



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2012년03월13일
(11) 등록번호 10-1121382
(24) 등록일자 2012년02월21일

(51) 국제특허분류(Int. Cl.)
G06F 9/46 (2006.01) G11B 20/10 (2006.01)
G11B 27/10 (2006.01) H04N 5/92 (2006.01)
(21) 출원번호 10-2005-7018474
(22) 출원일자(국제) 2004년11월09일
심사청구일자 2009년11월06일
(85) 번역문제출일자 2005년09월29일
(65) 공개번호 10-2007-0005449
(43) 공개일자 2007년01월10일
(86) 국제출원번호 PCT/JP2004/016598
(87) 국제공개번호 WO 2005/045840
국제공개일자 2005년05월19일
(30) 우선권주장
JP-P-2003-00380464 2003년11월10일 일본(JP)
JP-P-2004-00261376 2004년09월08일 일본(JP)
(56) 선행기술조사문헌
KR1020030007706 A
KR1020070017099 A
EP01357749 A1

(73) 특허권자
파나소닉 주식회사
일본 오오사카후 가도마시 오오아자 가도마 1006
반치
(72) 발명자
이와모토 히로아키
일본국 오오사카후 히라카타시 기쿠가오카미나미
마치 5-1-508
이케다 와타루
일본국 오오사카후 오오사카시 니시쿠 교마치보리
1-15-18-1202
(뒷면에 계속)
(74) 대리인
김영철

전체 청구항 수 : 총 3 항

심사관 : 황승희

(54) 발명의 명칭 기록매체, 재생장치, 재생방법

(57) 요약

BD-ROM에는 AVClip 및 플레이리스트 정보로 이루어지는 플레이리스트와, 애플리케이션과, BD-J Object가 기록되어 있다. 애플리케이션은 가상머신용 프로그래밍 언어로 기술된 프로그램이며, 가상머신에 의한 실행이 가능해지는 생존구간이 미리 규정되어 있다. 상기 BD-J Object에는 플레이리스트 관리테이블이 포함되어 있다. 플레이리스트 관리테이블은 애플리케이션의 생존구간에서, 각 애플리케이션의 실행과 동시에 행해야 할 플레이리스트의 재생제어를 나타낸다.

대표도 - 도11



(72) 발명자

오카다 도모유키

일본국 나라켄 나라시 도미오모토마치 1-8-19-303

오쿠보 마사후미

일본국 오오사카후 오오사카시 요도가와쿠 노나카
미나미1-4-40-440

특허청구의 범위

청구항 1

삭제

청구항 2

애플리케이션과, 디지털 스트림과, 상기 애플리케이션 및 상기 디지털 스트림에 관련하는 타이틀과, 상기 디지털 스트림의 재생경로를 규정하는 플레이리스트정보를 복수 포함하는 관리정보가 기록된 기록매체로,

상기 애플리케이션은,

가상머신용 프로그래밍 언어로 기술된 프로그램이고,

상기 기록매체에 기록된 타이틀의 재생구간에 있어서, 상기 가상머신의 워크 메모리 상에 상기 애플리케이션이 판독되어, 상기 가상머신에 의한 상기 애플리케이션의 실행이 가능해지는 생존구간이 미리 규정되어 있으며,

상기 관리정보는, 상기 생존구간에 있어서, 애플리케이션의 실행과 동시에, 자동으로 재생을 개시하는 취지를 나타내는 재생속성이 부가된 상기 플레이리스트정보에 나타나는 재생경로에 따라서 상기 타이틀에 관련하는 디지털 스트림의 재생제어를 나타내는 정보를 포함하는 것을 특징으로 하는 기록매체.

청구항 3

기록매체로부터 워크 메모리 상에 판독된 애플리케이션을 실행하는 가상머신부와,

상기 기록매체에 기록된 디지털 스트림으로, 상기 기록매체에 기록된 타이틀에 관련하는 디지털 스트림을 재생하는 재생제어엔진부와,

상기 기록매체에 기록된 타이틀의 재생구간에 있어서, 상기 가상머신부의 워크 메모리 상에 상기 애플리케이션이 판독되어, 상기 가상머신부에 의한 상기 애플리케이션의 실행이 가능해지는 생존구간이 도래하면, 상기 기록매체에 기록된 애플리케이션을 상기 워크 메모리에 판독하여 상기 가상머신부에 실행시키고, 상기 가상머신부에 의한 실행과 동시에, 상기 기록매체에 기록된 관리정보에 포함되는 상기 디지털 스트림의 재생경로를 규정하는 복수의 플레이리스트정보 중 자동으로 재생을 개시하는 취지를 나타내는 재생속성이 부가된 플레이리스트정보에 나타나는 재생경로에 따라서 상기 타이틀에 관련하는 디지털 스트림의 재생을 상기 재생제어엔진부에 실행시키는 애플리케이션 매니저부를 구비하는 것을 특징으로 하는 재생장치.

청구항 4

기록매체로부터 워크 메모리 상에 판독된 애플리케이션을 실행하는 가상머신부와, 상기 기록매체에 기록된 디지털 스트림으로, 상기 기록매체에 기록된 타이틀에 관련하는 디지털 스트림을 재생하는 재생제어엔진부를 가진 컴퓨터에 대한 재생방법으로,

상기 기록매체에 기록된 타이틀의 재생구간에 있어서, 상기 가상머신부의 워크 메모리 상에 상기 애플리케이션이 판독되어, 상기 가상머신부에 의한 상기 애플리케이션의 실행이 가능해지는 생존구간이 도래하면, 상기 기록매체에 기록된 애플리케이션을 상기 워크 메모리에 판독하여 상기 가상머신부에 실행시키는 단계와,

상기 가상머신부에 의한 실행과 동시에, 상기 기록매체에 기록된 관리정보에 포함되는 상기 디지털 스트림의 재생경로를 규정하는 복수의 플레이리스트정보 중 자동으로 재생을 개시하는 취지를 나타내는 재생속성이 부가된 플레이리스트정보에 나타나는 재생경로에 따라서 상기 타이틀에 관련하는 디지털 스트림의 재생을 상기 재생제어엔진부에 실행시키는 단계를 컴퓨터에 실행시키는 것을 특징으로 하는 재생방법.

청구항 5

삭제

청구항 6

삭제

청구항 7

삭제

청구항 8

삭제

청구항 9

삭제

청구항 10

삭제

청구항 11

삭제

청구항 12

삭제

청구항 13

삭제

청구항 14

삭제

청구항 15

삭제

청구항 16

삭제

청구항 17

삭제

청구항 18

삭제

청구항 19

삭제

청구항 20

삭제

청구항 21

삭제

명세서

기술분야

[0001] 본 발명은 가상머신을 대상으로 한 애플리케이션의 실행을 제어하는 애플리케이션 제어기술 분야에 속하는 발명이며, 본 제어기술을 BD-ROM 등의 영화작품 배포를 위한 기록매체, BD-ROM을 대상으로 한 재생장치 등에 응용하는 경우의 응용기술에 깊이 관계한다.

배경 기술

[0002] Java(등록상표) 프로그래밍 등, 가상머신을 대상으로 한 애플리케이션 제어기술은 퍼스널 컴퓨터 소프트웨어 업계에 깊이 침투되어 있다. 퍼스널 컴퓨터 소프트웨어로부터 더 발전하여, 현재 Blu-ray Disc Read Only Memory(BD-ROM)용 재생장치에서의 재생제어를 Java(상표등록) 프로그래밍으로 실현하기 위한 여러 연구가 검토되고 있다.

[0003] 이와 같은 재생장치에 대한 선행기술로는 일본국 특허 2813245호 공보에 기재된 것이 알려져 있다.

[0004] 그런데 Java(등록상표) 프로그래밍으로 작성된 애플리케이션의 동작이라는 것은, 리소스(Resource)의 사용상황이나 CPU의 부하에 따라 변동되는 불안정한 것이다. 리소스가 고갈되어 있음에 따라, 애플리케이션의 기동에 실패하거나, 애플리케이션이 이상종료하거나 하여 장치가 블랙아웃(black out) 되는 것은 퍼스널컴퓨터 소프트웨어 업계에서는 "흔히 있는 일"로 치부될지 모르지만, BD-ROM 재생장치의 제조에 관계하는 민생기기 분야에서는 품질 문제가 될지도 모른다. 품질 문제에 대한 과급이 있을 수 있으므로, 각 제조회사는 Java(등록상표) 프로그래밍에 의한 장치제어의 실현에 주저하는 일이 많았다(주; 블랙아웃이란, 장치의 소프트웨어가 갑자기 멈추어서(freeze) 표시화면이 검게 되는 상태를 말한다).

발명의 상세한 설명

[0005] 본 발명의 목적은 기록매체에 대한 제어를 실현하는 애플리케이션이 이상종료한 경우, 또는 기동에 실패한 경우의 패일 세이프(Fail Safe)를 실현할 수 있는 기록매체를 제공하기 위한 것이다.

[0006] 상기 목적을 달성하기 위해, 본 발명에 관한 기록매체는, 애플리케이션과, 디지털 스트림과, 상기 애플리케이션 및 상기 디지털 스트림에 관한 타이틀과, 상기 디지털 스트림의 재생경로를 규정하는 플레이리스트정보를 복수 포함하는 관리정보가 기록된 기록매체로, 상기 애플리케이션은, 가상머신용 프로그래밍 언어로 기술된 프로그램이고, 상기 기록매체에 기록된 타이틀의 재생구간에 있어서, 상기 가상머신의 워크 메모리 상에 상기 애플리케이션이 판독되어, 상기 가상머신에 의한 상기 애플리케이션의 실행이 가능해지는 생존구간이 미리 규정되어 있으며, 상기 관리정보는, 상기 생존구간에 있어서, 애플리케이션의 실행과 동시에, 자동으로 재생을 개시하는 취지를 나타내는 재생속성이 부가된 상기 플레이리스트정보에 나타나는 재생경로에 따라서 상기 타이틀에 관한 디지털 스트림의 재생제어를 나타내는 정보를 포함하는 것을 특징으로 하고 있다.

[0007] 본 발명에 관한 기록매체에 의하면, 애플리케이션 실행과 동시에 행해야 할 디지털 스트림의 재생제어가 생존구간별로 규정되어 있다. 그 때문에, 애플리케이션의 기동에 실패하거나, 애플리케이션의 실행 도중에 설령 애플리케이션이 이상종료했다 해도, 동시에 이루어지고 있는 디지털 스트림 재생을 계속함으로써, "표시화면에 무엇인가가 표시되고 있는 상태"를 제공할 수 있다. 이로써, 장치가 블랙아웃 되는 최악의 케이스를 회피할 수 있으므로, 장치를 제조하는 제조회사에 최저한도의 안도감을 줄 수 있다.

[0008] 이러한 안도감이 주어지므로, 품질 문제로 과급되는 점에 신경을 쓰고 있는 각 제조회사에 대해서도, Java(등록상표) 애플리케이션에 의한 재생장치의 제어의 개발을 적극 후원할 수 있다. 이러한 적극적인 후원에 의해, 재생장치의 저가격화, 다양화가 진행되면, BD-ROM 콘텐츠의 충실화를 도모할 수 있으므로, 콘텐츠 관련산업의 발달을 적극적으로 선도할 수 있다.

실시 예

[0108] (제 1 실시 예)

[0109] 이후, 본 발명에 관한 기록매체의 실시 예에 대해 설명한다. 우선 먼저, 본 발명에 관한 기록매체의 실시행위 중, 사용행위에 대한 예를 설명한다. 도 1은 본 발명에 관한 기록매체의 사용행위에 대한 예를 나타내는 도면이다. 도 1에서, 본 발명에 관한 기록매체는 BD-ROM(100)이며, BD-ROM(100)은, 재생장치(200), 리모컨(300), TV(400)에 의해 형성되는 홈시어터 시스템에 저작물을 공급하는 용도로 제공된다.

[0110] 이 중 리모컨(400)에는, 재생개시(Play), 재생정지(Stop), 일시정지(Pause On), 일시정지 해제(Pause Off), 스틸 기능 해제(still off), 속도지정부 빨리 감기(Forward Play(speed)), 속도지정부 되감기(Backward

Play(speed)), 음성 전환(Audio Change), 부영상 전환(Subtitle Change), 앵글전환(Angle Change)을 접수하는 키나, 메뉴 조작시의 포커스 이동 조작을 접수하는 MoveUp 키, MoveDown 키, MoveRight 키, MoveLeft 키, 메뉴 표시조작을 접수하는 Pop-up 키, 수치 입력을 접수하는 Numeric 키가 존재한다.

- [0111] 이상이 본 발명에 관한 기록매체의 사용 예에 대한 설명한다.
- [0112] 이어, 본 발명에 관한 기록매체의 생산 행위에 대해 설명한다. 본 발명에 관한 기록매체는 BD-ROM의 파일시스템 상에서의 개량에 의해 실현할 수 있다. 도 2는 BD-ROM에서의 파일 디렉터리 구성을 나타내는 도면이다. 본 도면에서 BD-ROM에는 Root 디렉터리 아래에 BDMV 디렉터리가 있다.
- [0113] BDMV 디렉터리에는 확장자 bdmv가 부여된 파일(index.bdmv, MovieObject. bdmv, BD-J Object.bdmv)이 있다. 그리고 이 BDMV 디렉터리 밑에는 PLAYLIST 디렉터리, CLIPINF 디렉터리, STREAM 디렉터리, BDA 디렉터리라고 불리는 4개의 서브 디렉터리가 더 존재한다.
- [0114] PLAYLIST 디렉터리에는 확장자 mpls가 부여된 파일(00001.mpls, 00002. mpls, 00003mpls)이 있다.
- [0115] CLIPINF 디렉터리에는 확장자 clpi가 부여된 파일(00001.clpi, 00002.clpi, 00003.clpi)이 있다.
- [0116] STREAM 디렉터리에는 확장자 m2ts가 부여된 파일(00001.m2ts, 00002.m2ts, 00003.m2ts)이 있다.
- [0117] BDA 디렉터리에는 확장자 jar이 부여된 파일(00001.jar, 00002.jar, 00003jar)이 있다. 이상의 디렉터리 구조에 의해 서로 다른 종별의 복수 파일이 BD-ROM 상에 배치되어 있음을 알 수 있다.
- [0118] 본 도면에서 확장자. m2ts가 부여된 파일(00001.m2ts, 00002.m2ts, 00003.m2ts, ……)은 AVClip를 저장하고 있다. AVClip에는 MainClip, SubClip이라고 하는 종별이 있다. MainClip은, 비디오 스트림, 오디오 스트림, 자막을 구성하는 프레젠테이션 그래픽스 스트림(Presentation Graphics Stream, PG 스트림), 메뉴를 구성하는 인터랙티브 그래픽스 스트림(Interactive Graphics Stream, IG 스트림)과 같은 복수 엘리멘터리 스트림(Elementary Stream)을 다중화함으로써 얻어진 디지털 스트림이다.
- [0119] SubClip은, 오디오 스트림, 그래픽스 스트림, 텍스트 자막 스트림(TextSTStream) 등, 하나의 엘리멘터리 스트림만에 해당하는 디지털 스트림이다.
- [0120] 확장자 "clpi"가 부여된 파일(00001.clpi, 00002.clpi, 00003.clpi, ……)은 AVClip의 각각에 1대 1로 대응하는 관리정보이다. 관리정보이므로, Clip 정보는 AVClip에서의 스트림의 부호화 형식, 프레임 레이트, 비트 레이트, 해상도 등의 정보나 GOP의 선두위치를 나타내는 EP_map을 가지고 있다.
- [0121] 확장자 "mpls"가 부여된 파일(00001.mpls, 00002.mpls, 00003.mpls, ……)은 플레이리스트 정보를 저장한 파일이다. 플레이리스트 정보는 AVClip을 참조하여 플레이리스트를 정의하는 정보이다. 도 3은 PL 정보의 구성을 나타내는 도면이며, 본 도면의 좌측에 나타낸 바와 같이, 플레이리스트 정보는 『MainPath 정보』, 『PLMark 정보』, 『SubPath 정보』로 구성된다.
- [0122] MainPath 정보(MainPath())는 점선 화살표 mp1로 나타내는 바와 같이, 복수의 PlayItem 정보(PlayItem())로 이루어진다. PlayItem이란, 하나 이상의 AVClip 시간 축 상에서 In_Time, Out_Time을 지정함으로써 정의되는 재생 구간이다. PlayItem 정보를 복수 배치함으로써, 복수 재생구간으로 이루어지는 플레이리스트(PL)가 정의된다. 도면 중의 점선 mp2는 PlayItem 정보의 내부 구성을 클로즈업하고 있다. 본 도면에 나타내는 바와 같이, PlayItem 정보는, 대응하는 AVClip을 나타내는 『Clip_information_file_name』과, 『In_time』과, 『Out_time』으로 이루어진다. 도 4는 AVClip과 PL의 관계를 나타내는 도면이다. 제 1단계는 AVClip이 갖는 시간 축을 나타내고, 제 2단계는 PL이 갖는 시간 축을 나타낸다. PL 정보는 PlayItem #1, #2, #3이라는 3개의 PlayItem 정보를 포함하고 있으며, 이들 PlayItem #1, #2, #3의 In_time, Out_time에 의해 3개의 재생구간이 정의되게 된다. 이들 재생구간을 배열시키면, AVClip 시간 축과는 다른 시간 축이 정의되게 된다. 이것이 제 2단계에 나타내는 PL 시간 축이다. 이와 같이, PlayItem 정보의 정의에 의해 AVClip과는 다른 시간 축의 정의가 가능해진다.
- [0123] AVClip에 대한 지정은, 원칙적으로는 하나이지만, 복수 AVClip에 대한 일괄지정도 있을 수 있다. 이 일괄지정은 PlayItem 정보에서의 복수의 Clip_Information_ file_name에 의해 이루어진다. 도 5는 4개의 Clip_Information_file_name에 의해 이루어진 일괄지정을 나타내는 도면이다. 본 도면에서 제 1단계~제 4단계는 4개의 AVClip 시간 축(AVClip #1, #2, #3, #4의 시간 축)을 나타내고, 제 5단계는 PL 시간 축을 나타낸다. PlayItem 정보가 갖는 4개의 Clip_Information_file_name으로 이들 4개의 시간 축이 지정되어 있다. 이렇

게 함으로써, PlayItem이 갖는 In_time, Out_time에 의해 택일적으로 재생 가능한 4개의 재생구간이 정의되게 된다. 이로써, PL 시간 축에는 절환 가능한 복수 앵글 영상으로 이루어지는 구간(이른바 멀티앵글구간)이 정의되게 된다.

[0124] PLmark 정보(PLmark())는 PL 시간 축 중, 임의의 구간을 캡처로서 지정하는 정보이다. 도 6은 PLmark 정보의 내부 구성을 나타내는 도면이며, 본 도면의 인출선 pm1으로 나타내는 바와 같이, PLmark 정보는 『ref_to_PlayItem_Id』와 『Mark_time_stamp』를 포함한다. 도 7은 PLmark에 의한 캡처 정의를 나타내는 도면이다. 본 도면에서 제 1단계는 AVClip 시간 축을 나타내고, 제 2단계는 PL 시간 축을 나타낸다. 도면 중의 화살표 pk1, 2는, PLmark에서의 PlayItem 지정(ref_to_PlayItem_Id)과, 일 시점의 지정(mark_time_stamp)을 나타낸다. 이들의 지정에 의해 PL 시간 축에는 3개의 캡처(Chapter #1, #2, #3)가 정의되게 된다. 이상이 PLmark에 대한 설명이다. 이어, SubPath 정보에 대해 설명한다.

[0125] SubPath 정보(SubPath())는 SubClip의 시간 축 상에 In_Time, Out_Time을 지정함으로써 하나 이상의 재생구간을 정의하는 정보이며, 도 8에 도시하는 내부 구성을 가지고 있다. 본 도면에 나타내는 바와 같이, SubPath 정보는, 점선의 인출선 sh1으로 나타내는 것과 같이 복수의 SubPlayItem 정보(SubPlayItem())로 이루어진다. 점선 sh2를 이용하여 클로즈업하고 있는 바와 같이, SubPlayItem 정보는, 『Clip_information_file_name』과 『In_time』과 『Out_time』과 『Sync_PlayItem_Id』와 『Sync_start_Pts_of_PlayItem』으로 이루어진다. SubClip의 시간 축 상에 대한 In_Time, Out_Time의 지정은 『Clip_information_file_name』, 『In_time』, 『Out_time』에 의해 이루어진다. 『Sync_PlayItem_Id』 및 『Sync_start_Pts_of_PlayItem』은 SubClip 시간 축 상의 재생구간과 PL 시간 축을 동기시키는 동기지정을 이룬다. 이 동기지정에 의해 PL 시간 축과 SubPlayItem 시간 축은 동기하여 진행하게 된다.

[0126] 도 9는 SubPlayItem 시간 축 상의 재생구간 정의와, 동기지정을 나타내는 도면이다. 본 도면에서 제 1단계는 PL 시간 축을 나타내고, 제 2단계는 SubPlayItem 시간 축을 나타낸다. 도면 중의 SubPlayItem.IN_time은 재생구간의 시작점을, SubPlayItem.Out_time은 재생구간의 끝점을 각각 나타낸다. 이에 의해, SubClip 시간 축 상에도 재생구간이 정의되어 있음을 알 수 있다. 화살표 Sn1에서 Sync_PlayItem_Id는 PlayItem에 대한 동기지정을 나타내고, 화살표 Sn2에서 sync_start_PTS_of_PlayItem은 PL 시간 축에서의 PlayItem 상의 일 시점의 지정을 나타낸다.

[0127] 복수 AVClip의 절환을 가능하게 하는 멀티앵글 구간이나 AVClip-SubClip을 동기시킬 수 있는 동기구간의 정의를 가능하게 하는 것이 BD-ROM에서의 플레이리스트 정보의 특징이다. 이상의 Clip 정보 및 플레이리스트 정보는 "정적 시나리오"로 분류된다. 왜냐하면, 이상의 Clip 정보 및 플레이리스트 정보에 의해 정적 재생 단위인 PL이 정의되기 때문이다. 이상으로 정적 시나리오에 대한 설명을 마친다.

[0128] 이어, "동적 시나리오"에 대해 설명한다. 동적 시나리오란, AVClip의 재생제어를 동적으로 규정하는 시나리오 데이터이다. "동적으로"라는 것은 재생장치에서의 상태 변화나 사용자로부터의 키 이벤트에 의해 재생제어의 내용이 바뀌는 것을 말한다.

[0129] BD-ROM에서는, 이 재생제어의 동작환경으로 2개의 모드를 상정하고 있다. 첫 번째는 DVD 재생장치의 동작환경과 매우 비슷한 동작환경이며, 코맨드 베이스(command-based)의 실행 환경이다. 두 번째는 Java(등록상표) 가상머신의 동작환경이다. 이들 2개의 동작환경 중 첫 번째는 HDMV 모드라 불린다. 두 번째는 BD-J 모드라 불린다. 이들 2개의 동작환경이 있으므로, 동적 시나리오는 이중 어느 하나의 동작환경을 상정하여 기술된다. HDMV 모드를 상정한 동적 시나리오는 Movie Object라 불리며, 관리정보에 의해 정의된다. 한편, BD-J 모드를 상정한 동적 시나리오는 BD-J Object라 불린다.

[0130] 우선 먼저, Movie Object에 대해 설명한다.

[0131] <Movie Object>

[0132] Movie Object는 MovieObject.bdmv라고 하는 파일에 저장된다. 도 10은 MovieObject.bdmv의 내부 구성을 나타내는 도면이다. 본 도면의 좌측 단에 나타난 바와 같이, MovieObject.bdmv는, 코드 열 "MOBJ"를 나타내는 『type_indicator』와, 『version_number』와, 하나 이상의 MovieObject인 『MovieObjects()』로 이루어진다. 도면 중의 인출선 vh1은 MovieObjects의 내부 구성을 클로즈업하고 있다. MovieObjects()는 자신의 데이터 길이인 『length』와, 자신에 포함되는 MovieObject의 개수인 『number_of_mobjs』와, number_of_mobjs 개의 MovieObject인 『MovieObjects』로 이루어진다. 이들 number_of_mobjs 개의 MovieObject는 식별자 mobj_id로서 식별된다. 도면 중의 인출선 vh2는 식별자 mobj_id에 의해 특정되는 임의의 MovieObject[mobj_id]()의 내부 구

성을 클로즈업하고 있다.

- [0133] 이 인출 선으로 나타낸 바와 같이, MoveiObject는 MenuCall이 이루어졌을 때, MenuCall 후의 재생 재개를 의도하고 있는지의 여부를 나타내는 『resume_intention_flag』, MenuCall을 마스크 할 것인가 여부를 나타내는 정보 『menu_call_mask』, 타이틀 서치 기능을 마스크 할 것인가를 나타내는 『title_search_flag』, 내비게이션 코맨드(Navigation command)의 개수인 『number_of_navigation_command』, number_of_navigation_command 개의 『내비게이션 코맨드』로 이루어진다.
- [0134] 내비게이션 코맨드 열은 조건 분기, 재생장치에서의 상태 레지스터의 설정, 상태 레지스터의 설정 값 취득 등을 실현하는 코맨드 열로 이루어진다. Movie Object에서 기술 가능한 코맨드를 이하에 나타낸다.
- [0135] PlayPL 코맨드
- [0136] 서식: PlayPL(제 1 인수, 제 2 인수)
- [0137] 제 1 인수는 플레이리스트의 번호로, 재생해야 할 PL을 지정할 수 있다. 제 2 인수는 그 PL에 포함되는 PlayItem이나, 그 PL에서의 임의의 시각, Chapter, Mark를 이용하여 재생개시 위치를 지정할 수 있다.
- [0138] PlayItem에 의해 PL 시간 축 상의 재생개시 위치를 지정한 PlayPL 기능을 PlayPLatPlayItem(), Chapter에 의해 PL 시간 축 상의 재생개시 위치를 지정한 PlayPL 기능을 PlayPLatChapter(), 시각 정보에 의해 PL 시간 축 상의 재생개시 위치를 지정한 PlayPL 기능을 PlayPLatSpecified Time()이라고 한다.
- [0139] JMP 코맨드
- [0140] 서식: JMP 인수
- [0141] JMP 코맨드는 현재의 동적 시나리오를 도중에 폐기하고(discard), 인수(引數)인 분기처 동적 시나리오를 실행하는 분기이다. JMP 명령의 형식에는 분기처 동적 시나리오를 직접 지정하고 있는 직접 참조인 것과, 분기처 동적 시나리오를 간접 참조하고 있는 간접 참조인 것이 있다.
- [0142] Movie Object에서의 내비게이션 코맨드의 기술은 DVD에서의 내비게이션 코맨드의 기술 방식과 매우 유사하므로, DVD 상의 디스크 콘텐츠를 BD-ROM에 이식하는 작업을 효율적으로 행할 수 있다. Movie Object에 대해서는 이하의 국제공개공보에 기재된 선행기술이 존재한다. 상세한 내용에 대해서는 본 국제공개공보 W02004/074976을 참조하길 바란다.
- [0143] 이상으로 Movie Object에 대한 설명을 끝낸다. 이어 BD-J Object 에 대해 설명한다.
- [0144] <BD-J Object>
- [0145] BD-J Object는 Java(등록상표) 프로그래밍 환경에서 기술된 BD-J 모드의 동적 시나리오이다.
- [0146] 도 11은 BD-J Object.bdmv의 내부 구성을 나타내는 도면이다. 본 도면의 좌단에 나타내는 바와 같이, BD-J Object.bdmv는, 코드 열 "BOBJ"를 나타내는 『type_indicater』와, 『version_number』와, 하나 이상의 BD-J Object인 『BD-J Objects()』로 이루어진다. 도면 중의 인출선 bh1은 BD-J Objects의 내부 구성을 클로즈업하고 있다. BD-J Objects()는 자신의 데이터 길이인 『length』와, 자신에 포함되는 BD-J Object의 개수인 『number_of_bobj』와, number_of_bobj개의 BD-J Object인 『BD-J Objects』로 이루어진다. 이들 number_of_bobj개의 BD-J Object는 식별자 bobj_id로 식별된다. 도면 중의 인출선 bh2는 식별자 bobj_id에 의해 특정되는 임의의 BD-J Object[bobj_id]()의 내부 구성을 클로즈업하고 있다.
- [0147] 이 인출 선으로 나타내는 바와 같이, 본 도면에 도시는 것과 같은 BD-J Object는, 『resume_intention_flag[bobj_id]』와, 『menu_call_mask[bobj_id]』와, 『title_search_flag[bobj_id]』와, 『Application_Management_Table[bobj_id]』와, 『Playlist_Management_Table[bobj_id]』로 이루어진다. 『resume_intention_flag』, 『menu_call_mask』, 『title_search_flag』를 포함하고 있다는 점에서 BD-J Object는 Movie Object와 거의 동일하다.
- [0148] Movie Object와의 차이는, BD-J Object에 코맨드가 직접 기술되어 있지 않은 점이다. 즉, Movie Object에서 제어순서는 내비게이션 코맨드에 의해 직접 기술되어 있었다. 이에 대해 BD-J Object에서는 Java(등록상표) 애플리케이션에 대한 지정을 『Application_Management_Table[bobj_id]』상에 기재함으로써, 간접적으로 제어순서를 규정하고 있다. 이러한 간접적인 규정에 의해 복수 동적 시나리오에서 제어순서를 공통화한다고 하는 제어순서의 공통화를 효율적으로 행할 수 있다.

