모듈(120) 및 복수의 개별 기능 모듈(130)을 기억하고 있다.

개별 기능 모듈(130)은, 네트워크 프린터에서의 리소스의 상한값을 저장한 리소스 제한 정보를 포함하여 구성되어 있다.

도 4는 리소스 제한 정보(400)의 데이터 구조를 나타내는 도면이다.

리소스 제한 정보(400)는, 도 4에 나타내는 바와 같이, 개별 기능 모듈(130) 및 공통 기능 모듈(120)이 그 개별 기능 모듈(130)의 실행에 사용 가능한 메모리(RAM(54))의 상한값을 저장한 필드(402)와, 개별 기능 모듈(130) 및 공통 기능 모듈(120)이 그 개별 기능 모듈(130)의 실행에 기동 가능한 클래스수를 저장한 필드(404)를 포함하여 구성되어 있다.

개별 기능 모듈(130)은, 또, 개별 기능 모듈(130)에 관한 모듈 정보를 포함하여 구성되어 있다.

도 5는 모듈 정보(420)의 데이터 구조를 나타내는 도면이다.

모듈 정보(420)는, 도 5에 나타내는 바와 같이, 개별 기능 모듈(130)이 사용하는 리소스의 양을 제한하는 대상(이하, 리소스 관리 대상이라 함)으로 할 지의 여부를 저장한 필드(422)와, 개별 기능 모듈(130)이 실행 가능한 네트워크 프린터의 기종을 저장한 필드(424)와, 전자 서명 정보를 저장한 필드(426)를 포함하여 구성되어 있다.

도 5의 예에서, 필드(422)에는 「유효」가 저장되어 있다. 이는, 개별 기능 모듈(130)을 리소스 관리 대상으로서 관리하는 것을 나타낸다. 또, 필드(424)에는 「TypeA」가, 필드(426)에는 「X사」가 각각 저장되어 있다. 이는, 개별 기능 모듈(130)이 실행 가능한 기종이「TypeA」이고, X사의 전자 서명을 받고 있는 것을 나타낸다.

기억 장치(62)는, 또, 공통 기능 모듈(120)의 실행 환경을 나타내는 실행 환경 정보를 등록한 실행 환경 정보 등록 테이블을 기억하고 있다.

도 6은 실행 환경 정보 등록 테이블(440)의 데이터 구조를 나타내는 도면이다.

실행 환경 정보 등록 테이블(440)은, 도 6에 나타내는 바와 같이, 개별 기능 모듈(130)의 기동 가능한 개수의 상한값을 등록한 필드(442)와, 실행해야 할 개별 기능 모듈(130)의 명칭을 등록한 필드(444)와, 삭제해야 할 개별 기능 모듈(130)의 명칭을 등록한 필드(446)와, 에뮬레이션하는 네트워크 프린터의 기종을 등록한 필드(448)와, 대응 가능한 전자 서명 정보를 등록한 필드(450)를 포함하여 구성되어 있다.

도 6의 예에서, 필드(442)에는 「5」가, 필드(444)에는 「개별 기능 모듈(b, d)」이, 필드(446)에는 「개별 기능 모듈(c)」가 각각 등록되어 있다. 이는, 개별 기능 모듈(130)을 최대 5개까지 기동할 수 있고, 공통 기능 모듈(120)의 기동 시에 개별 기능 모듈(b, d)을 실행하며, 개별 기능 모듈(c)을 삭제해야 함을 나타낸다. 또, 필드(448)에는 「TypeA」가, 필드(450)에는 「X사」가 각각 등록되어 있다. 이는, 에뮬레이션하는 네트워크 프린터의 기종이 「TypeA」이고, X사의 전자 서명 정보를 포함하는 개별 기능 모듈(130)을 실행할 수 있음을 나타낸다.

기억 장치(62)는, 또, 리소스 환산 테이블(22)을 기억하고 있다.

도 7은 리소스 환산 테이블(22)의 데이터 구조를 나타내는 도면이다.

리소스 환산 테이블(22)에는, 도 7에 나타내는 바와 같이, 리소스의 종별 또는 사용 형태별로 하나의 레코드가 등록되어 있다. 각 레코드는, 리소스의 명칭을 등록하는 필드(502)와, 환산율을 등록하는 필드(504)를 포함하여 구성되어 있다.

도 7의 예에서, 제1 단째의 레코드에는, 리소스의 명칭으로서 「메모리 사용 형태A」가, 환산율로서 「1」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 메모리를 사용 형태 A에서 사용하는 경우는, 개별 기능 모듈(130)이 호스트 단말(100)에서 사용하는 메모리량에 환산율「1」을 승산함으로써, 네트워크 프린터에서 사용하는 메모리량으로의 환산을 행하는 것을 나타낸다. 마찬가지로, 개별 기능 모듈(130)이 메모리를 사용 형태 B, C에서 사용하는 경우는, 사용 형태 B, C에 대응하는 환산율을 채용한다.

메모리의 사용 형태 A～C는, 공통 기능 모듈(120) 및 개별 기능 모듈(130)이 사용하는 함수나 라이브러리에 의해 결정된다. 예를 들어, 정수형의 변수를 다루는 라이브러리 등을 사용하는 경우는 사용 형태 A가 되고, 더블형의 변수를 다루는 라이브러리 등을 사용하는 경우는 사용 형태 B가 된다.

또한, 제4 단째의 레코드에는, 리소스의 명칭으로서「클래스수」가, 환산율로서 「1」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 호스트 단말(100)에 기동하는 클래스수에 환산율 「1」을 승산함으로써, 네트워크 프린터에서 기동하는 클래스수로의 환산을 행하는 것을 나타낸다.

또, 리소스 환산 테이블(22)에는, 복수의 테스트 모듈의 각각에 대해 호스트 단말(100) 및 네트워크 프린터에서 그 테스트 모듈이 사용하는 리소스의 양에 기초하여 환산율을 결정하고, 각 테스트 모듈에 대해서 결정된 환산율 중 최대값이 등록되어 있다.

기억 장치(62)는, 또, 리소스 관리 대상이 되는 각 개별 기능 모듈(130)마다 그 개별 기능 모듈(130)이 사용하는 리소스의 양을 관리하는 리소스 관리 테이블을 기억하고 있다. 리소스 관리 테이블은, 개별 기능 모듈(130)이 리소스 관리 대상인 경우에 그 기동에 따라 생성된다.

도 8은 리소스 관리 테이블(460)의 데이터 구조를 나타내는 도면이다.

리소스 관리 테이블(460)에는, 도 8에 나타내는 바와 같이, 리소스의 종별마다 하나의 레코드가 등록되어 있다. 각 레코드는, 리소스의 명칭을 등록하는 필드(462)와, 개별 기능 모듈(130)이 네트워크 프린터에서의 리소스의 상한값을 등록하는 필드(464)와, 개별 기능 모듈(130)이 호스트 단말(100)에서 현재 사용하고 있는 리소스의 양을 등록하는 필드(466)와, 필드(466)의 값을 네트워크 프린터에서 사용하는 리소스의 양으로 환산한 값을 등록하는 필드(468)를 포함하여 등록되어 있다.

도 8의 예에서, 제1 단째의 레코드에는, 리소스의 명칭으로서 「메모리」가, 상한값으로서 「1000000」이, 현재값으로서 메모리의 사용 형태 A～C마다 「200000」, 「100000」 및 「150000」이, 환산값으로서 「650000」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 네트워크 프린터에서의 메모리량의 상한값이 1000000[byte]이고, 현재 메모리를 사용 형태 A에서 200000[byte], 사용 형태 B에서 100000[byte], 사용 형태C에서 150000[byte] 사용하고 있음을 나타낸다. 또, 네트워크 프린터에서 사용하는 메모리량으로 환산한 값(이하, 환산 메모리량이라 함)이 650000[byte]임을 나타낸다. 환산 메모리량은, 도 7의 리소스 환산 테이블(22)을 참조하여, 200000×1+ 100000×1.5+ 150000×2=650000으로서 산출할 수 있다.

또한, 제2 단째의 레코드에는, 리소스의 명칭으로서 「클래스수」가, 상한값으로서 「100」이, 현재값으로서 「20」이, 환산값으로서 「20」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 네트워크 프린터에서 기동 가능한 클래스수의 상한값이 100개이고, 현재 클래스를 20개 기동하고 있음을 나타낸다. 또, 네트워크 프린터에서 기동하는 클래스수로 환산한 값(이하, 환산 클래스수라 함)이 20개임을 나타낸다. 환산 클래스수는, 도 7의 리소스 환산 테이블(22)을 참조하여, 20×1=20으로서 산출할 수 있다.

기억 장치(62)는, 또, 개별 기능 모듈(130)이 수신하는 이벤트를 처리하는 이벤트 리스너를 등록하는 이벤트 리스너 테이블(480)을 기억하고 있다.

도 9는 이벤트 리스너 테이블(480)의 데이터 구조를 나타내는 도면이다.

이벤트 리스너 테이블(480)에는, 도 9에 나타내는 바와 같이, 개별 기능 모듈(130)이 등록하는 이벤트 리스너마다 레코드가 등록된다. 각 레코드는, 이벤트 리스너의 명칭을 등록하는 필드(482)를 포함하여 등록되어 있다.

도 3으로 되돌아가서, CPU(50)는, 마이크로 프로세싱 유닛 등으로 이루어지고, ROM(52)의 소정 영역에 저장되어 있는 소정의 프로그램을 기동시키며, 그 프로그램에 따라 도 10, 도 14, 도 15, 도 17 및 도 18의 플로우차트에 나타내는 개별 기능 모듈 제어 처리, 클래스 판독 처리, 이벤트 리스너 제어 처리, 인스턴스 삭제 처리 및 모듈 인증 처리를 공통 기능 모듈(120)의 처리로서 각각 시분할로 실행한다.

처음에, 개별 기능 모듈 제어 처리를 설명한다.

도 10은 개별 기능 모듈 제어 처리를 나타내는 플로우차트이다.

개별 기능 모듈 제어 처리는, 개별 기능 모듈(130)의 삭제 및 실행을 제어하는 처리로서, CPU(50)에서 실행되면, 도 10에 나타내는 바와 같이, 우선 단계 S100으로 이행한다.

단계 S100에서는, 실행해야 할 개별 기능 모듈(130)의 명칭 및 삭제해야 할 개별 기능 모듈(130)의 명칭을 실행 환경 정보 등록 테이블(440)로부터 취득하고, 단계 S102로 이행하여 삭제해야 할 개별 기능 모듈(130)이 존재하는지의 여부를 판정하며, 삭제해야 할 개별 기능 모듈(130)이 존재한다고 판정했을 때(Yes)는 단계 S104로 이행한다.

단계 S104에서는, 취득한 명칭에 기초하여 해당 개별 기능 모듈(130)을 기억 장치(62)로부터 삭제하고, 단계 S106으로 이행하여 해당 개별 기능 모듈(130)에 포함되는 모듈 정보(420)에 기초하여 해당 개별 기능 모듈(130)이 리소스 관리 대상인지의 여부를 판정하며, 해당 개별 기능 모듈(130)이 리소스 관리 대상이라고 판정했을 때(Yes)는 단계 S108로 이행한다.

단계 S108에서는, 해당 개별 기능 모듈(130)에 대응하는 리소스 관리 테이블(460)을 기억 장치(62)로부터 삭제하고, 단계 S110으로 이행하여 현재 기동 중인 모듈수를 나타내는 변수의 값에서 「1」을 감산하며, 단계 S102로 이행한다.

한편, 단계 S106에서 해당 개별 기능 모듈(130)이 리소스 관리 대상이 아니라고 판정했을 때(No)는, 단계 S102로 이행한다.

한편, 단계 S102에서 삭제해야 할 개별 기능 모듈(130)이 존재하지 않는다고 판정했을 때(No)는, 단계 S112로 이행하여 실행해야 할 개별 기능 모듈(130)이 존재하는지의 여부를 판정하고, 실행해야 할 개별 기능 모듈(130)이 존재한다고 판정했을 때(Yes)는 단계 S114로 이행한다.

단계 S114에서는, 현재 기동 중인 모듈수를 나타내는 변수의 값이 소정의 상한값 미만인지의 여부를 판정하고, 소정의 상한값 미만이라고 판정했을 때(Yes)는 단계 S116으로 이행한다.

단계 S116에서는, 취득한 명칭에 기초하여 해당 개별 기능 모듈(130)을 기억 장치(62)로부터 판독하고, 단계 S118로 이행하여 판독한 개별 기능 모듈(130)의 실행 여부를 판정하는 실행 여부 판정 처리를 실행하며, 단계 S120으로 이행한다.

단계 S120에서는, 실행 여부 판정 처리로부터 개별 기능 모듈(130)의 실행을 허가하는 것을 나타내는 반환값이 반환되었는지의 여부를 판정하고, 실행을 허가하는 것을 나타내는 반환값이 반환되었다고 판정했을 때(Yes)는 단계 S122로 이행한다.

단계 S122에서는, 해당 개별 기능 모듈(130)에 포함되는 모듈 정보(420)에 기초하여 해당 개별 기능 모듈(130)이 리소스 관리 대상인지의 여부를 판정하고, 해당 개별 기능 모듈(130)이 리소스 관리 대상이라고 판정했을 때(Yes)는 단계 S124로 이행한다.

단계 S124에서는, 해당 개별 기능 모듈(130)에 대응하는 리소스 관리 테이블(460)을 생성하고, 해당 개별 기능 모듈(130)에 포함되는 리소스 제한 정보(400)로부터 상한값을 취득하며, 취득한 상한값을 생성한 리소스 관리 테이블(460)에 등록하고, 단계 S126으로 이행하여 현재 기동 중인 모듈수를 나타내는 변수의 값에 「1」을 가산하며, 단계 S128로 이행한다.

단계 S128에서는, 생성한 리소스 관리 테이블(460)의 어드레스를 리소스 확보처의 참조 포인터로 설정하고, 단계 S130으로 이행하여 해당 개별 기능 모듈(130)을 기동하는 모듈 기동 처리를 실행하며, 단계 S132로 이행하여 리소스 확보처의 참조 포인터를 클리어하고, 단계 S112로 이행한다.

한편, 단계 S122에서 해당 개별 기능 모듈(130)이 리소스 관리 대상이 아니라고 판정했을 때(No)는, 단계 S134로 이행하여 단계 S130과 같은 모듈 기동 처리를 실행하고, 단계 S112로 이행한다.

한편, 단계 S120에서 실행 여부 판정 처리로부터 개별 기능 모듈(130)의 실행을 허가하지 않는 것을 나타내는 반환값이 반환되었다고 판정했을 때(No)는, 단계 S112로 이행한다.

한편, 단계 S114에서 현재 기동 중인 모듈수를 나타내는 변수의 값이 소정의 상한값 이상이라고 판정했을 때(No)는, 단계 S136으로 이행하여 모듈수가 상한에 도달한 것을 나타내는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S112에서 실행해야 할 개별 기능 모듈(130)이 존재하지 않는다고 판정했을 때(No)는, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 단계 S118의 실행 여부 판정 처리를 설명한다.

도 11은 실행 여부 판정 처리를 나타내는 플로우차트이다.

실행 여부 판정 처리는, 단계 S118에서 실행되면, 도 11에 나타내는 바와 같이, 우선 단계 S200으로 이행한다.

단계 S200에서는, 개별 기능 모듈(130)에 포함되는 모듈 정보(420)로부터 기종 정보를 취득하고, 단계 S202로 이행하여 취득한 기종 정보와 실행 환경 정보 등록 테이블(440)의 기종 정보가 일치하는지의 여부를 판정하며, 그들 기종 정보가 일치한다고 판정했을 때(Yes)는 단계 S204로 이행한다.

단계 S204에서는, 개별 기능 모듈(130)에 포함되는 모듈 정보(420)로부터 전자 서명 정보를 취득하고, 단계 S206으로 이행하여 실행 환경 정보 등록 테이블(440)에 기초하여 취득한 전자 서명 정보가 대응가능한 것인지의 여부를 판정하며, 대응가능한 전자 서명 정보라고 판정했을 때(Yes)는 단계 S208로 이행한다.

단계 S208에서는, 개별 기능 모듈(130)에 포함되는 모듈 정보(420)에 기초하여 개별 기능 모듈(130)이 리소스 관리 대상인지의 여부를 판정하고, 리소스 관리 대상이라고 판정했을 때(Yes)는 단계 S209로 이행한다.

단계 S209에서는, 실행 환경 정보 등록 테이블(440)에 대응하는 리소스 환산 테이블(22)이 존재하는지의 여부를 판정하고, 대응의 리소스 환산 테이블(22)이 존재한다고 판정했을 때(Yes)는, 단계 S210으로 이행하여 대응의 리소스 환산 테이블(22)을 기억 장치(62)로부터 판독하며, 단계 S211로 이행한다.

단계 S211에서는, 개별 기능 모듈(130)에 포함되는 리소스 제한 정보(400)로부터 상한값을 취득하고, 단계 S212로 이행하여 상한값의 취득에 성공했는지의 여부를 판정하며, 상한값의 취득에 성공했다고 판정했을 때(Yes)는 단계 S214로 이행한다.

단계 S214에서는, 취득한 상한값이 총 메모리 잔량 미만인지의 여부를 판정하고, 총 메모리 잔량 미만이라고 판정했을 때(Yes)는, 단계 S216으로 이행하여 개별 기능 모듈(130)의 실행을 허가하는 것을 나타내는 반환값을 반환하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S214에서 취득한 상한값이 총 메모리 잔량 이상이라고 판정했을 때(No)는, 단계 S217로 이행하여 상한값이 총 메모리 잔량 이상임을 나타내는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 단계 S218로 이행하여 개별 기능 모듈(130)의 실행을 허가하지 않는 것을 나타내는 반환값을 반환하고, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S212에서 상한값의 취득에 실패했다고 판정했을 때(No), 단계 S

206에서 대응가능한 전자 서명 정보가 아니라고 판정했을 때(No) 및 단계 S202에서 기종 정보가 일치하지 않는다고 판정했을 때(No)는, 모두 단계 S218로 이행한다.

한편, 단계 S209에서 실행 환경 정보 등록 테이블(440)에 대응하는 리소스 환산 테이블(22)이 존재하지 않는다고 판정했을 때(No)는, 단계 S211로 이행한다.