- [0149] 또, MovieObject에서의 PL 재생은 PL 재생을 명하는 내비게이션 코맨드(PlayPl 코맨드)의 기술에 의해 이루어졌지만, BD-J Object에서의 PL 재생은 PL 재생순서를 나타내는 『Playlist_Management_Table[bobj_id]』를 BD-J Object에 포함함으로써 기술이 가능해진다. 그뿐 아니라, 애플리케이션 관리테이블에서 참조되는 애플리케이션에 PL 재생순서를 포함함으로써 기술이 가능해진다. 즉, 플레이리스트 관리테이블의 기술, 애플리케이션에서의 PL 재생순서의 기술 중 어느 것이나 플레이리스트 재생의 순서를 포함시켜 둘 수 있다.
- [0150] 여기서, Java(등록상표) 애플리케이션에 대해 설명한다. Java(등록상표) 애플리케이션은 가상머신의 힙영역(heap area)(워크 메모리라고도 불린다)에 로드된 하나 이상의 xlet 프로그램으로 이루어진다. 이 워크 메모리에 로드된 xlet 프로그램 및 데이터로부터 애플리케이션은 구성되게 된다. 이상이 Java(등록상표) 애플리케이션의 구성이다.
- [0151] 이 Java(등록상표) 애플리케이션의 실체에 해당하는 것이 도 2에서의 BDMV 디렉터리 밑의 BDJA 디렉터리에 저장된 Java(등록상표) 아카이브 파일(00001.jar, 00002.jar)이다. 이후, Java(등록상표) 아카이브 파일에 대해 도 12를 참조하면서 설명한다.
- [0152] <Java(등록상표) 아카이브 파일>
- [0153] Java(등록상표) 아카이브 파일(도 2의 00001.jar, 00002.jar)은 하나 이상의 클래스파일(class file), 하나 이상의 데이터 파일 등을 하나로 정리함으로써 얻어지는 파일이다. 도 12(a)는 아카이브 파일에 의해 저장되어 있는 프로그램, 데이터를 나타내는 도면이다. 본 도면에서의 데이터는, 프레임 내에 나타내는 디렉터리 구조가 배치된 복수 파일을 Java(등록상표) 아카이버(archiver)에 의해 정리한 것이다. 프레임 내에 나타내는 디렉터리 구조는 Root 디렉터리, Java(등록상표) 디렉터리, image 디렉터리로 이루어지며, Root 디렉터리에 common.pkg가, Java(등록상표) 디렉터리에 클래스파일(aaa.class, bbb.class)이, image 디렉터리에 menu.jpg가 배치되어 있다. Java(등록상표) 아카이브 파일은 이들을 Java(등록상표) 아카이버에 의해 정리함으로써 얻어진다. 이러한 클래스파일 및 데이터는 BD-ROM에서 캐시(cache)로 관독될 때 전개되고, 캐시 상에서 디렉터리에 배치된 복수 파일로서 취급된다. Java(등록상표) 아카이브 파일의 파일명에서의 "zzzzz"라는 5 자릿수의 수치는 애플리케이션 ID(applicationID)를 나타낸다. 본 Java(등록상표) 아카이브 파일이 캐시로 관독되었을 때, 이 파일명에서의 수치를 참조함으로써 임의의 Java(등록상표) 애플리케이션을 구성하는 프로그램, 데이터를 추출할 수 있다.
- [0154] 본 도면에서의 클래스파일(도면 중의 aaa.class, bbb.class)은 상술한 xlet 프로그램에 대응하는 클래스파일이다. BD-J 모드에서의 재생순서는 이 클래스파일의 인스턴스(instance)에 해당하는 xlet 프로그램에 의해 규정된다.
- [0155] xlet 프로그램이란, JMF(Java(등록상표) Media FrameWork) 방식의 인터페이스를 이용할 수 있는 Java(등록상표) 프로그램이며, JMF 등의 방식에 따라 키 이벤트에 기초하는 처리를 행한다. xlet 프로그램은 JMF 방식의 처리가 가능하므로, MPLS 파일에 대한 인스턴스(JMF 플레이어 인스턴스)를 생성함으로써, 플레이리스트 재생을 재생장치에 명할 수 있다. 그 이외에도 xlet 프로그램에서는 기능 API의 콜(call)을 기술함으로써 BD-ROM 재생장치 특유의 처리를 실행시킬 수 있다.
- [0156] 또, xlet 프로그램은 WWW 사이트를 액세스하여 콘텐츠를 다운로드 하는 절차를 실행할 수도 있다. 이로써, 다운로드 콘텐츠와 플레이리스트 재생을 혼합한 작품을 재생시킬 수 있다.
- [0157] xlet 프로그램의 클래스파일에 대해 설명한다. 도 12(b)는 클래스파일의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, 클래스파일은 통상의 클래스파일과 마찬가지로 『콘스탄트 풀(constant pool)』, 『인터페이스』, 『메소드(method) 1, 2, 3, ..., n』으로 이루어진다. 클래스파일에서의 메소드에는 거동(舉動)의 트리거(tirgger)가 되는 키 이벤트가 미리 등록되어 있는 메소드(EventListner)와, JMF의 재생순서를 명하는 메소드(JMF 플레이어 인스턴스의 메소드), BD-ROM 재생장치 측의 기능 API를 콜 하는 메소드가 있다. 이들 메소드는 자신에게 할당된 로컬변수나 자신을 콜 할 때의 인수를 이용하는 것에 의해 연산 등의 순서가 기술되어 있다. 이상이 Java(등록상표) 아카이브 파일에 대한 설명이다. 또, 본 실시 예에서 애플리케이션을 구성하는 프로그램, 데이터는 Java(등록상표) 아카이브 파일에 정리되었으나, LZH 파일, zip 파일이어도 된다.
- [0158] 이상이, BD-J 모드에서의 동적 시나리오에 대한 설명이다.
- [0159] <BD-ROM에서의 상태천이>
- [0160] DVD-Video와 같은 관독 전용 디스크에서 공급되는 디스크 콘텐츠는 톱 메뉴를 중핵으로 한 구조로 되어 있다.

그 톱 메뉴에서 개개의 저작물로 분기하여 재생을 행하고, 그 후 다시 TopMenu Title로 되돌아간다는 독특한 상태천이를 이룬다. 도 13은 디스크 콘텐츠에서의 상태천이를 나타내는 도면이다. 본 도면에서의 사각 틀은 타이틀이다. 타이틀이란, 디스크 콘텐츠 특유의 상태천이에서, 하나의 "상태"에 해당하는 재생 단위이다. 타이틀에는 BD-ROM의 로딩시에 최초로 재생되는 『FirstPlayTitle』, Top-Menu를 구성하는 『Top_menuTitle』, 이들 이외의 일반적인 『Title』이 있다. 또, 도면 중의 화살표 jh1, 2, 3, 4, 5, 6, 7, 8은 타이틀 사이의 분기를 상징적으로 나타낸다. 본 도면에 도시되는 상태천이란, BD-ROM 로딩시에, 『FirstPlayTitle』이 재생되고, 『Top_menuTitle』로의 분기가 발생하여, 톱 메뉴에 대한 선택 대기가 된다고 하는 것이다. BD-ROM과 같은 영화작품 배포용 기록매체 업계에서는, 동적 상표를 로딩시에 재생하는 관습이 정착되어 있다. 이 동적 상표는 영화작품의 제작자나 배포자를 상징하는 것이며, FirstPlayTitle은 BD-ROM이 로딩 되었을 때, 무엇보다도 먼저 이 동적 상표를 재생시킨다는 역할 분담을 담당한다.

[0161] 그리고 사용자에게 의한 메뉴에 대한 선택 조작이 있으면, 선택에 따라서 해당 타이틀의 재생을 행하고, 다시 TopMenu Title로 되돌아가는 처리를 BD-ROM의 이젝트(eject)가 이루어질 때까지 계속 반복하는 것이 디스크 콘텐츠 특유의 상태천이이다.

[0162] 이와 같은 상태천이를 이루는 타이틀은, HDMV 모드의 동적 시나리오, BD-J 모드의 동적 시나리오로 구성된다. 도 14는 HDMV 모드의 동적 시나리오로 구성되는 2개의 타이틀을 나타내는 도면이다. 본 도면의 제 1단계는 식별자 title_id에 의해 식별되는 임의의 Title(title_id)을 나타낸다. 제 2단계는 그 타이틀을 구성하고 있는 하나 이상의 MovieObject로 이루어지는 MovieObject 열을 나타낸다. 제 3단계는 MovieObject를 구성하는 내비게이션 코멘드를 나타낸다.

[0163] 다른 타이틀로의 점프(jump)를 재생장치에 명하는 내비게이션 코멘드(JumpTitle 코멘드)를 MovieObject에 기술해 둠으로써, 도 13에 나타내는 것과 같은, 어떤 타이틀로부터 다른 타이틀로의 분기가 실현되게 된다. 또, PL 재생을 명하는 내비게이션 코멘드(PlayPL 코멘드)를 MovieObject에 기술해 둠으로써, 본 도면 제 4단계에 나타내는 PL이 타이틀로 귀속하게 된다.

[0164] PL을 귀속시킴으로써, HDMV 모드에서의 타이틀에서는 동화상 재생을 수반하는 영화작품을 정의할 수 있다. 이것이 HDMV 모드의 동적 시나리오에 의해 정의되는 타이틀의 구성이다.

[0165] 이어, BD-J 모드의 동적 시나리오에 의해 구성되는 타이틀의 내부 구성에 대해 설명한다. 도 15는 BD-J 모드의 동적 시나리오(BD-J Object)에 의해 구성되는 타이틀의 내부 구성을 나타내는 도면이다.

[0166] 제 1단계는 식별자 Title_id에 의해 식별되는 임의의 타이틀을 나타내고, 제 2단계는 그 타이틀을 구성하는 유일한 BD-J Object를 나타낸다. 제 3단계는 그 BD-J Object 내부에 있는 애플리케이션 관리테이블, 플레이리스트 관리테이블을 나타낸다. 제 4단계는 제 3단계의 애플리케이션 관리테이블에 의해 동작하게 되는 애플리케이션을 나타낸다. 이 애플리케이션은, 제 5단계에 나타내는 바와 같이, 다른 타이틀로의 Jump를 재생장치에 명하는 메소드(JumpTitleAPI를 콜하는 메소드)를 포함하고 있으므로, JumpTitleAPI의 콜 메소드에 의해 도 13에 나타내는 다른 타이틀로의 분기가 실현되게 된다. 한편, 제 3단계에는 플레이리스트 관리테이블이 기술되어 있으므로, 제 4단계에서는 애플리케이션의 실행과 함께 PL이 재생되게 된다.

[0167] BD-J Object에는 애플리케이션 관리테이블, 플레이리스트 관리테이블이 포함되어 있으므로, 제 4단계에 나타내는 바와 같이, 애플리케이션의 실행과 PL 재생이 동시에 이루어지게 된다. 이러한 애플리케이션의 실행과 PL 재생의 동시 실행이 BD-J 모드에서의 타이틀의 특징이다.

[0168] 플레이리스트 관리테이블은 모든 BD-J Object에 존재하는 것은 아니다. 플레이리스트 관리테이블은 임의적인 구성요소이며, 플레이리스트 관리테이블을 갖는 BD-J Object도 있고, 플레이리스트 관리테이블을 갖지 않는 BD-J Object도 있다. 도 16은 플레이리스트 관리테이블을 갖지 않는 타이틀을 나타내는 도면이다. 애플리케이션 관리테이블만 존재하므로, 플레이리스트 관리테이블을 갖지 않는 BD-J Object에서는, 제 4단계에 나타내는 바와 같이, 애플리케이션 동작만이 규정된다. 이상에 의한 애플리케이션 동작의 규정에 따라 PL 재생을 수반하지 않는 제어순서만의 타이틀이 규정되게 된다.

[0169] 또, 도 14에서는, HDMV 모드의 타이틀로부터 HDMV 모드의 타이틀로의 분기가 기술되어 있었으나, 도 17에 나타내는 바와 같이, HDMV 모드의 타이틀로부터 BD-J 모드의 타이틀로의 분기도 있을 수 있다. 마찬가지로, 도 15에서는 BD-J 모드의 타이틀로부터 BD-J 모드의 타이틀로의 분기가 기술되어 있었으나, 도 18에 도시하는 바와 같이, BD-J 모드의 타이틀로부터 HDMV 모드의 타이틀로의 분기도 있을 수 있다.

[0170] 상술한 타이틀의 내부 구성에서, 어떤 타이틀의 구성요소가 되는 MovieObject는 어느 것인지, 또는 어떤 타이틀

의 구성요소가 되는 BD-J Object는 어느 것인지를 정의하는 것이 도 2에 나타낸 index.bdmv이다. 이후 index.bdmv에 대해 설명한다.

- [0171] Index.bdmv는 타이틀을 구성하는 Movie Object 또는 BD-J Object를 나타내는 테이블이다.
- [0172] 도 19는 index.bdmv의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, index.bdmv는, 값 "INDX"를 나타내는 값을 갖는 『type_indicator』와, 『version_number』와, 본 파일의 선두에서 Indexes까지의 상대 어드레스를 나타내는 『Indexes_start_address』와, 『Indexes()』로 이루어진다. 『indexes』는 각 타이틀에 대한 인덱스이며, 점선 인출선 ix1에 의해 클로즈업되어 있는 바와 같이, 『Indexes』는 『length』, 『FirstPlayback()』{『FirstPlayback_mobj_id_ref』}, 『TopMenu()』{『TopMenu_mobj_id_ref』}, 『number_of_Titles』, 『Title[0]()』~『Title[number_of_Titles-1]()』로 이루어진다.
- [0173] 『FirstPlayback()』{『FirstPlayback_mobj_id_ref』}는 FirstPlayTitle에 대한 Index이며, FirstPlayTitle을 구성하는 MovieObject 식별자의 참조 값(FirstPlayback_mobj_id_ref)가 저장된다.
- [0174] 『TopMenu()』{『TopMenu_mobj_id_ref』}는 Top-MenuTitle에 대한 Index이며, Top-menuTitle을 구성하는 MovieObject 식별자의 참조 값(TopMenu_mobj_id_ref)이 저장된다.
- [0175] 『Title[0]()』~『Title[number_of_Titles-1]()』은 FirstPlayTitle, Top-MenuTitle 이외의 타이틀에 대한 Index이며, number_of_Title 개 존재하고 있다. 이들은 식별자 title_id로 식별된다.
- [0176] 여기서, 식별자 title_id에 의해 특정되는 인덱스를 Title[title_id]()라고 한다. 도면 중의 인출선 ix2는 Title[title_id]()에 대한 내부 구성을 클로즈업하고 있다.
- [0177] 본 도면에 나타내는 바와 같이, "Title[title_id]()"는 "Title[title_id]가 분기를 가지고 있는가 여부 등, 타이틀 재생의 유형을 나타내는 『Title_Playback_Type[title_id]』와, 이 타이틀에 대한 서치(search) 기능의 실행이 허가되어 있는지의 여부를 나타내는 『Title_access_Flag[title_id]』와, 타이틀을 구성하는 MovieObject를 일의적으로 나타내는 『title_mobj_id_ref[title_id]』로 이루어진다. 여기서, 타이틀을 구성하는 동적 시나리오가 BD-J Object이면, 『title_bobj_id_ref[title_id]』가 『title_mobj_id_ref[title_id]』의 대신이 된다. 『title_bobj_id_ref[title_id]』는 타이틀을 구성하는 BD-J Object를 일의적으로 나타낸다.
- [0178] Index.bdmv에 대해서는 이하의 국제공개공보에 상세한 내용이 기재되어 있다. 상세한 내용에 대해서는 국제공개공보 W02004/025651 A1을 참조하기 바란다.
- [0179] <애플리케이션 관리테이블>
- [0180] BD-J Object에 포함되는 애플리케이션 관리테이블 및 플레이리스트 관리테이블은 본 실시 예의 주안이 되는 구성요소가 되며, 그 상세한 내용에 대해 설명한다. 먼저, 애플리케이션 관리테이블(AMT)에 대해 설명한다.
- [0181] 도 20(a)은 애플리케이션 관리테이블의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, 애플리케이션 관리테이블은 『life_cycle』과 『apli_id_ref』와 『run_attribute』와 『run_priority』로 이루어진다.
- [0182] 도 20(b)은 애플리케이션 관리테이블을 구성하는 정보요소의 의미 내용을 나타낸다.
- [0183] 『life_cycle』은 애플리케이션의 "생존구간(life cycle)"을 나타낸다.
- [0184] 『apli_id_ref』는 "애플리케이션 식별자"에 대한 참조 값이 기술됨으로써, 좌측에 기재한 생존구간을 갖는 애플리케이션이 어느 것인지를 나타낸다. 애플리케이션 식별자는 Java(등록상표) 아카이브 파일에서, 파일명으로서 부여된 5 자릿수의 수치 zzzzz로 표현된다. 『apli_id_ref』에는 이 5 자릿수의 수치가 기술된다.
- [0185] 『run_attribute』는 당해 생존구간에서의 애플리케이션의 "기동속성(run attribute)"이 기술된다. 기동속성에는 AutoRun, Present, Susped라는 중별이 있다.
- [0186] 『run_priority』는 당해 생존구간에서의 애플리케이션의 "기동 우선도"가 기술된다. BD-J Object에서는 이들 정보를 이용하여 애플리케이션의 거동을 제어한다.
- [0187] <생존구간>
- [0188] 애플리케이션 관리테이블에 규정되는 정보 중, 생존구간에 대해 설명한다.
- [0189] 생존구간이란, BD-ROM에 기록된 콘텐츠 전체의 시간 축에서, 가상머신의 워크 메모리 상에서 애플리케이션이 생

존할 수 있는 구간을 나타낸다. 워크 메모리에서의 "생존"이란, 그 애플리케이션을 구성하는 xlet 프로그램이 워크 메모리에 판독되고, 가상머신에 의한 실행이 가능하게 되어 있는 상태를 말한다.

[0190] Java(등록상표) 가상머신에서 애플리케이션을 동작시키는 경우, 시간 축의 어디에서 애플리케이션에 의한 서비스를 개시하고, 시간 축의 어디에서 애플리케이션에 의한 서비스를 종료하는가라고 하는 "서비스의 개시 점?종료 점"을 명확히 규정하는 것이 중요해진다. 이 서비스의 개시 점?종료 점을 규정하는 것이 애플리케이션 관리 테이블에서의 생존구간이다.

[0191] 그러면, 도 13과 같은 상태천이를 이루는 디스크 콘텐츠에서, 타이틀의 생존구간은 어떻게 규정되는 것인가. BD-ROM의 로딩이 이루어진 후, 도 13에서 화살표 jh1, 2, 3, 4, ... 로 나타낸 참조 번호의 수치 순으로 분기가 이루어지고, BD-ROM이 이젝트된 것으로 한다. 그렇게 하면, BD-ROM이 로딩되고 나서 이젝트 될 때까지의 연속 시간대를 하나의 시간 축으로 동일시할 수 있다. 이 시간 축을 디스크 전체의 시간 축으로 한다. 도 21(a)는 디스크 전체의 시간 축을 나타내는 도면이며, 도 21(b)는 이 시간 축에서의 구성을 나타낸다. 도 21(b)에 나타내는 바와 같이, 디스크 전체의 시간 축은 FirstPlayTitle이 재생되고 있는 구간, TopMenu Title이 재생되고 있는 구간, title #1이 재생되고 있는 구간 등으로 이루어진다. 이들 타이틀의 재생구간은 어떻게 규정되어 있는가 하면, 타이틀은 하나 이상의 MovieObject 또는 유일한 BD-J Object로 구성되므로, 어느 하나의 MovieObject 또는 BD-J Object가 유효해져 있는 기간을 타이틀의 재생구간으로 생각할 수 있다.

[0192] 즉, FirstPlay Title, TopMenu Title, 그 이외의 타이틀은 모두 동적 시나리오로 구성되므로, 타이틀을 구성하는 MovieObject 또는 BD-J Object 중 어느 하나가 현재 MovieObject, 또는 현재 BD-J Object로서 활성화(Activated) 되며, 재생장치 내에서 해독?실행에 제공되고 있는 기간을 타이틀의 재생구간으로 정의할 수 있다. 도 22(a)는 BD-ROM 전체의 시간 축에서, 식별자 bobj_id에 의해 특정되는 BD-J Object로 특정되는 타이틀 재생구간을 나타내는 도면이다. 여기서, 식별자 bobj_id에 의해 특정되는 BD-J Object가 하나의 타이틀을 구성하고 있으면, 그 식별자 bobj_id에 의해 특정되는 BD-J Object가 유효해져 있는 BD-ROM 시간 축 상의 한 구간을 타이틀의 재생구간으로 생각할 수 있다.

[0193] 마찬가지로, 식별자 mobj_id에 의해 특정되는 MovieObject가 하나의 타이틀을 구성하고 있으면, 그 식별자 mobj_id에 의해 특정되는 MovieObject가 유효해져 있는 BD-ROM 시간 축 상의 한 구간을 타이틀의 재생구간으로 생각할 수 있다.

[0194] 여기서, MovieObject, BD-J Object가 Activate 되어 있는 기간의 종료 기간은 타이틀 분기(JumpTitle)가 이루어질 때까지이다. 즉, 타이틀 분기(JumpTitle)가 이루어질 때까지 실행의 대상이 되어 있는 동적 시나리오는 현재 MovieObject 및 현재 BD-J Object로서 취급되기 때문에, 그 MovieObject 또는 BD-J Object에서 JumpTitle이 발생할 때까지의 하나의 구간을 타이틀 재생구간으로 취급한다.

[0195] 이어, 타이틀 구간과 PL 시간 축의 관계에 대해 설명한다. 상술한 바와 같이, MovieObject, BD-J Object에서는 하나의 처리순서로서 PL 재생순서를 기술할 수 있다. PL 재생순서의 기술이 있으면, 상술한 PL 시간 축의 전부 또는 일부가 타이틀 구간에 귀속하게 된다. 도 22(a)의 일례에서 BD-J Object에 플레이리스트 관리테이블이 기술되어 있는 것으로 한다. 이 경우, BD-J Object에 대응하는 타이틀 구간에는, 도 22(b)에 나타내는 바와 같이, PL 시간 축이 귀속한다. 이 PL 시간 축에는 또한, 복수 챕터(Chapter #1, #2, #3)가 정의될 수 있으므로, BD-ROM 상의 시간 축에는 BD-ROM 전체-Title-PL-챕터라는 도메인이 존재하게 된다. 이들 도메인을 이용하여 애플리케이션의 생존구간을 기술할 수 있다. 또한, 플레이리스트 재생은 애플리케이션의 실행과 동시에 이루어지므로, 플레이리스트 재생 도중에 타이틀 분기가 발생하는 경우가 있다. 이 경우, 하나의 타이틀 재생구간 내에는 플레이리스트 시간 축 전체가 아니라, 플레이리스트 시간 축의 일부분만이 귀속하게 된다. 즉, 하나의 타이틀 재생구간에서, 플레이리스트 시간 축의 전체가 귀속하는가, 그 일부분이 귀속하는가는 타이틀 분기가 언제 발생하는가에 따라서 변화한다.

[0196] 도 23은 도 22(b)의 시간 축 상에 규정되는 생존구간의 전형을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, 애플리케이션에는 타이틀을 생존구간으로 한 "타이틀 바운더리 애플리케이션(Title Boundary Application)", Title 내에서의 챕터를 생존구간으로 한 "챕터 바운더리 애플리케이션(Chapter Boundary Application)", BD-ROM 전체의 시간 축을 생존구간으로 한 "타이틀 언바운더리 애플리케이션(Title Unboundary Application)"이라는 3개의 전형이 있다.

[0197] 이 중 타이틀 바운더리 애플리케이션의 생존구간은 그 타이틀의 식별자를 이용하여 정의할 수 있다. 또, 챕터 바운더리 애플리케이션의 생존구간은 챕터가 속하는 타이틀의 식별자와, 그 챕터 식별자와의 쌍을 이용하여 정

의할 수 있다.

- [0198] 플랫폼(flatform)이 동작하고 있다고 해도, 타이틀이나 챗터의 생존구간이 끝나면, 리소스를 애플리케이션으로 부터 회수할 수 있다. 리소스 회수의 기회를 보증하므로, 플랫폼의 동작을 안정화시킬 수 있다.
- [0199] 가까운 장래에 실시될 디스크 콘텐츠를 제재(題材)로 선택하여, 애플리케이션 관리테이블에서의 생존구간 기술에 대해 구체 예를 들어 설명한다. 여기서, 제재로 하는 디스크 콘텐츠는 영상 본편을 구성하는 본편 타이틀(Title #1), 온라인 쇼핑을 구성하는 온라인 쇼핑 타이틀(Title #2), 게임 애플리케이션을 구성하는 게임 타이틀(title #3)이라는, 성격이 다른 3개의 타이틀을 포함하는 것이다. 도 24는 본편 타이틀, 온라인 쇼핑 타이틀, 게임 타이틀이라는 3개의 타이틀을 포함하는 디스크 콘텐츠를 나타내는 도면이다. 본 도면의 우측에는 Index.bdmv를 기술하고 있으며, 좌측에는 3개의 타이틀을 기술하고 있다.
- [0200] 우측의 점선 틀은 각 애플리케이션이 어느 타이틀에 속하고 있는가 하는 귀속관계를 나타낸다. 3개의 타이틀 중 title #1은, application #1, application #2, application #3 이라는 3개의 애플리케이션으로 이루어진다. title #2는, application #3, application #4라는 2개의 애플리케이션, title #3은 application #5를 포함한다. 도 24의 일 예에서 application #3은 title #1, title #2의 쌍방에 의해 기동된다.
- [0201] 도 24의 점선으로 나타나는 귀속 관계로부터 각 애플리케이션의 생존구간을 그래프화하면 도 25(a)와 같이 된다. 본 도면에서 가로축은 타이틀 재생구간이며, 세로축 방향으로 각 애플리케이션의 생존구간을 배치하고 있다. 여기서, application #1, application #2는 title #1만에 귀속하고 있으므로, 이들의 생존구간은 title #1 내에 한정되어 있다. application #4는 title #2만에 귀속하고 있으므로, 이들의 생존구간은 title #2 내에 한정되어 있다. application #5는 title #3만에 귀속하고 있으므로, 이들의 생존구간은 title #3 내에 한정되어 있다. application #3은 title #1 및 title #2에 귀속하고 있으므로, 이들의 생존구간은 title #1-title #2에 걸쳐 있다. 이 생존구간에 기초하여 애플리케이션 관리테이블을 기술하면, title #1, #2, #3의 애플리케이션 관리테이블은 도 25(b)와 같이 된다. 이와 같이 애플리케이션 관리테이블이 기술되면, title #1의 재생개시시에 application #1, application #2, application #3을 워크 메모리에 로드해 둔다. 그리고 title #2의 개시시에 application #1, application #2를 워크 메모리로부터 삭제하여 application #3만으로 하는 제어를 행한다. 이와 동시에 title #2의 재생개시시에 application #4를 워크 메모리에 로드해 두고, title #3의 개시시에 application #3, #4를 워크 메모리로부터 삭제하는 제어를 행할 수 있다.
- [0202] 또, title #3의 재생중에 application #5를 워크 메모리에 로드해 두고, title #3의 재생 종료시에 application #5를 워크 메모리로부터 삭제하는 제어를 행할 수 있다.
- [0203] 타이틀 간 분기가 있었던 경우라도, 분기원(branch source)-분기처(branch destination)에서 생존하고 있는 애플리케이션은 워크 메모리 상에 저장해 두고, 분기원에는 없고 분기처에만 존재하는 애플리케이션을 워크 메모리에 판독하면 되므로, 애플리케이션을 워크 메모리에 판독하는 횟수는 필요 최저 수가 된다. 이와 같이, 판독 횟수를 적게 함으로써 타이틀의 경계를 인식시키지 않는 애플리케이션, 즉 언바운더리 한 애플리케이션을 실현할 수 있다.
- [0204] 이어서, 애플리케이션의 기동속성에 대해 보다 상세하게 설명한다. 기동속성에는 자동적인 기동을 나타내는 「AutoRun」, 자동 기동의 대상은 아니지만, 가상머신의 워크 메모리에 두어도 됨을 나타내는 「Present」, 가상머신의 워크 메모리에는 놓이지만, CPU 파워의 할당은 불가능해지는 「Suspend」가 있다.
- [0205] 「AutoRun」은, 대응하는 타이틀의 분기와 동시에, 그 애플리케이션을 워크 메모리에 판독하고, 또한 실행한다는 취지를 나타내는 속성이다. 어떤 타이틀로부터 다른 타이틀로의 분기가 있으면, 애플리케이션 관리를 행하는 관리 주체(애플리케이션 매니저)는 그 분기처 타이틀에서 생존하고 있고, 또한, 기동속성이 AutoRun으로 설정된 애플리케이션을 가상머신의 워크 메모리에 판독 실행한다. 이로써, 그 애플리케이션은 타이틀 분기와 함께 자동으로 기동되게 된다.
- [0206] 기동속성 「Present」는, 계속 속성(continuous attribute)이며, 분기원 타이틀에서의 애플리케이션의 상태를 계속하는 것을 나타낸다. 또, 대응하는 애플리케이션을 실행해도 됨을 나타내는 속성이다. 기동속성이 「Present」일 경우, 이 기동속성이 부여된 애플리케이션은 다른 애플리케이션으로부터의 호출이 허가되게 된다. 애플리케이션 관리를 행하는 관리 주체(애플리케이션 매니저)는 기동중의 애플리케이션으로부터 호출이 있으면, 그 애플리케이션의 applicationID가 애플리케이션 관리테이블에 기술되어 있고, 기동속성이 「Present」인지의 여부를 판정한다. 「Present」이면, 그 애플리케이션을 워크 메모리에 로드한다. 한편, 그 호출처 애플리케이션의 applicationID가 애플리케이션 관리테이블에 기술되어 있지 않은 경우, 그 애플리케이션은 워크 메모리에 로

드되지 않는다. 애플리케이션에 의한 호출은, 이 「Present」가 부여된 애플리케이션에 한정되게 된다. 「Present」는 기동속성을 명시적으로 지정하지 않는 경우에 부여되는 디폴트의 기동속성이므로, 어떤 애플리케이션의 기동속성이 무 지정(no specification) 「—」인 경우, 그 애플리케이션의 기동속성은 이 Present임을 의미한다.

- [0207] 「Suspend」란, 리소스는 할당되어 있으나, CPU 파워는 할당되지 않은 상태에 애플리케이션이 놓여지는 것을 말한다. 이러한 Suspend는, 예를 들어, 게임 타이틀 실행중에 사이드 경로(side path)를 경유하는 처리의 실현에 의의가 있다.
- [0208] 도 26은 기동속성이 취할 수 있는 3개의 형태(Present, AutoRun, Suspend)와, 직전 타이틀에서의 애플리케이션 상태의 3개의 형태(비 기동, 기동중, Suspend)가 취할 수 있는 조합을 나타내는 도면이다. 직전 상태가 "비 기동"인 경우, 기동속성이 "AutoRun"이면, 분기처 타이틀에서 그 애플리케이션은 기동되게 된다.
- [0209] 직전 상태가 "비 기동"이며, 기동속성이 "Present", "Suspend"이면, 분기처 타이틀에서 그 애플리케이션은 아무 것도 하지 않고 상태를 계속하게 된다.
- [0210] 직전 상태가 "기동중"인 경우, 기동속성이 "Present", "AutoRun"이면, 분기 타이틀에서 그 애플리케이션은 아무 것도 하지 않고 상태를 계속하게 된다.
- [0211] 기동속성이 "Suspend"이면, 애플리케이션의 상태는 Suspend되게 된다. 직전 상태가 "Suspend"인 경우, 분기처 타이틀의 기동속성이 "Suspend"이면 Suspend를 유지하게 된다. "Present" 또는 "AutoRun"이면, 분기처 타이틀에서 그 애플리케이션은 재개(resume)하게 된다. 애플리케이션 관리테이블에서 생존구간 및 기동속성을 정의함으로써, 타이틀 재생구간의 진행에 따라 Java(등록상표) 애플리케이션을 동작시키는 동기제어가 가능해지고, 영상재생과 프로그램 실행을 수반한 다양한 애플리케이션을 세상에 배출할 수 있다.
- [0212] 또한, 직전 상태가 "Suspend"이며, 분기처 타이틀의 기동속성이 "Present"인 경우는 직전 상태, 즉 서스펜드 상태를 유지해도 된다.
- [0213] 마지막으로, 각 애플리케이션에 대한 "기동 우선도"에 대해 설명한다.
- [0214] 이 기동 우선도는 0~255의 값을 취하며, 메모리 리소스 고갈이나 CPU 부하가 높아졌을 때, 어떤 애플리케이션을 강제로 종료시키는가, 또 어떤 애플리케이션으로부터 리소스를 빼앗는가라는 처리를 애플리케이션 매니저가 행할 때의 판단자료가 된다. 이 경우, 애플리케이션 매니저는 기동 우선도가 낮은 애플리케이션의 동작을 종료하고, 기동 우선도가 높은 애플리케이션의 동작을 계속시키는 처리를 행한다.
- [0215] 또, 기동 우선도는 재생중 PL에 대한 요구가 경합하는 경우의 애플리케이션 간의 조정에도 이용된다. 여기서, 어떤 애플리케이션이 어떤 플레이리스트를 빨리 감기를 하고 있는 것으로 한다. 여기서, 다른 애플리케이션이 같은 플레이리스트에 대한 포즈 요구를 행했다고 하면, 이들 애플리케이션에 부여된 기동 우선도를 비교한다. 그리고 빨리 감기를 명한 애플리케이션의 기동 우선도가 높으면, 관련 애플리케이션에 의한 빨리 감기를 계속하여 행한다. 반대로, 포즈를 명한 애플리케이션의 기동 우선도가 높으면 빨리 감기 중에 PL에 대한 포즈를 행한다.
- [0216] 이상의 생존구간?기동속성?기동 우선도에 의해 가상머신 상에서 동작할 수 있는 애플리케이션의 수를 소정 수 이하로 제한하도록 규정해 두는 것이 편집(authoring)시에 가능해진다. 그 때문에, 애플리케이션의 안정된 동작을 보증할 수 있다.
- [0217] <플레이리스트 관리테이블>
- [0218] 이상이 애플리케이션 관리테이블에 대한 설명이다. 이어, 플레이리스트 관리테이블에 대해 설명한다. 플레이리스트 관리테이블이란, 애플리케이션의 생존구간에서, 각 애플리케이션 실행과 동시에 행해야 할 재생제어를 나타내는 테이블이다. 애플리케이션의 동작은 불안정하여, 기동의 실패나 이상종료가 있을 수 있다. 그래서 기동 실패, 이상종료가 있는 경우의 패일 세이프(Fail Safe) 기구로서, 본 실시 예에서는 애플리케이션의 생존구간마다 플레이리스트 관리테이블을 설치하고 있다. 플레이리스트 관리테이블은 어떤 애플리케이션의 생존구간이 개시되었을 때, 이것과 동시에 행해야 할 재생제어를 규정하는 정보이다. 이 재생제어란, 플레이리스트 정보에 기초하는 AVClip 재생이며, 플레이리스트 정보에 의한 재생제어를 동시에 행함으로써, 애플리케이션 실행과 플레이리스트 재생이 동시에 이루어지게 된다. 플레이리스트 관리테이블은 애플리케이션의 생존구간마다 설치되도록 하였으나, 플레이리스트 관리테이블이 설치되는 애플리케이션은 타이틀 바운더리의 애플리케이션에 한정된다. 왜냐하면, 타이틀 언 바운더리 애플리케이션은 모든 타이틀을 생존구간으로 하고 있으므로, 애플리케이션 실행

과 동시에 플레이리스트 재생을 행하는 제어는 적합하지 않기 때문이다.

- [0219] 챗터 바운더리 애플리케이션은 하나의 플레이리스트 내의 챗터로부터 애플리케이션 실행을 개시한다는 전체 하에서 생존구간이 규정되어 있으므로, 플레이리스트 재생을 규정할 필요는 없기 때문이다. 이상으로부터 플레이리스트 관리테이블은 하나 이상의 타이틀로 이루어지는 생존구간으로 정의되게 된다.
- [0220] 도 27(a)는 플레이리스트 관리테이블의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, 플레이리스트 관리테이블은 『PL_id_ref』와 『Playback_Attribute』로 이루어진다.
- [0221] 도 27(b)는 플레이리스트 관리테이블을 구성하는 정보요소의 의미 내용을 나타낸다. 『PL_id_ref』는 PL 식별자에 대한 "참조 값"이 기술됨으로써, 애플리케이션의 생존구간에서 재생 가능하게 되는 PL이 어느 것인지를 나타낸다. PL 식별자는 파일 YYYY.MPLS에서 파일명으로 부여된 5 자릿수의 수치 YYYY로 표현된다. 이 YYYY가 기술됨으로써, 『PL_id_ref』는 대응하는 타이틀에서 재생 가능하게 되는 PL이 어느 것인지를 나타낸다.
- [0222] 『Playback_Attribute』는 애플리케이션 관리테이블에서의 기동속성을 모방한 속성이며, 『PL_id_ref』에 기술된 PL을, 타이틀 개시시에, 어떻게 재생할지를 규정하는 재생속성이다. PL에 대한 재생속성에는 『AutoPlay』, 『Present』라는 중별이 있다.
- [0223] 『AutoPlay』란, 대응하는 타이틀의 분기와 동시에, 그 플레이리스트를 재생시킴을 나타내는 속성이다. 어떤 타이틀로부터 다른 타이틀로의 분기가 있으면, 애플리케이션 관리를 행하는 관리주체(애플리케이션 매니저)는 그 분기처 타이틀에서 재생 가능하고, 또한, 재생속성이 AutoPlay에 설정된 플레이리스트의 재생을 개시한다. 이로써, 기동속성이 AutoPlay로 설정된 플레이리스트는 타이틀 분기와 함께 자동으로 기동되게 된다.
- [0224] 『Present』란, 기동속성에서의 Present와 마찬가지로 계속속성이며, 분기된 타이틀에서의 PL의 상태를 계속함을 나타낸다. 또, 대응하는 플레이리스트를 재생해도 됨을 나타내는 속성이다. 예를 들어, 연속하여 재생되는 2개의 타이틀이 있고, 분기된 타이틀 측의 플레이리스트 관리테이블에서는 어떤 플레이리스트의 재생속성이 AutoPlay로 설정되고, 분기처 타이틀 측의 플레이리스트 관리테이블에서는 그 플레이리스트의 재생속성이 Present로 설정되어 있는 것으로 한다. 여기서, 플레이리스트의 재생시간이 2시간 길이며, 이 중 1시간이 경과된 시점에서 분기가 발생한 것으로 한다. 이 경우 분기처 타이틀에서는 재생속성이 Present로 설정되어 있으므로, 분기처 타이틀에서 그 플레이리스트는 1시간이라는 재생완료구간의 직후에 재생되게 된다. 이렇게 재생속성을 Present로 설정해 두면, 타이틀 간의 분기가 있었던 경우에도 플레이리스트 재생을 그 나머지 부분에서 개시할 수 있다. 이로써, 서로 분기하는 일련의 타이틀에서, 공통의 플레이리스트를 재생하는 "타이틀 사이에서의 플레이리스트 재생의 공통화"를 용이하게 실현할 수 있다. 또, 분기처 타이틀이 복수인 경우, 이들 복수 타이틀의 재생속성을 모두 Present로 해 두면 복수 중 어느 것으로 분기했다 해도, 하나의 공통 플레이리스트 재생을 계속시킬 수 있다.
- [0225] 또한, 타이틀의 경계는, 심리스(seamless) 재생을 보증하지 않아도 되므로, 상술한 바와 같이 복수 타이틀 사이에서 하나의 플레이리스트를 재생하려고 하는 경우, 분기 전후에 플레이리스트 재생을 중단시키는 것은 허용된다.
- [0226] 또, 재생속성이 「Present」인 경우, 이 재생속성이 부여된 플레이리스트는 다른 애플리케이션으로부터의 재생요구에 의해 재생되게 된다. 애플리케이션 관리를 행하는 관리주체(애플리케이션 매니저)는 기동중의 애플리케이션으로부터 플레이리스트의 재생요구가 있으면, 요구를 받은 플레이리스트의 PL_id_ref가 플레이리스트 관리테이블에 기술되어 있고, 재생속성이 「AutoPlay」나 「Present」중 어느 하나인지의 여부를 판정한다. 「AutoPlay」나 「Present」중 어느 하나이면, 그 플레이리스트를 재생한다. 한편, 요구를 받은 플레이리스트의 PL_id_ref가 플레이리스트 관리테이블에 기술되어 있지 않은 경우, 그 플레이리스트를 재생하지 않는다. 애플리케이션의 요구에 의한 플레이리스트 재생은 이 「AutoPlay」나 「Present」중 어느 하나가 부여된 플레이리스트에 한정되게 된다. 「Present」는 재생속성을 명시적으로 지정하지 않는 경우에 부여되는 디폴트의 재생속성이므로, 어떤 플레이리스트의 재생속성이 무지정「—」이면, 그 플레이리스트의 재생속성은 이 Present임을 의미한다.
- [0227] 도 28은 분기처 타이틀이 취할 수 있는 3가지 형태(플레이리스트 관리테이블 없음(i), 플레이리스트 관리테이블이 있으면서 AutoPlay(ii), 플레이리스트 관리테이블이 있으면서 Present(iii))와, 직전 타이틀에서의 PL 상태(비 재생상태, 재생중인 상태)가 취할 수 있는 6가지의 조합을 나타내는 도면이다.
- [0228] 본 도면에서의 6가지의 조합 중, "직전 상태 = 비 재생상태"와 "분기처 타이틀 = 플레이리스트 관리테이블이 있고, 또한 분기처 타이틀의 재생속성 = AutoPlay"의 조합에 있어서, 분기처 타이틀에서의 PL의 재생은 자동으로

개시하게 된다.

- [0229] 또, "직전상태 = 재생중 상태"와 "분기처 타이틀 = 플레이리스트 관리테이블 없음"의 조합에 있어서, 분기처 타이틀에서의 PL의 재생은 자동으로 정지하게 된다.
- [0230] 그리고 이들 2개의 조합 이외에는 모두 분기처 타이틀의 상태를 계속하게 된다. 플레이리스트 관리테이블에 기초하는 플레이리스트 재생의 개시는, 분기처 타이틀에 있어서 비 재생상태이고, 분기처 타이틀에 있어서 AutoPlay 속성이 부여되어 있는 경우에 한정되므로, 타이틀의 분기가 발생할 때마다, 플레이리스트 재생을 개시하도록 할 필요는 없다. 타이틀 간의 분기가 다수 발생했다 해도, 플레이리스트 재생을 개시하도록 하는 횟수를 필요 최저 수로 할 수 있다.
- [0231] 플레이리스트 관리테이블 및 애플리케이션 관리테이블의 기술 예에 대해 도 29(a)를 참조하면서 설명한다. 여기서, 상정하는 구체 예는 2개의 연속하는 Title(title #1, title #2)이며, 그 중 title #1에서는 AutoRun 애플리케이션으로서 application #1, application #2가 기술되어 있다. title #2에서는 AutoRun 애플리케이션으로서 application #2, application #3이 기술되어 있다. 한편, title #1의 플레이리스트 관리테이블에는 AutoPlay 플레이리스트로서 Playlist #1, title #2의 플레이리스트 관리테이블에는 AutoPlay 플레이리스트로서 Playlist #2가 기술되어 있는 것으로 한다. 도 29(b)는, 도 29(a)와 같이 기술된 애플리케이션 관리테이블, 플레이리스트 관리테이블에 의해 플레이리스트 재생, 애플리케이션 실행이 어떻게 진행되는가를 나타내는 도면이다.
- [0232] title #1에서 애플리케이션 관리테이블, 플레이리스트 관리테이블은 상술한 바와 같이 설정되어 있으므로, title #1의 개시시에는 application #1, application #2가 자동으로 기동되고, Playlist #1의 재생이 자동으로 개시된다.
- [0233] title #2에서 애플리케이션 관리테이블, 플레이리스트 관리테이블은 상술한 바와 같이 설정되어 있으므로, title #1 측에 기재는 있으나 title #2 측에는 기재가 없는 application #1의 실행은 정지된다. 동일하게 title #1 측에 기재는 있으나 title #2 측에 기재가 없는 Playlist #1의 재생도 정지시키게 된다.
- [0234] title #1 측에 기재는 없으나 title #2 측에 기재가 있는 Playlist #2, application #3은 재생 및 실행이 자동으로 개시되게 된다. 타이틀 분기가 있으면, 그 분기를 계기로, 재생해야 할 플레이리스트를 다른 플레이리스트로 전환할 수 있다. 이와 같이, 애플리케이션 관리테이블, 플레이리스트 관리테이블을 이용함으로써 분기를 계기로 하여 플레이리스트 재생을 전환하는 처리를 편집 단계에서 규정해 둘 수 있다.
- [0235] 또, 도 29에서는 application #1, application #2, application #3에는 각각 200, 128, 200의 기동 우선도가 주어져 있다. 이들의 기동 우선도를 부여함으로써, Playlist #1, Playlist #2에 대한 제어 요구가 경합한 경우의 조정을 행하게 할 수 있다. 여기서, application #1이 Playlist #1에 대해 빨리 감기를 명하고 있는 것으로 한다. 한편, application #2가 포즈(pause) 요구를 행한 것으로 한다. 이 경우, 애플리케이션 관리테이블에 각 애플리케이션에 대한 기동 우선도가 규정되어 있으므로, 이 기동 우선도에 따라 양 애플리케이션에 대한 조정이 이루어지게 된다. 그 결과, application #2에 의한 요구를 받아들이지 않고, application #1에 의한 제어를 계속하는 처리를 편집시에 규정해 둘 수 있다. 기동 우선도를 플레이리스트 관리테이블과 아울러 이용함으로써, 플레이리스트에 대한 제어가 경합한 경우의 조정까지도 재생장치가 행하도록 할 수 있다.
- [0236] 플레이리스트 관리테이블의 기술의 다른 구체 예에 대해 설명한다. 도 30(a)는 플레이리스트 관리테이블의 다른 기술 예를 나타내는 도면이다. 본 도면에서 상정하고 있는 것은, 2개의 연속하는 타이틀(title #1, title #2)에서, title #1 측의 플레이리스트 관리테이블에는, AutoPlay 플레이리스트로서 Playlist #1이, 재생가능한 플레이리스트로서 Playlist #2가 기술되고, title #1 측의 애플리케이션 관리테이블에는, AutoPlay 애플리케이션인 application #1과, 실행 가능한 애플리케이션으로서 application #2가 기술되어 있다. 한편, 타이틀 #2 측의 플레이리스트 관리테이블에는 재생 가능한 플레이리스트로서 Playlist #2, Playlist #3이 기술되고, 애플리케이션 관리테이블에는 AutoRun 애플리케이션으로서 application #3이 기술되어 있다. 도 30(b)는 도 30(a)의 케이스에 기초하는 애플리케이션의 실행 및 플레이리스트 재생의 진행을 나타내는 도면이다. title #1의 애플리케이션 관리테이블에는 AutoRun 애플리케이션으로서 application #1이 기술되어 있으므로, title #1의 개시시에는 application #1이 자동으로 기동된다. 한편, title #1의 애플리케이션 관리테이블에는 실행 가능 애플리케이션으로서 application #2가 기술되어 있으므로, application #1로부터의 호출 yd1에 의해 application #2가 기동된다.
- [0237] title #2 측의 애플리케이션 관리테이블에서 application #1, application #2가 비 생존으로 되어 있고, 대신

AutoRun 애플리케이션으로서 application #3이 기술되어 있다. 그 때문에 title #1 - title #2의 경계부에서는 application #1, application #2를 정지하고 application #3을 자동으로 기동하는 처리가 이루어진다. 플레이리스트 관리테이블을 참조하면, title #1 측의 플레이리스트 관리테이블은 Playlist #1, Playlist #2가 재생 가능하다고 기술되어 있으며, 그 중 Playlist #1은 AutoPlay 속성으로 되어 있다. 그 때문에, Playlist #1은 Title #1의 개시시에 자동으로 재생된다.

[0238] title #1 측의 플레이리스트 관리테이블에는 Playlist #1 이외에 Playlist #2가 재생 가능하다고 기술되어 있으므로, application #1은 Playlist #1의 재생을 정지시키고, 대신 Playlist #2의 재생을 요구함으로써 플레이리스트 교대를 실행할 수 있다.

[0239] title #2 측의 플레이리스트 관리테이블에는 재생 가능한 플레이리스트로서 Playlist #2, Play #3이 기술되어 있다. 그리고 AutoPlay 속성이 부여된 플레이리스트는 없다. 그 때문에, 가령 title #1 개시시에 자동 재생된 Playlist #1의 재생이 title #2까지 계속했다 하더라도 Playlist #1의 재생은 자동으로 종료하게 된다.

[0240] 그러나 Playlist #2 재생이 계속된 채 title #2에 이르면, Playlist #2 재생은 title #2 개시 이후에도 계속한다. title #2의 플레이리스트 관리테이블에는 재생 가능한 플레이리스트로서 Playlist #2, Playlist #3이 기술되어 있다. 그 때문에, title #2에서 실행중이 되는 application #3은, Playlist #2의 재생을 정지하고, 대신 Playlist #3의 재생을 요구함으로써 재생중의 플레이리스트를 교대시킬 수 있다.

[0241] 다음으로, 플레이리스트 관리테이블에 의해 타이틀 재생구간이 어떻게 규정되는지를 도 31을 참조하면서 설명한다. 도 31(a)는 재생속성이 AutoPlay로 설정된 타이틀의 타이틀 재생구간을 나타내는 도면이다. 재생속성이 AutoPlay를 나타내도록 설정되면, 타이틀의 재생개시와 동시에 AutoPlayPL의 재생을 개시한다. 여기서, 애플리케이션이 정상으로 동작하고, 정상으로 종료되었다 해도, 이 타이틀 재생구간은 PL 시간 축을 기준으로 하여 정해진다.

[0242] 도 31(b)는, 플레이리스트 관리테이블에서 재생속성이 "AutoPlay"를 나타내도록 설정되고, 애플리케이션이 이상 종료한 경우를 나타낸다. 이러한 이상종료에 의해 어떤 애플리케이션도 동작하고 있지 않은 상태가 되나, AutoPlayPL의 재생은 계속된다. 이 경우에도 AutoPlayPL의 PL 시간 축이 타이틀 재생구간이 된다.

[0243] 도 31(c)는, 플레이리스트 관리테이블에서 재생속성이 "AutoPlay"를 나타내도록 설정되고, 메인 애플리케이션의 기동에 실패한 케이스를 나타낸다. 이 경우도 AutoPlayPL 재생은 애플리케이션의 기동 실패와는 관계없이 행해지므로, AutoPlay PL의 시간 축이 타이틀 재생구간이 된다.

[0244] 이상과 같이 플레이리스트 관리테이블의 재생속성을 "AutoPlay"로 설정해 두면, Java(등록상표) 애플리케이션의 기동에 5~10초라는 시간이 걸렸다 해도, 그 기동이 이루어지고 있는 동안 "우선 무엇인가가 표시되고 있는 상태"가 된다. 타이틀 실행 개시시에 애플리케이션 기동에 시간이 걸렸다 해도, 화면은 "우선 무엇인가가 표시되고 있는 상태"가 된다. 이로써, 애플리케이션 기동에 시간이 걸림에 따른 기동의 지연(start-up delay)의 장기화를 보충할 수 있다.

[0245] 애플리케이션 관리테이블 및 플레이리스트 관리테이블을 정의함으로써, 타이틀 재생구간의 진행에 따라서 Java(등록상표) 애플리케이션을 동작시키는 동기제어가 가능해지며, 영상재생과 프로그램 실행을 수반한 여러 애플리케이션을 세상에 배출할 수 있다. 이상이 기록매체에 대한 설명이다. 이어서, 본 발명에 관한 재생장치에 대해 설명한다.

[0246] 도 32는 본 발명에 관한 재생장치의 내부 구성을 나타내는 도면이다. 본 발명에 관한 재생장치는 본 도면에 나타내는 내부에 의거하여 공업적으로 생산된다. 본 발명에 관한 재생장치는 주로 시스템 LSI와 드라이브 장치라는 2개의 파트로 이루어지며, 이들 파트를 장치의 캐비닛 및 기관으로 실장 함으로써 공업적으로 생산할 수 있다. 시스템 LSI는 재생장치의 기능을 담당하는 다양한 처리부를 집적한 집적회로이다. 이렇게 하여 생산되는 재생장치는, BD-ROM 드라이브(1), 리드 버퍼(2), 디멀티플렉서(3), 비디오 디코더(4), 비디오 플레인(5), P-Graphics 디코더(6), 프레젠테이션 그래픽스 플레인(7), 합성부(8), 폰트 제너레이터(9), I-Graphics 디코더(10), 스위치(11), 인터랙티브 그래픽스 플레인(12), 합성부(13), CLUT부(14), CLUT부(15), 오디오 디코더(16), 네트워크 디바이스(17), Local Storage(18), 리드 버퍼(19), 디멀티플렉서(20), 명령 ROM(21), 사용자 이벤트 처리부(22), PSR 세트(23), CPU(24), 시나리오 메모리(25), 로컬 메모리(26), 스위치(27)로 구성된다.

[0247] 우선 먼저, BD-ROM에 기록된 AVClip 재생에 관한 구성요소(BD 드라이브(1)~오디오 디코더(16))에 대해 설명한다.

- [0248] BD-ROM 드라이브(1)는, BD-ROM의 로딩/이젝트를 행하고, BD-ROM에 대한 액세스를 실행한다.
- [0249] 리드 버퍼(2)는 FIFO 메모리이며, BD-ROM으로부터 판독된 TS 패킷이 선입선출 식으로 저장된다.
- [0250] 디멀티플렉서(De-MUX)(3)는 리드버퍼(2)로부터 TS 패킷을 취출하여, 이 TS 패킷을 구성하는 TS 패킷을 PES 패킷으로 변환한다. 그리고 변환에 의해 얻어진 PES 패킷 중, CPU(24)에서 설정된 PID를 갖는 것을 비디오 디코더(4), P-Graphics 디코더(6), I-Graphics 디코더(10), 오디오 디코더(16) 중 어느 하나에 출력한다.
- [0251] 비디오 디코더(4)는 디멀티플렉서(3)로부터 출력된 복수 PES 패킷을 복호하여 비 압축 형식의 픽처를 얻어서 비디오 플레인(5)에 기록한다.
- [0252] 비디오 플레인(5)은 비 압축 형식의 픽처를 저장해 두기 위한 플레인이다. 플레인이란, 재생장치에서 1 화면 분의 화소 데이터를 저장해 두기 위한 메모리 영역이다. 비디오 플레인(5)에서의 해상도는 1920×1080이며, 이 비디오 플레인(5)에 저장된 픽처 데이터는 16비트의 YUV 값으로 표현된 화소 데이터에 의해 구성된다. 비디오 플레인(5)에서는 비디오 스트림에서의 1프레임 별 재생영상을 스케일링(scaling)할 수 있다. 스케일링이란, 1프레임 별 재생화상을 비디오 플레인(5) 전체의 1/4(쿼터라 한다), 1/1(풀 스케일이라 한다) 중 어느 하나로 변화시키는 것이다. 이러한 스케일링을 BD-J 모드에서 CPU(24)로부터의 지시에 따라 실행하므로, 비디오 스트림의 재생화상을 화면의 구석으로 물거나, 전면적으로 출력하는 화면 연출이 가능해진다.
- [0253] P-Graphics 디코더(6)는 BD-ROM에서 판독된 프레젠테이션 그래픽스 스트림을 디코딩하여, 비 압축 그래픽스를 프레젠테이션 그래픽스 플레인(7)에 기록한다. 그래픽스 스트림의 디코딩에 의해 자막이 화면상에 나타나게 된다.
- [0254] 프레젠테이션 그래픽스 플레인(7)은 1 화면 분의 영역을 가진 메모리이며, 1 화면 분의 비 압축 그래픽스를 저장할 수 있다. 본 플레인에서의 해상도는 1920×1080이며, 프레젠테이션 그래픽스 플레인(7) 중의 비 압축 그래픽스의 각 화소는 8비트의 인덱스 컬러로 표현된다. CLUT(Color Lookup Table)를 이용하여 이러한 인덱스 컬러를 변환함으로써, 프리젠테이션 그래픽스 플레인(7)에 저장된 비 압축 그래픽스는 표시에 제공된다.
- [0255] 합성부(8)는 비디오 플레인(5)에 저장된 비 압축 상태의 픽처 데이터(i)를 프리젠테이션 그래픽스 플레인(7)의 저장 내용과 합성한다.
- [0256] 폰트 제네레이터(9)는 문자 폰트를 이용하여 textST 스트림에 포함되는 텍스트 코드를 비트맵으로 전개하여 프리젠테이션 그래픽스 플레인(7)에 기록한다.
- [0257] I-Graphics 데이터(10)는 HDMV 모드에서 BD-ROM 또는 Local Storage(18)로부터 판독된 IG 스트림을 디코딩하여, 비 압축 그래픽스를 인터랙티브 그래픽스 플레인(12)에 기록한다.
- [0258] 스위치(11)는, 폰트 제너레이터(9)가 생성한 폰트 열, P-Graphics 디코더(6)의 디코딩에 의해 얻어진 그래픽스 중 어느 하나를 선택적으로 프리젠테이션 그래픽스 플레인(7)에 기록하는 스위치이다.
- [0259] 인터랙티브 그래픽스 플레인(12)은 I-Graphics 디코더(10)에 의한 디코딩에 의해 얻어진 비 압축 그래픽스가 기록된다. 또, 인터랙티브 그래픽스 플레인(12)에는 BD-J 모드에서, 애플리케이션에 의해 렌더링(描畵)된 문자나 그래픽스가 기록된다.
- [0260] 합성부(13)는 인터랙티브 그래픽스 플레인(12)의 저장 내용과, 합성부(8)의 출력인 합성화상(비 압축 상태의 픽처 데이터와, 프리젠테이션 그래픽스 플레인(7)의 저장 내용을 합성한 것)을 합성한다. 이러한 합성에 의해 애플리케이션이 I-Graphics 디코더(10)에 기록한 문자·그래픽스를 비 압축 상태의 픽처 데이터 상에 오버레이(overlay)하여 표시할 수 있다.
- [0261] CLUT부 14는 비디오 플레인(5)에 저장된 비 압축 그래픽스에서의 인덱스 컬러를 Y, Cr, Cb 값으로 변환한다.
- [0262] CLUT부 15는 인터랙티브 그래픽스 플레인(12)에 저장된 비 압축 그래픽스에서의 인덱스 컬러를 Y, Cr, Cb값으로 변환한다.
- [0263] 오디오 디코더(16)는 디멀티플렉서(3)로부터 출력된 PES 패킷을 복호하여 비 압축 형식의 오디오 데이터를 출력한다.
- [0264] 이상이 AVClip 재생에 관한 구성요소이다. 이어, BD-J 모드에서의 동작에 관한 구성요소(네트워크 디바이스(17)~De-mux(20))에 대해 설명한다.