한편, 단계 S208에서 개별 기능 모듈(130)이 리소스 관리 대상이 아니라고 판정했을 때(No)는, 단계 S216으로 이행한다.

다음에, 단계 S130, S134의 모듈 기동 처리를 설명한다.

도 12는 모듈 기동 처리를 나타내는 플로우차트이다.

모듈 기동 처리는, 단계 S130, S134에서 실행되면, 도 12에 나타내는 바와 같이, 우선 단계 S300으로 이행한다.

단계 S300에서는, 개별 기능 모듈(130)로부터 클래스를 판독해야 할 클래스판독 명령을 출력하고, 단계 S302로 이행하여 클래스의 판독에 성공했는지의 여부를 판정하며, 클래스의 판독에 성공했다고 판정했을 때(Yes)는 단계 S304로 이행한다.

단계 S304에서는, 리소스 확보처의 참조 포인터가 설정되어 있는지의 여부를 판정하고, 리소스 확보처의 참조 포인터가 설정되어 있다고 판정했을 때(Yes)는, 단계 S305로 이행하여 판독한 클래스의 실행에 필요한 메모리량을 산출하며, 단계 S306으로 이행한다.

단계 S306에서는, 판독한 클래스가 사용하는 라이브러리 등으로부터 메모리의 사용 형태를 특정하고, 특정한 메모리의 사용 형태에 대응하는 환산율을 판독한 리소스 환산 테이블(22)로부터 취득하며, 산출한 메모리량에 취득한 환산율을 승산함으로써, 네트워크 프린터에서 사용하는 메모리량으로의 환산을 행한다.

다음에, 단계 S307로 이행하여 리소스 확보처의 참조 포인터가 지시하는 리소스 관리 테이블(460)(이하, 참조 리소스 관리 테이블(460)이라 함)의 사용 메모리량에 환산 메모리량을 가산하고, 단계 S308로 이행하여 가산한 합계의 메모리량이 참조 리소스 관리 테이블(460)의 상한값 미만인지의 여부를 판정하며, 상한값 미만이라고 판정했을 때(Yes)는 단계 S310으로 이행한다.

단계 S310에서는, 판독한 클래스의 인스턴스를 메모리 상에 생성하고, 단계 S312로 이행하여 리소스 확보처의 참조 포인터의 값을 나타내는 리소스 확보처 참조 정보를 생성한 인스턴스에 저장하며, 단계 S313으로 이행하여 환산 메모리량을 생성한 인스턴스에 저장하고, 단계 S314로 이행한다.

단계 S314에서는, 인스턴스의 생성에 성공했는지의 여부를 판정하고, 인스턴스의 생성에 성공했다고 판정했을 때(Yes)는, 단계 S316으로 이행하여 판독한 클래스의 기능을 호출하는 클래스기능 호출 처리를 실행하며, 단계 S318로 이행하여 개별 기능 모듈(130)의 이벤트 리스너를 등록하는 이벤트 리스너 등록 처리를 실행하고, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S308에서 합계의 메모리량이 상한값 이상이라고 판정했을 때(No)는, 단계 S320으로 이행하여 참조 리소스 관리 테이블(460)의 사용 메모리량에서 환산 메모리량을 감산하고, 단계 S321로 이행한다.

단계 S321에서는, 개별 기능 모듈(130)이 사용하는 메모리량이 상한에 도달한 것을 나타내는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 단계 S322로 이행하여 에러를 통지하고, 단계 S314로 이행한다.

한편, 단계 S304에서 리소스 확보처의 참조 포인터가 설정되어 있지 않다고 판정했을 때(No)는, 단계 S324로 이행하여 판독한 클래스의 인스턴스를 메모리 상에 생성하고, 단계 S314로 이행한다.

한편, 단계 S314에서 인스턴스의 생성에 실패했다고 판정했을 때(No) 및 단계 S302에서 클래스의 판독에 실패했다고 판정했을 때(No)는, 모두 단계 S318로 이행한다.

다음에, 단계 S318의 이벤트 리스너 등록 처리를 설명한다.

도 13은 이벤트 리스너 등록 처리를 나타내는 플로우차트이다.

이벤트 리스너 등록 처리는, 단계 S318에서 실행되면, 도 13에 나타내는 바와 같이, 우선 단계 S400으로 이행한다.

단계 S400에서는, 개별 기능 모듈(130)로부터 이벤트 리스너 클래스를 판독해야 할 클래스 판독 명령을 출력하고, 단계 S402로 이행하여 이벤트 리스너 클래스의 판독에 성공했는지의 여부를 판정하며, 이벤트 리스너 클래스의 판독에 성공했다고 판정했을 때(Yes)는 단계 S404로 이행한다.

단계 S404에서는, 리소스 확보처의 참조 포인터가 설정되어 있는지의 여부를 판정하고, 리소스 확보처의 참조 포인터가 설정되어 있다고 판정했을 때(Yes)는, 단계 S405로 이행하여 판독한 이벤트 리스너 클래스의 실행에 필요한 메모리량을 산출하며, 단계 S406으로 이행한다.

단계 S406에서는, 판독한 이벤트 리스너 클래스가 사용하는 라이브러리 등으로부터 메모리의 사용 형태를 특정하고, 특정한 메모리의 사용 형태에 대응하는 환산율을 판독한 리소스 환산 테이블(22)로부터 취득하며, 산출한 메모리량에 취득한 환산율을 승산함으로써, 네트워크 프린터에서 사용하는 메모리량으로의 환산을 행한다.

다음에, 단계 S407로 이행하여 참조 리소스 관리 테이블(460)의 사용 메모리량에 환산 메모리량을 가산하고, 단계 S408로 이행하여 가산한 합계의 메모리량이 참조 리소스 관리 테이블(460)의 상한값 미만인지의 여부를 판정하며, 상한값 미만이라고 판정했을 때(Yes)는 단계 S410으로 이행한다.

단계 S410에서는, 판독한 이벤트 리스너 클래스의 인스턴스를 메모리 상에 생성하고, 단계 S412로 이행하여 리소스 확보처의 참조 포인터의 값을 나타내는 리소스 확보처 참조 정보를 생성한 인스턴스에 저장하며, 단계 S413으로 이행하여 환산 메모리량을 생성한 인스턴스에 저장하고, 단계 S414로 이행한다.

단계 S414에서는, 인스턴스의 생성에 성공했는지의 여부를 판정하고, 인스턴스의 생성에 성공했다고 판정했을 때(Yes)는, 단계 S416으로 이행하여 생성한 인스턴스의 이벤트 리스너를 이벤트 리스너 실행 리스트에 등록하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S408에서 합계의 메모리량이 상한값 이상이라고 판정했을 때(No)는, 단계 S418로 이행하여 참조 리소스 관리 테이블(460)의 사용 메모리량에서 환산 메모리량을 감산하고, 단계 S419로 이행한다.

단계 S419에서는, 개별 기능 모듈(130)이 사용하는 메모리량이 상한에 도달한 것을 나타내는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 단계 S420으로 이행하여 에러를 통지하고, 단계 S414로 이행한다.

한편, 단계 S404에서 리소스 확보처의 참조 포인터가 설정되어 있지 않다고 판정했을 때(No)는, 단계 S422로 이행하여 판독한 이벤트 리스너 클래스의 인스턴스를 메모리 상에 생성하고, 단계 S414로 이행한다.

한편, 단계 S414에서 인스턴스의 생성에 실패했다고 판정했을 때(No) 및 단계 S402에서 이벤트 리스너 클래스의 판독에 실패했다고 판정했을 때(No)는, 모두 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 클래스 판독 처리를 설명한다.

도 14는 클래스 판독 처리를 나타내는 플로우차트이다.

클래스 판독 처리는, 클래스 판독 명령에 따라 클래스를 판독하는 처리로서, CPU(50)에서 실행되면, 도 14에 나타내는 바와 같이, 우선 단계 S500으로 이행한다.

단계 S500에서는, 클래스 판독 명령을 취득했는지의 여부를 판정하고, 클래스 판독 명령을 취득했다고 판정했을 때(Yes)는 단계 S502로 이행하지만, 그렇지 않다고 판정했을 때(No)는 클래스 판독 명령을 취득할 때까지 단계 S500에서 대기한다.

단계 S502에서는, 클래스 판독 명령에 관한 클래스가 캐쉬 테이블에 등록되어 있는지의 여부를 판정하고, 캐쉬 테이블에 등록되어 있지 않다고 판정했을 때(No)는 단계 S504로 이행한다.

단계 S504에서는, 클래스 판독 명령에 관한 클래스가 속하는 개별 기능 모듈(130)을 특정하고, 단계 S506으로 이행하여 특정한 해당 개별 기능 모듈(130)에 포함되는 모듈 정보(420)에 기초하여 해당 개별 기능 모듈(130)이 리소스 관리 대상인지의 여부를 판정하며, 리소스 관리 대상이라고 판정했을 때(Yes)는 단계 S508로 이행한다.

단계 S508에서는, 해당 개별 기능 모듈(130)에 대응하는 리소스 관리 테이블(460)의 어드레스를 리소스 확보처의 참조 포인터로 설정하고, 단계 S509로 이행한다.

단계 S509에서는, 클래스수에 대응하는 환산율을 판독한 리소스 환산 테이블(22)로부터 취득하고, 클래스 판독 명령에 관한 클래스의 수 「1」에 취득한 환산율을 승산함으로써, 네트워크 프린터에서 기동하는 클래스수로의 환산을 행한다.

다음에, 단계 S510으로 이행하여 참조 리소스 관리 테이블(460)의 기동 클래스수에 환산 클래스수를 가산하고, 단계 S512로 이행하여 가산한 합계의 클래스수가 참조 리소스 관리 테이블(460)의 상한값 미만인지의 여부를 판정하며, 상한값 미만이라고 판정했을 때(Yes)는 단계 S514로 이행한다.

단계 S514에서는, 클래스 판독 명령에 관한 클래스를 개별 기능 모듈(130)로부터 판독하고, 단계 S516으로 이행하여 판독한 클래스를 캐쉬 테이블에 등록하며, 단계 S517로 이행한다.

단계 S517에서는, 환산 클래스수를 판독한 클래스에 저장하고, 단계 S518로 이행하여 리소스 확보처의 참조 포인터를 클리어하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S512에서 합계의 클래스수가 상한값 이상이라고 판정했을 때(No)는, 단계 S520으로 이행하여 참조 리소스 관리 테이블(460)의 기동 클래스수에서 환산 클래스수를 감산하고, 단계 S521로 이행한다.

단계 S521에서는, 개별 기능 모듈(130)이 기동하는 클래스수가 상한에 도달한 것을 나타내는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 단계 S522로 이행하여 에러를 통지하고, 단계 S518로 이행한다.

한편, 단계 S506에서 해당 개별 기능 모듈(130)이 리소스 관리 대상이 아니라고 판정했을 때(No)는, 단계 S524로 이행하여 클래스 판독 명령에 관한 클래스를 개별 기능 모듈(130)로부터 판독하고, 단계 S526으로 이행하여 판독한 클래스를 캐쉬 테이블에 등록하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S502에서 클래스 판독 명령에 관한 클래스가 캐쉬 테이블에 등록되어 있다고 판정했을 때(Yes)는, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 이벤트 리스너 제어 처리를 설명한다.

도 15는 이벤트 리스너 제어 처리를 나타내는 플로우차트이다.

이벤트 리스너 제어 처리는, 이벤트 리스너의 실행을 제어하는 처리로서, CPU(50)에서 실행되면, 도 15에 나타내는 바와 같이, 우선 단계 S600으로 이행한다.

단계 S600에서는, 이벤트 리스너 실행 리스트를 취득하고, 단계 S602로 이행하여 취득한 이벤트 리스너 실행 리스트에 기초하여 실행해야 할 이벤트 리스너가 존재하는지의 여부를 판정하며, 실행해야 할 이벤트 리스너가 존재한다고 판정했을 때(Yes)는 단계 S604로 이행한다.

단계 S604에서는, 해당 이벤트 리스너의 생성원이 된 개별 기능 모듈(130)을 특정하고, 단계 S606으로 이행하여 특정한 해당 개별 기능 모듈(130)에 포함되는 모듈 정보(420)에 기초하여 해당 개별 기능 모듈(130)이 리소스 관리 대상인지의 여부를 판정하며, 리소스 관리 대상이라고 판정했을 때(Yes)는 단계 S608로 이행한다.

단계 S608에서는, 해당 개별 기능 모듈(130)에 대응하는 리소스 관리 테이블(460)의 어드레스를 리소스 확보처의 참조 포인터로 설정하고, 단계 S610으로 이행하여 해당 이벤트 리스너를 실행하는 이벤트 리스너 실행 처리를 실행하며, 단계 S612로 이행하여 리소스 확보처의 참조 포인터를 클리어하고, 단계 S614로 이행한다.

단계 S614에서는, 해당 이벤트 리스너를 이벤트 리스너 실행 리스트로부터 삭제하고, 단계 S602로 이행한다.

한편, 단계 S606에서, 해당 개별 기능 모듈(130)이 리소스 관리 대상이 아니라고 판정했을 때(No)는 단계 S616으로 이행하여 해당 이벤트 리스너를 실행하고, 단계 S614로 이행한다.

한편, 단계 S602에서 실행해야 할 이벤트 리스너가 존재하지 않는다고 판정했을 때(No)는, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 단계 S610의 이벤트 리스너 실행 처리를 설명한다.

도 16은 이벤트 리스너 실행 처리를 나타내는 플로우차트이다.

이벤트 리스너 실행 처리는, 단계 S610에서 실행되면, 도 16에 나타내는 바와 같이, 우선 단계 S700으로 이행한다.

단계 S700에서는, 이벤트 리스너에 포함되는 명령 리스트의 선두로 프로그램 포인터를 이동하고, 단계 S702로 이행하여 프로그램 포인터가 지시하는 어드레스에 실행해야 할 명령이 존재하는지의 여부를 판정하며, 실행해야 할 명령이 존재한다고 판정했을 때(Yes)는 단계 S703으로 이행하여 명령의 실행에 필요한 메모리량을 산출하고, 단계 S704로 이행한다.

단계 S704에서는, 명령의 실행에 사용하는 라이브러리 등으로부터 메모리의 사용 형태를 특정하고, 특정한 메모리의 사용 형태에 대응하는 환산율을 판독한 리소스 환산 테이블(22)로부터 취득하며, 산출한 메모리량에 취득한 환산율을 승산함으로써, 네트워크 프린터에서 사용하는 메모리량으로의 환산을 행한다.

다음에, 단계 S705로 이행하여 참조 리소스 관리 테이블(460)의 사용 메모리량에 환산 메모리량을 가산하고, 단계 S706으로 이행하여 가산한 합계의 메모리량이 참조 리소스 관리 테이블(460)의 상한값 미만인지의 여부를 판정하며, 상한값 미만이라고 판정했을 때(Yes)는 단계 S708로 이행한다.

단계 S708에서는 메모리를 확보하고, 단계 S710으로 이행하여 프로그램 포인터가 지시하는 어드레스의 명령을 실행하며, 단계 S712로 이행하여 이벤트 리스너에 포함되는 명령 리스트의 다음으로 프로그램 포인터를 이동하고, 단계 S713으로 이행하여 소정의 대기 시간만큼 처리를 대기하며, 단계 S702로 이행한다.

한편, 단계 S706에서 합계의 메모리량이 상한값 이상이라고 판정했을 때(No)는, 단계 S714로 이행하여 참조 리소스 관리 테이블(460)의 사용 메모리량에서 환산 메모리량을 감산하고, 단계 S715로 이행한다.

단계 S715에서는, 개별 기능 모듈(130)이 사용하는 메모리량이 상한에 도달한 것을 나타내는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 단계 S716으로 이행하여 에러를 통지하고, 단계 S712로 이행한다.

한편, 단계 S702에서 실행해야 할 명령이 존재하지 않는다고 판정했을 때(No)는, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 인스턴스 삭제 처리를 설명한다.

도 17은 인스턴스 삭제 처리를 나타내는 플로우차트이다.

인스턴스 삭제 처리는, 인스턴스를 삭제하는 처리로서, CPU(50)에서 실행되면, 도 17에 나타내는 바와 같이, 우선 단계 S800으로 이행한다.

단계 S800에서는, 삭제해야 할 인스턴스를 등록한 인스턴스 삭제 리스트를 취득하고, 단계 S802로 이행하여 취득한 인스턴스 삭제 리스트에 기초하여 삭제해야 할 인스턴스가 존재하는지의 여부를 판정하며, 삭제해야 할 인스턴스가 존재한다고 판정했을 때(Yes)는 단계 S804로 이행한다.

단계 S804에서는, 해당 인스턴스로부터 리소스 확보처 참조 정보를 취득하고, 단계 S805로 이행하여 리소스 확보처 참조 정보의 취득에 성공했는지의 여부를 판정하며, 리소스 확보처 참조 정보의 취득에 성공했다고 판정했을 때(Yes)는 단계 S806으로 이행한다.

단계 S806에서는, 취득한 리소스 확보처 참조 정보에 기초하여 리소스 확보처의 참조 포인터를 설정하고, 단계 S808로 이행하여 해당 인스턴스를 삭제하며, 단계 S810으로 이행하여 해당 인스턴스의 실행에 필요한 메모리량을 참조 리소스 관리 테이블(460)의 사용 메모리량에서 감산하고, 단계 S812로 이행한다.

단계 S812에서는 리소스 확보처의 참조 포인터를 클리어하고, 단계 S814로 이행하여 해당 인스턴스를 인스턴스 삭제 리스트로부터 삭제하며, 단계 S802로 이행한다.

한편, 단계 S805에서 리소스 확보처 참조 정보의 취득에 실패했다고 판정했을 때(No)는, 단계 S816으로 이행하여 해당 인스턴스를 삭제하고, 단계 S814로 이행한다.

한편, 단계 S802에서 삭제해야 할 인스턴스가 존재하지 않는다고 판정했을 때(No)는, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 모듈 인증 처리를 설명한다.

도 18은 모듈 인증 처리를 나타내는 플로우차트이다.

모듈 인증 처리는, CPU(50)에서 실행되면, 도 18에 나타내는 바와 같이, 우선 단계 S900으로 이행한다.