- [0265] 네트워크 디바이스(17)는 재생장치에서의 통신기능을 실현하는 것이며, BD-J 모드에서 URL 지정이 Java(등록상표) 애플리케이션으로부터 주어지면, 그 URL에 해당하는 웹사이트와의 TCP 커넥션(TCP connection), FTP 커넥션(FTP connection) 등을 확립한다. 이러한 커넥션의 확립에 의해 웹사이트로부터의 다운로드를 Java(등록상표) 애플리케이션이 행하도록 한다.
- [0266] Local Storage(18)는 네트워크 디바이스(17)에 의해 확립된 커넥션을 통해 web 사이트에서 다운로드 된 콘텐츠 등, BD-ROM 이외의 기록매체, 통신매체로부터 공급된 콘텐츠를 메타 데이터(metadata)와 함께 저장해 두기 위한 하드디스크이다. 이 메타데이터는 다운로드 콘텐츠를 Local Storage(18)에 바인드하여 관리하기 위한 정보이며, 이 Local Storage(18)를 액세스함으로써 BD-J 모드에서의 애플리케이션은 다운로드 콘텐츠 길이를 이용한 여러 처리를 행할 수 있다.
- [0267] 리드 버퍼(19)는 FIFO 메모리이며, Local Storage(18)에 저장된 다운로드 콘텐츠에 SubClip이 포함되어 있는 경우, 이 SubClip을 구성하는 TS 패킷을 선입선출 식으로 저장한다.
- [0268] 디멀티플렉서(De-MUX)(20)는, 리드 버퍼(19)로부터 TS 패킷을 취출하고, TS 패킷을 PES 패킷으로 변환한다. 그리고 변환에 의해 얻어진 PES 패킷 중, 원하는 PID를 갖는 것을 폰트 제네레이터(9), I-Graphics 디코더(10), 오디오 디코더(16)에 출력한다.
- [0269] 이상의 네트워크 디바이스(17)~De-mux(20)에 의해, Java(등록상표) 애플리케이션이 네트워크를 통해 다운로드 한 콘텐츠를 BD-ROM에 기록된 콘텐츠와 동일하게 재생시킬 수 있다. 이어, 재생장치에서의 통합 제어를 실현하는 구성요소(명령 ROM(21)~스위치(27))에 대해 설명한다.
- [0270] 명령 ROM(21)은 재생장치의 제어를 규정하는 소프트웨어를 기억하고 있다.
- [0271] 사용자 이벤트 처리부(22)는 리모컨이나 재생장치의 프론트 패널에 대한 키 조작에 따라, 그 조작을 행하는 사용자 이벤트를 CPU(24)에 출력한다.
- [0272] PSR 세트(23)는 재생장치에 내장되는 레지스터이며, 64개의 Player Status Register(PSR)와, 4096개의 General Purpose Register(GPR)로 이루어진다. Player Status Register의 설정 값(PSR) 중, PSR4~PSR8은 현재의 재생시점을 표현하는데 이용된다.
- [0273] PSR4는 1~100의 값으로 설정됨으로써 현재의 재생시점이 속하는 타이틀을 나타내고, 0으로 설정됨으로써 현재의 재생시점이 탑 메뉴임을 나타낸다.
- [0274] PSR5는 1~999의 값으로 설정됨으로써 현재의 재생시점이 속하는 챕터 번호를 나타내고, 0xFFFF로 설정됨으로써 재생장치에서 챕터 번호가 무효임을 나타낸다.
- [0275] PSR6은 0~999의 값으로 설정됨으로써 현재의 재생시점이 속하는 PL(현재 PL)의 번호를 나타낸다.
- [0276] PSR7은 0~255의 값으로 설정됨으로써 현재의 재생시점이 속하는 PlayItem(현재 Play Item)의 번호를 나타낸다.
- [0277] PSR8은 0~0xFFFFFFFF의 값으로 설정됨으로써 45KHz의 시간 정밀도를 이용하여 현재의 재생시점(현재 PTM(Presentation Time))을 나타낸다. 이상의 PSR4~PSR8에 의해, 도 21(a)에서의 BD-ROM 전체의 시간 축에서, 현재의 재생시점은 어디인지를 특정할 수 있다.
- [0278] CPU(24)는 명령 ROM(21)에 저장되어 있는 소프트웨어를 실행하여 재생장치 전체의 제어를 실행한다. 이 제어의 내용은 사용자 이벤트 처리부(22)로부터 출력된 사용자 이벤트 및 PSR 세트(23)에서의 각 PSR의 설정 값에 따라 동적으로 변화한다.
- [0279] 시나리오 메모리(25)는 현재의 PL 정보나 현재의 Clip 정보를 저장해 두기 위한 메모리이다. 현재 PL 정보란, BD-ROM에 기록되어 있는 복수 PL 정보 중, 현재 처리 대상이 되어 있는 것을 말한다. 현재 Clip 정보란 BD-ROM에 기록되어 있는 복수 Clip 정보 중, 현재 처리 대상이 되어 있는 것을 말한다.
- [0280] 로컬 메모리(26)는, BD-ROM으로부터의 판독은 저속이므로, BD-ROM 기록 내용을 일시적으로 저장해 두기 위한 캐시 메모리(cache memory)이다. 이러한 로컬 메모리(26)가 존재함으로써, BD-J 모드에서의 애플리케이션 실행이 효율적으로 이루어지게 된다.
- [0281] 스위치(27)는, BD-ROM 및 Local Storage(18)로부터 판독된 각종 데이터를 리드 버퍼(2), 리드 버퍼(19), 시나리오 메모리(25), 로컬 메모리(26) 중 어느 하나에 선택적으로 투입하는 스위치이다.

- [0282] 이상인, 본 실시 예에 관한 재생장치의 하드웨어 구성이다. 이어, 본 실시 예에 관한 재생장치의 소프트웨어 구성에 대해 설명한다.
- [0283] 도 33은 ROM(24)에 저장된 소프트웨어와 하드웨어로 이루어지는 부분을 레이어 구성으로 치환하여 묘화한 도면이다. 본 도면에 나타내는 바와 같이, 재생장치의 레이어 구성은 이하의 a), b), c)로 이루어진다. 즉,
- [0284] a) BD Player Device의 제 1 계층,
- [0285] b) BD Play Model의 제 2 계층,
- [0286] c) Application Runtime Enviroment의 제 3 계층으로 이루어진다.
- [0287] 이들 계층 중 도 32에 나타내는 재생장치의 하드웨어 구성은 제 1 계층에 속하게 된다. 본 도면의 제 1 계층 "BD-Player Device"에는 도 32에 도시한 하드웨어 구성 중 비디오 디코더(4), P-Graphics 디코더(6), I-Graphics 디코더(10), 오디오 디코더(16)로 이루어지는 "디코더"와, 비디오 플레인(5), 프레젠테이션 그래픽스 플레인(7), 인터랙티브 그래픽스 플레인(12)으로 이루어지는 "플레인", BD-ROM 및 그 파일시스템, Local Storage(18) 및 그 파일시스템을 포함한다.
- [0288] 제 2 계층 "BD Player Model"은 이하의 b1), b2)의 층으로 이루어진다. 즉,
- [0289] b2) 플레이 백 컨트롤 엔진(32)의 층
- [0290] b1) 가상파일시스템(Virtual File System, 30) 및 프레젠테이션 엔진(31) 층으로 이루어지고, 자신보다 상위의 계층에 대해 기능 API를 제공한다.
- [0291] 제 3 계층 "Application Runtime Enviroment"는 이하의 c1), c2)의 계층으로 이루어진다. 즉,
- [0292] c1) 모듈매니저(34)가 존재하는 층,
- [0293] c2) HDMV 모듈(33), BD-J 모듈(35)이 존재하는 층으로 이루어진다. 도 33의 레이어 모델에서 이 모듈매니저(34)가 최상위 계층에 위치하나, 모듈매니저(34)는 HDMV 모듈(33) 및 BD-J 모듈(35)을 바이패스하여 플레이 백 컨트롤 엔진(32)에 이르는 우회로(ur1)를 갖고 있다. 이 우회로에 의해 모듈매니저(34)는 도 33의 레이어 모델에서 "L" 자형을 거꾸로 한 형상을 이루며, 사용자 이벤트 매니저(37)를 내장한 구성으로 되어 있다.
- [0294] BD-J 모듈(35)은 이른바 Java(등록상표) 플랫폼이며, Java(등록상표) 가상머신(38)을 중핵으로 한 구성으로 되어 있다. Java(등록상표) 가상머신(38) 내의 워크 메모리는 여러 시스템 프로그램이나 애플리케이션이 동작하고 있다. Java(등록상표) 가상머신(38)의 상위에 묘화한 애플리케이션 매니저(36), Event Listner Manager(39)(Default Operation Manager(40))는 그러한 시스템 프로그램의 하나이다. 이 중 애플리케이션 매니저(36) 중에는 PLMT 프로세서(41)가 존재하고 있다. 또, BD-J 모듈(35)과, 플레이 백 컨트롤 엔진(32)의 사이에는 퍼미션 컨트롤러(Permission Controller, 42)가 개재되어 있다.
- [0295] 우선 먼저, 제 2층에 속하는 가상파일시스템(30)~모듈매니저(34)에 대해 설명한다. 도 34는 가상파일시스템(30)~모듈매니저(34)에 의한 처리를 모식화 한 도면이다.
- [0296] 가상파일시스템(30)은 Local Storage(18)에 저장된 다운로드 콘텐츠를 BD-ROM에서의 디스크 콘텐츠와 일체적으로 취급하기 위한 가상적인 파일시스템이다. 여기서, Local Storage(18)에 저장된 다운로드 콘텐츠는 SubClip, Clip 정보, 플레이리스트 정보를 포함한다. 이 다운로드 콘텐츠에서의 플레이리스트 정보는 BD-ROM 및 Local Storage(18) 중 어느 하나에 존재하는 Clip 정보라도 지정할 수 있는 점에서 BD-ROM 상의 플레이리스트 정보와 다르다. 이 지정시에, 가상파일시스템(30) 상의 플레이리스트 정보는 BD-ROM이나 Local Storage(18)에서의 파일을 풀 패스(Pull Path)로 지정할 필요는 없다. BD-ROM 상의 파일시스템이나 Local Storage(18) 상의 파일시스템은 가상적인 하나의 파일시스템(가상파일시스템(30))으로서 인식되기 때문이다. 이 때문에, PlayItem 정보에서의 Clip_Information_file_name 및 SubPlayItem 정보의 Clip_Information_file_name은 Clip 정보가 저장한 파일의 파일 보디(file body)에 해당하는 5 자릿수의 수치를 지정함으로써, 가상파일시스템(30), BD-ROM 상의 AVClip을 지정할 수 있다. 가상파일시스템(30)을 통해 Local Storage(18)의 기록 내용을 판독하고, BD-ROM의 기록 내용과 동적으로 조합함으로써, 다양한 재생의 변화(variation)를 산출할 수 있다. Local Storage(18)와 BD-ROM을 조합하여 이루어지는 디스크 콘텐츠는 BD-ROM에서의 디스크 콘텐츠와 대등하게 취급되기 때문에, 본원에서의 "BD-ROM"은 Local Storage(18) + BD-ROM의 조합으로 이루어지는 가상적인 기록매체를 포함하게 된다.
- [0297] 프레젠테이션 엔진(31)은 AV 재생기능을 실행한다. 재생장치의 AV 재생기능이란, DVD 플레이어, CD 플레이어

부터 답습한 전통적인 기능 군이며, 재생개시(Play), 재생정지(Stop), 일시정지(Pause On), 일시정지의 해제(Pause Off), 스틸 기능의 해제(still off), 속도지정부 빨리 감기(Forward Play(speed)), 속도지정부 되감기(Backward Play(speed)), 음성 전환(Audio Change), 부영상 전환(Subtitle Change), 앵글전환(Angle Change)의 기능이다. AV 재생기능을 실현하기 위해, 프레젠테이션 엔진(31)은 리드 버퍼(2) 상에 판독된 AVClip 중 원하는 시각에 해당하는 부분의 디코드를 행하도록, 비디오 디코더(4), P-Graphics 디코더(6), I-Graphics 디코더(10), 오디오 디코더(16)를 제어한다. 원하는 시각으로 하여 PSR(현재 PTM)에 나타나는 개소의 디코드를 행하게 함으로써, AVClip에서 임의의 시점의 재생을 가능하게 할 수 있다.

[0298] 재생제어 엔진(Playback Control Engine(PCE))(32)은 플레이리스트에 대한 재생제어기능(i), PSR 세트(23)에서의 상태취득/설정 기능(ii)의 모든 기능을 실행한다. PL에 대한 재생제어기능이란, 프레젠테이션 엔진(31)이 행하는 AV 재생기능 중, 재생개시나 재생정지를 현재 PL 정보 및 Clip 정보에 따라 행하게 하는 것을 말한다. 이들 기능(i)~(ii)은 HDMV 모듈(33)~BD-J 모듈(35)로부터의 기능 콜에 따라 실행한다.

[0299] 즉, PL 재생을 명하는 기능 콜이 있으면, 플레이 백 컨트롤 엔진(32)은 재생이 명해진 플레이리스트 정보를 가상파일시스템(30)을 경유하여 BD-ROM 또는 Local Storage(18)로부터 판독한다. 그렇게 판독된 플레이리스트 정보 내의 PlayItem 정보를 참조하여, 그 PlayItem 정보에서 Clip_Information_file_name에 기재되어 있는 Clip 정보를 가상파일시스템(30)을 경유하여 BD-ROM 또는 Local Storage(18)로부터 판독한다. 도 34에서의 ◎ 1, 2, 3, 4는 가상파일시스템(30)을 경유한 플레이리스트 정보 판독(◎1), 플레이리스트 정보를 구성하는 PlayItem 정보의 해독(◎2), 가상파일시스템(30)을 경유한 Clip 정보 판독(◎3), Clip 정보의 해독(◎4)을 모식화 한 것이다. 이상의 과정을 거쳐 Clip 정보, 플레이리스트 정보가 해독되면, AVClip을 구성하는 TS 패킷을 가상파일시스템(30)을 통해 프레젠테이션 엔진(31)에 인계한다. 이렇게 하여 프레젠테이션 엔진(31)에 TS 패킷이 순차 인계되면, 프레젠테이션 엔진(31)은 AVClip을 구성하는 TS 패킷을 디코더에 출력하여 플레인에 표시시킨다. 도면 중의 ☆ 1, 2, 3, 4는 AVClip을 구성하는 TS 패킷의 판독(☆1, 2), 가상파일시스템(30)으로부터 프레젠테이션 엔진(31)으로의 TS 패킷 인계(☆3), 디코더로의 TS 패킷 투입(☆4), 디코더로부터 각종 플레인으로의 디코드 결과 출력(☆5)을 모식적으로 나타내고 있다.

[0300] HDMV 모듈(33)은 HDMV 모드의 실행주체이며, mobj_id에 의해 분기처 MovieObject를 지정한 활성화 요구(activate(mobj_id))가 모듈매니저(34)로부터 이루어지면, MovieObject(mobj_id)를 로컬 메모리(26)에 판독하여, 이 MovieObject에 기술된 내비게이션 코멘드를 해독하고, 해독 결과에 기초하여 플레이 백 컨트롤 엔진(32)에 대한 기능 콜을 실행한다. 도 28에서 ▽2, ▽3, ▽4가 부여된 화살표는 모듈 매니저(34)에 의한 activate(mobj_id)(2), MovieObject에 기술된 내비게이션 코멘드의 해독(3), 플레이 백 컨트롤 엔진(32)에 대한 기능 콜(4)을 모식적으로 나타내고 있다.

[0301] 모듈매니저(34)는, BD-ROM으로부터 판독된 Index.bdmv를 유지하며, 분기 제어를 행한다. 이 분기 제어는 현재 타이틀을 구성하는 동적 시나리오에 터미네이트 이벤트(Terminate Event) 발행하고, 분기처 타이틀을 구성하는 동적 시나리오에 액티베이트 이벤트(Activate Event)를 발행함으로써 이루어진다. title_id를 지정한 점프 타이틀 코멘드(JumpTitle(title_id))를 MovieObject가 실행한 경우, 현재 타이틀을 구성하는 MovieObject에 터미네이트 이벤트를 발행하고, title_id에 대응하는 타이틀을 구성하는 Movie Object를 활성화하도록 Activate(mobj_id) 이벤트를 발행한다. 도면 중의 ▽0, ▽1, ▽2가 부여된 화살표는 JumpTitle 코멘드의 실행(▽0), 모듈매니저(34)에 의한 Index.bdmv 참조(▽1), 분기처 타이틀을 구성하는 Movie Object의 활성화(▽2)의 통지를 모식적으로 나타내고 있다. JumpTitleAPI (JumpTitle(title_id))을 BD-J Object가 콜한 경우도 마찬가지이며, 현재 타이틀을 구성하는 BD-J Object에 터미네이트 이벤트를 발행하고, 그 title_id에 대응하는 타이틀을 구성하는 BD-J Object를 활성화하도록 activate(bobj_id)를 BD-J 모듈(35)에 발행한다.

[0302] 이상이 프레젠테이션 엔진(31)~모듈매니저(34)에 대한 설명이다. 이어서, 애플리케이션 매니저(36)에 대해 도 35를 참조하면서 설명한다. 도 35는 애플리케이션 매니저(36)를 나타내는 도면이다.

[0303] 애플리케이션 매니저(36)는 타이틀 분기가 발생할 때마다, 분기 전 타이틀에서는 실행되고 있지 않으나, 새로운 타이틀에서는 AutoRun의 기동속성을 갖는 애플리케이션을 기동하도록 Java(등록상표) 가상머신(38)에 지시한다. 그와 함께, 분기 전 타이틀에서는 실행되고 있었으나, 새로운 타이틀을 생존구간으로 하지 않는 애플리케이션을 종료시킨다. 이들 기동제어 및 종료제어는 현재 BD-J Object에서의 애플리케이션 관리테이블을 참조한 상태에서 이루어진다. 여기서, 타이틀 분기가 있었던 경우, 모듈매니저(34)로부터 activate(bobj_id)가 통지된다. 이러한 통지가 있으면, bobj_id에 대응하는 BD-J Object를 현재 BD-J Object로 하여, 그 현재 BD-J Object에서의 애플리케이션 관리테이블을 참조함으로써, 상술한 자동으로 기동해야 할 애플리케이션 및 자동으로 종료해야 할 애플

플리케이션을 특정한다. 도 35에서의 ☆0, ☆1, ☆2, ☆3은 TitleJump의 발생(☆0), activate(bobj_id)의 통지(☆1), 애플리케이션 관리테이블의 참조(☆2), Java(등록상표) 가상머신(38)에 대한 애플리케이션 기동지시(☆3)라는 일련의 과정을 모식화 하여 나타낸다. 이 기동지시에 의해 Java(등록상표) 가상머신(38)은 로컬 메모리(26)로부터 워크 메모리에 xlet 프로그램을 판독한다(☆4, 5).

[0304] 이상이 애플리케이션 매니저(36)에 대한 설명이다. 이어, 사용자 이벤트 매니저(37)~디폴트 오퍼레이션 매니저(40)에 대해 도 36을 참조하면서 설명한다.

[0305] 사용자 이벤트 매니저(37)는 사용자 이벤트 처리부(22)가 접수한 사용자 이벤트를 재생제어를 위한 사용자 이벤트와 키 이벤트로 분리한다. 재생제어를 위한 사용자 이벤트란, 재생개시(Play), 재생정지(Stop), 일시정지(Pause On), 일시정지의 해제(Pause Off), 스틸 기능의 해제(still off), 속도지정부 빨리 감기(Forward Play(speed)), 속도지정 되감기(Backward Play(speed)), 음성절환(Audio Change), 부영상 절환(Subtitle Change), 앵글절환(Angle Change) 중 어느 하나를 명하는 사용자 이벤트이다. 한편, 키 이벤트란, Pop-up 키, MoveUp 키, MoveDown 키, MoveRight키, MoveLeft 키, Numeric 키 중 어느 하나의 누름을 나타내는 사용자 이벤트이다. 그리고 재생제어를 위한 사용자 이벤트에 기초하여, 재생제어 기능을 플레이 백 컨트롤 엔진(32)에 실행시키기 위한 기능 콜을 행한다. 이 기능 콜은 U0(User Operation)라 불리며, HDMV 모듈(33), BD-J 모듈(35)을 통하지 않고, 모듈매니저(34)에서의 우회로에 있는 U0 컨트롤러(37a)를 이용하여 이루어진다. 이로써 재생개시(Play), 재생정지(Stop), 일시정지(Pause On), 일시정지의 해제(Pause Off)라는 재생제어는 지연 없이 이루어지게 된다. 도면 중의 ☆1, 2, 3은 사용자 이벤트 매니저(37)에 의한 재생제어 사용자 이벤트와, 키 이벤트의 분리(☆1, 2), 재생제어 사용자 이벤트에 기초하는 플레이 백 컨트롤 엔진(32)에 대한 기능 콜(☆3)을 모식적으로 나타내고 있다.

[0306] Java(등록상표) 가상머신(38)은 애플리케이션을 구성하는 xlet 프로그램을 워크 메모리에 로드하여 xlet 프로그램을 해독하고, 해독 결과에 따라서 하위층에 대한 제어를 행한다. 하위층에 대한 제어는 JMF 메소드를 BD 미들웨어(middleware, 도시 생략)로 발행하여 BD 재생장치가 대응하고 있는 기능 콜로 치환시키고, 치환 후의 기능 콜을 플레이 백 컨트롤 엔진(32)에 발행함으로써 이루어진다.

[0307] Event Listner Manager(39)는 키 이벤트를 해석하고, 이벤트의 분배를 행한다. 도면 중의 실선 화살표 ◇1, ◇2는 이 Event Listner Manager(39)에 의한 분배를 모식적으로 나타낸다. xlet 프로그램 내의 Event Listner에 등록된 키 이벤트라면, BD-J Object에 의해 간접 참조 되고 있는 xlet 프로그램에 관한 이벤트를 분배한다. xlet 프로그램의 Event Listner에는 JMF에 대응한 키 이벤트가 등록되어 있으므로, 본 키 이벤트에 의해 xlet 프로그램의 기동이 가능해진다. 키 이벤트가 Event Listner 비등록의 키 이벤트일 경우, 본 키 이벤트를 디폴트 오퍼레이션 매니저(40)로 분배한다. BD-ROM 재생장치에서 생기는 키 이벤트에는 Event Listner에 등록되어 있지 않은 다양한 것이 있으며, 이들 키 이벤트가 발생했다 해도 빠짐없는 처리를 실행하기 위함이다.

[0308] 디폴트 오퍼레이션 매니저(40)는 xlet 프로그램 내의 Event Listner에 등록되어 있지 않은 키 이벤트가 Event Listner Manager(39)로부터 배분되면, 그 Event Listner 비등록 이벤트에 대응하는 기능 콜을 플레이 백 컨트롤 엔진(32)에 대해 실행한다. 이 디폴트 오퍼레이션 매니저(40)에 의한 기능 콜을 모식적으로 나타낸 것이 도면 중의 화살표 ◇3이다.

[0309] PLMT 프로세서(41)는, 애플리케이션 매니저(36)의 한 구성요소이며, 모듈매니저(34)로부터 activate(bobj_id)가 이루어지면, 그 bobj_id에 의해 특정되는 BD-J Object의 플레이리스트 관리테이블을 참조한다. 그리고 재생속성 = AutoPlay의 PL이 그 BD-J Object의 플레이리스트 관리테이블 내에 존재하면, 그 AutoPlayPL을 재생하도록 플레이 백 컨트롤 엔진(32)에 출력한다. 한편, PL에 의한 재생 종료료를 나타내는 노티파이 이벤트(Notify Event)를 플레이 백 컨트롤 엔진(32)이 발행하면, 그 발행된 시점을 타이틀 종료 시점으로서 해석한다. 도면 중의 △1, 2는 플레이 백 컨트롤 엔진(32)에 대한 PlayPL 기능의 콜(△1), 플레이 백 컨트롤 엔진(32)으로부터의 노티파이 이벤트 출력(△2)을 모식화 하고 있다.

[0310] 이상으로 BD-J 모듈(35) 내의 레이어 구성에 대한 설명을 마친다. 또한, 퍼미션 컨트롤러(42)에 대해서는 본 실시 예에서의 설명을 생략한다. 퍼미션 컨트롤러(42)의 처리 내용은 본 실시 예보다도 제 3 실시 예에서 상세하게 설명하는 것이 바람직하기 때문이다.

[0311] <Java(등록상표) 가상머신(38)의 내부 구성>

[0312] Java(등록상표) 가상머신(38)의 내부 구성에 대해 설명한다. 도 37은 Java(등록상표) 가상머신(38)의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이 Java(등록상표) 가상머신(38)은, 도 32에 도시한 CPU(2

4)와, 사용자 클래스 로더(user class loader)(52), 메소드 영역(method area)(53), 워크 메모리(54), 스레드(thread)(55a, b, ..., n), Java(등록상표) 스택(56a, b, ..., n)으로 구성된다.

[0313] 사용자 클래스 로더(52)는 BDJA 디렉터리의 Java(등록상표) 아카이브 파일에서의 클래스파일을 로컬 메모리(26)등으로부터 판독하여 메소드 영역(53)에 저장한다. 이 사용자 클래스 로더(52)에 의한 클래스파일 판독은 파일 경로(path)를 지정한 판독을 애플리케이션 매니저(36)가 사용자 클래스 로더(52)에 지시함으로써 이루어진다. 파일 경로가 로컬 메모리(26)를 나타내고 있으면, 사용자 클래스 로더(52)는 애플리케이션을 구성하는 Java(등록상표) 아카이브 파일에서의 클래스파일을 로컬 메모리(26)로부터 워크 메모리(54)에 판독한다. 파일 경로가 가상파일시스템(30) 상의 디렉터리를 나타내고 있으면, 사용자 클래스 로더(52)는 애플리케이션을 구성하는 Java(등록상표) 아카이브 파일에서의 클래스파일을 BD-ROM 또는 Local Storage(18)로부터 워크 메모리(54)로 판독한다. 도 35에 나타내는 애플리케이션 기동제어(☆3, 4, 5)는 이 사용자 클래스 로더(52)에 의한 클래스파일 판독에 의해 실현된다. 판독이 지시된 클래스파일이 로컬 메모리(26)에 없는 경우, 사용자 클래스 로더(52)는 판독 실패를 애플리케이션 매니저(36)에 통지하게 된다.

[0314] 메소드 영역(53)은 사용자 클래스 로더(52)에 의해 로컬 메모리(26)로부터 판독된 클래스파일이 저장된다. 워크 메모리(54)는 이른바 힙영역(heap area)이며, 여러 클래스파일의 인스턴스가 저장된다. 도 33에 나타낸 애플리케이션 매니저(36), Event Listener Manager(39)는 이 워크 메모리(54)에 상주하는 레지던트 애플리케이션(resident application)이다. 워크 메모리(54)에는 이들 레지던트형 인스턴스 외에, 메소드 영역(53)에 판독된 클래스파일에 대응하는 인스턴스가 저장된다. 이 인스턴스가 애플리케이션을 구성하는 xlet 프로그램이다. 이러한 xlet 프로그램을 워크 메모리(54)에 배치함으로써 애플리케이션은 실행 가능한 상태가 된다. 도 33, 도 35, 도 36의 레이어 모델에서는 이 워크 메모리(54)를 Java(등록상표) 가상머신(38) 상위층에 기술하고 있었다. 이는 워크 메모리(54) 상의 애플리케이션 매니저(36) 및 Event Listener Manager(39)를 알기 쉽게 기술하기 위한 배려에 지나지 않는다. 애플리케이션 매니저(36) 및 Event Listener Manager(39)는 인스턴스로서 스레드(55a, b, ..., n)에 의해 실행된다는 것이 현실적인 기술이 된다.

[0315] 스레드(55a, b, ..., n)는 워크 메모리(54)에 저장된 메소드를 실행하는 논리적인 실행주체이며, 로컬변수나 오퍼랜드 스택(operand stack)에 저장된 인수를 오퍼랜드로 하여 연산을 행하고, 연산 결과를 로컬변수 또는 오퍼랜드 스택에 저장한다. 도면 중의 화살표 ky1, ky2, kyn은 워크 메모리(54)로부터 스레드(55a, b, ..., n)로의 메소드 공급을 상징적으로 나타내고 있다. 물리적인 실행주체가 유일하게 CPU 하나인데 대해, 논리적인 실행주체인 스레드는 최대 64개가 Java(등록상표) 가상머신(38) 내에 존재할 수 있다. 이 64개라는 수치 내에서 스레드를 신규로 작성하는 것도, 기존의 스레드를 삭제하는 것도 가능하며, 스레드의 동작 수는 Java(등록상표) 가상머신(38)의 동작 중 증감할 수 있다. 스레드의 수는 적절히 증가시킬 수 있으므로, 복수 스레드에 의해 하나의 인스턴스의 병렬 실행을 행하고, 인스턴스의 고속화를 도모할 수도 있다. 본 도면에서는 CPU(24)와 스레드의 대응관계는 1대 다(多)의 관계로 하고 있으나, CPU가 복수일 경우, CPU와 스레드의 대응 관계는 다 대 다의 관계가 될 수 있다. 스레드(55a, b, ..., n)에 의한 메소드 실행은 메소드를 이루는 바이트 코드(byte code)를 CPU(24)의 네이티브 코드(native code)로 변환한 다음 CPU(24)에 발행함으로써 이루어진다. 이 네이티브 코드 변환에 대해서는 본원의 주안점에서 벗어나므로 설명을 생략한다.

[0316] Java(등록상표) 스택(56a, b, ..., n)은 스레드(55a, b, ..., n)와 1대 1의 비율로 존재하고 있으며, 내부에 프로그램 카운터(도면 중의 PC)와 하나 이상의 프레임を 가지고 있다. "프로그램 카운터"는 인스턴스에서 현재 어느 부분이 실행되고 있는지를 나타낸다. "프레임"은 메소드에 대한 1회의 콜에 대해 할당된 스택시스템의 영역이며, 그 1회의 콜 시의 인수가 저장되는 "오퍼랜드 스택"과 콜된 메소드가 이용하는 "로컬변수 스택(도면 중의 로컬변수)"으로 이루어진다. 프레임은 콜이 1회 이루어질 때마다 Java(등록상표) 스택(56a, b, ..., n) 상에 누적되는 것이므로, 어느 메소드가 자신을 재귀적으로 호출하는 경우에도 이 프레임은 하나 누적되게 된다. JMF 플레이어 인스턴스의 재생 메소드를 콜하는 경우, 또는 JumpTitleAPI의 콜을 콜 하는 경우에도, 이들 콜에 대응하는 프레임이 Java(등록상표) 스택(56a, b, ..., n) 상에 누적된다. 그리고 이들 프레임의 오퍼랜드 스택에는 재생 메소드에 의한 재생의 대상이 되는 MPLS 파일의 파일명, 또는 JumpTitleAPI의 콜 점프 처를 나타내는 title_id 등이 인수로서 저장된다.

[0317] 이상과 같이 구성된 Java(등록상표) 가상머신(38)의 내부 구성에서, 애플리케이션 매니저(36) 및 그 구성요소인 PLMT 프로세서(41)가 어떠한 처리를 행하는지를 상세하게 설명한다.