단계 S900에서는 기억 장치(62)의 로그 파일로부터 로그 정보를 독출하고, 단계 S902로 이행하여 독출한 로그 정보에 기초하여 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달했는지의 여부를 판정하며, 기동 클래스수 또는 사용 메모리량이 상한에 도달하고 있지 않다고 판정했을 때(No)는 단계 S904로 이행한다.

단계 S904에서는, 독출한 로그 정보에 기초하여 개별 기능 모듈(130)이 네트워크 프린터에 인스톨 불가능한지의 여부를 판정하고, 개별 기능 모듈(130)이 인스톨 가능하다고 판정했을 때(No)는 단계 S906으로 이행한다.

단계 S906에서는, 기억 장치(62)의 로그 파일에 미처리의 로그 정보가 존재하는지의 여부를 판정하고, 미처리의 로그 정보가 존재하지 않는다고 판정했을 때(No)는, 단계 S908로 이행하여 개별 기능 모듈(130)의 실행 파일을 기억 장치(62)로부터 독출하며, 단계 S910으로 이행한다.

단계 S910에서는 독출한 실행 파일에 전자 서명 정보를 부가하고, 단계 S912로 이행하여 전자 서명 정보를 부가한 실행 파일을 기억 장치(62)에 저장하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S906에서 미처리의 로그 정보가 존재한다고 판정했을 때(Yes)는, 단계 S900으로 이행한다.

한편, 단계 S904에서 개별 기능 모듈(130)이 인스톨 불가능하다고 판정했을 때(Yes) 및 단계 S902에서 기동 클래스수 또는 사용 메모리량이 상한에 도달했다고 판정했을 때(Yes)는, 모두 단계 S914로 이행하여 개별 기능 모듈(130)이 인증 불가능한 것을 나타내는 메시지를 표시 장치(64)에 표시하고, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 본 실시 형태의 동작을 설명한다.

처음에, 리소스 관리 대상이 되는 개별 기능 모듈(130)을 실행하는 경우를 설명한다.

호스트 단말(100)에서는, 공통 기능 모듈(120)의 실행에 의해 개별 기능 모듈 제어 처리가 실행된다. 개별 기능 모듈 제어 처리에서는, 단계 S102~S110을 거쳐 삭제해야 할 개별 기능 모듈(130)이 존재하는 경우는, 그 개별 기능 모듈(130)이 삭

제된다. 다음에, 단계 S114를 거쳐 현재 기동 중인 모듈수가 소정의 상한값 미만이라고 판정되면, 단계 S116, S118을 거쳐 해당 개별 기능 모듈(130)이 판독되고, 판독된 개별 기능 모듈(130)의 실행 여부가 판정된다. 실행 여부 판정 처리에서는, 개별 기능 모듈(130)에 대해 일치하는 기능 정보 및 대응 가능한 전자 서명 정보를 가지면서, 사용 가능한 메모리량의 상한값이 총 메모리 잔량 미만인 경우에 실행이 허가된다.

개별 기능 모듈(130)의 실행이 허가되면, 단계 S124～S128을 거쳐 리소스 관리 테이블(460)이 생성되고, 현재 기동 중인 모듈수가 「1」 가산되며, 개별 기능 모듈(130)이 기동한다. 모듈 기동 처리에서는, 단계 S509, S510, S305～S307을 거쳐 개별 기능 모듈(130)의 기동 클래스수 및 사용 메모리량이 네트워크 프린터에서의 기동 클래스수 및 사용 메모리량으로 환산되어 가산된다. 이때, 기동 클래스수 및 사용 메모리량 중 어느 하나가 상한값 이상이 되면, 단계 S521, S522 또는 단계 S321, S322를 거쳐 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 생성되는 동시에 에러가 통지되고, 클래스의 판독 또는 인스턴스의 생성이 중지된다.

이에 대해, 기동 클래스수 및 사용 메모리량이 모두 상한값 미만인 경우는, 단계 S514, S310, S318을 거쳐 개별 기능 모듈(130)의 클래스가 판독되고, 판독된 클래스의 인스턴스가 생성되며, 개별 기능 모듈(130)의 이벤트 리스너가 등록된다. 이벤트 리스너 등록 처리에서는, 단계 S509, S510, S405～S407을 거쳐 개별 기능 모듈(130)의 기동 클래스수 및 사용 메모리량이 네트워크 프린터에서의 기동 클래스수 및 사용 메모리량으로 환산되어 가산된다. 이때, 기동 클래스수 및 사용 메모리량 중 어느 하나가 상한값 이상이 되면, 단계 S521, S522 또는 단계 S419, S420을 거쳐 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 생성되는 동시에 에러가 통지되고, 이벤트 리스너 클래스의 판독 또는 인스턴스의 생성이 중지된다.

이에 대해, 기동 클래스수 및 사용 메모리량이 모두 상한값 미만인 경우는, 단계 S514, S410, S416을 거쳐 이벤트 리스너 클래스가 판독되고, 이벤트 리스너 클래스의 인스턴스가 생성되며, 생성된 인스턴스의 이벤트 리스너가 이벤트 리스너 실행 리스트에 등록된다.

한편, 호스트 단말(100)에서는, 공통 기능 모듈(120)의 실행에 의해 이벤트 리스너 제어 처리가 실행된다. 이벤트 리스너 제어 처리에서는, 단계 S703～S705를 거쳐 실행해야 할 이벤트 리스너의 생성원이 된 개별 기능 모듈(130)의 사용 메모리량이 네트워크 프린터에서의 사용 메모리량으로 환산되어 가산된다. 이때, 사용 메모리량이 상한값 이상이 되면, 단계 S715, S716을 거쳐 개별 기능 모듈(130)이 사용하는 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 생성되는 동시에 에러가 통지되고, 이벤트 리스너의 실행이 중지된다.

이에 대해, 사용 메모리량이 상한값 미만인 경우는, 단계 S710을 거쳐 이벤트 리스너에 포함되는 명령이 실행된다.

한편, 호스트 단말(100)에서는, 공통 기능 모듈(120)의 실행에 의해 인스턴스 삭제 처리가 실행된다. 인스턴스 삭제 처리에서는, 삭제해야 할 인스턴스가 존재하는 경우는, 단계 S808, S810을 거쳐 그 인스턴스가 삭제되고, 그 이벤트 리스너의 생성원이 된 개별 기능 모듈(130)의 사용 메모리량이 감산된다.

도 19는 에러가 발생한 경우의 로그 파일의 내용을 나타내는 도면이다.

에러가 발생한 경우, 로그 파일에는, 도 19에 나타내는 바와 같이, 개별 기능 모듈(130)이 기동 및 정지한 것을 나타내는 로그 정보 외에, 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 기록된다.

도 20은 에러가 발생하지 않은 경우의 로그 파일의 내용을 나타내는 도면이다.

이에 대해, 에러가 발생하지 않은 경우, 로그 파일에는, 도 20에 나타내는 바와 같이, 개별 기능 모듈(130)이 기동 및 정지한 것을 나타내는 로그 정보만이 기록된다.

다음에, 리소스 관리 대상이 아닌 개별 기능 모듈(130)을 실행하는 경우를 설명한다.

호스트 단말(100)에서는, 개별 기능 모듈 제어 처리가 실행되면, 단계 S116, S118을 거쳐 해당 개별 기능 모듈(130)이 판독되고, 판독된 개별 기능 모듈(130)의 실행 여부가 판정된다.

개별 기능 모듈(130)의 실행이 허가되면, 단계 S134를 거쳐 개별 기능 모듈(130)이 기동한다. 모듈 기동 처리에서는, 단계 S524, S324, S318을 거쳐 개별 기능 모듈(130)의 클래스가 판독되고, 판독된 클래스의 인스턴스가 생성되며, 개별 기능

모듈(130)의 이벤트 리스너가 등록된다. 이벤트 리스너 등록 처리에서는, 단계 S524, S422, S416을 거쳐 이벤트 리스너 클래스가 판독되고, 이벤트 리스너 클래스의 인스턴스가 생성되며, 생성된 인스턴스의 이벤트 리스너가 이벤트 리스너 실행 리스트에 등록된다.

한편, 호스트 단말(100)에서는, 이벤트 리스너 제어 처리가 실행되면, 단계 S616을 거쳐 실행해야 할 이벤트 리스너에 포함되는 명령이 실행된다.

한편, 호스트 단말(100)에서는, 인스턴스 삭제 처리가 실행되면, 삭제해야 할 인스턴스가 존재하는 경우는 단계 S816을 거쳐 그 인스턴스가 삭제된다.

도 21은 리소스 관리 대상이 되는 개별 기능 모듈(b, c)을 병렬로 실행한 경우를 나타내는 타임차트이다.

도 21에 있어서, 실선은 개별 기능 모듈(b)의 스레드 및 공통 기능 모듈(120)의 스레드 중 개별 기능 모듈(b)의 실행에 이용되는 것을 나타낸다. 또한, 일점쇄선은 개별 기능 모듈(c)의 스레드 및 공통 기능 모듈(120)의 스레드 중 개별 기능 모듈(c)의 실행에 이용되는 것을 나타낸다.

개별 기능 모듈(b)이 실행되면, 공통 기능 모듈(120)의 AM 스레드(기동 처리부)가 실행되고, 개별 기능 모듈(b)이 기동하여 그 스레드가 실행된다. 또한, 공통 기능 모듈(120)의 AM 스레드가 실행되고, 개별 기능 모듈(b)의 이벤트 리스너가 생성된다. 그리고, 개별 기능 모듈(b)에 대응하는 이벤트가 발생하면, 공통 기능 모듈(120)의 AM 스레드(이벤트 처리부)가 실행되고, 개별 기능 모듈(b)의 클래스가 판독되며, 판독된 인스턴스가 생성된다. 개별 기능 모듈(b)이 불필요하게 된 경우는, 공통 기능 모듈(120)의 인스턴스 삭제 스레드가 실행되고, 개별 기능 모듈(b)의 인스턴스가 삭제된다. 이 일련의 처리에 있어서, 공통 기능 모듈(120) 및 개별 기능 모듈(b)의 스레드의 실행에 따라 증감하는 기동 클래스수 및 사용 메모리량은, 개별 기능 모듈(b)이 사용하는 리소스의 양으로서 관리되고, 개별 기능 모듈(b)에 설정된 소정의 상한값 미만이 되도록 제한된다.

이 동작은, 개별 기능 모듈(c)에 대해서도 동일하다. 단, 그 일련의 처리에 있어서, 공통 기능 모듈(120) 및 개별 기능 모듈(c)의 스레드의 실행에 따라 증감하는 기동 클래스수 및 사용 메모리량은, 개별 기능 모듈(c)이 사용하는 리소스의 양으로서 관리되고, 개별 기능 모듈(c)에 설정된 소정의 상한값 미만이 되도록 제한된다.

다음에, 개별 기능 모듈(130)을 인증하는 경우를 설명한다.

호스트 단말(100)에서는, 로그 파일이 생성되면, 단계 S900~S906을 반복하여 거쳐 로그 파일로부터 로그 정보가 순차적으로 독출되고, 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달했는지의 여부 및 개별 기능 모듈(130)이 네트워크 프린터에 인스톨 불가능한지의 여부가 판정된다. 도 20에 나타내는 로그 파일과 같이, 로그 파일에 포함되는 모든 로그 정보에 대해서, 기동 클래스수 또는 사용 메모리량이 상한에 도달하지 않으면서 인스톨 가능하다고 판정되면, 단계 S908~S912를 거쳐 개별 기능 모듈(130)의 실행 파일이 독출되고, 독출된 실행 파일에 전자 서명 정보가 부가되어 저장된다.

이에 대해, 도 19에 나타내는 로그 파일과 같이, 로그 파일에 포함되는 어느 하나의 로그 정보에 대해 기동 클래스수 또는 사용 메모리량이 상한에 도달하거나, 또는 인스톨 불가능하다고 판정되면, 단계 S914를 거쳐 인증 불가능함을 나타내는 메시지가 표시된다.

이와 같이 하여, 본 실시 형태에서는, 개별 기능 모듈(130)이 호스트 단말(100)에서 사용하는 리소스의 양을 측정하고, 측정한 리소스의 양을 네트워크 프린터에서 사용하는 리소스의 양으로 환산하며, 개별 기능 모듈(130)로부터 상한값을 취득하고, 환산한 리소스의 양 및 취득한 상한값에 기초하여 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하도록 되어 있다.

이에 의해, 개별 기능 모듈(130)이 네트워크 프린터에서 사용하는 리소스의 양이 리소스의 상한값에 도달하는지를 네트워크 프린터에 도입하기 전에 검증할 수 있다. 따라서, 종래에 비해, 소프트웨어의 개발을 용이하게 할 수 있는 동시에 안정성이 높은 소프트웨어를 개발할 수 있다.

또, 본 실시 형태에서는, 리소스 관리 대상이 되는 각 개별 기능 모듈(130)마다 그 개별 기능 모듈(130)이 사용하는 메모리량 및 공통 기능 모듈(120)이 그 개별 기능 모듈(130)의 실행에 사용하는 메모리량 및 기동하는 클래스수를 측정하도록 되어 있다.

이에 의해, 공통 기능 모듈(120)이 네트워크 프린터에서 사용하는 리소스의 양이 리소스의 상한값에 도달하는지를 개별 기능 모듈(130) 단위로 검증할 수 있다.

또, 본 실시 형태에서는, 환산한 리소스의 양 및 취득한 상한값에 기초하여 개별 기능 모듈(130)에 의한 리소스의 확보를 금지하도록 되어 있다.

이에 의해, 개별 기능 모듈(130)이 상한값을 초월하여 리소스의 양을 사용하는 것을 제한할 수 있다.

또, 본 실시 형태에서는, 호스트 단말(100)에서 소정 조건으로 사용되는 리소스의 양 및 네트워크 프린터에서 같은 소정 조건으로 사용되는 리소스의 양에 기초하여 결정되는 환산율을 등록한 리소스 환산 테이블(22)에 기초하여 환산을 행하도록 되어 있다.

이에 의해, 호스트 단말(100) 및 네트워크 프린터의 사이에서 리소스의 양을 비교적 정확하게 환산할 수 있다.

또, 본 실시 형태에서는, 공통 기능 모듈(120) 및 개별 기능 모듈(130)이 사용하는 리소스의 종별에 따라 리소스 환산 테이블(22)로부터 대응하는 환산율을 취득하고, 취득한 환산율에 기초하여 환산을 행하도록 되어 있다.

이에 의해, 공통 기능 모듈(120) 및 개별 기능 모듈(130)이 사용하는 리소스의 종별에 따른 환산을 행할 수 있으므로, 호스트 단말(100) 및 네트워크 프린터의 사이에서 리소스의 양을 더 정확하게 환산할 수 있다.

또, 본 실시 형태에서는, 공통 기능 모듈(120) 및 개별 기능 모듈(130)이 사용하는 리소스의 형태에 따라 리소스 환산 테이블(22)로부터 대응하는 환산율을 취득하고, 취득한 환산율에 기초하여 환산을 행하도록 되어 있다.

이에 의해, 공통 기능 모듈(120) 및 개별 기능 모듈(130)이 사용하는 리소스의 형태에 따른 환산을 행할 수 있으므로, 호스트 단말(100) 및 네트워크 프린터의 사이에서 리소스의 양을 더 정확하게 환산할 수 있다.

또, 본 실시 형태에서는, 리소스 환산 테이블(22)은, 복수의 테스트 모듈의 각각에 대해 호스트 단말(100) 및 네트워크 프린터에서 그 테스트 모듈이 사용하는 리소스의 양에 기초하여 환산율을 결정하고, 각 테스트 모듈에 대해서 결정된 환산율 중 최대값을 등록하였다.

이에 의해, 리소스 환산 테이블(22)에는, 각 테스트 모듈에 대해서 결정된 환산율 중 최대값이 등록되어 있으므로, 공통 기능 모듈(120) 및 개별 기능 모듈(130)이 사용하는 리소스의 양을 넉넉하게 추정할 수 있다. 따라서, 개별 기능 모듈(130)의 동작을 비교적 확실하게 보증할 수 있다.

또, 본 실시 형태에서는, 로그 파일로부터 로그 정보를 독출하고, 독출한 로그 정보에 기초하여 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달했는지의 여부를 판정하며, 리소스의 양이 상한에 도달하고 있지 않다고 판정했을 때는 개별 기능 모듈(130)의 실행 파일에 전자 서명 정보를 부가하도록 되어 있다.

이에 의해, 사용하는 리소스의 양이 상한에 도달하지 않는 개별 기능 모듈(130)에 대해서만 전자 서명 정보가 부가되므로, 개별 기능 모듈(130)의 동작을 비교적 확실하게 보증할 수 있다.

또, 본 실시 형태에서는, 로그 파일로부터 로그 정보를 독출하고, 독출한 로그 정보에 기초하여 개별 기능 모듈(130)이 네트워크 프린터에 인스톨 불가능한지의 여부를 판정하며, 인스톨 가능하다고 판정했을 때는 개별 기능 모듈(130)의 실행 파일에 전자 서명 정보를 부가하도록 되어 있다.

이에 의해, 네트워크 프린터에 인스톨 가능한 개별 기능 모듈(130)에 대해서만 전자 서명 정보가 부가되므로, 개별 기능 모듈(130)의 동작을 더 확실하게 보증할 수 있다.

상기 제1 실시 형태에 있어서, 리소스 측정부(16) 및 단계 S305, S405, S703은 형태 1, 5, 9 또는 10의 리소스 측정 수단에 대응하고, 리소스 환산부(24) 및 단계S306, S406, S509, S704는 형태 1, 5, 9 또는 10의 리소스 환산 수단에 대응한다. 또한, 상한값 취득부(18) 및 단계 S211은 형태 1, 5, 9 또는 10의 리소스 제한 정보 취득 수단에 대응하고, 로그 정보 생성부(26) 및 단계 S321, S419, S521, S715는 형태 1, 5, 9 또는 10의 로그 정보 생성 수단에 대응한다.