[0318] bobj_id의 BD-J Object의 활성화를 요구하는 이벤트(activatred[bobj_id])를 모듈매니저(34)가 출력하면, 워크 메모리(54) 상의 한 인스턴스인 애플리케이션 매니저(36)는 그 bobj_id를 갖는 BD-J Object를 새로이 현재 BD-J

Object로 한다. 그리고 애플리케이션 매니저(36)는 분기원 타이틀에서의 실행상태와, 현재 BD-J Object에서의 애플리케이션의 기동속성을 대조하여 자동으로 기동해야 할 애플리케이션(i), 자동으로 종료해야 할 애플리케이션(ii)을 판정한다.

- [0319] (i) 자동으로 기동해야 할 애플리케이션의 판정은 분기 전 타이틀에서 비 실행 상태이지만, 현재 BD-J Object에서 AutoPlay 속성을 갖는 애플리케이션의 apli_id_ref를 현재 BD-J Object의 애플리케이션 관리테이블로부터 검색함으로써 이루어진다. 이러한 apli_id_ref가 있으면, 애플리케이션 [apli_id_ref]에 대한 Java(등록상표) 아카이브 파일에 속하는 클래스파일을 메소드 영역(53)으로 판독하도록 사용자 클래스 로더(52)에 지시하고, 그 클래스파일에 대한 인스턴스를 워크 메모리(54) 상에 생성시킨다. 이렇게 하여 현재 타이틀을 생존구간으로 하고 있는 애플리케이션이 실행 가능한 상태가 된다. 그 후, 애플리케이션에서의 메소드를 스레드(55a, b, ..., n)에 실행시킴으로써, 애플리케이션을 실행시킨다.
- [0320] (ii) 자동으로 종료해야 할 애플리케이션의 판정은 분기원 타이틀에서 실행 상태이지만, 현재 타이틀에서 생존구간을 갖지 않는 애플리케이션의 apli_id_ref를 현재 BD-J Object의 애플리케이션 관리테이블로부터 검색함으로써 이루어진다. 이러한 apli_id_ref가 있으면, 이러한 애플리케이션을 구성하는 xlet 프로그램을 종료시킨다. 이로써 당해 애플리케이션이 워크 메모리(54) 상에서 차지하고 있던 영역이나 그 애플리케이션의 메소드 실행을 위한 Java(등록상표) 스택(56a, b, ..., n)에서의 프레임 등의 리소스가 회수되게 된다.
- [0321] 애플리케이션 매니저(36)의 구성요소인 PLMT 프로세서(41)는 분기원 타이틀에서의 재생상태와, 현재 타이틀에서의 플레이리스트의 재생속성을 대조하여, 자동으로 재생해야 할 플레이리스트(i), 자동으로 종료해야 할 플레이리스트(ii)를 판정한다.
- [0322] (i) 자동으로 재생해야 할 플레이리스트의 판정은, 분기원 타이틀에서 비 재생상태이며, 현재 타이틀에서 AutoPlay 속성으로 설정되어 있는 플레이리스트가 플레이리스트 관리테이블 내에 존재하는가 여부의 검색을 플레이리스트 관리테이블에 대해 행함으로써 이루어진다. 만약 있으면, 재생해야 할 플레이리스트의 PL_id_ref를 인수로 한 플레이리스트 재생의 기능 콜을 실행한다. 이 콜에 의해 인수인 PL_id_ref를 오퍼랜드 스택에 저장한 프레임이 Java(등록상표) 스택(56a, b, ..., n)에 생성된다. 그리고 플레이리스트 재생의 기능 콜을 스레드(55a, b, ..., n)가 실행한다. 이 기능 콜 실행에 의해 프레젠테이션 엔진(31)은 플레이리스트 재생을 개시한다.
- [0323] (ii) 자동으로 종료해야 할 플레이리스트의 판정은, 분기원 타이틀에서 재생중 상태이며, 현재 타이틀의 플레이리스트 관리테이블에 기술되어 있지 않은 플레이리스트가 존재하는가 여부의 검색을 플레이리스트 관리테이블에 대해 행함으로써 이루어진다. 만약 있으면, 그 플레이리스트의 재생정지를 명하는 기능 콜을 실행하여, 그 플레이리스트의 재생을 위한 기능 콜에 대응하는 프레임을 Java(등록상표) 스택(56a, b, ..., n)으로부터 삭제한다.
- [0324] 워크 메모리(54) 상에서의 애플리케이션 종료에는 4개의 형태가 있다. 도 38은 애플리케이션 종료의 4개의 형태를 나타내는 도면이다. 첫 번째는, 리소스가 고갈되었으므로 애플리케이션 매니저(36)에 의한 터미네이트 이벤트를 발행에 의해 애플리케이션이 종료하는 형태(☆1), 두 번째는, 다른 애플리케이션이 발행한 터미네이트 이벤트를 애플리케이션 매니저(36)를 통해 접수하여 종료하는 형태(☆2), 세 번째는, 애플리케이션 관리테이블에 기술된 생존구간이 종료되어 애플리케이션 매니저(36)가 발행한 터미네이트 이벤트에 의해 종료하는 형태(☆3), 네 번째는, 애플리케이션이 자신의 종료 프로세스를 기동함으로써 종료하는 형태이다(☆4). 이들 4개의 형태 중 3개가 애플리케이션 매니저(36)에 의해 이루어지고 있으므로, 애플리케이션 매니저(36)는 애플리케이션 동작 제어의 중추를 장악함을 알 수 있다. 가령, 터미네이트 이벤트의 발행으로도 애플리케이션이 종료하지 않는 경우, 애플리케이션 매니저(36)는 그 애플리케이션을 강제 종료시켜서, 사용하고 있는 리소스를 내어주도록 할 수 있다. 강권적인 리소스 해방을 담당하는 것도 애플리케이션 매니저(36)의 하나의 특징이다.
- [0325] 이상이 BD-J 모듈(35)의 구성요소이다.
- [0326] (플로차트의 설명)
- [0327] 이상의 애플리케이션 매니저(36)에 대한 설명은 그 개요만을 살펴본 것에 지나지 않는다. 애플리케이션 매니저(36)의 처리를 더 상세하게 나타낸 것이 도 39, 도 40의 플로차트이다. 이후, 이들 플로차트를 참조하여 애플리케이션 매니저(36)의 처리순서에 대해 보다 상세하게 설명한다.
- [0328] 도 39는 애플리케이션 매니저(36)에 의한 처리순서를 나타내는 도면이다. 본 도면에서의 처리순서는 단계 S1-단계 S2-단계S3-단계 S4로 이루어지는 메인 루프를 가지고 있다. 단계 S1은 TitleJump가 이루어졌는가 여부의 판정이며, 만약 이루어졌으면 타이틀 전환을 행하고(단계 S7), 분기처 타이틀의 애플리케이션 관리테이블을 참조함으로써, 분기원 타이틀에서 기동중이고, 분기처 타이틀에서 비 생존인 애플리케이션을 종료시킨다(단계 S8).

한편, 분기처 타이틀의 플레이리스트 관리테이블을 참조함으로써, 분기원 타이틀에서 재생중이며, 분기처 타이틀에서 비 생존이 되는 PL의 재생을 정지시킨다(단계 S9).

[0329] 그리고, 분기원 타이틀에서는 비 재생상태, 분기처 타이틀에서 AutoPlay 속성으로 되어 있는 PL을 PLMT 프로세서(41)가 판정하도록 한다(단계 S10). 만약 존재하면, PLMT 프로세서(41)는 AutoPlayPL을 재생하도록 플레이 백 컨트롤 엔진(32)에 지시한다(단계 S11). 만약 존재하지 않는다면, AutoPlayPL은 재생하지 않는다.

[0330] 이어, 단계 S12~단계 S18로 이루어지는 일련의 순서는 분기처 타이틀을 생존구간으로 하는 애플리케이션의 기동을 실현한다. 이 순서는 AutoRun 애플리케이션에 대해 기동지시를 행하고(단계 S14), 기동에 성공하면(단계 S15에서 Yes), AutoPlayPL의 재생화상을 쿼터(1/4)로 변환하는(단계 S18) 순서를 실현하는 것이다.

[0331] 한편, 이 단계 S15가 No이면, 단계 S14~단계 S17로 이루어지는 루프처리를 실행하게 된다. 본 루프처리에서의 제어변수는 재기동 카운터이다. 재기동 카운터는 애플리케이션의 재기동 횟수를 규정하는 카운터이다. 본 재기동 카운터는 단계 S12에서 리셋 되고, 단계 S16에서 0인가 여부의 판정이 이루어진다. 0이 아닌 경우, 단계 S17에서 재기동 카운터는 감소(decrement)된다. 이상의 단계 S14, 단계 S15, 단계 S16, 단계 S17로 이루어지는 루프처리에 의해, 재기동 카운터가 0이 아닌 한, AutoRun 애플리케이션의 기동은 반복되게 된다. 이러한 반복에 의해, 애플리케이션의 기동이 보증되게 된다.

[0332] 단계 S2는 메인이 되는 애플리케이션이 실행되고 있지 않은 상태인지의 여부의 판정이며, 만약 그렇다면, 단계 S5의 판정을 행한다. 단계 S5는 애플리케이션이 정상종료되었는가의 판정이다. 만약 이상종료되었으면, 단계 S19, 단계 S20의 처리를 실행한다. 정상종료되었으면, 단계 S19~단계 S20을 실행하지 않고 단계 S1~단계 S4로 이루어지는 메인 루프로 되돌아간다.

[0333] 단계 S19는 AutoPlayPL의 재생중인지의 여부의 판정이며, 만약 재생중이라면, AutoPlayPL의 재생화상을 풀 스크린화하도록 플레이 백 컨트롤 엔진(32)에 지시한다(단계 S20). 그 후, 단계 S16으로 이행한다. 단계 S16으로의 이행에 의해, 이상종료시에도 단계 S14~단계 S17로 이루어지는 루프처리가 실행되게 된다. 이로써, 단계 S12에서 설정된 재기동 카운터의 횟수가 0이 될 때까지 애플리케이션의 재기동은 반복되게 된다.

[0334] 단계 S4는 BD 드라이브(1)에 BD-ROM이 존재하고 있는지의 여부의 판정이며, 만약 BD-ROM이 존재하지 않으면, 모든 애플리케이션에 대해 종료 지시를 발한다(단계 S6).

[0335] 도 40은 플레이리스트 관리테이블, 애플리케이션 관리테이블의 구체 예를 나타낸다. 도 40의 제 1단계는 타이틀의 재생영상을 나타내고, 제 2단계는 타이틀의 시간 축을 나타낸다. 제 3단계는 PL 재생의 진행, 제 4단계는 애플리케이션 실행을 나타낸다. 제 4단계에서 application #1, 타이틀의 개시와 함께 기동되고 있으며, 그 후, 시점 t1에서 동작상태가 된다. 한편, Playlist #1은 타이틀의 개시와 함께 재생이 개시되고 있다. Playlist #1의 재생은 타이틀의 개시와 동일한 시점에 개시되고 있으므로, 제 1단계의 좌측에 나타내는 바와 같이, 타이틀의 재생개시 직후에서 애플리케이션이 동작상태가 될 때까지의 개시 지연(start up delay)에서, 플레이리스트의 재생화상 gj1이 풀 스크린 표시된다. 한편, application #1은 시점 t1에서 동작상태가 되므로, PL 재생화상을 하위 화면(child screen), 애플리케이션의 실행 화상을 상위 화면(parent screen)으로 한 합성화상 gj2가 시점 t1에서 표시되게 된다. 애플리케이션의 실행 화상은, Start 버튼, continue 버튼, POWER 인디케이터를 배치한 게임용 화면이며, 인터랙티브 그래픽스 플레인(12)에 대한 렌더링 처리를 Java(등록상표) 애플리케이션이 실행함으로써 이루어진다. Java(등록상표) 애플리케이션이 이러한 렌더링 처리를 행하기 위해서는, 그래픽스 렌더링이나 문자 렌더링을 위한 라이브러리(library)가 필요하게 된다. 이후 애플리케이션 실행과 PL 재생이 동시에 이루어지는 한, 상위/하위 화면의 표시는 계속된다.

[0336] 그 후, application #1이 이상종료되어, 시점 t2에서 애플리케이션 매니저(36)가 이 이상종료를 검지했다고 하자. 화살표 br1은 이 검지를 상징적으로 나타낸다. 그러면 단계 S20에서 애플리케이션 매니저(36)는 PL의 재생화상을 풀 스크린으로 한다. 도면 중의 시점 t3는 이 풀 스크린화의 타이밍을 나타낸다. 도면 중 제 1단계의 우측에 풀 스크린화로 얻어진 화상 gj3를 나타낸다.

[0337] 이상과 같이 본 실시 예에 의하면, 플레이리스트 관리테이블의 재생속성을 "AutoPlay"로 설정해 둬으로써, Java(등록상표) 애플리케이션이 동작상태가 될 때까지 5~10초라는 시간이 걸렸다고 해도, 그동안 "우선 무엇인가 표시되고 있는 상태"가 된다. 이 "우선 무엇인가가 표시되고 있는 상태"에 의해 타이틀 실행 개시시의 개시지연을 보충할 수 있다.

[0338] 또, 애플리케이션의 기동에 실패했거나, 애플리케이션이 정상이 아니었다 해도, 플레이리스트 관리테이블에서 규정된 플레이리스트의 재생을 계속함으로써, "우선 무엇인가가 표시되고 있는 상태"가 된다. 이로써, 장치가

블랙아웃(black out) 된다는 최악의 케이스를 회피할 수 있으므로, 장치를 제조하는 제조회사에게 최저한도의 안도감을 줄 수 있다.

[0339] (제 2 실시 예)

[0340] 제 2 실시 예는 에러 종료시에서의 리커버리(recovery) 처리를 편집시에 규정해 두기 위한 개량에 관한 것이다. 이러한 리커버리 처리를 규정하기 위해, 본 실시 예에 관한 기록매체에서는 하나의 BD-J Object 내에 에러관리 테이블을 설치하고 있다. 도 41(a)은 제 2 실시 예에 관한 BD-J Object의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, BD-J Object에는 애플리케이션 관리테이블, 플레이리스트 관리테이블 외에, 에러관리테이블(Error Management Table[bobj_id])이 추가되어 있다. 도 41(b)는 에러관리테이블의 내부 구성을 나타내는 도면이다. 본 도면에 나타내는 바와 같이, 에러관리테이블은 Number_of_recovery 개의 리커버리정보(recovery())로 이루어진다. 식별자 recovery_id로 특정되는 임의의 리커버리정보의 내부 구성을 인출선 em1에 의해 클로즈업하고 있다. 이 인출 선에 의하면, 리커버리정보는 리커버리정보에 대응하는 애플리케이션의 식별자를 일의적으로 특정하는 참조 값 『Apli_id_ref』 『Restart_Application_flag』, 『Continuous_Playback_flag』, 『Select_Title_flag』, 『Notify_Event_flag』, 『Reboot_flag』라는 5개의 플래그로 이루어진다. 도 42는 이들 5개의 플래그의 의미 내용을 나타낸다. 이후 에러관리테이블에서의 각 플래그의 의미 내용에 대해 설명한다.

[0341] 『Restart_Application_flag』는 0으로 설정됨으로써 이상종료시에 재기동하지 않음을 나타낸다. 0 이외의 정수 값 n으로 설정됨으로써 재기동을 n회 반복함을 나타낸다. 디폴트값으로서 본 플래그는 0으로 설정되어 있다.

[0342] 『Continuous_Playback_flag』는 0으로 설정됨으로써 이상종료시에 PL 재생을 계속하지 않음을 나타낸다. 정수 값 1로 설정됨으로써 이상종료시에 PL 재생을 계속함을 나타내고, 2로 설정됨으로써 이상종료시에 풀 스크린?1배속으로 PL 재생을 계속함을 나타낸다. 디폴트값으로서 본 플래그는 0으로 설정되어 있다.

[0343] 『Select_Title_flag』는 0으로 설정됨으로써 이상종료시에 타이틀 분기를 행하지 않음을 나타낸다. 0 이외의 정수 값 n으로 설정된 경우, 타이틀 번호 n에 의해 특정되는 타이틀로 Jump함을 나타낸다. 디폴트값으로서 본 플래그는 0으로 설정되어 있다.

[0344] 『Notify_Event_flag』는 0으로 설정됨으로써 이상종료시에 이벤트를 출력하지 않음을 나타낸다. 0 이외의 정수 값 n으로 설정된 경우, 이벤트 번호 n의 이벤트를 출력함을 나타낸다. 디폴트값으로서 본 플래그는 1로 설정되어 있다.

[0345] 『Reboot_flag』는 0으로 설정됨으로써 이상종료시에 재생장치의 부트스트랩(bootstrap)을 행하지 않음을 나타낸다. 정수 값 1로 설정된 경우 부트스트랩을 행함을 나타낸다.

[0346] 이상의 플래그가 존재하므로, 애플리케이션이 이상종료되었을 때, 어떤 리커버리 처리를 실행하는가를 편집시에 미리 규정해 둘 수 있다. 에러관리테이블 기술의 구체 예에 대해서 설명한다. 도 43(a)는 에러관리테이블이 기술된 두 개의 Title(title #1, title #2)을 나타내는 도면이다. title #1의 애플리케이션 관리테이블에는 AutoRun 애플리케이션으로서 application #1이 기술되어 있으며, title #1의 에러관리테이블에는 이 application #1이 이상종료 했을 때에 이용되어야 할 리커버리정보가 기술되어 있다. title #1의 플레이리스트 관리테이블에는 AutoPlay 플레이리스트로서 Playlist #1이 기술되어 있다.

[0347] title #2 측의 애플리케이션 관리테이블에는 AutoRun 애플리케이션으로서 application #2가 기술되어 있으며, title #2 측의 에러관리테이블에는 이 application #2에 대한 리커버리정보가 기술되어 있다.

[0348] 도 43(b)는, 도 43(a)와 같이 기재된 애플리케이션 관리테이블, 에러관리테이블에 기초하는 애플리케이션 실행 및 플레이리스트 재생의 진행을 나타낸다.

[0349] application #1에 대한 리커버리정보는 Continuous_Playback_Flag = 2로 기술되어 있으므로, application #1의 이상종료시에는 플레이리스트 재생이 1배속, 풀 스크린에 의해 계속되게 된다.

[0350] 한편, application #2에 대한 리커버리정보는 Notify_Event_Flag = 2로 기술되어 있으므로, application #2의 이상종료에는 번호 = 2의 이벤트를 출력하게 된다. 이상과 같은 리커버리정보의 기술에 의해, 애플리케이션이 이상종료했을 때의 처리를 타이틀별로, 애플리케이션별로 변화시킬 수 있다. BD-J Object에서 상술한 에러관리테이블이 추가되었으므로, 본 실시 예에 관한 애플리케이션 매니저(36)는 도 44, 도 45의 플로차트에 따라 처리를 행한다. 도 44는 제 2 실시 예에 관한 애플리케이션 매니저(36)의 처리순서를 나타내는 플로차트이다. 본 플로차트는 도 39와 동일하게, 단계 S1~단계 S4를 메인루프로 하고 있다. 이 메인 애플리케이션에서 타이틀이 선

택되었을 때의 처리는 단계 S21~단계 S27에 나타내는 것이 된다.

- [0351] 단계 S21에서는, 분기처 타이틀에 플레이리스트 관리테이블이 존재하는가 여부의 판정을 PLMT 프로세서(41)에 행하게 한다. 만약 존재하면, 분기원 타이틀에서 비 재생상태이나, 분기처 타이틀에서 AutoPlay 속성으로 되어 있는 PL의 재생을 플레이 백 컨트롤 엔진(32)에 개시시키고(단계 S22), 그 재생이 성공했는지의 여부를 판정한다(단계 S23). 만약 성공하면, 단계 S25~단계 S28의 처리를 행한다. 성공하지 않으면 도 45의 플로차트로 이행한다.
- [0352] 분기처 타이틀에 플레이리스트 관리테이블이 없는 경우에는 분기원 타이틀에서 재생상태였던 PL의 재생을 플레이 백 컨트롤 엔진(32)에 정지시키고(단계 S24), 단계 S25~단계 S28의 처리를 행한다.
- [0353] 단계 S25는 분기처 타이틀에 애플리케이션 관리테이블이 존재하는지의 여부의 판정이다. 만약 존재하면, 분기처 애플리케이션에서의 AutoRun 애플리케이션을 기동하여(단계 S26), 단계 S27에서 기동에 성공했는지의 여부를 판정한다. 성공하면, 단계 S1~단계 S4의 루프처리로 되돌아간다. 성공하지 않으면 도 45의 플로차트로 이행한다.
- [0354] 도 45의 플로차트는 애플리케이션이 이상종료한 경우의 처리순서를 나타내는 플로차트이다. 단계 S30은 이상종료한 애플리케이션이 속하는 타이틀에 에러관리테이블이 존재하는가 여부의 판정이며, 만약 존재하지 않으면 단계 S1~단계 S4으로 이루어지는 루프처리로 되돌아간다. 존재하면, 단계 S31~단계 S44의 처리를 거친 다음, 단계 S1~단계 S4로 이루어지는 루프처리로 되돌아간다. 단계 S31은 에러관리테이블에서의 Restart_Application_flag가 0이 아닌지의 판정이다. 0이 아니면, 단계 S40~단계 S44로 이루어지는 루프처리를 실행한다. 이 루프처리는 Restart_Application_flag의 값 n을 재기동 카운터로 설정하여(단계 S40), 단계 S41~단계 S44의 루프처리를 실행하는 것이다. 이 루프처리의 제어변수는 재기동 카운터이며, 이 재기동 카운터가 0이 될 것(단계 S41에서 Yes) 및 애플리케이션의 기동에 성공했을 것(단계 S44에서 Yes)이 본 루프처리의 종료 조건이다. 본 루프처리에서는 이들 단계 S41, 단계 S44가 No인 한, 재기동 카운터를 감소하여(단계 S42), AutoRun 애플리케이션을 기동하는 단계 S43을 반복한다. 이러한 반복에 의해, 이상종료에서 애플리케이션의 재기동이 이루어지는 것이다. Restart_Application_flag가 0이면 단계 S32를 실행한다.
- [0355] 단계 S32는 Continuous_Playback_flag가 0인지, 1인지, 2인지를 판정하는 단계이다. 만약 2이면, AutoPlayPL의 재생화상을 풀 스크린 표시로 하고(단계 S33), 단계 S1~단계 S4로 이루어지는 메인 루프로 되돌아간다. 만약 1이면, AutoPlayPL의 재생화상을 쿼터 표시로 한 채(단계 S34), 단계S1~단계 S4로 이루어지는 메인 루프로 되돌아간다.
- [0356] 만약 0이면, 단계 S35로 이행한다. 단계 S35는 에러관리테이블의 selevt_title_flag가 0이 아닌가 여부의 판정이며, 만약 0이면 단계 S37로 이행한다. 0이 아니면, selevt_title_flag의 값 n을 분기처 타이틀로 하고(단계 S36), 도 44의 단계 S7로 이행한다.
- [0357] 단계 S37은 에러관리테이블의 Notify_Event_flag가 0이 아닌가 여부의 판정이며, 만약 0이라면 단계 S39로 이행한다. 0이 아니라면, Notify_Event_flag의 값 n으로부터 특정되는 이벤트 n을 발생하고(단계 S38), 도 44의 단계 S1~단계 S4의 메일 루프로 이행한다. 단계 S39는 에러관리테이블의 Boot_flag가 0이 아닌가 여부의 판정이며, 만약 0이라면 단계 S1~단계 S4로 이루어지는 메인 루프로 이행한다. 0이 아니라면, 도 44의 선두로 이행하여 재생장치의 부트스트랩을 실행한다.
- [0358] 이상과 같이 본 실시 예에 의하면, 애플리케이션이 에러 종료되었을 때, 재생장치는 어떠한 처치를 취해야 하는지를 장치 측의 제조자가 아니라 편집 담당자가 규정해 둘 수 있다. 또한, 에러관리테이블을 갖지 않는 타이틀 재생시에 애플리케이션의 이상종료가 발생할 경우에, 리커버리 처리를 행하는 프로그램을 재생장치에 포함하고 있어도 된다.
- [0359] 또, Restart_Application_Flag~Rboot_Flag 중 어느 하나를 지정하는 인수를 TitleJumpAPI에 설치해 두고, 이 인수에 따른 리커버리 처리를 애플리케이션 매니저(36)에 실행시켜도 된다.
- [0360] (제 3 실시 예)
- [0361] 제 1 실시 예에서 Java(등록상표) 가상머신에서의 PL 재생은 BD-J Object 내의 플레이리스트 관리테이블을 이용하여 규정할 수 있었다. 이에 대해, 본 실시 예에서는 애플리케이션의 JMF 메소드에 의한 PL 재생을 주안으로 하고 있다. 여기서, 문제가 되는 것이 플레이리스트 관리테이블이다. 즉, PL을 재생해도 좋은지의 여부는 BD-J Object별로 플레이리스트 관리테이블에 기술되어 있으므로, 어떤 타이틀에서는 재생이 가능하지만, 다른 타이틀에서는 재생이 불가능해지는 경우가 있다. 또, PL 재생은 가능했다고 해도, 저작권 보호의 관점에서 어떤 종류

의 애플리케이션에서의 재생을 금지하고자 하는 경우가 있다. 이러한 PL 재생의 제한을 실현하기 위해, 제 3 실시 예에서는 퍼미션 컨트롤러(42), 애플리케이션 매니저(36)가 이하의 처리를 행한다.

- [0362] 퍼미션 컨트롤러(42)는, 어느 하나의 애플리케이션이 PL 재생을 요구한 경우, 그 애플리케이션과 상호 인증을 행하고, 요구된 애플리케이션에 PL의 재생권한이 있는가 여부를 판정한다. 만약 있으면, 당해 재생을 플레이 백 컨트롤 엔진(32)에 요구하고, 없으면 불허가를 나타내는 응답 이벤트를 요구된 애플리케이션에 출력한다. 이 퍼미션 컨트롤러(42)에 의한 허가/불허가 판정에 의해, 어떤 배급회사의 배급에 관한 PL을 다른 배급회사의 배급에 관한 애플리케이션이 요구했다 해도, 이러한 요구를 불허가로 할 수 있다. 그 때문에, 정당한 권한이 없는 애플리케이션에 의한 PL의 무단 인용을 피할 수 있다. 허가로 해야 할 PL과 애플리케이션의 조합, 불허가로 해야 할 PL과 애플리케이션의 조합은 별도 BD-ROM에 기록된 퍼미션 필드(permission field)에 규정되어 있으며, 퍼미션 컨트롤러(42)에 의한 판정은 이에 기초한다. 이에 관한 파일의 상세한 내용은 본원의 주안점이 아니므로 설명을 생략한다.
- [0363] 제 3 실시 예에서 애플리케이션 매니저(36)는 현재의 재생시점에서 재생 가능한 PL을 애플리케이션으로부터의 요구에 따라 통지한다. 도 46은 이 애플리케이션 매니저(36)에 의한 통지 처리의 처리순서를 나타내는 플로차트이다. 본 플로차트에서는 애플리케이션의 기동중에, 재생 가능한 PL의 통지를 요구하는 요구(GetPL)를 애플리케이션이 발행했는가 여부의 감시를 행하고 있다(단계 S45). 만약 발행되면, 현재의 재생시점이 속하고 있는 타이틀을 구성하는 BD-J Object에 플레이리스트 관리테이블이 존재하는가 여부를 판정한다(단계 S46). PL의 기술이 있으면, 플레이리스트 관리테이블에 기술된 PL을 재생 가능한 PL로서 요구원의 애플리케이션에 통지한다(단계 S47).
- [0364] PL의 기술이 없으면, PL 재생이 불가능함을 발행된 애플리케이션에 통지한다(단계 S48). 이상이 제 3 실시 예에 관한 애플리케이션 매니저(36)의 처리순서이다. 이어서, PL 재생이 요구된 경우의 애플리케이션 매니저(36)의 처리에 대해 설명한다. 제 3 실시 예에 관한 애플리케이션 매니저(36)는 도 47의 플로차트에 따라 처리를 행한다.
- [0365] 도 47에서 애플리케이션 매니저(36)는 PL 재생을 요구한 애플리케이션이 존재하는지의 여부의 판정을 행하고 있다(단계 S51). 어느 하나의 애플리케이션이 PL 재생을 요구하면, 요구된 애플리케이션에 PL 재생의 권리가 존재하는지의 여부의 인증을 퍼미션 컨트롤러(42)가 행하도록 한다(단계 S52). 만약 재생할 권리가 있으며, 플레이 백 컨트롤 엔진(32)에 재생개시를 지시하여(단계 S53), 플레이 백 컨트롤 엔진(32)으로부터의 성공(success) 응답을 기다린다(단계 S54).
- [0366] 이러한 재생요구가 있으면, 플레이 백 컨트롤 엔진(32)은 플레이리스트 정보의 정당성을 체크한다. 이러한 정당성 체크에는 플레이리스트 정보, Clip 정보, AVClip이 존재하는 BD-ROM 및 Local Storage(18)에서, 정당한 플레이리스트를 구성하고 있는가 라는 체크나 플레이리스트 정보에서의 clip_Information_file_name에 의해 지정되는 Clip 정보 및 AVClip이 BD-ROM 및 Local Storage(18)에 현존하는가 여부의 체크가 있다. clip_Information_file_name에 의해 정확한 파일이 참조되고 있지 않은 경우, 또는, BD-ROM 및 Local Storage(18)로 구성되는 가상적인 패키지에 모순이 있으며, 정확한 플레이리스트를 구성할 수 없는 경우, 플레이 백 컨트롤 엔진(32)은 실패(false)를 나타내는 응답을 되돌려주게 된다. 또, 요구된 애플리케이션보다 높은 기동 우선도를 갖는 애플리케이션이 그 PL을 재생하고 있으며, PL 재생을 실현하는 리소스에서 경합이 발생하고 있는 경우, 플레이 백 컨트롤 엔진(32)은 실패를 나타내는 응답을 되돌려주게 된다.
- [0367] 이상의 과정을 거쳐 성공의 응답이 있으면, PL 재생의 성공을 나타내는 이벤트를 요구된 애플리케이션에 출력한다(단계 S55). 성공의 응답이 없으면, PL 재생의 실패를 나타내는 이벤트를 요구된 애플리케이션에 출력한다(단계 S56). 한편, 단계 S52에서 요구된 애플리케이션을 재생하는 권리가 요구된 애플리케이션에 없으면, PL 재생 불가를 나타내는 이벤트를 요구된 애플리케이션에 출력한다(단계 S57).
- [0368] 이상과 같이 본 실시 예에 의하면, 플레이리스트 재생 여부가 각 타이틀마다 불균일하며, 플레이리스트 재생의 권한을 가진 애플리케이션이나 그렇지 않은 애플리케이션 등 다양한 애플리케이션이 있었다고 해도, 적절한 플레이리스트 재생을 애플리케이션으로부터의 요구에 따라 실행할 수 있다. 그 때문에, 애플리케이션 실행과 플레이리스트 재생을 조합한 다채로운 콘텐츠 실현이 가능해진다.
- [0369] (제 4 실시 예)
- [0370] 제 1 실시 예에서는 타이틀 개시시에, 재생을 개시하고자 하는 플레이리스트에 AutoPlay를 나타내는 재생속성을 부여하여, AutoPlayPL의 재생을 재생장치에 명했다. 이에 대해 본 실시 예에서는, 언바운더리 애플리케이션을

BD-ROM에 기록해 두고, 타이틀 개시시에 자동으로 재생을 개시해야 할 타이틀을 언바운더리 애플리케이션이 선택하도록 하는 개량에 관한 것이다.