또, 상기 제1 실시 형태에 있어서, 로그 정보 취득부(28) 및 단계 S900은 형태 1의 로그 정보 취득 수단에 대응하고, 단계 S900은 형태 5, 9 또는 10의 로그 정보 취득 단계에 대응하며, 동작 판정부(30) 및 단계 S902는 형태 1의 동작 판정 수단에 대응한다. 또, 단계 S900은 형태 5, 9 또는 10의 동작 판정 단계에 대응하고, 모듈 인증부(32) 및 단계 S910은 형태 1의 소프트웨어 인증 수단에 대응하며, 단계 S910은 형태 5, 9 또는 10의 소프트웨어 인증 단계에 대응한다.

또한, 상기 제1 실시 형태에 있어서, 전자 서명 정보는 형태 1, 5, 9 또는 10의 인증 정보에 대응하고, CPU(50)는 형태 10의 연산 수단에 대응한다.

다음에, 본 발명의 제2 실시 형태를 도면을 참조하면서 설명한다. 도 22 내지 도 25는 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법의 제2 실시 형태를 나타내는 도면이다.

본 실시 형태는, 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 도 22에 나타내는 바와 같이, 호스트 단말(100) 상의 JAVA(등록상표) 애플리케이션의 실행 환경에 있어서, 네트워크 프린터의 동작을 제어하기 위한 JAVA(등록상표) 클래스 세트를 에뮬레이션하는 경우에 대해서 적용한 것이고, 상기 제1 실시 형태와 다른 것은 리소스의 상한값을 환산하는 점에 있다. 또, 이하, 상기 제1 실시 형태와 다른 부분에 대해서만 설명하고, 상기 제1 실시 형태와 중복되는 부분에 대해서는 동일한 부호를 부여하여 설명을 생략한다.

우선, 본 발명을 적용하는 호스트 단말(100)의 기능 개요를 설명한다.

도 22는 호스트 단말(100)의 기능 개요를 나타내는 기능 블록도이다.

공통 기능 모듈(120)은, 도 22에 나타내는 바와 같이, 개별 기능 모듈 관리부(14), 리소스 측정부(16), 상한값 취득부(18), 리소스 제한부(20), 리소스 환산 테이블(22) 및 로그 정보 생성부(26) 외에, 리소스의 양을 환산하는 리소스 환산부(34)를 갖고 구성되어 있다.

리소스 환산부(34)는, 리소스 환산 테이블(22)에 기초하여 상한값 취득부(18)로 취득한 리소스의 양을 호스트 단말(100)에서의 리소스의 상한값으로 환산한다.

리소스 제한부(20)는, 리소스 측정부(16)로 측정한 리소스의 양이 리소스 환산부(34)로 환산한 상한값 미만이 되도록, 개별 기능 모듈(130)이 사용하는 리소스의 양 및 개별 기능 모듈 관리부(14)가 그 개별 기능 모듈(130)의 실행에 사용하는 리소스의 양을 제한한다.

로그 정보 생성부(26)는, 리소스 측정부(16)로 측정한 리소스의 양이 리소스 환산부(34)로 환산한 상한값 이상이라고 판정했을 때는, 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성한다.

다음에, 호스트 단말(100)의 구성을 설명한다.

기억 장치(62)는, 도 7의 리소스 환산 테이블(22) 대신에, 도 23의 리소스 환산 테이블(22)을 기억하고 있다.

도 23은 리소스 환산 테이블(22)의 데이터 구조를 나타내는 도면이다.

리소스 환산 테이블(22)에는, 도 23에 나타내는 바와 같이, 리소스의 종별 또는 사용 형태별로 하나의 레코드가 등록되어 있다. 각 레코드는, 리소스의 명칭을 등록하는 필드(502)와, 환산율을 등록하는 필드(504)를 포함하여 구성되어 있다.

도 23의 예에서, 제1 단째의 레코드에는, 리소스의 명칭으로서 「메모리」가, 환산율로서 「1」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 네트워크 프린터에서의 메모리량의 상한값을 환산율 「1」로 나눔으로써, 호스트 단말(100)에서의 메모리량의 상한값으로의 환산을 행하는 것을 나타낸다.

또한, 제2 단째의 레코드에는, 리소스의 명칭으로서 「클래스수」가, 환산율로서 「1」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 네트워크 프린터에서 기동 가능한 클래스수의 상한값을 환산율 「1」로 나눔으로써, 호스트 단말(100)에서 기동 가능한 클래스수로의 환산을 행하는 것을 나타낸다.

또, 리소스 환산 테이블(22)에는, 복수의 테스트 모듈의 각각에 대해 호스트 단말(100) 및 네트워크 프린터에서 그 테스트 모듈이 사용하는 리소스의 양에 기초하여 환산율을 결정하고, 각 테스트 모듈에 대해서 결정된 환산율 중 최대값이 등록되어 있다.

기억 장치(62)는, 또, 도 8의 리소스 관리 테이블(460) 대신에, 도 24의 리소스 관리 테이블(460)을 기억하고 있다.

도 24는 리소스 관리 테이블(460)의 데이터 구조를 나타내는 도면이다.

리소스 관리 테이블(460)에는, 도 24에 나타내는 바와 같이, 리소스의 종별마다 하나의 레코드가 등록되어 있다. 각 레코드는, 리소스의 명칭을 등록하는 필드(462)와, 개별 기능 모듈(130)이 네트워크 프린터에서의 리소스의 상한값을 등록하는 필드(464)와, 필드(464)의 값을 호스트 단말(100)에서의 리소스의 상한값으로 환산한 값을 등록하는 필드(470)와, 개별 기능 모듈(130)이 호스트 단말(100)에서 현재 사용하고 있는 리소스의 양을 등록하는 필드(466)를 포함하여 등록되어 있다.

도 24의 예에서, 제1 단째의 레코드에는, 상한값으로서 「1000000」이, 환산값으로서 「666666」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 네트워크 프린터에서의 메모리량의 상한값이 1000000[byte]이고, 호스트 단말(100)에서의 메모리량의 상한값으로 환산한 값(이하, 환산 메모리 상한값이라 함)이 666666[byte]임을 나타낸다. 환산 메모리 상한값은 도 23의 리소스 환산 테이블(22)을 참조하여, 1000000/1.5=666666으로서 산출할 수 있다.

또한, 제2 단째의 레코드에는, 상한값으로서 「100」이, 환산값으로서 「20」이 각각 등록되어 있다. 이는, 개별 기능 모듈(130)이 네트워크 프린터에서 기동 가능한 클래스수의 상한값이 100개이고, 호스트 단말(100)에서 기동 가능한 클래스수의 상한값으로 환산한 값(이하, 환산 클래스 상한값이라 함)이 20개임을 나타낸다. 환산 클래스 상한값은, 도 23의 리소스 환산 테이블(22)을 참조하여, 20/1=20으로서 산출할 수 있다.

다음에, 호스트 단말(100)에서 실행되는 처리를 설명한다.

CPU(50)는, 도 11의 실행 여부 판정 처리에 대신하여, 도 25의 플로우차트에 나타내는 실행 여부 판정 처리를 실행한다.

도 25는 실행 여부 판정 처리를 나타내는 플로우차트이다.

실행 여부 판정 처리는, 단계 S118에서 실행되면, 도 25에 나타내는 바와 같이, 우선 단계 S200~S212를 거쳐 단계 S213으로 이행한다.

단계 S213에서는, 취득한 상한값에 대응하는 환산율을 판독한 리소스 환산 테이블(22)로부터 취득하고, 취득한 상한값을 취득한 환산율로 나눔으로써, 호스트 단말(100)에서의 상한값으로의 환산을 행한다.

다음에, 단계 S214로 이행하여 환산 메모리 상한값이 총 메모리 잔량 미만인지의 여부를 판정하고, 총 메모리 잔량 미만이라고 판정했을 때(Yes)는 단계 S216으로 이행하지만, 그렇지 않다고 판정했을 때(No)는 단계 S217로 이행한다.

또, 상기 제1 실시 형태에 있어서는, 단계 S306, S406, S509, S704의 처리를 설치하였지만, 본 실시 형태에서는 그들의 처리가 불필요하게 된다.

또한, 상기 제1 실시 형태에 있어서는, 단계 S313, S320, S418, S517, S520, S714의 처리에서 환산 메모리량 및 환산 클래스수를 취급하였지만, 본 실시 형태에서는, 호스트 단말(100)에서의 사용 메모리량 및 기동 클래스수를 취급한다.

또, 상기 제1 실시 형태에 있어서는, 단계 S308, S408, S512, S706에서 네트워크 프린터에서의 사용 메모리량 및 기동 클래스수를 취급하였지만, 본 실시 형태에서는, 환산 메모리 상한값 및 환산 메모리를 취급한다.

다음에, 본 실시 형태의 동작을 설명한다.

처음에, 리소스 관리 대상이 되는 개별 기능 모듈(130)을 실행하는 경우를 설명한다.

호스트 단말(100)에서는, 공통 기능 모듈(120)의 실행에 의해 개별 기능 모듈 제어 처리가 실행된다. 개별 기능 모듈 제어 처리에서는, 단계 S102~S110을 거쳐 삭제해야 할 개별 기능 모듈(130)이 존재하는 경우는, 그 개별 기능 모듈(130)이 삭제된다. 다음에, 단계 S114를 거쳐 현재 기동 중인 모듈수가 소정의 상한값 미만이라고 판정되면, 단계 S116, S118을 거쳐 해당 개별 기능 모듈(130)이 판독되고, 판독된 개별 기능 모듈(130)의 실행 여부가 판정된다. 실행 여부 판정 처리에서는, 단계 S213을 거쳐 리소스 제한 정보(400)로부터 취득된 상한값이 호스트 단말(100)에서의 상한값으로 환산된다. 그리고, 개별 기능 모듈(130)에 대해서, 일치하는 기종 정보 및 대응가능한 전자 서명 정보를 가지면서, 환산 메모리 상한값이 총 메모리 잔량 미만인 경우에 실행이 허가된다.

개별 기능 모듈(130)의 실행이 허가되면, 단계 S124~S128을 거쳐 리소스 관리 테이블(460)이 생성되고, 현재 기동 중인 모듈수가 「1」 가산되며, 개별 기능 모듈(130)이 기동한다. 모듈 기동 처리에서는, 단계 S510, S305, S307을 거쳐 개별 기능 모듈(130)의 기동 클래스수 및 사용 메모리량이 가산된다. 이때, 기동 클래스수 및 사용 메모리량 중 어느 하나가 환산 상한값(환산 클래스수 또는 환산 메모리 상한값을 말함. 이하 동일) 이상이 되면, 단계 S521, S522 또는 단계 S321, S322를 거쳐 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 생성되는 동시에 에러가 통지되고, 클래스의 판독 또는 인스턴스의 생성이 중지된다.

이에 대해, 기동 클래스수 및 사용 메모리량이 모두 환산 상한값 미만인 경우는, 단계 S514, S310, S318을 거쳐 개별 기능 모듈(130)의 클래스가 판독되고, 판독된 클래스의 인스턴스가 생성되며, 개별 기능 모듈(130)의 이벤트 리스너가 등록된다. 이벤트 리스너 등록 처리에서는, 단계 S510, S405, S407을 거쳐 개별 기능 모듈(130)의 기동 클래스수 및 사용 메모리량이 가산된다. 이때, 기동 클래스수 및 사용 메모리량 중 어느 하나가 환산 상한값 이상이 되면, 단계 S521, S522 또는 단계S419, S420을 거쳐 개별 기능 모듈(130)의 기동 클래스수 또는 사용 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 생성되는 동시에 에러가 통지되고, 이벤트 리스너 클래스의 판독 또는 인스턴스의 생성이 중지된다.

이에 대해, 기동 클래스수 및 사용 메모리량이 모두 환산 상한값 미만인 경우는, 단계 S514, S410, S416을 거쳐 이벤트 리스너 클래스가 판독되고, 이벤트 리스너 클래스의 인스턴스가 생성되며, 생성된 인스턴스의 이벤트 리스너가 이벤트 리스너 실행 리스트에 등록된다.

한편, 호스트 단말(100)에서는, 공통 기능 모듈(120)의 실행에 의해 이벤트 리스너 제어 처리가 실행된다. 이벤트 리스너 제어 처리에서는, 단계 S703, S705를 거쳐 실행해야 할 이벤트 리스너의 생성원이 된 개별 기능 모듈(130)의 사용 메모리량이 가산된다. 이때, 사용 메모리량이 환산 상한값 이상이 되면, 단계 S715, S716을 거쳐 개별 기능 모듈(130)이 사용하는 메모리량이 상한에 도달한 것을 나타내는 로그 정보가 생성되는 동시에 에러가 통지되고, 이벤트 리스너의 실행이 중지된다.

이에 대해, 사용 메모리량이 환산 상한값 미만인 경우는, 단계 S710을 거쳐 이벤트 리스너에 포함되는 명령이 실행된다.

이와 같이 하여, 본 실시 형태에서는, 개별 기능 모듈(130)이 호스트 단말(100)에서 사용하는 리소스의 양을 측정하고, 개별 기능 모듈(130)로부터 상한값을 취득하며, 취득한 상한값을 호스트 단말(100)에서의 리소스의 상한값으로 환산하고, 측정한 리소스의 양 및 환산한 상한값에 기초하여 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하도록 되어 있다.

이에 의해, 개별 기능 모듈(130)이 네트워크 프린터에서 사용하는 리소스의 양이 리소스의 상한값에 도달하는지를 네트워크 프린터에 도입하기 전에 검증할 수 있다. 따라서, 종래에 비해, 소프트웨어의 개발을 용이하게 할 수 있는 동시에 안정성이 높은 소프트웨어를 개발할 수 있다.

상기 제2 실시 형태에 있어서, 리소스 측정부(16) 및 단계 S305, S405, S703은 형태 2, 6, 11 또는 12의 리소스 측정 수단에 대응하고, 리소스 환산부(34) 및 단계 S213은 형태 2, 6, 11 또는 12의 리소스 환산 수단에 대응한다. 또한, 상한값 취득부(18) 및 단계 S211은 형태 2, 6, 11 또는 12의 리소스 상한값 취득 수단에 대응하고, 로그 정보 생성부(26) 및 단계 S321, S419, S521, S715는 형태 2, 6, 11 또는 12의 로그 정보 생성 수단에 대응한다.

또, 상기 제2 실시 형태에 있어서, 로그 정보 취득부(28) 및 단계 S900은 형태 2의 로그 정보 취득 수단에 대응하고, 단계 S900은 형태 6, 11 또는 12의 로그 정보 취득 단계에 대응하며, 동작 판정부(30) 및 단계 S902는 형태 2의 동작 판정 수단에 대응한다. 또한, 단계 S900은 형태 6, 11 또는 12의 동작 판정 단계에 대응하고, 모듈 인증부(32) 및 단계 S910은 형태 2의 소프트웨어 인증 수단에 대응하며, 단계 S910은 형태 6, 11 또는 12의 소프트웨어 인증 단계에 대응한다.

또, 상기 제2 실시 형태에 있어서, 전자 서명 정보는 형태 2, 6, 11 또는 12의 인증 정보에 대응하고, CPU(50)는 형태 12의 연산 수단에 대응한다.

다음에, 본 발명의 제3 실시 형태를 도면을 참조하면서 설명한다. 도 26 내지 도 30은 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법의 제3 실시 형태를 나타내는 도면이다.

본 실시 형태는, 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 도 26에 나타내는 바와 같이, 호스트 단말(100) 상의 JAVA(등록상표) 애플리케이션의 실행 환경에 있어서, 네트워크 프린터의 동작을 제어하기 위한 JAVA(등록상표) 클래스 세트를 에뮬레이션하는 경우에 대해서 적용한 것이고, 상기 제1 및 제2 실시 형태와 다른 것은 파일 조작에 관한 로그 정보에 기초하여 개별 기능 모듈(130)을 인증하는 점에 있다. 또, 이하, 상기 제1 및 제2 실시 형태와 다른 부분에 대해서만 설명하고, 상기 제1 및 제2 실시 형태와 중복되는 부분에 대해서는 동일한 부호를 부여하여 설명을 생략한다.

우선, 본 발명을 적용하는 호스트 단말(100)의 기능 개요를 설명한다.

도 26은 호스트 단말(100)의 기능 개요를 나타내는 기능 블록도이다.

공통 기능 모듈(120)은, 도 26에 나타내는 바와 같이, 개별 기능 모듈 관리부(14) 및 로그 정보 생성부(26) 외에, 개별 기능 모듈(130)이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시부(36)를 갖고 구성되어 있다.

로그 정보 생성부(26)는, 리소스 감시부(36)의 감시 결과에 기초하여 리소스 사용 상황을 나타내는 로그 정보를 생성한다.

다음에, 호스트 단말(100)의 구성을 설명한다.

CPU(50)는, 도 16의 이벤트 리스너 실행 처리에 대신하여, 도 27의 플로우차트에 나타내는 이벤트 리스너 실행 처리를 실행한다. 또한, 도 18의 모듈 인증 처리에 대신하여, 도 28의 플로우차트에 나타내는 모듈 인증 처리를 실행한다.

처음에, 이벤트 리스너 실행 처리를 설명한다.

도 27은 이벤트 리스너 실행 처리를 나타내는 플로우차트이다.

이벤트 리스너 실행 처리는, 단계 S610에서 실행되면, 도 27에 나타내는 바와 같이, 우선 단계 S1000으로 이행한다.

단계 S1000에서는, 이벤트 리스너에 포함되는 명령 리스트의 선두로 프로그램 포인터를 이동하고, 단계 S1002로 이행하여 프로그램 포인터가 지시하는 어드레스에 실행해야 할 명령이 존재하는지의 여부를 판정하며, 실행해야 할 명령이 존재한다고 판정했을 때는 단계 S1004로 이행한다.

단계 S1004에서는, 실행해야 할 명령이 파일 조작을 행하는 파일 조작명령인지의 여부를 판정하고, 파일 조작명령이라고 판정했을 때(Yes)는 단계 S1006으로 이행한다. 파일 조작으로서는, 예를 들어 파일의 작성, 파일의 판독, 파일의 기입, 파일의 삭제, 파일명의 확인, 파일명의 변경, 파일 속성의 확인, 파일 속성의 변경, 디렉토리의 작성, 디렉토리의 삭제, 디렉토리명의 확인, 디렉토리명의 변경, 디렉토리 속성의 확인 및 디렉토리 속성의 변경을 들 수 있다. 파일 또는 디렉토리의 속성으로서는, 판독 전용 등의 액세스권의 설정에 관한 속성이 포함된다.

단계 S1006에서는, 파일 조작의 대상이 되는 파일명 및 패스명을 포함하는 로그 정보를 생성하고, 생성한 로그 정보를 기억 장치(62)의 로그 파일에 기록하며, 단계S1008로 이행한다.

단계 S1008에서는 프로그램 포인터가 지시하는 어드레스의 명령을 실행하고, 단계 S1010으로 이행하여 이벤트 리스너에 포함되는 명령 리스트의 다음으로 프로그램 포인터를 이동하며, 단계 S1002로 이행한다.

한편, 단계 S1004에서 파일 조작명령이 아니라고 판정했을 때(No)는, 단계 S1008로 이행한다.

한편, 단계 S1002에서 실행해야 할 명령이 존재하지 않는다고 판정했을 때(No)는, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 모듈 인증 처리를 설명한다.

도 28은 모듈 인증 처리를 나타내는 플로우차트이다.

모듈 인증 처리는, CPU(50)에서 실행되면, 도 28에 나타내는 바와 같이,우선 단계 S1100으로 이행한다.

단계 S1100에서는 기억 장치(62)의 로그 파일로부터 로그 정보를 독출하고, 단계 S1102로 이행하여 독출한 로그 정보가 파일 조작에 관한 로그 정보인지의 여부를 판정하며, 파일 조작에 관한 로그 정보라고 판정했을 때(Yes)는 단계 S1104로 이행한다.

단계 S1104에서는 독출한 로그 정보로부터 파일명을 취득하고, 단계 S1106으로 이행하여 취득한 파일명이 소정 길이(네트워크 프린터에서 대응 가능한 길이) 이하인지의 여부를 판정하며, 파일명이 소정 길이 이하라고 판정했을 때(Yes)는 단계 S1108로 이행한다.

단계 S1108에서는, 취득한 파일명에 네트워크 프린터에서 사용할 수 없는 문자 종류(한자 등)의 문자가 포함되어 있는지의 여부를 판정하고, 파일명에 사용 불가능한 문자 종류의 문자가 포함되어 있지 않다고 판정했을 때(No)는 단계 S1110으로 이행한다.

단계 S1110에서는 독출한 로그 정보로부터 패스명을 취득하고, 단계 S1112로 이행하여 취득한 패스명이 소정 길이 이하인지의 여부를 판정하며, 패스명이 소정 길이 이하라고 판정했을 때(Yes)는 단계 S1114로 이행한다.

단계 S1114에서는, 취득한 패스명에 사용 불가능한 문자 종류의 문자가 포함되어 있는지의 여부를 판정하고, 패스명에 사용 불가능한 문자 종류의 문자가 포함되어 있지 않다고 판정했을 때(No)는 단계 S1116으로 이행한다.

단계 S1116에서는 기억 장치(62)의 로그 파일에 미처리의 로그 정보가 존재하는지의 여부를 판정하고, 미처리의 로그 정보가 존재하지 않는다고 판정했을 때(No)는, 단계 S1118로 이행하여 개별 기능 모듈(130)의 실행 파일을 기억 장치(62)로부터 독출하며, 단계 S1120으로 이행한다.

단계 S1120에서는 독출한 실행 파일에 전자 서명 정보를 부가하고, 단계 S1122로 이행하여 전자 서명 정보를 부가한 실행 파일을 기억 장치(62)에 저장하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S1116에서 미처리의 로그 정보가 존재한다고 판정했을 때(Yes)는, 단계 S1100으로 이행한다.

한편, 단계 S1106에서 파일명이 소정 길이보다도 크다고 판정했을 때(No), 단계 S1108에서 파일명에 사용 불가능한 문자 종류의 문자가 포함되어 있다고 판정했을 때(Yes), 단계 S1112에서 패스명이 소정 길이보다도 크다고 판정했을 때(No), 및 단계 S1114에서 패스명에 사용 불가능한 문자 종류의 문자가 포함되어 있다고 판정했을 때(Yes)는, 모두 단계 S1124로 이행하여 개별 기능 모듈(130)이 인증 불가능한 것을 나타내는 메시지를 표시 장치(64)에 표시하고, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S1102에서 독출한 로그 정보가 파일 조작에 관한 로그 정보가 아니라고 판정했을 때(No)는, 단계 S1116으로 이행한다.

다음에, 본 실시 형태의 동작을 설명한다.

호스트 단말(100)에서는, 개별 기능 모듈(130)의 실행에 의해 파일 조작명령이 실행되면, 단계 S1006을 거쳐 파일 조작의 대상이 되는 파일명 및 패스명을 포함하는 로그 정보가 생성된다.

도 29 및 도 30은 파일 조작이 행해진 경우의 로그 파일의 내용을 나타내는 도면이다.

파일 조작이 행해진 경우, 로그 파일에는, 도 29 및 도 30에 나타내는 바와 같이, 개별 기능 모듈(130)이 기동 및 정지한 것을 나타내는 로그 정보와, 네트워크 프린터에서의 파일명 및 패스명의 길이의 상한값을 나타내는 로그 정보와, 파일 조작의 대상이 된 파일명 및 패스명을 포함하는 로그 정보가 기록된다. 도 29의 예에서는 제3～제5 단째의 레코드가, 도 30의 예에서는 제3 단째의 레코드가 파일 조작에 관한 로그 정보를 저장하고 있다.

호스트 단말(100)에서는, 로그 파일이 생성되면, 단계 S1100~S1116을 반복하여 거쳐 로그 파일로부터 로그 정보가 순차적으로 독출되고, 독출된 로그 정보로부터 파일명 및 패스명이 취득되며, 취득된 파일명 및 패스명이 소정 길이 이하인지의 여부 및 취득된 파일명 및 패스명에 사용불가능한 문자 종류의 문자가 포함되어 있는지의 여부가 판정된다. 도 30에 나타내는 로그 파일과 같이, 로그 파일에 포함되는 파일 조작에 관한 모든 로그 정보에 대해서, 파일명 및 패스명이 소정 길이 이하이면서, 파일명 및 패스명에 사용 불가능한 문자 종류의 문자가 포함되어 있지 않다고 판정되면, 단계 S1118~S1122를 거쳐 개별 기능 모듈(130)의 실행 파일이 독출되고, 독출된 실행 파일에 전자 서명 정보가 부가되어 저장된다.

이에 대해, 도 29에 나타내는 로그 파일과 같이, 로그 파일에 포함되는 파일 조작에 관한 어느 하나의 로그 정보에 대해서, 파일명 또는 패스명이 소정 길이보다도 크다고 판정되면, 단계 S1124를 거쳐 인증 불가능함을 나타내는 메시지가 표시된다. 도 29의 예에서는, 제4 및 제5 단째의 레코드가 이에 해당한다.

이와 같이 하여, 본 실시 형태에서는, 파일 조작에 관한 로그 정보를 로그 파일로부터 독출하고, 독출한 로그 정보에 기초하여 파일명 및 패스명이 소정 길이 이하인지의 여부를 판정하며, 소정 길이 이하라고 판정했을 때는, 개별 기능 모듈(130)의 실행 파일에 전자 서명 정보를 부가하도록 되어 있다.

이에 의해, 파일 조작의 대상이 되는 파일명 및 패스명이 네트워크 프린터에서 대응 가능한 길이 이하의 개별 기능 모듈(130)에 대해서만 전자 서명 정보가 부가되므로, 개별 기능 모듈(130)의 동작을 더 확실하게 보증할 수 있다.

또, 본 실시 형태에서는, 파일 조작에 관한 로그 정보를 로그 파일로부터 독출하고, 독출한 로그 정보에 기초하여 파일명 및 패스명에 사용 불가능한 문자 종류의 문자가 포함되어 있는지의 여부를 판정하며, 사용 불가능한 문자 종류의 문자가 포함되어 있지 않다고 판정했을 때는, 개별 기능 모듈(130)의 실행 파일에 전자 서명 정보를 부가하도록 되어 있다.

이에 의해, 파일 조작의 대상이 되는 파일명 및 패스명에 네트워크 프린터에서 사용할 수 없는 문자 종류의 문자를 포함하지 않는 개별 기능 모듈(130)에 대해서만 전자 서명 정보가 부가되므로, 개별 기능 모듈(130)의 동작을 더 확실하게 보증할 수 있다.

상기 제3 실시 형태에 있어서, 리소스 감시부(36) 및 단계 S1004는 형태 3, 7, 13 또는 14의 리소스 감시 수단에 대응하고, 로그 정보 생성부(26) 및 단계 S1006은 형태 3, 7, 13 또는 14의 로그 정보 생성 수단에 대응하며, 로그 정보 취득부(28) 및 단계 S1100은 형태 3의 로그 정보 취득 수단에 대응한다. 또한, 단계 S1100은 형태 7, 13 또는 14의 로그 정보 취득 단계에 대응하고, 동작 판정부(30) 및 단계 S1106, S1108, S1112, S1114는 형태 3의 동작 판정 수단에 대응하며, 단계 S1106, S1108, S1112, S1114는 형태 7, 13 또는 14의 동작 판정 단계에 대응한다.

또, 상기 제3 실시 형태에 있어서, 모듈 인증부(32) 및 단계 S1120은 형태 3의 소프트웨어 인증 수단에 대응하고, 단계 S1120은 형태 7, 13 또는 14의 소프트웨어 인증 단계에 대응하며, 전자 서명 정보는 형태 3, 7, 13 또는 14의 인증 정보에 대응한다. 또, CPU(50)는 형태 14의 연산 수단에 대응한다.

다음에, 본 발명의 제4 실시 형태를 도면을 참조하면서 설명한다. 도 31 및 도 32는 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법의 제4 실시 형태를 나타내는 도면이다.

본 실시 형태는, 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 호스트 단말(100) 상의 JAVA(등록상표) 애플리케이션의 실행 환경에 있어서, 네트워크 프린터의 동작을 제어하기 위한 JAVA(등록상표) 클래스 세트를 에뮬레이션하는 경우에 대해 적용한 것이고, 상기 제1 내지 제3 실시 형태와 다른 것은, 개별 기능 모듈(130)의 실행 파일에 기초하여 개별 기능 모듈(130)을 인증하는 점에 있다. 또, 이하, 상기 제1 내지 제3 실시 형태와 다른 부분에 대해서만 설명하고, 상기 제1 내지 제3 실시 형태와 중복되는 부분에 대해서는 동일한 부호를 부여하여 설명을 생략한다.

우선, 호스트 단말(100)의 구성을 설명한다.

CPU(50)는, 도 18의 모듈 인증 처리에 대신하여, 도 31의 플로우차트에 나타내는 모듈 인증 처리를 실행한다.

도 31은 모듈 인증 처리를 나타내는 플로우차트이다.

모듈 인증 처리는, CPU(50)에서 실행되면, 도 31에 나타내는 바와 같이, 우선 단계 S1200으로 이행한다.

단계 S1200에서는 개별 기능 모듈(130)의 실행 파일을 독출하고, 단계 S1201로 이행하여 독출한 개별 기능 모듈(130)의 실행 파일로부터 클래스 파일리스트를 취득하며, 단계 S1202로 이행하여 취득한 클래스 파일리스트에 기초하여 클래스의 검증을 행하는 클래스 검증 처리를 실행하고, 단계 S1204로 이행한다.

단계 S1204에서는, 클래스 검증 처리로부터 클래스 부존재 통지가 출력되었는지의 여부를 판정하고, 클래스 부존재 통지가 출력되지 않는다고 판정했을 때(No)는 단계 S1206으로 이행한다.

단계 S1206에서는, 기억 장치(62)의 로그 파일에 미처리의 로그 정보가 존재하는지의 여부를 판정하고, 미처리의 로그 정보가 존재하지 않는다고 판정했을 때(No)는, 단계 S1208로 이행하여 개별 기능 모듈(130)의 실행 파일을 기억 장치(62)로부터 독출하며, 단계 S1210으로 이행한다.

단계 S1210에서는 판독한 실행 파일에 전자 서명 정보를 부가하고, 단계 S1212로 이행하여 전자 서명 정보를 부가한 실행 파일을 기억 장치(62)에 저장하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S1206에서 미처리의 로그 정보가 존재한다고 판정했을 때(Yes)는, 단계 S1202로 이행한다.

한편, 단계 S1204에서 클래스 부존재 통지가 출력되었다고 판정했을 때(Yes)는, 단계 S1214로 이행하여 개별 기능 모듈(130)이 인증 불가능함을 나타내는 메시지를 표시 장치(64)에 표시하고, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 단계 S1202의 클래스 검증 처리를 설명한다.

도 32는 클래스 검증 처리를 나타내는 플로우차트이다.

클래스 검증 처리는, 단계 S1202에서 실행되면, 도 32에 나타내는 바와 같이, 우선 단계 S1300으로 이행한다.

단계 S1300에서는, 취득한 클래스 파일 리스트에 기초하여 클래스 파일을 판독하고, 단계 S1302로 이행하여 판독한 클래스 파일로부터 클래스 정의 리스트를 생성하며, 단계 S1304로 이행한다.

단계 S1304에서는, 생성한 클래스 정의 리스트에 미검증의 클래스 정의가 존재하는지의 여부를 판정하고, 미검증의 클래스 정의가 존재한다고 판정했을 때(Yes)는, 단계 S1306으로 이행하여 클래스의 판독에 성공했는지의 여부를 판정하며, 클래스의 판독에 성공했다고 판정했을 때(Yes)는 단계 S1308로 이행한다.

단계 S1308에서는, 클래스 정의에 기초하여 클래스 해결패스 상에 클래스가 존재하는지의 여부를 판정하고, 클래스 해결패스 상에 클래스가 존재하지 않는다고 판정했을 때(No)는, 단계 S1310으로 이행하여 클래스 부존재 통지를 출력하며, 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

한편, 단계 S1308에서 클래스 해결패스 상에 클래스가 존재한다고 판정했을 때(Yes)는, 단계 S1304로 이행한다.

한편, 단계 S1306에서 클래스의 판독에 실패했다고 판정했을 때(No) 및 단계 S1304에서 미검증의 클래스 정의가 존재하지 않는다고 판정했을 때(No)는, 모두 일련의 처리를 종료하여 원래의 처리로 복귀시킨다.

다음에, 본 실시 형태의 동작을 설명한다.

호스트 단말(100)에서는, 단계 S1202~S1206을 반복하여 거쳐 개별 기능 모듈(130)이 사용하는 클래스가 네트워크 프린터에서 실행 가능한 클래스만으로 이루어지는지의 여부가 판정된다. 그 결과, 네트워크 프린터에서 실행 가능한 클래스만으로 이루어진다고 판정되면, 단계 S1208~S1212를 거쳐 개별 기능 모듈(130)의 실행 파일이 독출되고, 독출된 실행 파일에 전자 서명 정보가 부가되어 저장된다.

이에 대해, 개별 기능 모듈(130)이 사용하는 어느 하나의 클래스가 네트워크 프린터에서 실행할 수 없는 클래스라고 판정되면, 단계 S1214를 거쳐 인증 불가능함을 나타내는 메시지가 표시된다.

이와 같이 하여, 본 실시 형태에서는, 개별 기능 모듈(130)의 실행 파일을 취득하고, 취득한 실행 파일에 기초하여 개별 기능 모듈(130)이 사용하는 클래스가 네트워크 프린터에서 실행 가능한 클래스만으로 이루어지는지의 여부를 판정하며, 네트워크 프린터에서 실행 가능한 클래스만으로 이루어진다고 판정했을 때는, 개별 기능 모듈(130)의 실행 파일에 전자 서명 정보를 부가하도록 되어 있다.

이에 의해, 네트워크 프린터에서 실행 가능한 클래스만을 사용하는 개별 기능 모듈(130)에 대해서만 전자 서명 정보가 부가되므로, 개별 기능 모듈(130)의 동작을 더 확실하게 보증할 수 있다. 예를 들어, 악의를 가진 개발자가 호스트 단말(100)로부터 네트워크 프린터로 이행할 때에, 클래스의 바이너리를 고쳐 쓰기해버리면 네트워크 프린터에서 올바르게 동작하지 않게 된다. 이러한 부정 행위도 방지할 수 있다.

상기 제4 실시 형태에 있어서, 단계 S1200은 형태 4의 실행 파일 취득 수단 또는 형태 8 또는 15의 실행 파일 취득 단계에 대응하고, 단계 S1204는 형태 4의 제2 동작 판정 수단 또는 형태 8 또는 15의 제2 동작 판정 단계에 대응한다. 또한, 단계 S1210은 형태 4의 소프트웨어 인증 수단 또는 형태 8 또는 15의 소프트웨어 인증 단계에 대응한다.

또, 상기 제1 및 제2 실시 형태에 있어서, 리소스 환산 테이블(22)은 각 테스트 모듈에 대해 결정된 환산율 중 최대값을 등록하여 구성하였지만, 이에 한정되지 않고, 각 테스트 모듈에 대해서 결정된 환산율 중 평균값을 등록하여 구성할 수도 있다.

이에 의해, 리소스 환산 테이블(22)에는, 각 테스트 모듈에 대해 결정된 환산율 중 평균값이 등록되어 있으므로, 기능 모듈의 동작을 어느 정도의 확실성을 갖고 보증할 수 있는 동시에 네트워크 프린터에서 사용되는 리소스의 양을 억제할 수 있다.

또한, 상기 제1 및 제2 실시 형태에 있어서는, 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하도록 구성하였지만,이에 한정되지 않고, 개별 기능 모듈(130)이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 메시지를 표시 장치(64)에 표시하도록 구성할 수도 있다.

또, 상기 제1 및 제2 실시 형태에 있어서는, 리소스 환산 테이블(22)에 환산율로서 「1」을 등록한 예를 나타냈지만, 이에 한정되지 않고, 호스트 단말(100)과 네트워크 프린터의 사양의 차이에 따라서는 그 이외의 값을 등록할 수도 있다.

또한, 상기 제1 및 제2 실시 형태에 있어서는, 리소스의 양으로서 사용 메모리량 및 기동 클래스수를 제한하도록 구성하였지만, 이에 한정되지 않고, 소켓 접속수, 파일 접속수, 파일수, 파일용량, 클래스 사이즈, ZIP 메모리 용량, CPU 이용량, 소켓 통신량 및 파일 판독/기입량을 제한하도록 구성할 수도 있다.