- [0371] 언바운더리 애플리케이션은, 플레이 백 컨트롤 엔진(32)과 같은, 재생장치에서의 레지던트 애플리케이션과 대등한 입장에 있는 애플리케이션이며, 플레이리스트 관리테이블에 기술되어 있는 복수 플레이리스트 정보 중에서 재생장치 측의 PSR 설정 값에 합치하는 것을 선택하고, 통지하는 처리를 플레이 백 컨트롤 엔진(32)으로부터의 요구에 따라 실행한다.
- [0372] PL 선택을 언바운더리 애플리케이션이 행하도록 하는 경우, 이와 같은 선택이 필요한 타이틀에 있어서는, 플레이리스트 관리테이블에서의 재생속성을 모두 무 지정으로 설정해 둔다. 이는 "모두가 무 지정"임을 신호로, 플레이 백 컨트롤 엔진(32)이 PL 선택을 타이틀 언바운더리 애플리케이션에 요구하도록 하기 위한 것이다.
- [0373] 이 언바운더리 애플리케이션에 의한 선택은 편집시에 규정된 선택 알고리즘에 기초한다. 도 48(a)~(c)는 언바운더리 애플리케이션에 포함된 선택 알고리즘의 내용을 표 형식으로 나타내는 도면이다. 이 표는 PSR 값이 취할 수 있는 값의 범위와 PSR이 그들 값이 되었을 때, 재생해야 할 PL을 대응시켜서 나타내고 있다. 이 중 도 48(a)은 시청제한레벨(Parental level)에 기초하는 선택 알고리즘의 내용을 나타낸다. 여기서, 시청제한레벨은 재생장치에서 PSR(14)에 제시되어 있다. 구체적으로 말하면, PSR(14)에는 사용자의 연령을 나타내는 정수 값이 설정되어 있으며, 이들을 재생장치는 시청제한레벨로서 해석한다. 도 48(a)에서, PSR(14)이 취할 수 있는 값은 14세 미만, 14세 이상 18세 미만, 18세 이상의 3개의 범위로 나뉘어져 있다. 그리고 이들 범위별로 재생해야 할 PL이 대응되어 있다. 언바운더리 애플리케이션이 이러한 선택 알고리즘에 의한 선택을 행하면, PSR의 설정 값이 14세 미만이면 Playlist #1이, 14세 이상 18세 미만이면 Playlist #2가, 18세 이상이면 Playlist #3이 각각 선택되게 된다.
- [0374] 도 48(b)은 Language for Audio에 기초하는 선택 알고리즘의 내용을 나타낸다. 여기서, Language for Audio는 재생장치에서 PSR(16)에 나타나 있다. 구체적으로 말하면, PSR(16)에는 정수 값이 설정되어 있으며, 이를 재생장치는 음성 재생용 언어 설정으로서 해석한다. 도 48(b)에서, PSR(16)이 취할 수 있는 값은 영어를 나타내는 값, 일본어를 나타내는 값, 그 외의 값의 3개의 범위로 나뉘어져 있다. 그리고 이들 범위마다 재생해야 할 PL이 대응되어 있다. 언바운더리 애플리케이션이 이러한 선택 알고리즘에 의한 선택을 행하면, PSR(16)의 설정 값이 영어를 나타낸다면 Playlist #1, 일본어를 나타낸다면 Playlist #2가, 영어, 일본어 이외의 값이면 Playlist #3이 각각 선택되게 된다.
- [0375] 도 48(c)은 Player Configuration for Video에 기초하는 선택 알고리즘의 내용을 나타낸다. 여기서, Player Configuration for Video는 재생장치에서 PSR(14)에 나타나 있다. 구체적으로 말하면, PSR(14)에는 정수 값이 설정되어 있으며, 이들 재생장치는 영상재생용 환경설정으로서 해석한다. 도 48(c)에서, PSR(14)이 취할 수 있는 값은 해상도 525×600 TVsystem LetterBox, 해상도 525×600 TVsystem, 1920×1080 TVsystem의 3개의 범위로 나뉘어져 있다. 그리고 이들 범위별로 재생해야 할 PL이 대응되어 있다. 언바운더리 애플리케이션이 이러한 선택 알고리즘에 따라 선택을 행하면, PSR(14)의 설정 값이 해상도 525×600 TVsystem LetterBox를 나타내는 것이라면 Playlist #1이, 해상도 525×600을 나타내는 것이라면 Playlist #2가, TVsystem, 1920×1080 TVsystem을 나타내는 것이라면 Playlist #3이 각각 선택되게 된다. 도 48(a)~(c)에 나타낸 것과 같은 선택 알고리즘은, 이들 도면에 나타나는 조건 분기를 컴퓨터 기술언어로 기술함으로써 작성할 수 있다.
- [0376] 이상이, 본 실시 예에 관한 기록매체의 개량이다. 이어, 본 실시 예에 관한 재생장치의 개량에 대해 설명한다. 본 실시 예에서의 개량 점은 주로, 애플리케이션 매니저(36), 플레이 백 컨트롤 엔진(32)에 있다.
- [0377] 애플리케이션 매니저(36)는 타이틀의 분기가 발생되고, 플레이리스트 관리테이블을 참조했을 때, 그 플레이리스트 관리테이블에 AutoPlayPL이 존재하는지의 여부를 판정한다. AutoPlayPL이 없으면, 플레이리스트 관리테이블을 플레이 백 컨트롤 엔진(32)에 인계하여, 이 플레이리스트 관리테이블에 기재되어 있는 PL 중, 어느 하나를 자동으로 재생하도록 플레이 백 컨트롤 엔진(32)에 요구한다.
- [0378] 플레이 백 컨트롤 엔진(32)은 플레이리스트 관리테이블의 인계를 받으면, 언바운더리 애플리케이션에 대해 PL 선택을 행하도록 요구한다. 이 요구에 따라 언바운더리 애플리케이션으로부터 재생 가능한 PL의 리스트가 통지되면, 그 PL 리스트에 기재된 PL 중, PlayItem으로부터 인계된 플레이리스트 관리테이블에 존재하는 것을 판정한다. 언바운더리 애플리케이션에 의해 선택된 PL 중에 플레이리스트 관리테이블에 기재된 것이 있으면 그것의 재생을 자동으로 개시한다.
- [0379] 도 49는 타이틀 언바운더리 애플리케이션에 PL 선택을 행하게 하는 과정을 모식적으로 묘사한 도면이다. 본 도

면의 좌측은 재생장치에서의 소프트웨어의 레이어 구성을 나타내고, 본 도면의 우측에 BD-ROM의 기록내용을 나타낸다. 도면 중의 ㉠, 2, 3, 4는 AutoPlay가 없는 플레이리스트 관리테이블을 발견한 경우의 애플리케이션 매니저(36)로부터의 통지(㉠), 플레이 백 컨트롤 엔진(32)에 의한 재생 가능 PL의 문의(㉡), 타이틀 언바운더리 애플리케이션에 의한 PSR 설정 값 취득(㉢), 타이틀 언바운더리 애플리케이션으로부터 플레이 백 컨트롤 엔진(32)으로의 재생 가능한 PL의 통지(㉣)를 모식적으로 묘사하고 있다.

[0380] 또한, 도 49에서, 타이틀 언바운더리 애플리케이션은 BD-ROM 상에 기술했지만, 이는 타이틀 언바운더리 애플리케이션을 알기 쉽게 기술하기 위한 배려에 지나지 않는다. 타이틀 언바운더리 애플리케이션은 Java(등록상표) 애플리케이션이므로, Java(등록상표) 가상머신(38) 내의 워크 메모리(54)에서 인스턴스로서 스레드(55)에 의해 실행된다는 것이 현실적인 기술이 된다.

[0381] 이상과 같이 본 실시 예에 의하면, 타이틀의 바운더리에서 생존하고 있는 애플리케이션에 상술한 판정을 행하게 하므로, 재생장치 측의 플레이 백 컨트롤 엔진(32)은 BD-ROM에서의 복수 PL 중 재생장치 측의 상태설정예 따른 것은 어느 것인지를 타이틀 개시시의 빠른 단계에서 알 수 있다. 재생속성 = AutoPlay의 애플리케이션을 결정해 두지 않아도, 타이틀 개시시에 재생을 개시해야 할 PL을 결정할 수 있으므로, 언어 신뢰(language credit)나 시청제한 잠금(lock)의 재생제어를 BD-J 모드에서도 실현할 수 있다.

[0382] 또한, 본 실시 예에서의 선택 알고리즘은 PSR이 취할 수 있는 값을 플레이리스트에 대응시켰으나, 재생장치에서의 PSR의 설정 값이 상정하지 않은 값인 경우에, 재생장치에 재생시키는 플레이리스트를 미리 규정해 두어도 된다.

[0383] (제 5 실시 예)

[0384] 제 4 실시 예에서 타이틀 언바운더리 애플리케이션은 PSR의 설정 값에 따라 재생해야 할 PL을 선택하는 선택 알고리즘을 갖고 있으나, 본 실시 예에 관한 타이틀 언바운더리 애플리케이션은 하나의 PL 내에 멀티앵글 구간이 있는 경우, 당해 멀티앵글 구간에서의 복수 앵글 중 어느 하나를 선택하는 처리를 타이틀 언바운더리 애플리케이션이 행하도록 하는 개량에 관한 것이다. 본 실시 예에 관한 타이틀 언바운더리 애플리케이션은 PSR이 취할 수 있는 값의 복수의 범위와, 그들 범위별로 재생해야 할 앵글을 대응시킨 것이다. 본 실시 예에서 현재의 재생 시점이 멀티앵글 구간이면 플레이 백 컨트롤 엔진(32)은 타이틀 언바운더리 애플리케이션에 대해 어떤 앵글을 재생할지의 문의를 행한다. 이러한 문의가 있으면, 타이틀 언바운더리 애플리케이션은 현재의 PSR 설정 값을 취득하고, 선택 알고리즘을 실행하여 취득한 설정 값에 따른 앵글을 선택한다. 그 선택 결과를 플레이 백 컨트롤 엔진(32)에 통지하여, 그 앵글의 재생을 플레이 백 컨트롤 엔진(32)이 행하도록 한다.

[0385] 이상과 같이 본 실시 예에 의하면, PSR이 어떤 값일 때 어느 앵글을 선택한다고 하는 알고리즘을 편집 담당자가 규정해 둘 수 있으므로, 앵글을 응용한 여러 애플리케이션을 편집 담당자가 생성할 수 있다.

[0386] (제 6 실시 예)

[0387] BD-J 모드에서, PL 재생과의 동기를 어떻게 실현하는가에 대한 개량에 관한 것이다. 플레이 백 컨트롤 엔진(32)은 PlayPLAPI 기능이 콜 되면, PL 정보에 기초하는 처리순서를 실행한다. PL이 2시간이라고 하는 재생시간을 갖는다면, 이 2시간 동안 상술한 처리는 계속하게 된다. 여기서 문제가 되는 것은 Java(등록상표) 가상머신(38)이 성공의 응답을 되돌려 주는 시간과, 플레이 백 컨트롤 엔진(32)이 실제로 처리를 끝내는 시간과의 겹이다. Java(등록상표) 가상머신(38)은 이벤트 드리븐(event driven)의 처리 주체이므로 콜 직후에 재생 성공인지, 재생 실패인지를 나타내는 응답을 되돌려 주나, 플레이 백 컨트롤 엔진(32)에 의한 실제의 처리 종료는 2시간 경과 후이므로, 성공의 응답을 애플리케이션에 되돌려 주는 시간을 기준으로 하는 경우에는 2시간 경과 후에 해당 하는 처리결과를 감지할 수 없다. PL 재생에서 빨리 감기, 되감기, 스킵(Skip)이 행해지면, 이 2시간이라는 재생기간은 2시간 전후로 변동하게 되어, 처리 종결의 감지는 더 곤란해진다.

[0388] 플레이 백 컨트롤 엔진(32)은 애플리케이션과 스탠드 어론(stand alone)으로 동작하므로, 애플리케이션 매니저(36)는 PL 재생의 종료 시점을 정확하게 해석할 수 없다. 그래서 본 실시 예에서는 애플리케이션이 종료되든 말든 워크 메모리에 JMF 플레이어 인스턴스가 있는 한, 즉, 프레젠테이션 엔진(31)의 제어권을 BD-J 모듈(35)이 장악하고 있는 동안, 플레이 백 컨트롤 엔진(32)으로부터 노티파이 이벤트를 갖는다. 그리고 노티파이 이벤트가 있으면, 타이틀이 종료되었다고 해석하여 다음의 타이틀로의 분기를 행하도록 모듈매니저(34)에 통지한다. 이렇게 함으로써, 플레이 백 컨트롤 엔진(32)이 PL 재생을 종결한 시점을 타이틀의 종단으로 할 수 있다.

[0389] 이후, 도 50~도 54의 플로차트를 참조하여 플레이 백 컨트롤 엔진(32)에 의한 구체적인 제어순서를 설명한다.

- [0390] 도 50은 플레이 백 컨트롤 엔진(32)에 의한 PL 재생순서를 나타내는 플로차트이다. 이 재생순서는 프레젠테이션 엔진(31)에 대한 제어(단계 S106)와, BD-ROM 드라이브(1) 또는 Local Storage(18)에 대한 제어(단계 S108)를 주로 포함한다. 본 플로차트에서 처리 대상인 PlayItem을 PlayItem #x로 한다. 본 플로차트는 현재 PL 정보(.mpls)의 판독을 행하고(단계 S101), 그 후 단계 S102~단계 S110의 처리를 실행하는 것이다. 여기서, 단계 S102~단계 S110은 단계 S109가 Yes가 될 때까지, 현재 PL 정보를 구성하는 각각의 PI 정보에 대해 단계 S103~단계 S110의 처리를 반복하는 루프처리를 구성하고 있다. 이 루프처리에서 처리 대상이 되는 PlayItem을 PlayItem #x(PI #x)이라 부른다. 이 PlayItem #x는 현재 PL의 선두의 PlayItem으로 설정됨으로써, 초기화된(단계 S102). 상술한 루프처리의 종료 요건은, 이 PlayItem #x가 현재 PL의 최후 PlayItem이 되는 것이며(단계 S109), 만약 최후의 PlayItem이 아니면, 현재 PL에서의 다음 PlayItem이 PlayItem #x로 설정된다(단계 S110).
- [0391] 루프처리에서 반복 실행되는 단계 S103~단계 S110은 PlayItem #x의 Clip_information_file_name에서 지정되는 Clip 정보를 시나리오 메모리(25)에 판독하고(단계 S103), PlayItem #x의 In_time을 현재 Clip 정보의 EPmap을 이용하여, I 픽처 어드레스 u로 변환하며(단계 S104), PlayItem #x의 Out_time을 현재 Clip 정보의 EP_map을 이용하여 I 픽처 어드레스 v로 변환하여(단계 S105), 이들 변환으로 얻어진 주소 v의 다음 I 픽처를 구하여 그 어드레스의 하나 앞을 어드레스 w로 설정하고(단계 S107), 그렇게 하여 산출된 어드레스 w를 이용하여, I 픽처 어드레스 u로부터 어드레스 w까지의 TS 패킷의 판독을 BD-ROM 드라이브(1) 또는 Local Storage(18)에 명하는 것이다(단계 S108).
- [0392] 한편, 프레젠테이션 엔진(31)에 대해서는 현재 PLMak의 mark_time_stamp로부터 PlayItem #x의 Out_time까지의 출력을 명한다(단계 S106). 이상의 단계 S105~단계 S108에 의해 AVClip에서 PlayItem #x에 의해 지정되어 있는 부분의 재생이 이루어지게 된다. 그 후, PlayItem #x가 현재 PL의 최후 PI인지의 판정이 이루어진다(단계 S109).
- [0393] PlayItem #x가 현재 PL의 최후의 PI가 아니면, 현재 PL에서의 다음 PlayItem을 PlayItem #x로 설정하고(단계 S110), 단계 S103으로 되돌아간다. 이상의 단계 S103~단계 S110을 반복함으로써, PL을 구성하는 PI는 순차 재생되게 된다.
- [0394] 도 51은 앵글절환 순서 및 SkipBack, SkipNext의 순서를 나타내는 플로차트이다. 본 플로차트는 도 50의 처리순서와 병행하여 이루어지는 것이며, 단계 S111~S112로 이루어지는 루프처리를 반복하는 것이다. 본 루프에서의 단계 S111은 앵글절환을 요구하는 API가 Java(등록상표) 가상머신(38)으로부터 콜 되었는지의 여부의 판정이며, 앵글절환 API의 콜이 있으면, 현재 Clip 정보를 절환하는 조작을 실행한다.
- [0395] 도 51의 단계 S115는 판정 단계이며, PlayItem #x의 is_multi_angles가 온(ON)인지의 여부의 판정을 행한다. is_multi_angles란, PlayItem #x가 멀티앵글에 대응하고 있는지의 여부를 나타내는 그래프이며, 만약 단계 S115가 No이면 단계 S113으로 이행한다. 단계 S115가 Yes이면 단계 S116~단계 S119를 실행한다. 단계 S116~단계 S119는 절환 후의 앵글 번호를 변수 y에 대입하여(단계 S116), PlayItem #x에서의 y번째의 Clip_information_file_name으로 지정되어 있는 Clip 정보를 시나리오 메모리(21)에 판독하고(단계 S117), 현재 PTM을 현재 Clip 정보의 EP_map을 이용하여 I 픽처 어드레스 u로 변환하고(단계 S118), PlayItem #x의 Out_time을 현재 Clip 정보의 EP_map을 이용하여 I 픽처 어드레스 v로 변환한다(단계 S119)는 것이다. 이렇게 하여 I 픽처 어드레스 u, v를 변환한 후, 동시 실행되고 있는 도 50 측의 처리를 정지시킨 후 단계 S106으로 이행한다. 단계 S106으로의 이행에 의해, 다른 AVClip으로부터 TS 패킷이 판독되므로, 영상 내용이 절환되게 된다.
- [0396] 한편, 도 51의 루프에서의 단계 S112는, SkipBack/SkipNext를 의미하는 API가 Java(등록상표) 가상머신(38)으로부터 콜 되었는지의 여부의 판정이며, 만약 콜 되었으면, 도 52의 플로차트의 처리순서를 실행한다. 도 52는 SkipBack, SkipNextAPI가 콜되었을 때의 처리순서를 나타내는 플로차트이다. SkipBack, SkipNext를 실행할 때의 처리순서는 다종다양한 것이다. 여기서, 설명하는 것은 어디까지나 일 예에 지나지 않는 것에 유의하길 바란다.
- [0397] 단계 S121은 PSR로 나타나는 현재 PI 번호 및 현재 PTM을 변환함으로써, 현재 마크정보(Mark Information)를 얻는다. 단계 S122는 눌러진 것이 SkipNext 키인가, SkipBack 키인가의 판정이며, SkipNext 키라면 단계 S123에서 방향 플래그를 +1로 설정하고, SkipBack 키라면 단계 S124에서 방향 플래그를 -1로 설정한다.
- [0398] 단계 S125는 현재 PLMark의 번호에 방향 플래그 값을 더한 번호를 현재 PLMark의 번호로 설정한다. 여기서, SkipNext 키이면 방향 플래그는 +1로 설정되어 있으므로, 현재 PLMark는 증가하게(increment) 된다. SkipBack

키이면 방향 플래그는 -1로 설정되어 있으므로, 현재 PLMark는 감소하게 된다.

- [0399] 단계 S126에서는 현재 PLMark의 ref_to_PlayItem_Id에 기술되어 있는 PI를 PlayItem #x로 설정하고, 단계 S127에서는 PlayItem #x의 Clip_information_file_name에서 지정되는 Clip 정보를 판독한다. 단계 S128에서는 현재 Clip 정보의 EP_map을 이용하여, 현재 PLMark의 mark_time_stamp를 I 픽처 어드레스 u로 변환한다. 한편, 단계 S129에서는 PlayItem #x의 Out_time을 I 픽처 어드레스 u로 변환한다. 한편, 단계 S129에서는 PlayItem #x의 Out_time을 현재 Clip 정보의 EP_map를 이용하여 I 픽처 어드레스 v로 변환한다. 단계 S130은 현재 PLMark의 mark_time_stamp로부터 PlayItem #x의 Out_time까지의 출력을 프레젠테이션 엔진(31)에 명하여, 동시 실행하고 있는 도 50 측의 처리를 정지시킨 후, 도 50의 단계 S107로 이행한다. 이렇게 하여 I 픽처 어드레스 u, v를 변화시켜서 다른 부분의 재생을 명한 다음 단계 S107로 이행하므로, 다른 AVClip으로부터 TS 패킷이 판독되게 되어 영상 내용의 전환이 실현된다.
- [0400] 도 53은 프레젠테이션 엔진(31)에 의한 처리순서의 상세한 설명을 나타내는 플로차트이다. 본 플로차트는 I 픽처의 PTS를 현재 PTM으로 설정한 후(단계 S131), 단계 S132~단계 S137로 이루어지는 루프처리를 실행하는 것이다.
- [0401] 이어, 단계 S132~단계 S137에서의 루프처리에 대해 설명한다. 이 루프처리는 현재 PTM에 해당하는 픽처, 오디오의 재생 출력과, 현재 PTM의 갱신을 반복하는 것이다. 본 루프처리에서의 단계 S136은 루프처리의 종료 요건을 규정하고 있다. 즉, 단계 S136은 현재 PTM이 PI #x의 Out_time인 것을 루프처리의 종료 요건으로 하고 있다.
- [0402] 단계 S133은 빨리 감기 API, 또는 빨리 감기 API가 Java(등록상표) 가상머신(38)으로부터 콜 되었는가 여부의 판정이다. 만약 콜 되면, 단계 S138에서 빨리 감기인가 빨리 되감기인가의 판정을 행하고, 빨리 감기이면 다음 I 픽처의 PTS를 현재 PTM으로 설정한다(단계 S139). 이렇게 현재 PTM을 다음 I 픽처의 PTS로 설정함으로써 1초 걸러 AVClip를 재생해 갈 수 있다. 이로써, 2배속 등으로 AVClip은 순방향으로 빨리 재생되게 된다. 빨리 되감기라면, 현재 PTM이 PlayItem #x의 Out_time에 도달했는지를 판정한다(단계 S140). 만약 도달하고 있지 않다면 하나 앞의 I 픽처의 PTS를 현재 PTM으로 설정한다(단계 S141). 이렇게 판독처 어드레스 A를 하나 앞의 I 픽처로 설정함으로써 AVClip을 후 방향으로(backward) 1초 걸러 재생해 갈 수 있다. 이로써, 2배속 등으로 AVClip은 역방향으로 재생되게 된다. 또한, 빨리 감기, 되감기를 실행할 때의 처리순서는 다종다양한 것이다. 여기서 설명하는 것은 어디까지나 일례에 지나지 않음을 유의하기 바란다.
- [0403] 단계 S134는 메뉴 콜 API가 콜되었는지의 여부를 판정이며, 만약 콜되면, 현재의 재생처리를 서스펜드(suspend, 단계 S142)하여, 메뉴 처리용 메뉴 프로그램을 실행한다(단계 S143). 이상의 처리로, 메뉴 콜이 이루어진 경우는, 재생 처리를 중단한 다음, 메뉴 표시를 위한 처리가 실행되게 된다.
- [0404] 단계 S135는, sync_PlayItem_id에 의해, PlayItem #x를 지정한 subPlayItem #y가 존재하는가 여부의 판정이며, 만약 존재하면, 도 54의 플로차트로 이행한다. 도 54는, SubPlayItem의 재생순서를 나타내는 플로차트이다. 본 플로차트에서는, 먼저, 단계 S146에서 현재 PTM은 SubPlayItem #y의 sync_start_PTS_of_playItem인지의 여부를 판정한다. 만약 그렇다면, 단계 S153에서 SubPlayItem #y에 기초한 재생처리를 행하도록 플레이 백 컨트롤 엔진(32)에 통지한다.
- [0405] 단계 S136이 Yes로 판정된 경우, 단계 S144, 단계 S145를 실행한다. 단계 S144는 가상파일시스템(30)에서 Notify End Of File 이벤트가 출력되고, 또한, 디코더로부터 Notify End Of Decoding 이벤트가 출력되었는지의 여부를 판정한다. 만약 출력되면, Notify End Of Stream 이벤트를 플레이 백 컨트롤 엔진(32)에 출력한다.
- [0406] 도 54의 단계 S147~단계 S152는 SubPlayItem #y에 기초한 재생처리를 나타내는 플로차트이다. 단계 S147에서는, SubPlayItem #y의 Clip_information_file_name에서 지정되는 Clip정보를 판독한다. 단계 S148에서는 현재 Clip 정보의 EP_map을 이용하여, SubPlayItem #y의 In_time을 어드레스로 α 로 변환한다. 한편, 단계 S149에서는, SubPlayItem #y의 Out_time을 현재 Clip정보의 EP_map을 이용하여 어드레스 β 로 변환한다. 단계 S150은 SubPlayItem #y의 In_time에서 SubPlayItem #y의 Out_time까지의 출력을 디코더에 명한다. 이들의 변환으로 얻어진 어드레스 β 의 다음 I 픽처를 구하고, 그 어드레스의 하나 앞을 어드레스 γ 로 설정하며(단계 S151), 그렇게 산출된 어드레스 γ 를 이용하여, SubClip #z에서의 어드레스 α 에서 어드레스 γ 까지의 TS 패킷의 판독을 BD-ROM드라이브(1) 또는 Local Storage(18)에 명하는 것이다(단계 S152).
- [0407] 또한, 도 50으로 되돌아가서, 플레이 백 컨트롤 엔진(32)의 처리의 설명을 행한다. 단계 S133은 프레젠테이션 엔진(31)에 의한 재생제어가 완료되었는지의 여부를 판정이며, 최후의 PlayItem #x에 대하여 도 53의 플로차트

의 처리가 행해지고 있는 한, 단계 S113이 No가 된다. 도 53의 플로차트의 처리가 종료되면 비로소 단계 S113은 Yes가 되어 단계 S114로 이행한다. 단계 S114는, Java(등록상표) 가상머신(38)으로의 노티파이 이벤트의 출력이며, 이 출력에 의해 2시간이라는 재생시간의 경과를 Java(등록상표) 가상머신(38)은 알 수 있다.

[0408] 이상과 같이 본 실시 예에 의하면, 2시간이라는 재생시간의 경과시점을 애플리케이션 매니저(36)가 파악할 수 있으므로, 플레이리스트의 재생종료와 동기한 처리를 Java(등록상표) 가상머신(38)에 실행시킬 수 있다.

[0409] (비고)

[0410] 이상의 설명은, 본 발명의 모든 실시 예를 나타내고 있는 것은 아니다, 하기 (A) (B) (C) (D) 의 변경을 실시한 실시행위의 예에 의해서도, 본 발명의 실시는 가능해진다. 본원의 청구항에 기재된 각 발명은 이상에 기재한 복수의 실시 예 및 그들의 변형 예를 확장한 기재 내지 일반화한 기재이다. 확장 내지 일반화의 정도는 본 발명의 기술분야의 출원 당시의 기술수준의 특성에 의거한다.

[0411] (A) 모든 실시 예에서는, 본 발명에 관한 광디스크를 BD-ROM으로 실시하였지만, 본 발명의 광디스크는 기록되는 동적 시나리오, 인덱스 테이블(index table)에 특징이 있고, 이 특징은 BD-ROM의 물리적 성질에 의존하는 것은 아니다. 동적 시나리오, 인덱스 테이블을 기록할 수 있는 기록매체라면, 어떠한 기록매체라도 된다. 예를 들어, DVD-ROM, DVD-RAM, DVD-RW, DVD-R, DVD+RW, DVD+R, CD-R, CD-RW 등의 광디스크, PD, MO 등의 광자기 디스크라도 된다. 또한, 콤팩트 플래시(등록상표) 카드, 스마트미디어, 메모리스틱, 멀티미디어카드, PCM-CIA 카드 등의 반도체 메모리 카드라도 된다. 플렉시블 디스크(flexible disk), SuperDisk, Zip, Cluk! 등의 자기기록 디스크(i), ORB, Jaz, SparQ, SyJet, EZFley, 마이크로 드라이브 등의 착탈식(removable) 하드디스크 드라이브(ii)라도 된다. 또한, 기기 내장형 하드디스크라도 된다.

[0412] (B) 모든 실시 예에서의 재생장치는 BD-ROM에 기록된 AVClip을 디코딩한 후에 TV에 출력하고 있으나, 재생장치를 BD-ROM 드라이브만으로 하고, 이 이외의 구성요소를 TV에 구비시켜도 된다. 이 경우, 재생장치와 TV를 IEEE1394로 접속된 홈 네트워크에 포함시킬 수 있다. 또, 실시 예에서의 재생장치는 TV와 접속하여 이용되는 타입이었으나, 디스플레이와 일체형으로 된 재생장치여도 된다. 또한, 각 실시 예의 재생장치에서, 처리의 본질적 부분을 이루는 부분만을 재생장치로 해도 된다. 이들 재생장치는 모두 본원 명세서에 기재된 발명이므로, 이들의 어떠한 형태라도 제 1 실시 예~제 3 실시 예에 나타난 재생장치의 내부 구성을 토대로 재생장치를 제조하는 행위는 본원의 명세서에 기재된 발명의 실시 예가 된다. 제 1 실시 예~제 3 실시 예에 나타난 재생장치의 유상?무상에 의한 양도(유상의 경우는 판매, 무상의 경우는 증여가 된다), 대여, 수입하는 행위도 본 발명의 실시행위이다. 점포 전시, 카탈로그 권유, 팸플릿 배포에 의해 이들의 양도나 대여를 일반 사용자에게 제의하는 행위도 본 재생장치의 실시행위이다.

[0413] (C) 각 플로차트에 나타난 프로그램에 의한 정보 처리는 하드웨어 자원을 이용하여 구체적으로 실현되어 있기 때문에, 상기 플로차트에 처리 순서를 나타낸 프로그램은 단일체로 발명으로 성립한다. 모든 실시 예는 재생장치에 포함된 형태로, 본 발명에 관한 프로그램의 실시행위에 대한 실시 예를 제시하였으나, 재생장치로부터 분리하여, 제 1 실시 예~제 3 실시 예에 제시한 프로그램 단일체를 실시해도 된다. 프로그램 단일체의 실시행위에는 이들의 프로그램을 생산하는 행위(1)나 유상?무상에 의해 프로그램을 양도하는 행위(2), 대여하는 행위(3), 수입하는 행위(4), 쌍방향의 전자통신 회선을 통해 공중(公衆)에 제공하는 행위(5), 점포 전시, 카탈로그 권유, 팸플릿 배포에 의해 프로그램의 양도나 대여를 일반 사용자에게 제의하는 행위(6)가 있다.

[0414] (D) 각 플로차트에서 시계열로 실행되는 각 단계의 「시(時)」의 요소를 발명을 특정하기 위한 필수 사항이라고 생각한다. 그렇게 하면, 이들 플로차트에 의한 처리 순서는 재생방법의 사용 예를 개시하고 있음을 알 수 있다. 각 단계의 처리를 시계열로 행함으로써 본 발명의 본래의 목적을 달성하고, 작용 및 효과가 있도록 이들의 플로차트의 처리를 행하는 것이라면, 본 발명에 관한 기록방법의 실시행위에 해당하는 것은 말할 필요도 없다.

[0415] (E) BD-ROM에 기록할 때, AVClip을 구성하는 각 TS 패킷에는 확장 헤더를 부여해 두는 것이 바람직하다. 확장 헤더는 TP_extra_header라 불리며, 『Arribval_Time_Stamp』와 『copy_permission_indicator』를 포함하는 4 바이트의 데이터 길이를 갖는다. TP_extra_header가 붙은 TS 패킷(이하, EX 부착 TS 패킷이라 약칭한다)은 32개 마다 그룹화되어 3개의 섹터에 기록된다. 32개의 EX 부착 TS 패킷으로 이루어지는 그룹은 6144 바이트(=32×192)이며, 이는 3개의 섹터 사이즈 6144 바이트(=2048×3)와 일치한다. 3개의 섹터에 포함된 32개의 EX 부착 TS 패킷을 "Aligned Unit"이라 한다.

[0416] IEEE1394를 통해 접속된 홈 네트워크에서의 이용시, 재생장치(200)는 이하와 같은 송신처리로 Aligned Unit의 송신을 행한다. 즉, 송신자 측의 기기는 Aligned Unit에 포함되는 32개의 EX 부착 TS 패킷의 각각으로부터

TP_extra_header을 제거하고, TS 패킷 본체를 DTCP 규격에 기초하여 암호화하여 출력한다. TS 패킷 출력시에는, TS 패킷 사이의 여러 개소에 isochronous 패킷을 삽입한다. 이 삽입 개소는 TX_extra_header의 Arrival_Time_Stamp에 나타나는 시각에 기초한 위치이다. TS 패킷의 출력에 따라 재생장치(200)는 DTCP_Descriptor를 출력한다. DTCP_Descriptor는 TP_extra_header에서의 복제(copy)허가 여부의 설정을 나타낸다. 여기서 「복제 금지」를 나타내도록 DTCP_Descriptor를 기술해 두면, IEEE1394를 통해 접속된 홈 네트워크에서의 이용시에 TS 패킷은 다른 기기에 기록되지 않는다.

- [0417] (F) 각 실시 예에서의 디지털 스트림은 BD-ROM 규격의 AVClip이었으나, DVD-Video 규격, DVD-Video Recording 규격의 VOB(Video Object)라도 된다. VOB는 비디오 스트림, 오디오 스트림을 다중화함으로써 얻어진 ISO/IEC13818-1 규격 준거의 프로그램 스트림이다. 또한, AVClip에서의 비디오 스트림은 MPEG4나 WMV 방식이어도 된다. 또한, 오디오 스트림은 Linear-PCM 방식, Dolby-AC 3 방식, MP3 방식, MPEG-AAC 방식이라도 된다,
- [0418] (G) 각 실시 예에서의 영화 작품은 아날로그 방송으로 방송된 아날로그 영상신호를 인코딩함으로써 얻어진 것이어도 된다. 디지털 방송에서 방송된 트랜스포트 스트림으로 구성되는 스트림 데이터라도 된다.
- [0419] 또한, 비디오 테이프에 기록되어 있는 아날로그/디지털의 영상신호를 인코딩하여 콘텐츠를 얻어도 된다. 또한, 비디오 카메라로부터 직접 취득한 아날로그/디지털의 영상신호를 인코딩하여 콘텐츠를 얻어도 된다. 이 외에도, 배신(配信; distribution) 서버에 의해 배신되는 디지털 저작물이라도 된다.
- [0420] (H) BD-J 모듈(35)은 위성방송 수신을 위해 기기에 포함된 Java(등록상표) 플랫폼이어도 된다. BD-J모듈(35)이 이러한 Java(등록상표) 플랫폼이라면, 본 발명에 관한 재생장치는 MHP용 STB로서의 처리를 겸용하게 된다. 또한, 휴대전화 등의 처리 제어를 위해서 기기에 포함된 Java(등록상표) 플랫폼이라도 된다. 이러한 BD-J 모듈(35)이 이러한 Java(등록상표) 플랫폼이면, 본 발명에 관한 재생장치는 휴대전화로서의 처리를 겸용하게 된다.
- [0421] (I) 레이어 모델에서, BD-J 모드 상에 HDM 모드를 배치하여도 된다. 특히 HDMV모드에서의 동적 시나리오의 해석이나, 동적 시나리오에 기초하는 제어순서의 실행은 재생장치에 대한 부담이 가벼우므로, HDMV 모드를 BD-J모드 상에서 실행시켜도 아무런 문제도 발생하지 않기 때문이다. 또한, 재생장치나 영화작품의 개발에 있어서 동작 보증이 하나의 모드가 되기 때문이다.
- [0422] 또한 BD-J 모드만으로 재생처리를 실행하여도 된다. 제 5 실시 예에 나타난 바와 같이, BD-J 모드라도 PL의 재생과 동기한 재생제어가 가능하게 되므로, 굳이 HDMV 모드를 설치하지 않아도 되기 때문이다.
- [0423] (J) AVClip에 다중화되어야 할 인터랙티브 그래픽 스트림에 내비게이션 코멘트를 설치하고, 어떤 PL에서 다른 PL로의 분기를 실현하여도 된다.
- [0424] (K) 제 1 실시 예에서는, 하나의 BD-ROM에 속하는 복수 타이틀의 전부를 생존구간으로 하는 타이틀로, 타이틀 언바운더리 애플리케이션을 규정하였다. 이 외에도, 복수의 BD-ROM에 속하는 타이틀의 전부를 생존구간으로 하는 타이틀 언바운더리 애플리케이션을 규정하여도 된다.
- [0425] (L) 제 1 실시 예에서 애플리케이션 관리테이블을 작성시, 동시 실행할 수 있는 애플리케이션 수는, 예를 들어, 4개 이하로 제한하는 것이 바람직하다.
- [0426] 애플리케이션의 동기 실행 수를 4개 이하로 제한하는 이유는 다음과 같다.
- [0427] BD-ROM의 재생장치는 디지털 방송의 튜너기능을 구비하고 있는 것이 많고, 그 튜너 기능을 실현하는 애플리케이션이 메모리에 상주(레지던트)하고 있는 것도 많다. 이러한 레지던트 애플리케이션이 동작하는 여지를 만들기 위하여, 애플리케이션의 수는 4개 이하로 제한되어 있다. 4개의 애플리케이션의 내역으로, 첫 번째를 타이틀 언바운더리 애플리케이션, 두 번째를 타이틀 바운더리 애플리케이션, 세 번째를 챕터 바운더리 애플리케이션으로 해 두는 것이 바람직하다.
- [0428] (M) 제 2 실시 예에 있어서 에러관리테이블은, 하나의 애플리케이션이 이상종료했을 때, 하나의 리커버리 처리를 실행하도록 규정되어 있다. 그러나 하나의 애플리케이션이 이상종료했을 때, 복수의 리커버리 처리를 실행하도록 해도 된다. 즉, 어떤 애플리케이션이 이상종료했을 때, 플레이리스트 재생의 계속과, 애플리케이션의 재기동과, 이벤트 출력을 재생장치에 실행시켜도 된다.
- [0429] 또한, 애플리케이션 별이 아니라, 하나의 타이틀에 하나의 리커버리 처리를 규정하도록 에러관리테이블을 구성하여도 된다.
- [0430] (N) AVClip에는 메뉴를 표시하여 대화적인 조작을 접수하기 위한 인터랙티브 그래픽스 스트림을 다중화해 둘 수

있으므로, 탑 메뉴를 표시하여 대화적인 조작을 접수하는 AVClip을 단순히 재생시키는 내비게이션 코멘드를 Movie Object에 기술해둠으로써, 탑 메뉴 타이틀을 제작하여도 된다.

- [0431] (0) 각 실시 예에서의 디렉터리?파일구성 및 파일 내의 데이터 구조는 일례이며, 본 발명의 특징인 관리정보는 디렉터리?파일구성 및 파일 내의 데이터 구조에 의존하지 않는다. 예를 들어, BD-J모드의 동적 시나리오인 BD-J Object는, 식별자 bobj_id와, 식별자 BD-J가 부여된 파일(ZZZZZ. BD-J)로서 BDJA 디렉터리에 배치하고, BD-J Object. bdmv의 BD-J Object[n]()에는 식별자 bobj_id만을 저장하여도 된다.

산업상 이용 가능성

- [0432] 본 발명에 관한 기록매체 및 재생장치는 홈시어터 시스템에서의 이용과 같이 개인적인 용도로 이용될 수 있다. 그러나 본 발명은 상기 실시 예에 내부 구성이 개시되어 있으며, 이 내부 구성에 의거하여 양산하는 것이 명백하므로, 그 자체로서 공업상 이용할 수 있다. 이 점에서 본 발명에 관한 기록매체 및 재생장치는 산업상 이용 가능성을 갖는다.

도면의 간단한 설명

- [0009] 도 1은 본 발명에 관한 재생장치의 사용행위에 대한 예를 나타내는 도면이다.
- [0010] 도 2는 BD-ROM에서의 파일 디렉터리 구성을 나타내는 도면이다.
- [0011] 도 3은 PL 정보의 구성을 나타내는 도면이다.
- [0012] 도 4는 AVClip 시간 축과 PL 시간 축의 관계를 나타내는 도면이다.
- [0013] 도 5는 4개의 Clip_Information_file_name에 의해 이루어진 일괄지정을 나타내는 도면이다.
- [0014] 도 6은 PLmark 정보의 내부 구성을 나타내는 도면이다.
- [0015] 도 7은 PLmark에 의한 챕터 정의를 나타내는 도면이다.
- [0016] 도 8은 SubPath 정보의 내부 구성을 나타내는 도면이다.
- [0017] 도 9는 SubPlayItem 시간 축 상의 재생구간 정의와, 동기지정을 나타내는 도면이다.
- [0018] 도 10은 Movie Object의 내부 구성을 나타내는 도면이다.
- [0019] 도 11은 BD-J Object의 내부 구성을 나타내는 도면이다.
- [0020] 도 12(a)는 Java(등록상표) 아카이브(archive) 파일에 저장되어 있는 프로그램, 데이터를 나타내는 도면이다.
- [0021] 도 12(b)는 클래스파일의 내부 구성을 나타내는 도면이다.
- [0022] 도 13은 디스크 콘텐츠에서의 상태천이를 나타내는 도면이다.
- [0023] 도 14는 HDMV 모드의 동적 시나리오로 구성되는 타이틀을 나타내는 도면이다.
- [0024] 도 15는 BD-J 모드의 동적 시나리오(BD-J Object)에 의해 구성되는 타이틀의 내부 구성을 나타내는 도면이다.
- [0025] 도 16은 플레이리스트 관리테이블을 갖지 않는 타이틀을 나타내는 도면이다.
- [0026] 도 17은 HDMV 모드의 타이틀로부터 BD-J 모드의 타이틀로의 분기를 나타내는 도면이다.
- [0027] 도 18은 BD-J 모드의 타이틀로부터 HDMV 모드의 타이틀로의 분기를 나타내는 도면이다.
- [0028] 도 19는 index.bdmv의 내부 구성을 나타내는 도면이다.
- [0029] 도 20(a)는 애플리케이션 관리테이블의 내부 구성을 나타내는 도면이다.
- [0030] 도 20(b)는 애플리케이션 관리테이블을 구성하는 정보요소의 의미 내용을 나타내는 도면이다.
- [0031] 도 21(a)는 BD-ROM 전체의 시간 축을 나타내는 도면이다.
- [0032] 도 21(b)는 BD-ROM 전체의 시간 축에서의 구성을 나타내는 도면이다.
- [0033] 도 22(a)(b)는 BD-ROM 전체의 시간 축에서 식별자 bobj_id에 의해 특정되는 BD-J Object에서 특정되는 타이틀 재생구간을 나타내는 도면이다.

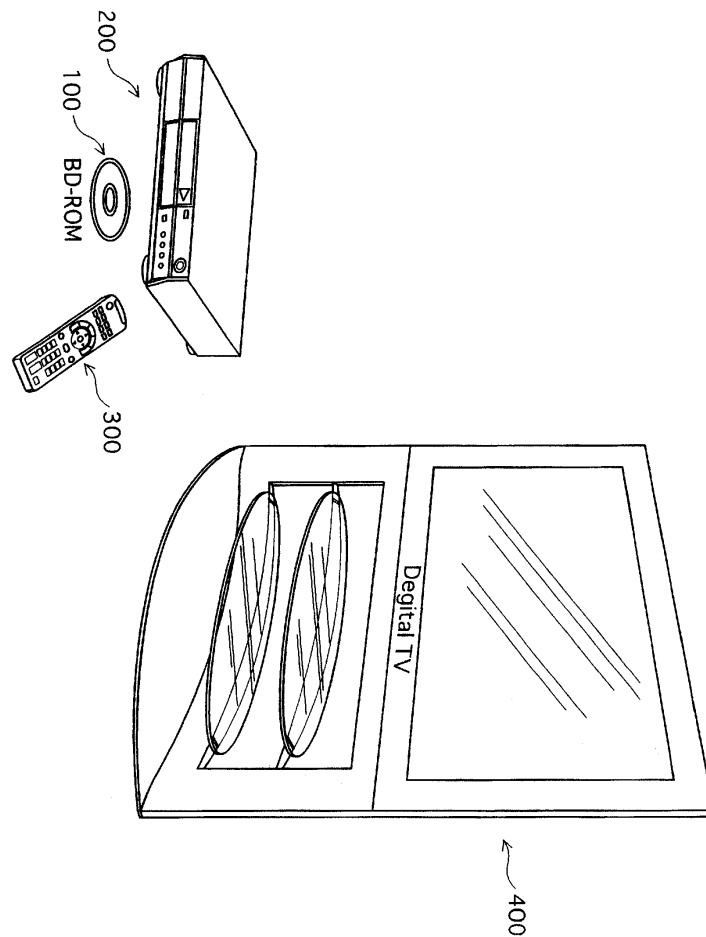
- [0034] 도 23은 도 22(b)의 시간 축 상에 규정되는 생존구간의 전형을 나타내는 도면이다.
- [0035] 도 24는 본편 타이틀, 온라인 쇼핑 타이틀, 게임 타이틀이라는 3개의 타이틀을 포함하는 디스크 콘텐츠를 나타내는 도면이다.
- [0036] 도 25(a)(b)는 애플리케이션 관리테이블, 생존구간의 일례를 나타내는 도면이다.
- [0037] 도 26은 기동속성이 취할 수 있는 3가지 형태(Present, AutoRun, Suspend)와, 직전 타이틀에서의 애플리케이션 상태의 3가지 형태(비 기동, 기동중, Suspend)가 취할 수 있는 조합을 나타내는 도면이다.
- [0038] 도 27(a)는 플레이리스트 관리테이블의 내부 구성을 나타내는 도면이다.
- [0039] 도 27(b)는 플레이리스트 관리테이블을 구성하는 정보요소의 의미 내용을 나타내는 도면이다.
- [0040] 도 28은 분기처 타이틀이 취할 수 있는 3가지 형태(플레이리스트 관리테이블 없음, 플레이리스트 관리테이블이 있으면서 Present, 플레이리스트 관리테이블이 있으면서 AutoPlay)와, 직전 타이틀에서의 PL 상태(비 재생상태, 재생중 상태)가 취할 수 있는 6가지의 조합을 나타내는 도면이다.
- [0041] 도 29(a)는 플레이리스트 관리테이블 및 애플리케이션 관리테이블의 기술 예를 나타내는 도면이다.
- [0042] 도 29(b)는 도 29(a)와 같이 기술된 애플리케이션 관리테이블, 플레이리스트 관리테이블에 의해 플레이리스트 재생, 애플리케이션 실행이 어떻게 진행되는지를 나타내는 도면이다.
- [0043] 도 30(a)는 플레이리스트 관리테이블의 다른 기술 예를 나타내는 도면이다.
- [0044] 도 30(b)는 도 30(a)의 케이스에 기초하는 애플리케이션 실행 및 플레이리스트 재생의 진행을 나타내는 도면이다.
- [0045] 도 31(a)~(c)는 플레이리스트 시간 축과 타이틀의 재생구간과의 관계를 나타내는 도면이다.
- [0046] 도 32는 본 발명에 관한 재생장치의 내부 구성을 나타내는 도면이다.
- [0047] 도 33은 ROM(24)에 저장된 소프트웨어와 하드웨어로 이루어지는 부분을 레이어 구성으로 치환하여 묘사한 도면이다.
- [0048] 도 34는 프레젠테이션 엔진(Presentation Engine, 31)~모듈매니저(34)에 의한 처리를 모식화 한 도면이다.
- [0049] 도 35는 애플리케이션 매니저(36)에 의한 처리를 모식화 한 도면이다.
- [0050] 도 36은 사용자 이벤트 매니저(User Event Manager, 37)~디폴트 오퍼레이션 매니저(Default Operation Manager, 40)를 나타내는 도면이다.
- [0051] 도 37은 Java(등록상표) 가상머신(38)의 내부 구성을 나타내는 도면이다.
- [0052] 도 38은 애플리케이션 종료의 4가지 형태를 나타내는 도면이다.
- [0053] 도 39는 애플리케이션 매니저(36)에 의한 처리순서를 나타내는 도면이다.
- [0054] 도 40은 플레이리스트 관리테이블, 애플리케이션 관리테이블의 구체 예를 나타내는 도면이다.
- [0055] 도 41(a)는 제 2 실시 예에 관한 BD-J Object의 내부 구성을 나타내는 도면이다.
- [0056] 도 41(b)는 에러관리테이블의 내부 구성을 나타내는 도면이다.
- [0057] 도 42는 에러관리테이블에서의 5개의 플래그의 의미 내용을 나타낸다.
- [0058] 도 43(a)는 에러관리테이블이 기술된 2개의 타이틀(title #1, title #2)을 나타내는 도면이다.
- [0059] 도 43(b)는 도 43(a)와 같이 기재된 애플리케이션 관리테이블, 에러관리테이블에 기초하는 애플리케이션 실행 및 플레이리스트 재생의 진행을 나타내는 도면이다.
- [0060] 도 44는 제 2 실시 예에 관한 애플리케이션 매니저(36)의 처리순서를 나타내는 플로차트이다.
- [0061] 도 45는 제 2 실시 예에 관한 애플리케이션 매니저(36)의 처리순서를 나타내는 플로차트이다.
- [0062] 도 46은 애플리케이션 매니저(36)에 의한 통지처리의 처리순서를 나타내는 플로차트이다.

[0063]	도 47은 제 3 실시 예에 관한 애플리케이션 매니저(36)의 처리순서를 나타내는 도면이다.
[0064]	도 48(a)는 시청제한레벨에 의한 선택 알고리즘의 내용을 나타낸다.
[0065]	도 48(b)는 Language for Audio에 기초하는 선택 알고리즘의 내용을 나타낸다.
[0066]	도 48(c)는 Player Configuration for Video에 기초하는 선택 알고리즘의 내용을 나타낸다.
[0067]	도 49는 타이틀 언바운더리 애플리케이션(Title unboundary application)이 PL 선택을 행하도록 하는 과정을 모식적으로 묘사한 도면이다.
[0068]	도 50은 플레이 백 컨트롤 엔진(Playback Control Engine, 32)에 의한 PL 재생순서를 나타내는 플로차트이다.
[0069]	도 51은 앵글절환, SkipBack, SkipNext의 접수순서를 나타내는 플로차트이다.
[0070]	도 52는 SkipBack, SkipNextAPI가 콜(Call) 된 때의 처리순서를 나타내는 플로차트이다.
[0071]	도 53은 프레젠테이션 엔진(31)에 의한 처리순서의 상세한 내용을 나타내는 플로차트이다.
[0072]	도 54는 SubPlayItem의 재생순서를 나타내는 플로차트이다.
[0073]	*부호의 설명*
[0074]	1 BD-ROM 드라이브
[0075]	2 리드 버퍼
[0076]	3 디멀티플렉서
[0077]	4 비디오 디코더
[0078]	5 비디오 플레인(Video Plane)
[0079]	6 P-Graphics 디코더
[0080]	7 Presentation Graphics 플레인
[0081]	8 합성부
[0082]	9 폰트 제너레이터
[0083]	10 I-Graphics 디코더
[0084]	11 스위치
[0085]	12 Interactive Graphics 플레인
[0086]	13 합성부
[0087]	14 CLUT부
[0088]	15 CLUT부
[0089]	16 오디오 디코더
[0090]	22 사용자 이벤트 처리부
[0091]	23 PSR 세트
[0092]	24 CPU
[0093]	25 시나리오 메모리
[0094]	26 로컬 메모리
[0095]	33 HDMV 모듈
[0096]	34 모듈 매니저
[0097]	35 BD-J 모듈

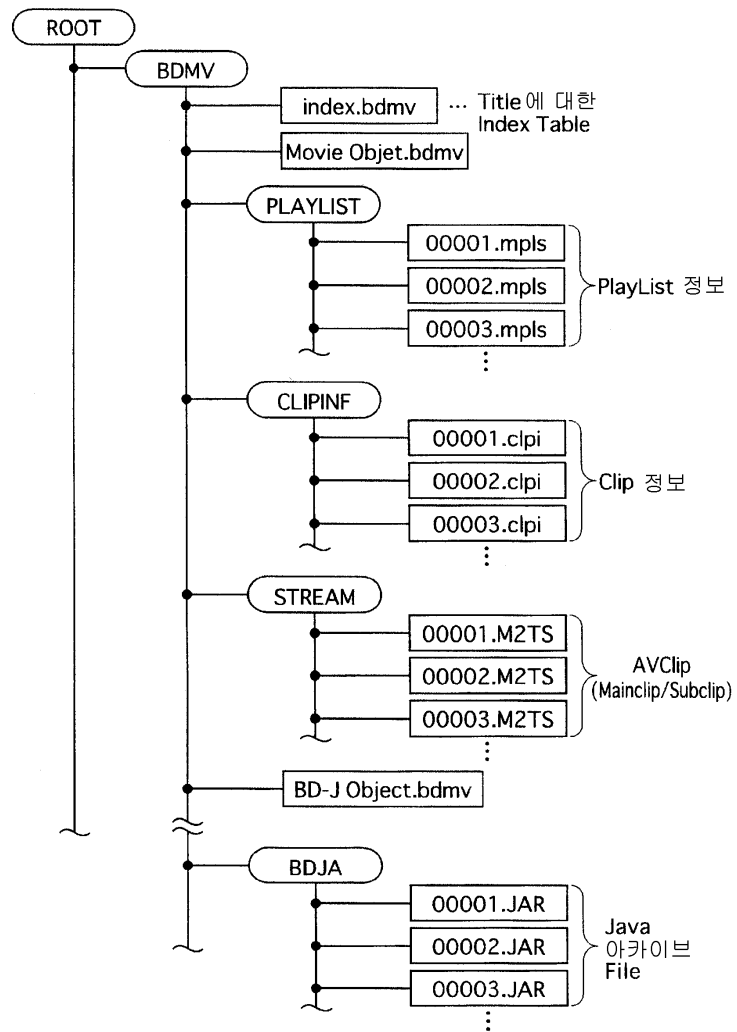
[0098]	36	애플리케이션 매니저
[0099]	37	UO 컨트롤러
[0100]	38	Java(등록상표) 가상머신
[0101]	41	PLMT 프로세서
[0102]	42	퍼미션 컨트롤러(permission controller)
[0103]	52	사용자 클래스 로더(User Class Loader)
[0104]	53	메소드영역(method area)
[0105]	54	워크 메모리(work memory)
[0106]	55a, b...n	스레드(thread)
[0107]	56a, b...n	Java(등록상표) 스택(stack)