또, 상기 제1 내지 제4 실시 형태를 각각 개별로 구성하였지만, 이에 한정되지 않고, 그들의 형태를 복합적으로 조합하여 구성할 수 있다. 즉, 단계 S902, S904의 판정 처리, 단계 S1106, S1108, S1112, S1114의 판정 처리 및 단계 S1204의 판정 처리를 임의로 조합할 수 있다. 가장 바람직한 것은, 이들 판정 처리에 관한 모든 인증 조건을 충족시킨 경우에만 개별 기능 모듈(130)에 전자 서명 정보를 부가하는 것이다.

또한, 상기 제1 내지 제4 실시 형태에 있어서, 도 10 내지 도 18, 도 25, 도 27, 도 28, 도 31 및 도 32의 플로우차트에 나타내는 처리를 실행함에 있어서는, 모두 ROM(52)에 미리 저장되어 있는 제어 프로그램을 실행하는 경우에 대해 설명하였지만, 이에 한정되지 않고, 이들의 순서를 나타낸 프로그램이 기억된 기억 매체로부터 그 프로그램을 RAM(54)으로 판독하여 실행하도록 해도 된다.

여기서, 기억 매체란, RAM, ROM 등의 반도체 기억 매체, FD, HD 등의 자기기억형 기억 매체, CD, CDV, LD, DVD 등의 광학적 판독 방식 기억 매체, MO 등의 자기 기억형/광학적 판독 방식 기억 매체로서, 전자적, 자기적, 광학적 등의 판독 방법의 여하에 상관없이, 컴퓨터로 판독 가능한 기억 매체이면, 모든 기억 매체를 포함하는 것이다.

또, 상기 제1 내지 제4 실시 형태에 있어서는, 본 발명에 관한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 호스트 단말(100) 상의 JAVA(등록상표) 애플리케이션의 실행 환경에 있어서, 네트워크 프린터의 동작을 제어하기 위한 JAVA(등록상표) 클래스 세트를 에뮬레이션하는 경우에 대하여 적용하였지만, 이에 한정되지 않고, 본 발명의 주지(主旨)를 일탈하지 않는 범위에서 다른 경우에도 적용 가능하다. 네트워크 프린터에 대신하여 예를 들어, 프로

젝터, 전자 페이퍼, 홈 게이트웨이, 퍼스널컴퓨터, PDA(Personal Digital Assistant), 네트워크 스토리지, 오디오 기기, 휴대전화, PHS(등록상표)(Personal Handyphone System), 시계형 PDA, STB(Set Top Box), POS(Point Of Sale) 단말, FAX기, 전화(IP 전화 등도 포함), 그 밖의 디바이스에 적용할 수 있다.

발명의 효과

이상 설명한 바와 같이, 본 발명에 의하면, 소프트웨어의 동작을 그 실행 환경에 도입하기 전에 검증함으로써, 소프트웨어의 개발을 용이하게 하는 동시에 안정성이 높은 소프트웨어를 개발하는 데에 매우 적합한 소프트웨어 인증 시스템, 소프트웨어 인증 프로그램 및 소프트웨어 인증 방법을 얻을 수 있다.

(57) 청구의 범위

청구항 1.

기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 시스템으로서,

상기 로그 정보를 취득하는 로그 정보 취득 수단과, 상기 로그 정보 취득 수단으로 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 수단과, 상기 동작 판정 수단이 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 수단을 구비하는 것을 특징으로 하는 소프트웨어 인증 시스템.

청구항 2.

기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 시스템으로서,

상기 로그 정보를 취득하는 로그 정보 취득 수단과, 상기 로그 정보 취득 수단으로 취득한 로그 정보에 기초하여 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 수단과, 상기 동작 판정 수단이 상한에 도달하고 있지 않다고 판정했을 때는 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 수단을 구비하는 것을 특징으로 하는 소프트웨어 인증 시스템.

청구항 3.

제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 시스템으로서,

상기 로그 정보를 취득하는 로그 정보 취득 수단과, 상기 로그 정보 취득 수단으로 취득한 로그 정보에 기초하여 상기 리소스 사용 상황이 상기 제1 실행 환경에 적합한지를 판정하는 동작 판정 수단과, 상기 동작 판정 수단이 적합하다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 수단을 구비하는 것을 특징으로 하는 소프트웨어 인증 시스템.

청구항 4.

제1항 내지 제3항 중 어느 한 항에 있어서,

상기 기능 모듈의 실행에 필요한 실행 파일을 취득하는 실행 파일 취득 수단과, 상기 실행 파일 취득 수단으로 취득한 실행 파일에 기초하여, 상기 기능 모듈을 구성하는 명령 또는 명령군이 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어지는지를 판정하는 제2 동작 판정 수단을 구비하고,

상기 소프트웨어 인증 수단은, 상기 제2 동작 판정 수단이 상기 제1 실행 환경에서 실행 가능한 명령 또는 명령군만으로 이루어진다고 판정했을 때는, 상기 소프트웨어에 상기 인증 정보를 부가하도록 되어 있는 것을 특징으로 하는 소프트웨어 인증 시스템.

청구항 5.

기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체로서,

상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계로 이루어지는 처리를 컴퓨터로 실행시키기 위한 프로그램을 포함하는 것을 특징으로 하는 소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체.

청구항 6.

기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체로서,

상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계로 이루어지는 처리를 컴퓨터로 실행시키기 위한 프로그램을 포함하는 것을 특징으로 하는 소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체.

청구항 7.

제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여, 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체로서,

상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여, 상기 리소스 사용 상황이 상기 제1 실행 환경에 적합한지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 적합하다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계로 이루어지는 처리를 컴퓨터로 실행시키기 위한 프로그램을 포함하는 것을 특징으로 하는 소프트웨어 인증 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체.

청구항 8.

기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양을 상기 기능 모듈이 제2 실행 환경에서 사용하는 리소스의 양으로 환산하는 리소스 환산 수단과, 리소스의 제한 조건을 나타내는 리소스 제한 정보를 취득하는 리소스 제한 정보 취득 수단과, 상기 리소스 환산 수단으로 환산한 리소스의 양 및 상기 리소스 제한 정보 취득 수단으로 취득한 리소스 제한 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서,

상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 하는 소프트웨어 인증 방법.

청구항 9.

기능 모듈이 제1 실행 환경에서 사용하는 리소스의 양을 측정하는 리소스 측정 수단과, 제2 실행 환경에서의 리소스의 상한값을 취득하는 리소스 상한값 취득 수단과, 상기 리소스 상한값 취득 수단으로 취득한 리소스의 상한값을 상기 제1 실행 환경에서의 리소스의 상한값으로 환산하는 리소스 환산 수단과, 상기 리소스 측정 수단으로 측정한 리소스의 양 및 상기 리소스 환산 수단으로 환산한 리소스의 상한값에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달한 것을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서,

상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여, 상기 기능 모듈이 사용하는 리소스의 양이 상한에 도달하고 있지 않은지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 상한에 도달하고 있지 않다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 하는 소프트웨어 인증 방법.

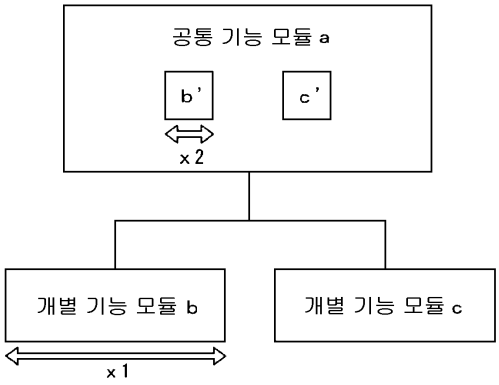
청구항 10.

제1 실행 환경과는 다른 제2 실행 환경에서 기능 모듈이 사용하는 리소스의 사용 상황을 감시하는 리소스 감시 수단과, 상기 리소스 감시 수단의 감시 결과에 기초하여, 상기 리소스 사용 상황을 나타내는 로그 정보를 생성하는 로그 정보 생성 수단을 구비하는 리소스 관리 시스템에 의해 생성된 상기 로그 정보에 기초하여 상기 기능 모듈을 포함하는 소프트웨어를 인증하는 소프트웨어 인증 방법으로서,

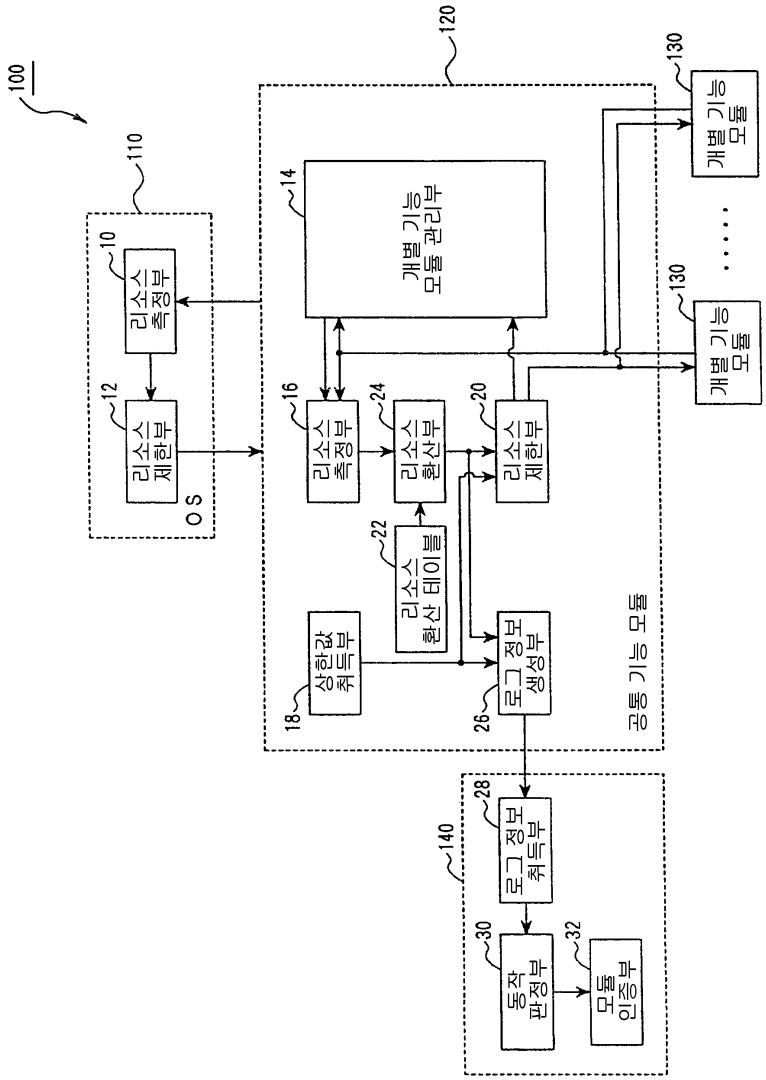
상기 로그 정보를 취득하는 로그 정보 취득 단계와, 상기 로그 정보 취득 단계에서 취득한 로그 정보에 기초하여, 상기 리소스 사용 상황이 상기 제1 실행 환경에 적합한지를 판정하는 동작 판정 단계와, 상기 동작 판정 단계에서 적합하다고 판정했을 때는, 상기 소프트웨어에 인증 정보를 부가하는 소프트웨어 인증 단계를 포함하는 것을 특징으로 하는 소프트웨어 인증 방법.

도면

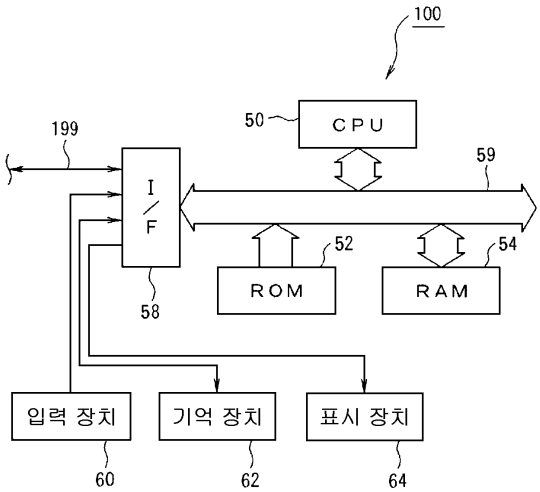
도면1



도면2


100
120
130
개별 기능 모듈
110
14
개별 기능 모듈부
10
리소스 측정부
12
리소스 제한부
OS
16
리소스 측정부
24
리소스 환산부
20
리소스 제한부
22
리소스 환산 테이블
상한값 취득부
18
로그 정보 생성부
26
공통 기능 모듈
140
28
로그 정보 취득부
30
동작 판정부
32
모듈 인증부

도면3


100
199
50
CPU
59
I/F
52
54
ROM
RAM
58
입력 장치
기억 장치
표시 장치
60
62
64

도면4

400

402 리소스명	404 상한값
메모리	1,000,000
클래스수	100

도면5

420

항목	값
422 리소스 관리	유효
424 기종 정보	TypeA
426 전자 서명 정보	X사

도면6

440

항목	값
442 모듈수 상한	5
444 실행 모듈	개별 기능 모듈 b 개별 기능 모듈 d
446 삭제 모듈	개별 기능 모듈 c
448 기종 정보	TypeA
450 전자 서명 정보	X사

도면7

22

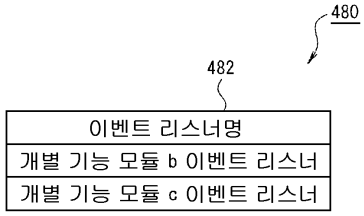
502 리소스명	504 환산율
메모리 사용 형태 A	1
메모리 사용 형태 B	1.5
메모리 사용 형태 C	2
클래스수	1

도면8

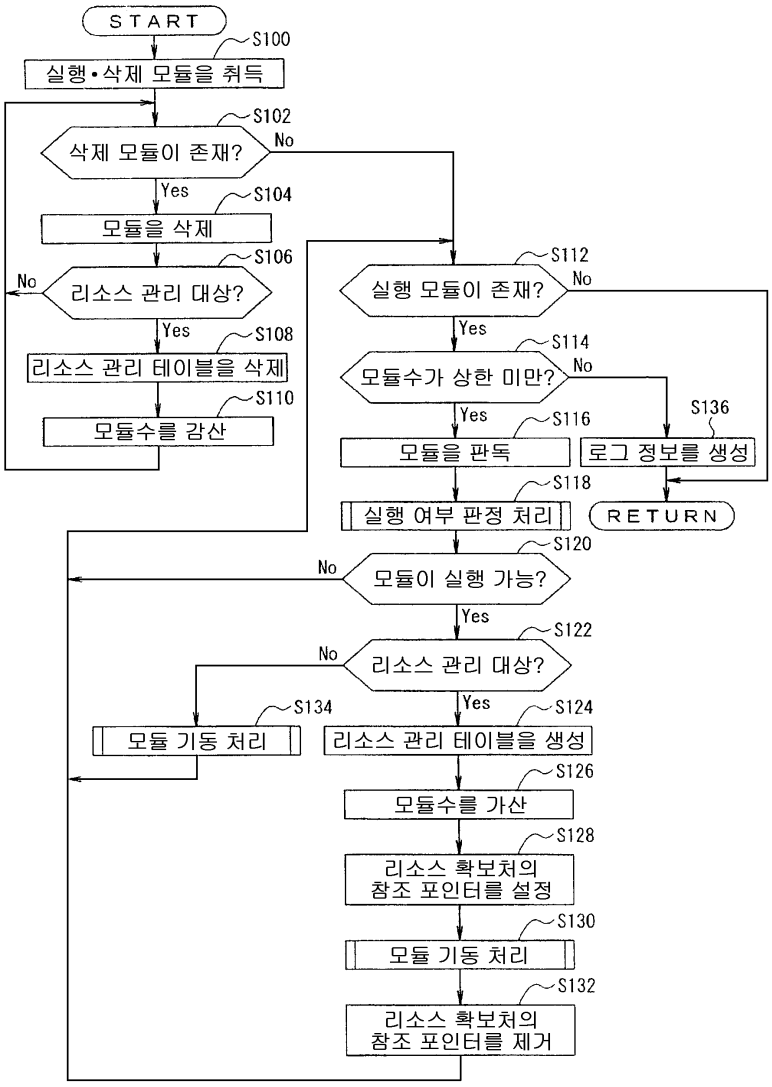
460

462 리소스명	464 상한값	466 현재값	468 환산값
메모리	1,000,000	메모리 사용 형태 A 200,000 메모리 사용 형태 B 100,000 메모리 사용 형태 C 150,000	650,000
클래스수	100	20	20

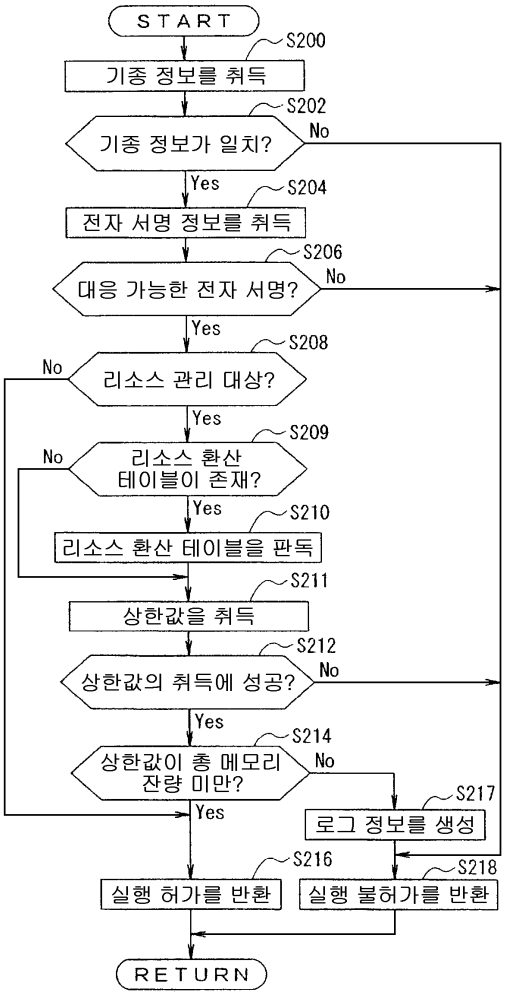
도면9


480
482
이벤트 리스너명
개별 기능 모듈 b 이벤트 리스너
개별 기능 모듈 c 이벤트 리스너

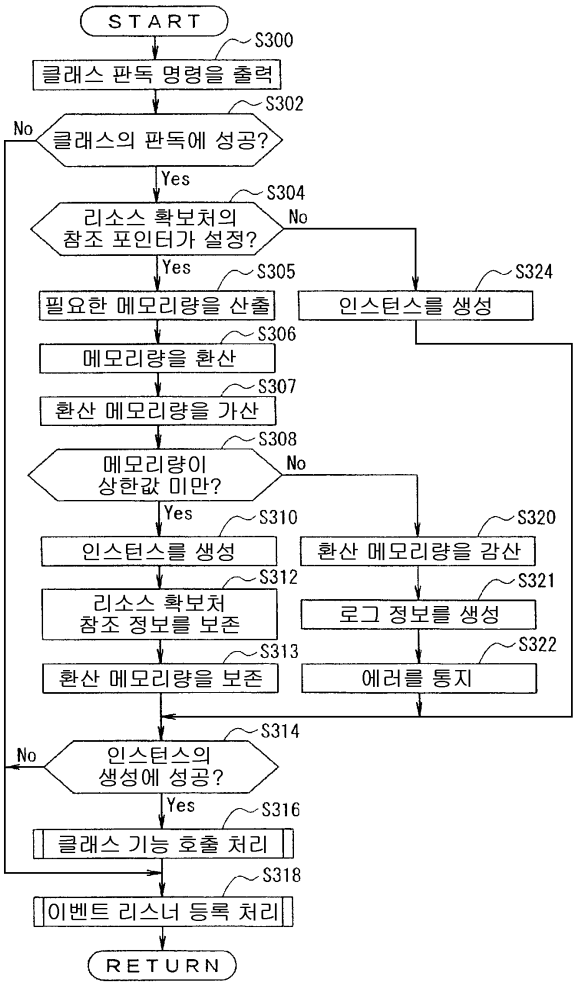
도면10


START
S100
실행•삭제 모듈을 취득
S102
삭제 모듈이 존재?
No
Yes
S104
모듈을 삭제
S106
No
리소스 관리 대상?
Yes
S108
리소스 관리 테이블을 삭제
S110
모듈수를 감산
S112
실행 모듈이 존재?
No
Yes
S114
모듈수가 상한 미만?
No
Yes
S116
모듈을 판독
S136
로그 정보를 생성
S118
실행 여부 판정 처리
RETURN
S120
No
모듈이 실행 가능?
Yes
S122
No
리소스 관리 대상?
Yes
S134
모듈 기동 처리
S124
리소스 관리 테이블을 생성
S126
모듈수를 가산
S128
리소스 확보처의
참조 포인터를 설정
S130
모듈 기동 처리
S132
리소스 확보처의
참조 포인터를 제거

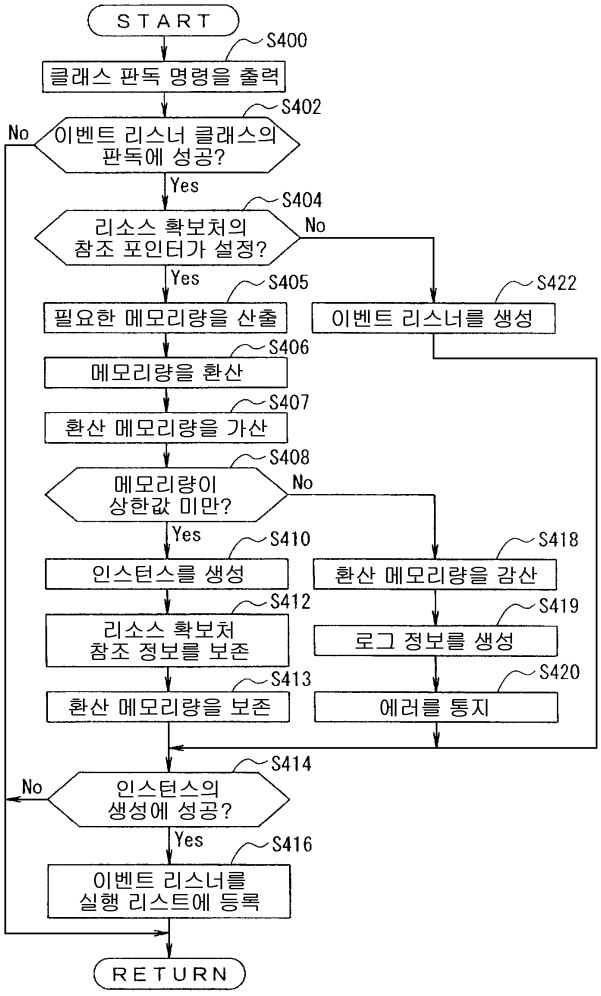
도면11


START
S200
기종 정보를 취득
S202
기종 정보가 일치?
No
Yes
S204
전자 서명 정보를 취득
S206
대응 가능한 전자 서명?
No
Yes
S208
No
리소스 관리 대상?
Yes
S209
No
리소스 환산 테이블이 존재?
Yes
S210
리소스 환산 테이블을 판독
S211
상한값을 취득
S212
상한값의 취득에 성공?
No
Yes
S214
상한값이 총 메모리 잔량 미만?
No
S217
로그 정보를 생성
Yes
S216
실행 허가를 반환
S218
실행 불허가를 반환
RETURN

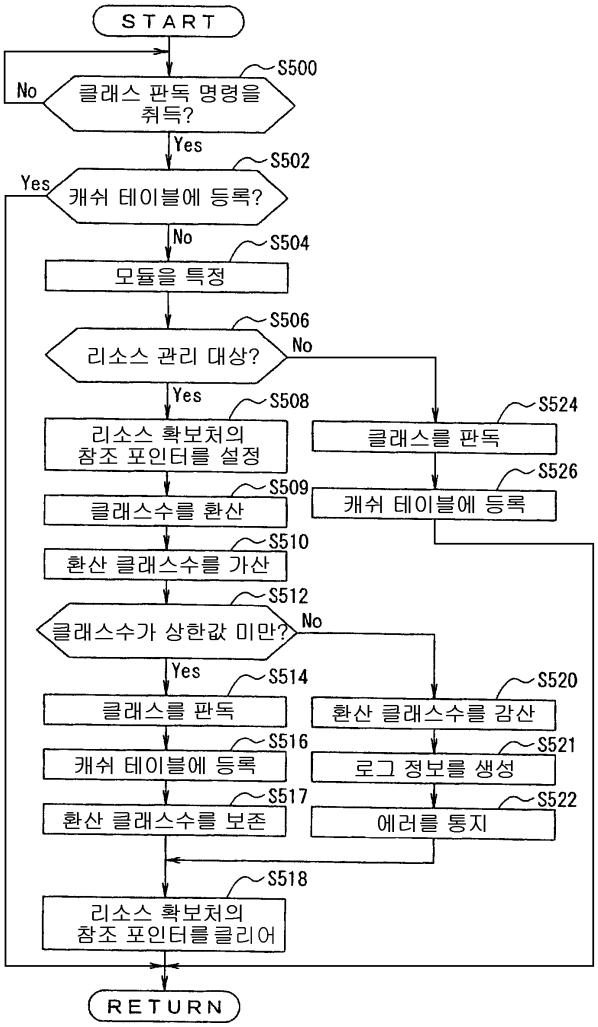
도면12


START
S300
클래스 판독 명령을 출력
S302
No
클래스의 판독에 성공?
Yes
S304
리소스 확보처의
참조 포인터가 설정?
No
Yes
S305
필요한 메모리량을 산출
S324
인스턴스를 생성
S306
메모리량을 환산
S307
환산 메모리량을 가산
S308
메모리량이
상한값 미만?
No
Yes
S310
인스턴스를 생성
S320
환산 메모리량을 감산
S312
리소스 확보처
참조 정보를 보존
S321
로그 정보를 생성
S313
환산 메모리량을 보존
S322
에러를 통지
S314
No
인스턴스의
생성에 성공?
Yes
S316
클래스 기능 호출 처리
S318
이벤트 리스너 등록 처리
RETURN

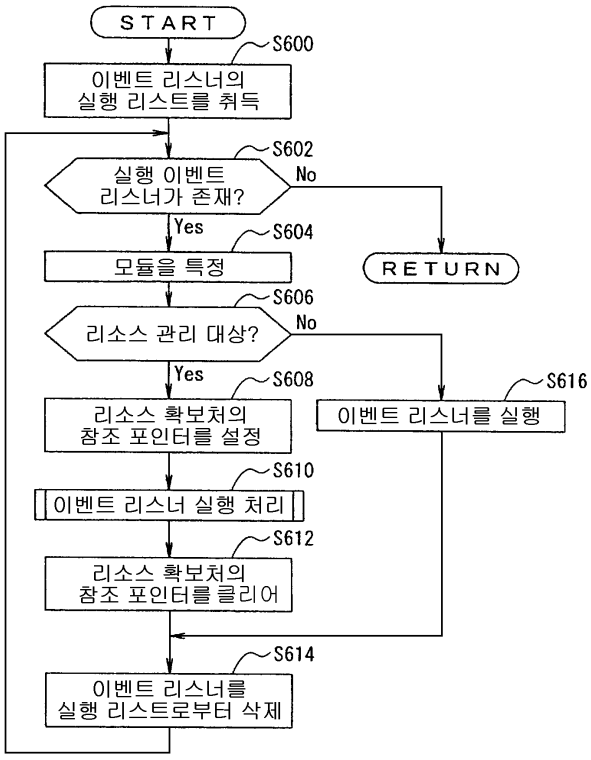
도면13


START
S400
클래스 판독 명령을 출력
S402
이벤트 리스너 클래스의 판독에 성공?
No
Yes
S404
리소스 확보처의 참조 포인터가 설정?
No
Yes
S405
필요한 메모리량을 산출
S422
이벤트 리스너를 생성
S406
메모리량을 환산
S407
환산 메모리량을 가산
S408
메모리량이 상한값 미만?
No
Yes
S410
인스턴스를 생성
S418
환산 메모리량을 감산
S412
리소스 확보처 참조 정보를 보존
S419
로그 정보를 생성
S413
환산 메모리량을 보존
S420
에러를 통지
S414
인스턴스의 생성에 성공?
No
Yes
S416
이벤트 리스너를 실행 리스트에 등록
RETURN

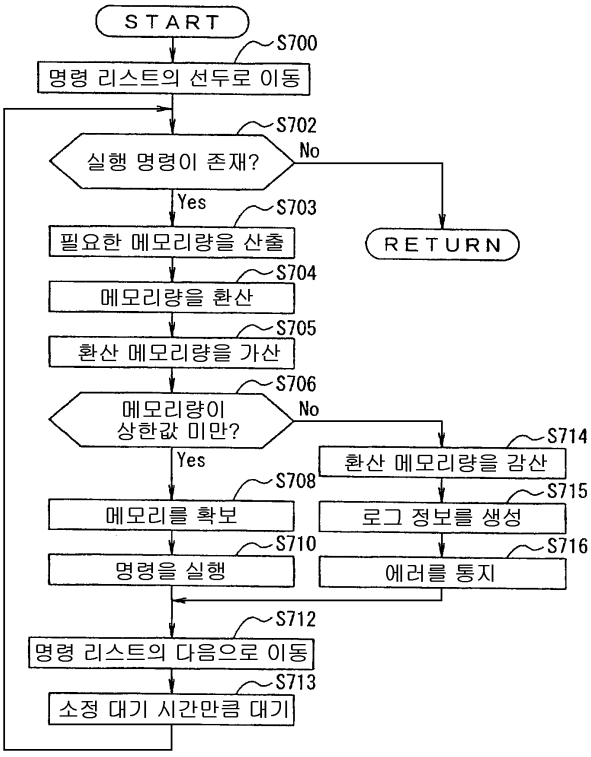
도면14


START
S500
No
클래스 판독 명령을 취득?
Yes
S502
Yes
캐쉬 테이블에 등록?
No
S504
모듈을 특정
S506
리소스 관리 대상?
No
Yes
S508
리소스 확보처의 참조 포인터를 설정
S509
클래스수를 환산
S510
환산 클래스수를 가산
S512
클래스수가 상한값 미만?
No
Yes
S514
클래스를 판독
S516
캐쉬 테이블에 등록
S517
환산 클래스수를 보존
S518
리소스 확보처의 참조 포인터를 클리어
RETURN
S524
클래스를 판독
S526
캐쉬 테이블에 등록
S520
환산 클래스수를 감산
S521
로그 정보를 생성
S522
에러를 통지

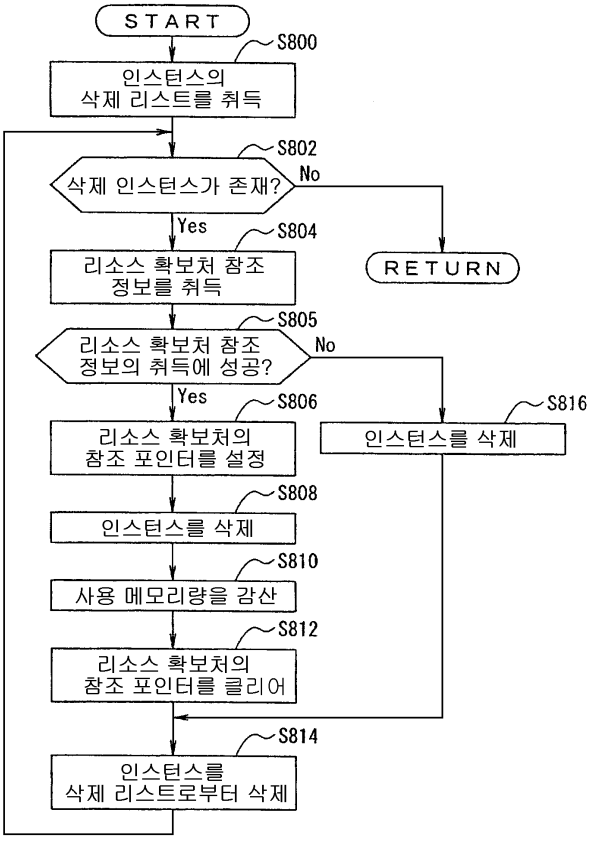
도면15


START
S600
이벤트 리스너의 실행 리스트를 취득
S602
실행 이벤트 리스너가 존재?
No
RETURN
Yes
S604
모듈을 특정
S606
리소스 관리 대상?
No
Yes
S608
리소스 확보처의 참조 포인터를 설정
S616
이벤트 리스너를 실행
S610
이벤트 리스너 실행 처리
S612
리소스 확보처의 참조 포인터를 클리어
S614
이벤트 리스너를 실행 리스트로부터 삭제

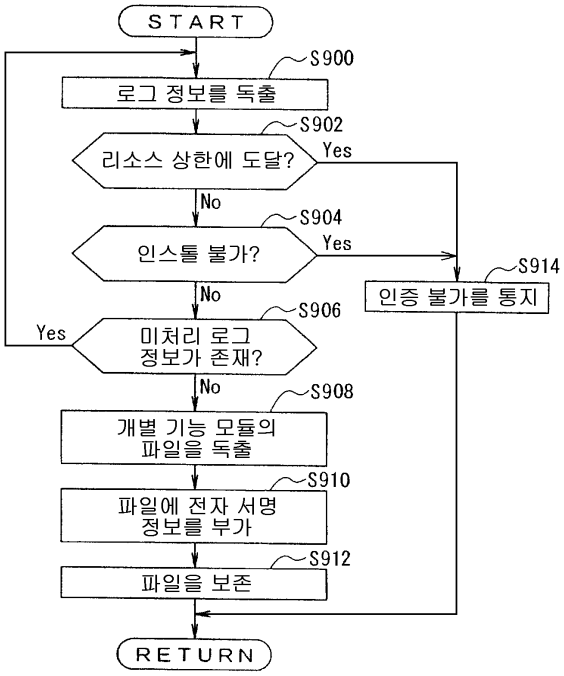
도면16


START
S700
명령 리스트의 선두로 이동
S702
실행 명령이 존재?
No
RETURN
Yes
S703
필요한 메모리량을 산출
S704
메모리량을 환산
S705
환산 메모리량을 가산
S706
메모리량이 상한값 미만?
No
S714
환산 메모리량을 감산
Yes
S708
메모리를 확보
S715
로그 정보를 생성
S710
명령을 실행
S716
에러를 통지
S712
명령 리스트의 다음으로 이동
S713
소정 대기 시간만큼 대기

도면17


START
S800
인스턴스의 삭제 리스트를 취득
S802
삭제 인스턴스가 존재?
No
Yes
RETURN
S804
리소스 확보처 참조 정보를 취득
S805
리소스 확보처 참조 정보의 취득에 성공?
No
Yes
S806
리소스 확보처의 참조 포인터를 설정
S816
인스턴스를 삭제
S808
인스턴스를 삭제
S810
사용 메모리량을 감산
S812
리소스 확보처의 참조 포인터를 클리어
S814
인스턴스를 삭제 리스트로부터 삭제

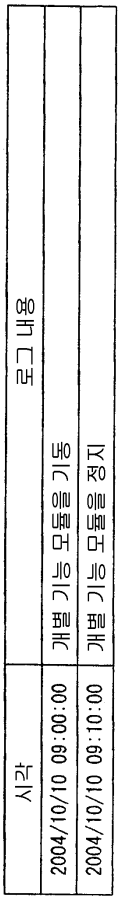
도면18


START
S900
로그 정보를 독출
S902
리소스 상한에 도달?
Yes
No
S904
인스톨 불가?
Yes
S914
인증 불가를 통지
No
S906
Yes
미처리 로그 정보가 존재?
No
S908
개별 기능 모듈의 파일을 독출
S910
파일에 전자 서명 정보를 부가
S912
파일을 보존
RETURN

도면19

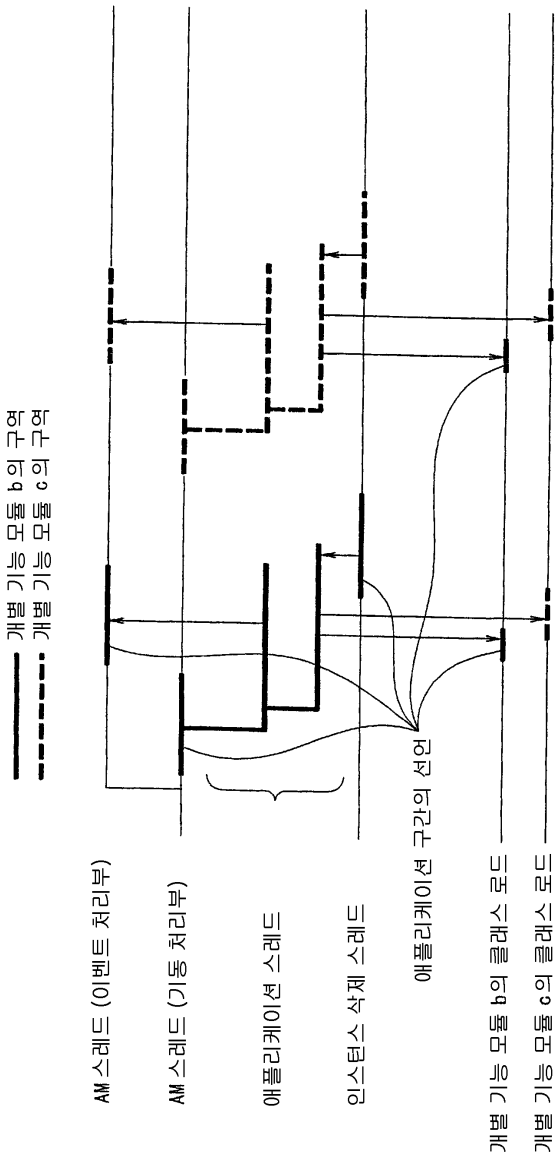
시각	로그 내용
2004/10/10 09:00:00	개별 기능 모듈을 기동
2004/10/10 09:01:00	메모리량이 상한에 도달하였습니다. (상한값 1,000,000 < 사용량 1,000,200)
2004/10/10 09:05:00	클래스수가 상한에 도달하였습니다. (상한값 100 < 사용량 101)
2004/10/10 09:06:00	클래스수가 상한에 도달하였습니다. (상한값 100 < 사용량 101)
2004/10/10 09:10:00	개별 기능 모듈을 정지

도면20

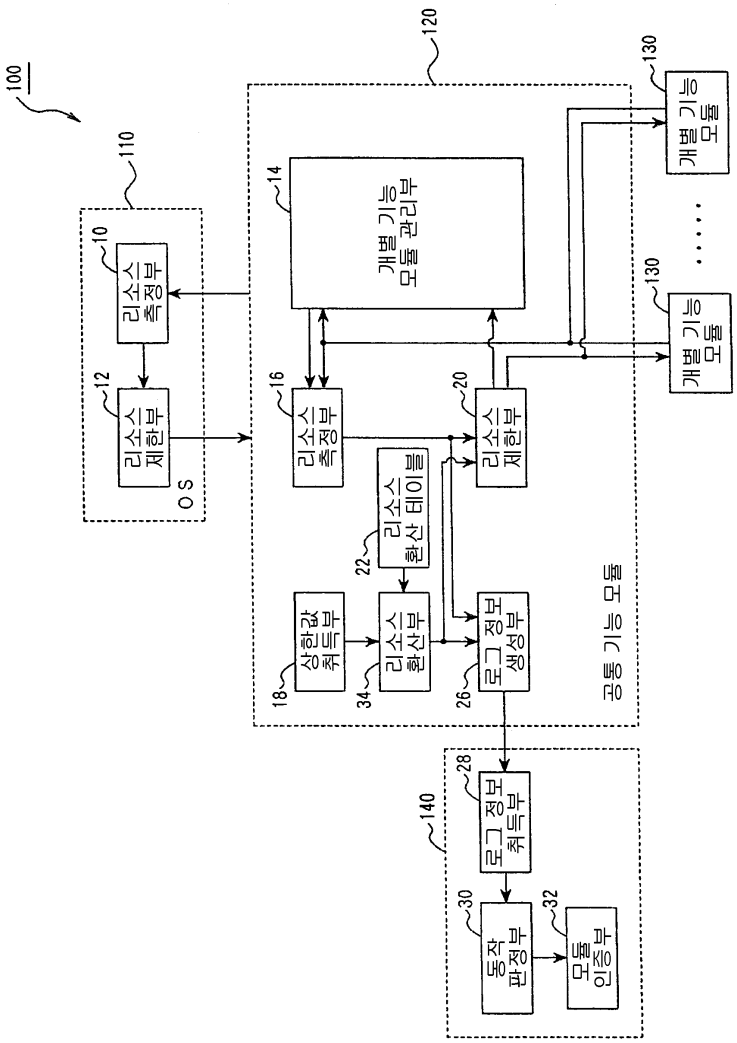


시각	로그 내용
2004/10/10 09:00:00	개별 기능 모듈을 기동
2004/10/10 09:10:00	개별 기능 모듈을 정지

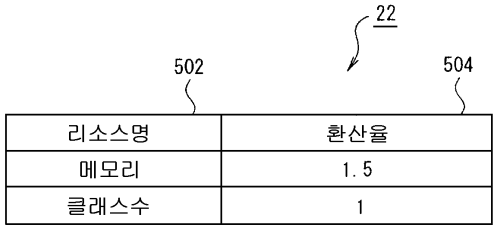
도면21


개별 기능 모듈 b의 구역
개별 기능 모듈 c의 구역
AM 스레드 (이벤트 처리부)
AM 스레드 (기동 처리부)
애플리케이션 스레드
인스턴스 삭제 스레드
애플리케이션 구간의 선언
개별 기능 모듈 b의 클래스 로드
개별 기능 모듈 c의 클래스 로드

도면22

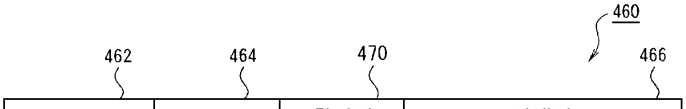


도면23



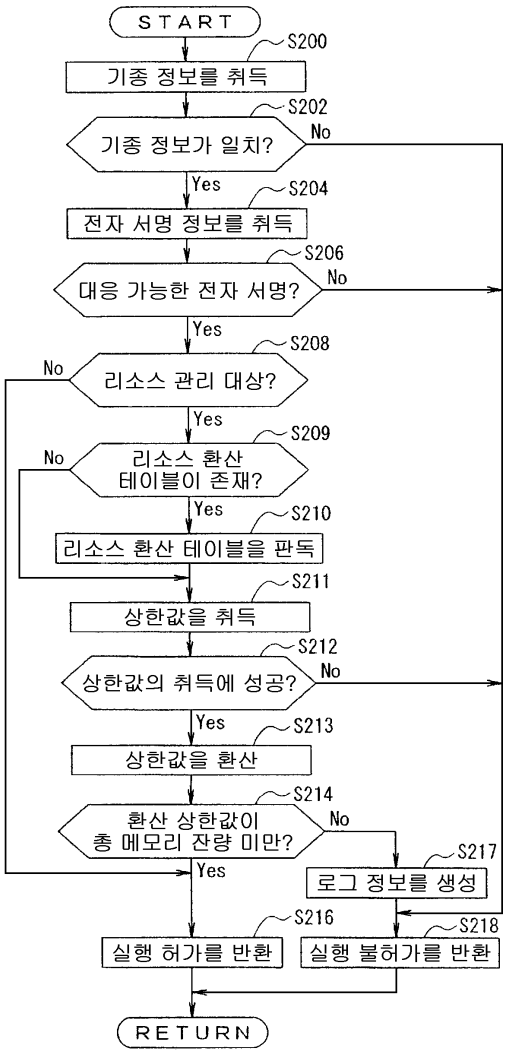
리소스명	환산율
메모리	1.5
클래스수	1

도면24

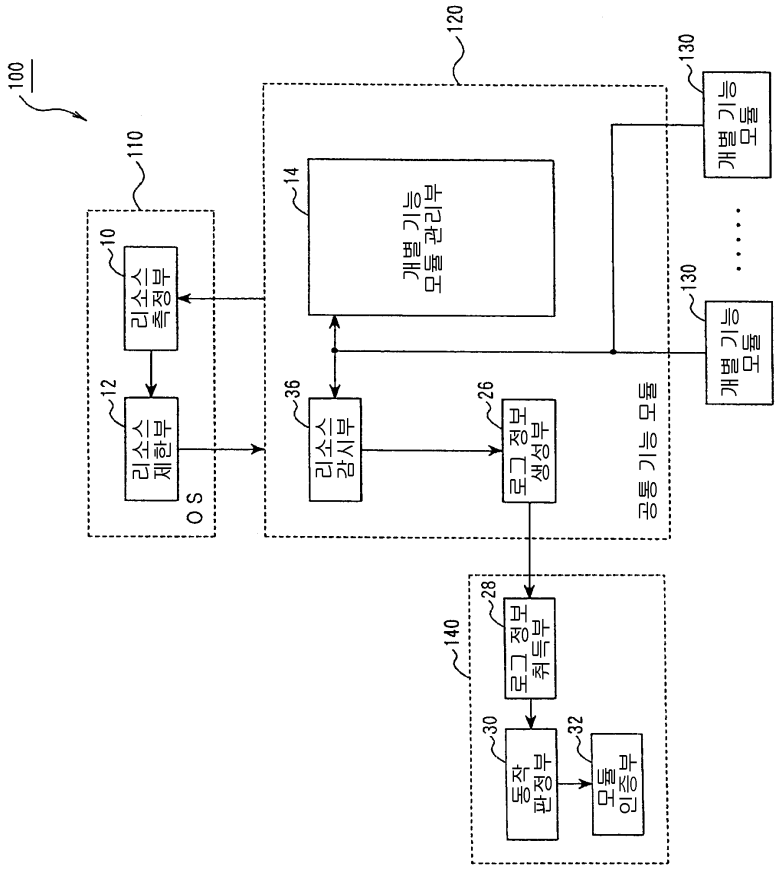


리소스명	상한값	환산값	현재값
메모리	1,000,000	666,666	450,000 (내역 메모리 사용 형태 A 200,000 메모리 사용 형태 B 100,000 메모리 사용 형태 C 150,000)
클래스수	100	20	20

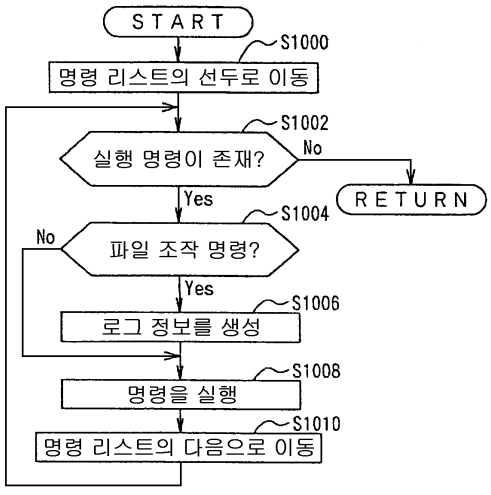
도면25


START
S200
기종 정보를 취득
S202
기종 정보가 일치?
No
Yes
S204
전자 서명 정보를 취득
S206
대응 가능한 전자 서명?
No
Yes
S208
No
리소스 관리 대상?
Yes
S209
No
리소스 환산
테이블이 존재?
Yes
S210
리소스 환산 테이블을 판독
S211
상한값을 취득
S212
상한값의 취득에 성공?
No
Yes
S213
상한값을 환산
S214
환산 상한값이
총 메모리 잔량 미만?
No
Yes
S217
로그 정보를 생성
S216
S218
실행 허가를 반환
실행 불허가를 반환
RETURN

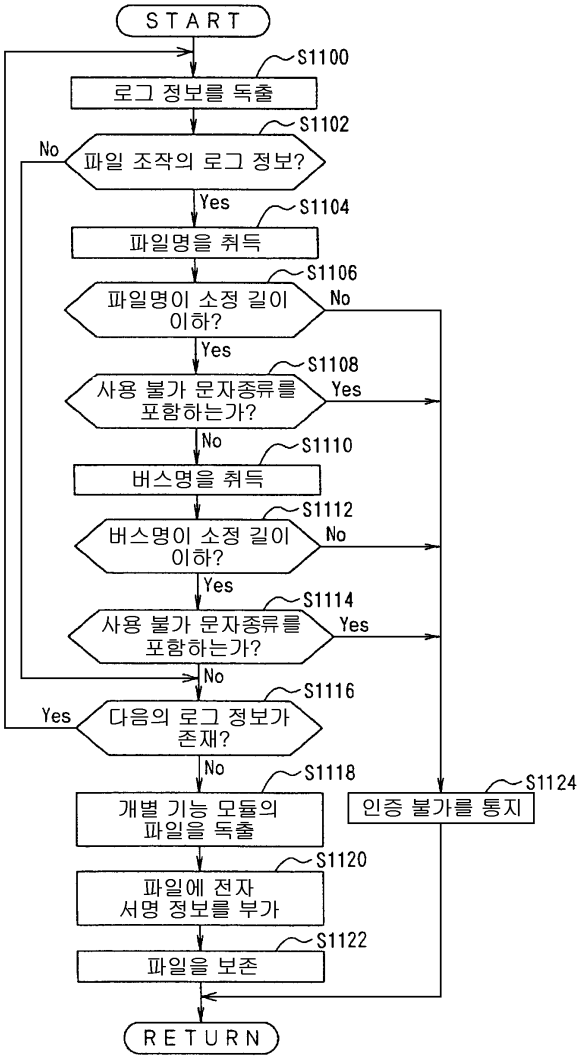
도면26


100
110
120
130
14
10
12
36
26
28
140
30
32
OS
리소스 측정부
리소스 제한부
리소스 감시부
로그 정보 생성부
개별 기능 모듈 관리부
개별 기능 모듈
개별 기능 모듈
공통 기능 모듈
로그 정보 취득부
동작 판정부
모듈 인증부

도면27


START
S1000
명령 리스트의 선두로 이동
S1002
실행 명령이 존재?
No
RETURN
Yes
S1004
No
파일 조작 명령?
Yes
S1006
로그 정보를 생성
S1008
명령을 실행
S1010
명령 리스트의 다음으로 이동

도면28


START
S1100
로그 정보를 독출
S1102
No
파일 조작의 로그 정보?
Yes
S1104
파일명을 취득
S1106
파일명이 소정 길이 이하?
No
Yes
S1108
사용 불가 문자종류를 포함하는가?
Yes
No
S1110
버스명을 취득
S1112
버스명이 소정 길이 이하?
No
Yes
S1114
사용 불가 문자종류를 포함하는가?
Yes
No
S1116
Yes
다음의 로그 정보가 존재?
No
S1118
개별 기능 모듈의 파일을 독출
S1120
파일에 전자 서명 정보를 부가
S1122
파일을 보존
S1124
인증 불가를 통지
RETURN

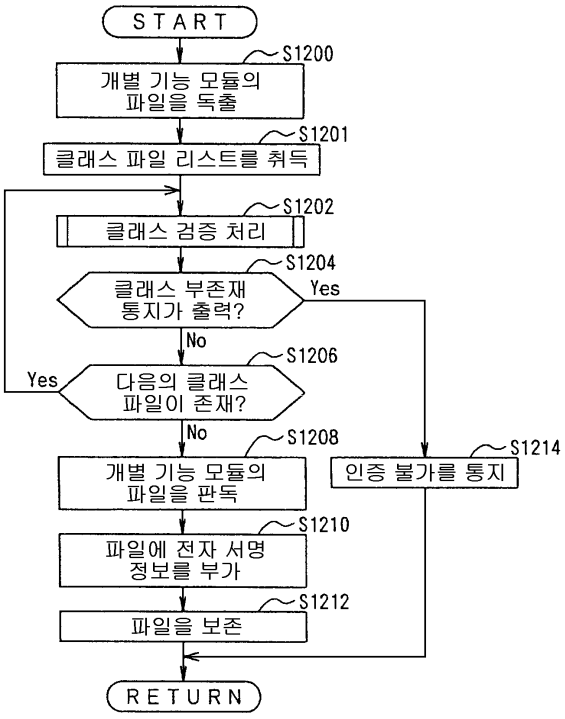
도면29

시각	로그 내용
2004/10/10 09:00:00	개별 기능 모듈을 기동
2004/10/10 09:00:00	파일명 상한값 :16, 버스명 상한값 :20
2004/10/10 09:01:00	파일명/path/fileA.dat 작성
2004/10/10 09:05:00	파일명 /path/longlonglonglonglonglongdirectory/fileB.dat 작성
2004/10/10 09:06:00	파일명/path/longlonglonglonglonglonglongfileC.dat 작성
2004/10/10 09:10:00	개별 기능 모듈을 정지

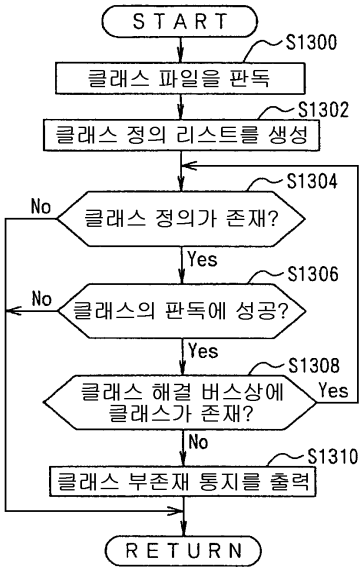
도면30

시각	로그 내용
2004/10/10 09:00:00	개별 기능 모듈을 기동
2004/10/10 09:00:00	파일명 상한값 :16, 버스명 상한값 :20
2004/10/10 09:01:00	파일명 /path/fileA.dat 작성
2004/10/10 09:10:00	개별 기능 모듈을 정지

도면31



도면32


START
S1300
클래스 파일을 판독
S1302
클래스 정의 리스트를 생성
S1304
No
클래스 정의가 존재?
Yes
S1306
No
클래스의 판독에 성공?
Yes
S1308
클래스 해결 버스상에
클래스가 존재?
Yes
No
S1310
클래스 부존재 통지를 출력
RETURN