도면

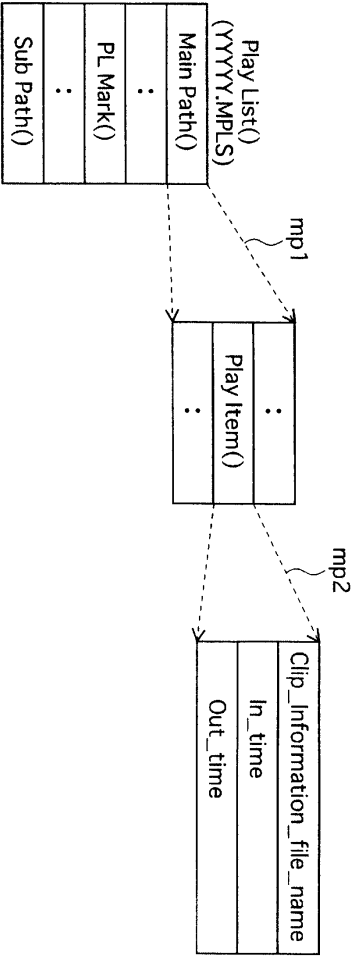
도면1



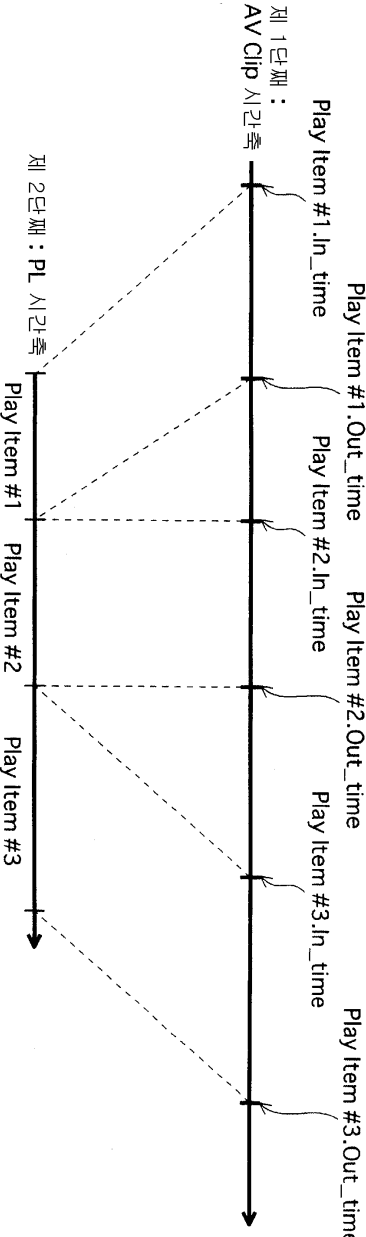
도면2

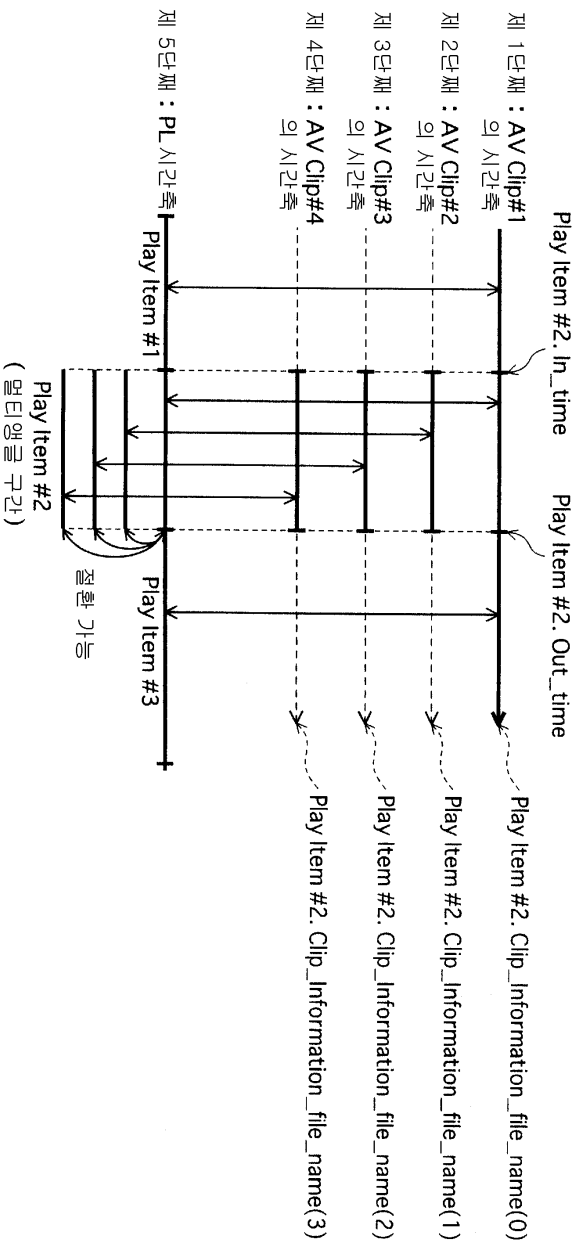


도면3

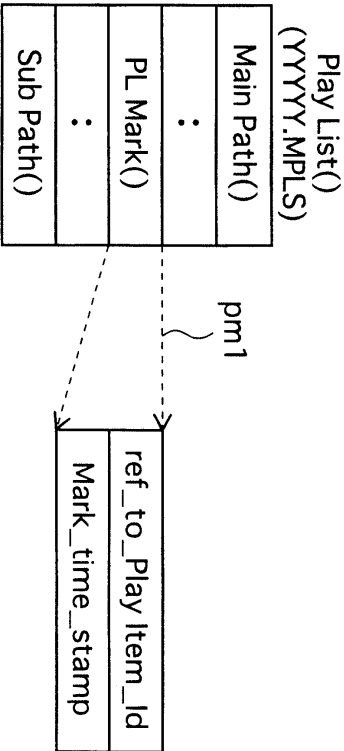


도면4

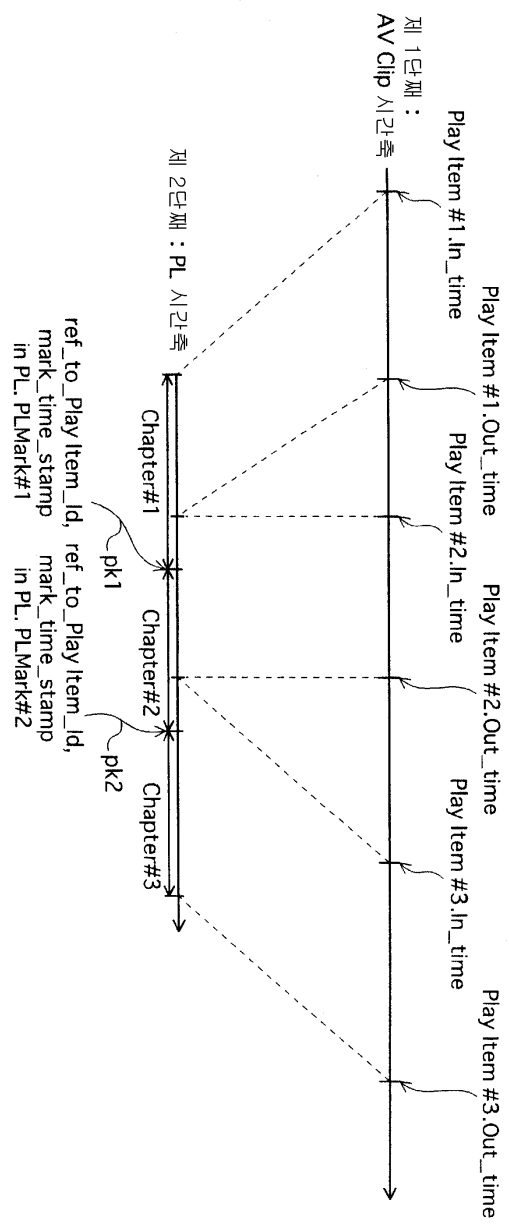




도면5

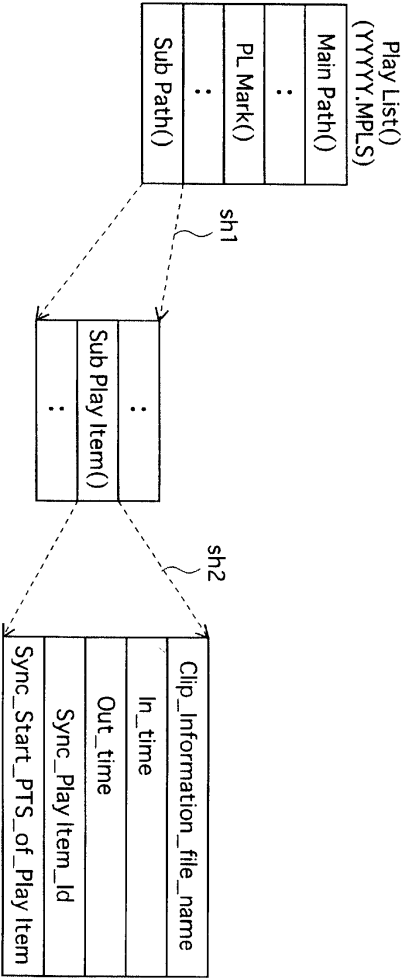


도면6

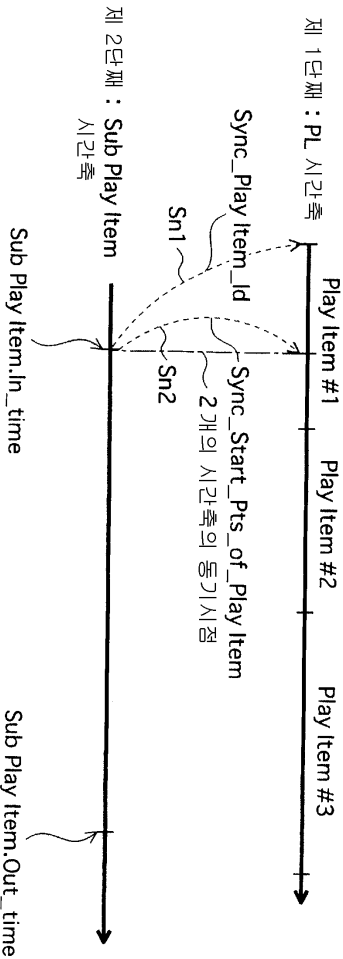


도면7

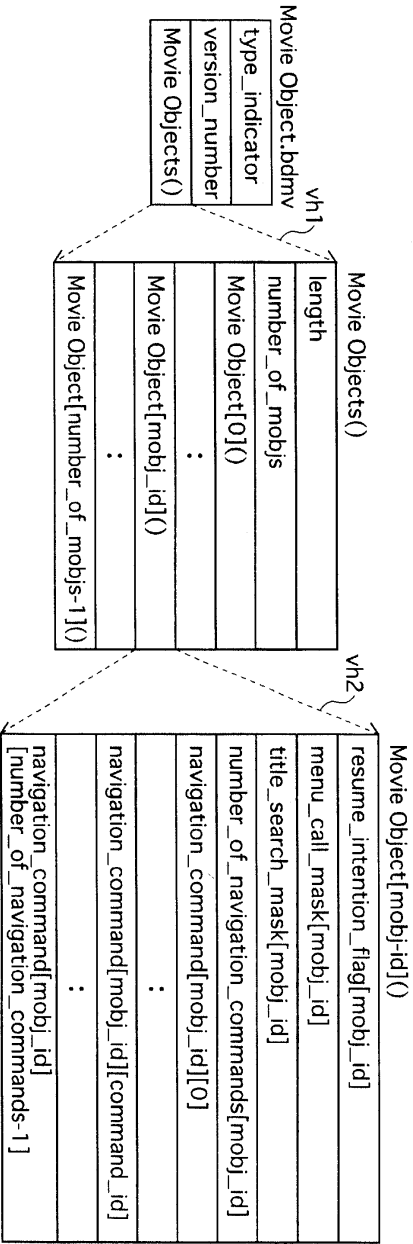
도면8



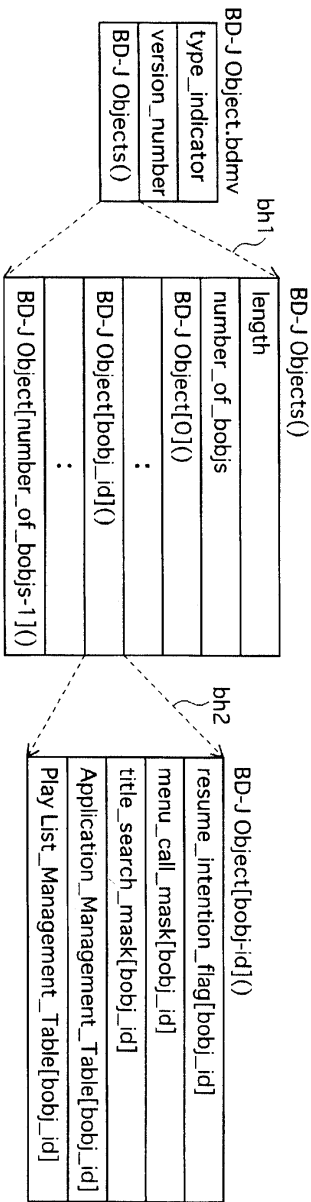
도면9



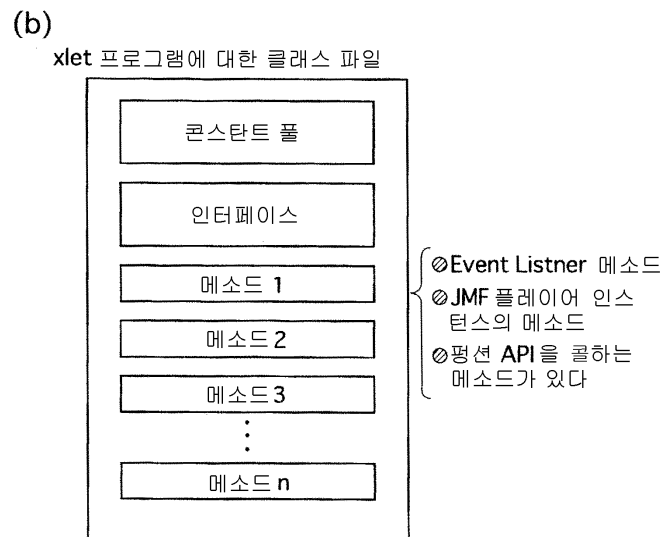
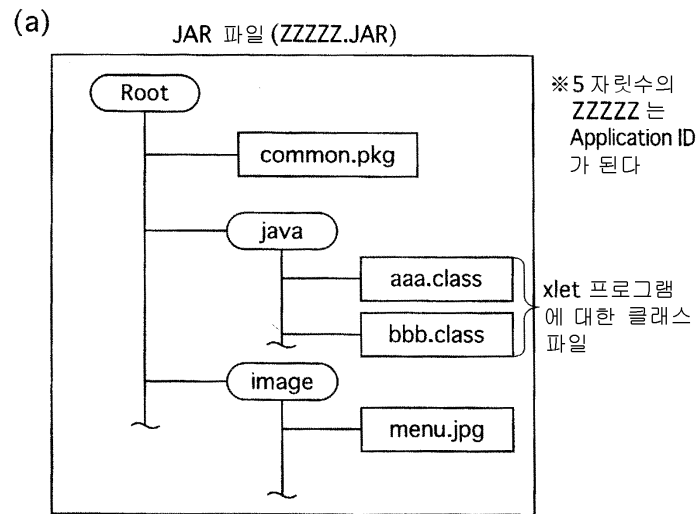
도면10



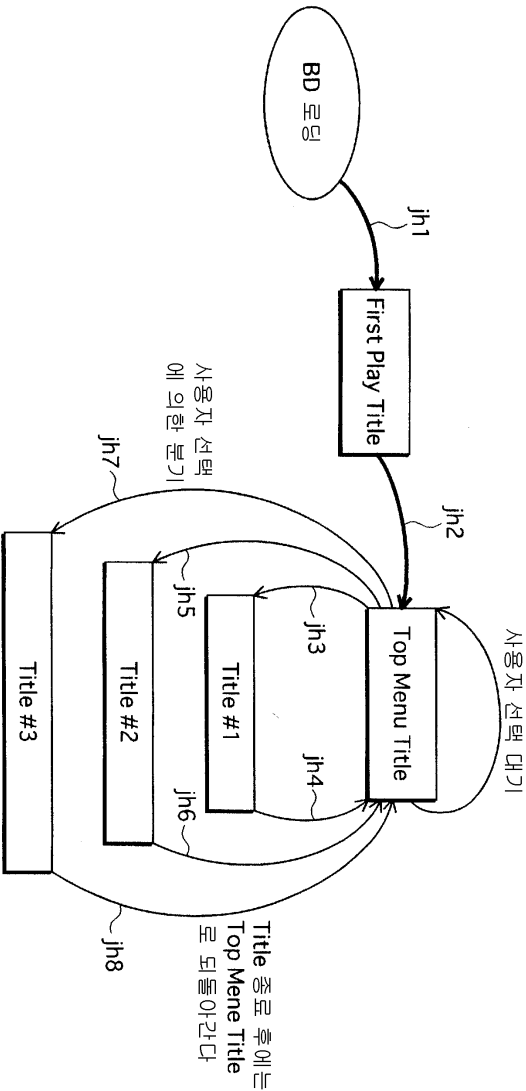
도면11



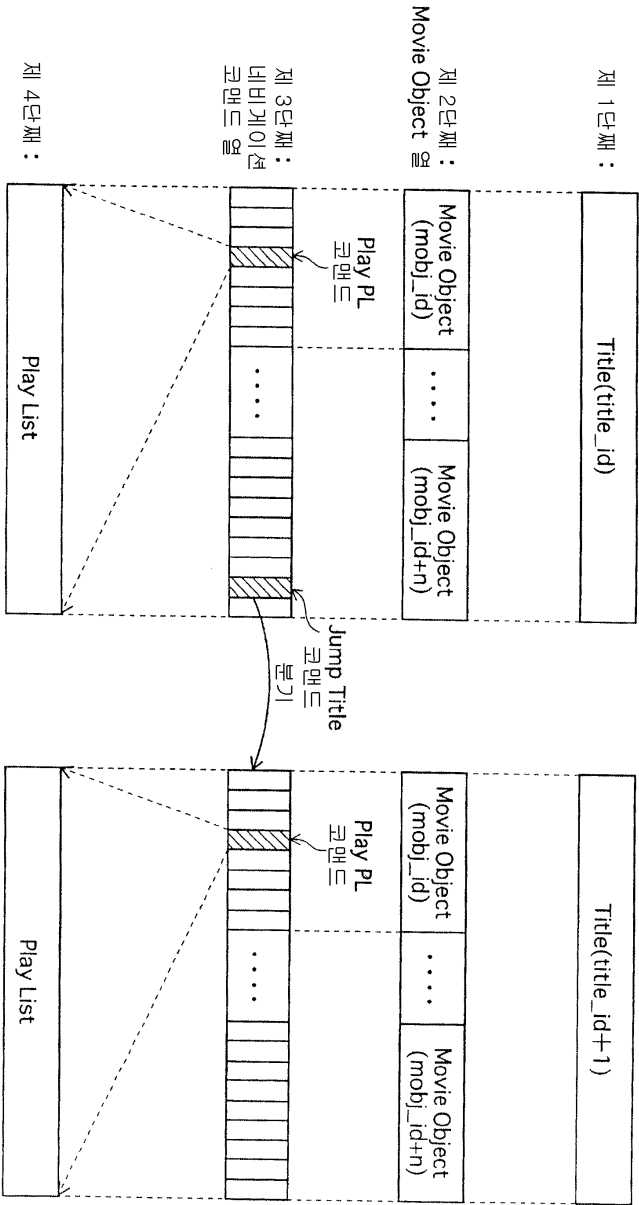
도면12



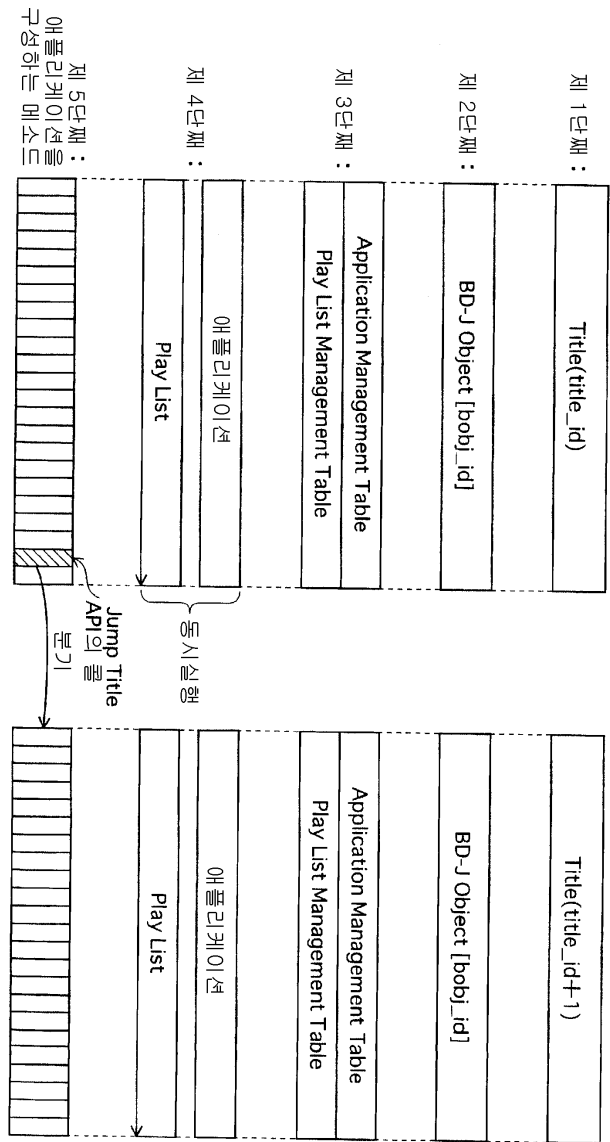
도면13



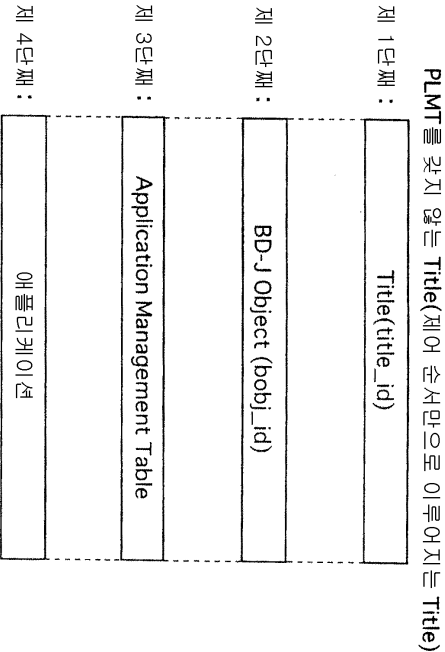
도면14

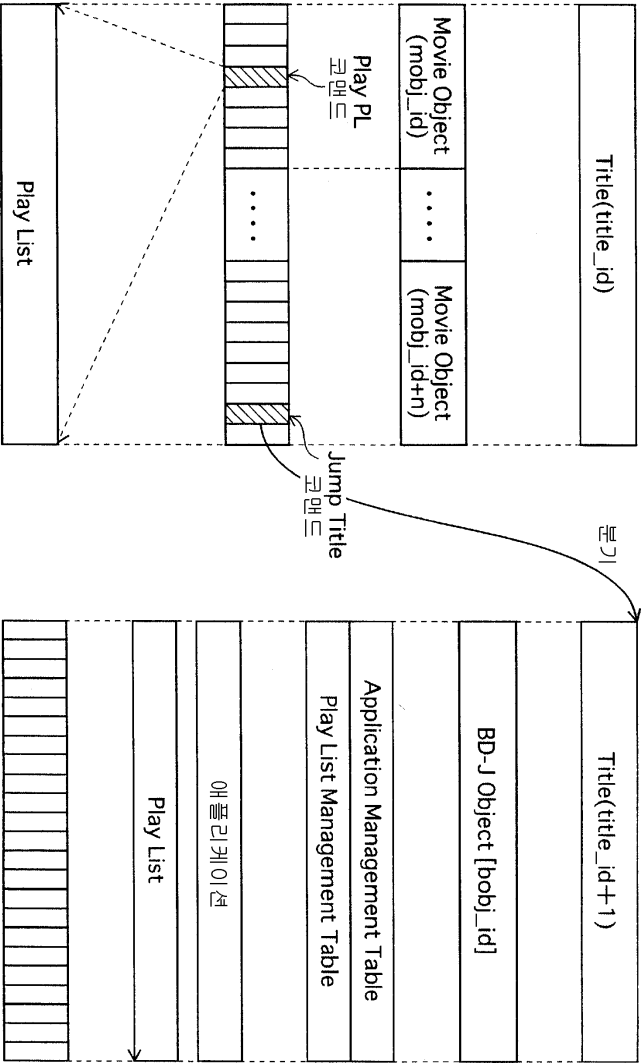


도면15



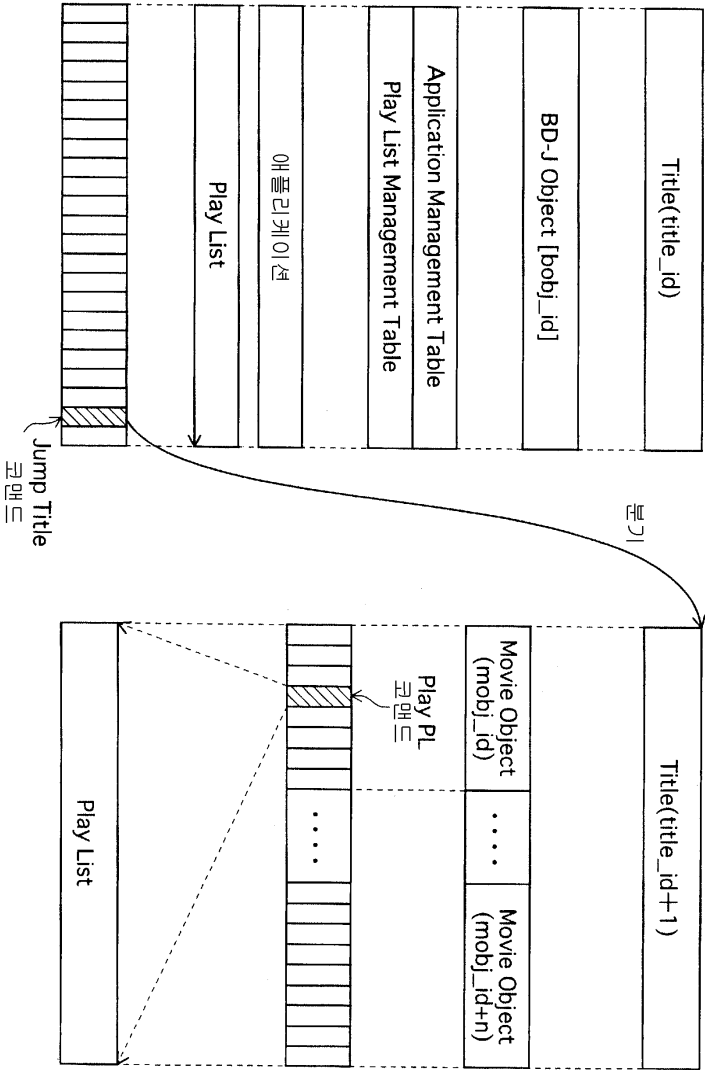
도면16

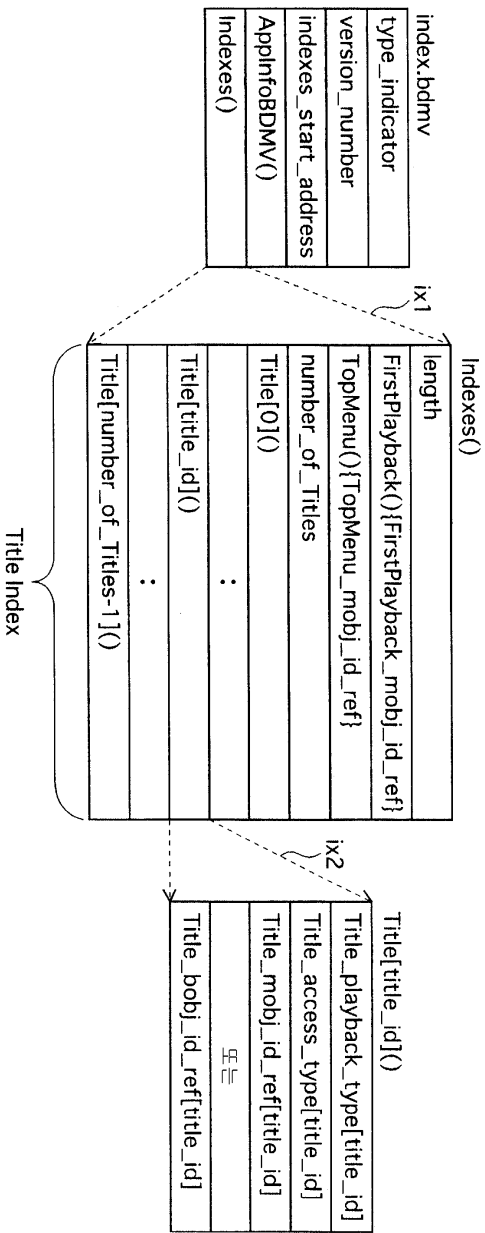




도면17

도면18





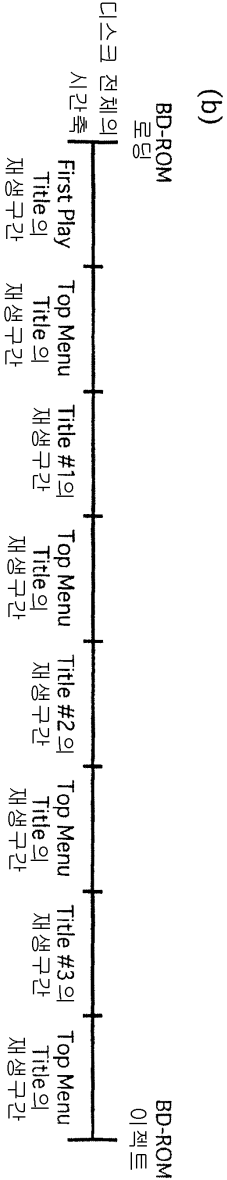
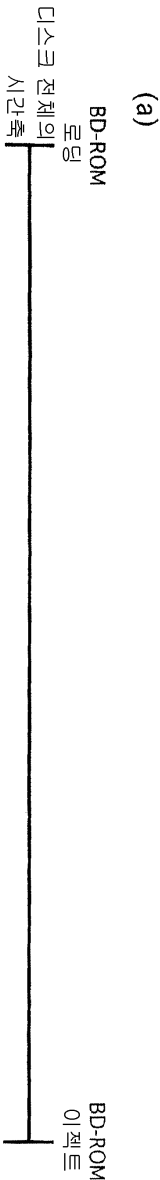
도면19

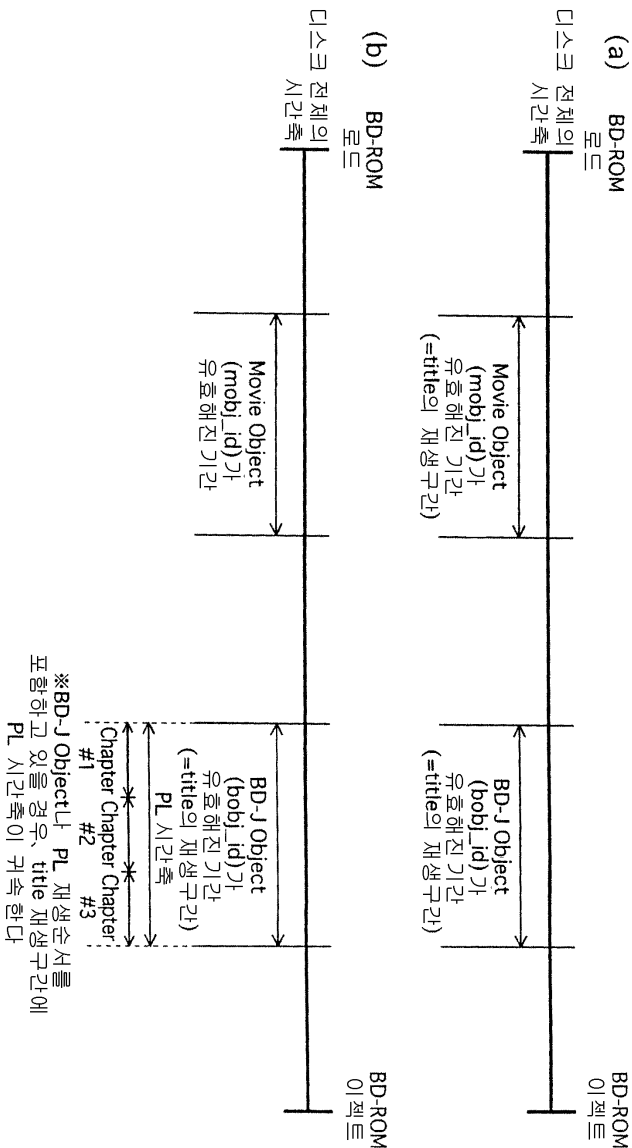
(a) Application Management Table (AMT)

life_cycle
apli_id_ref
run_attribute
run_priority

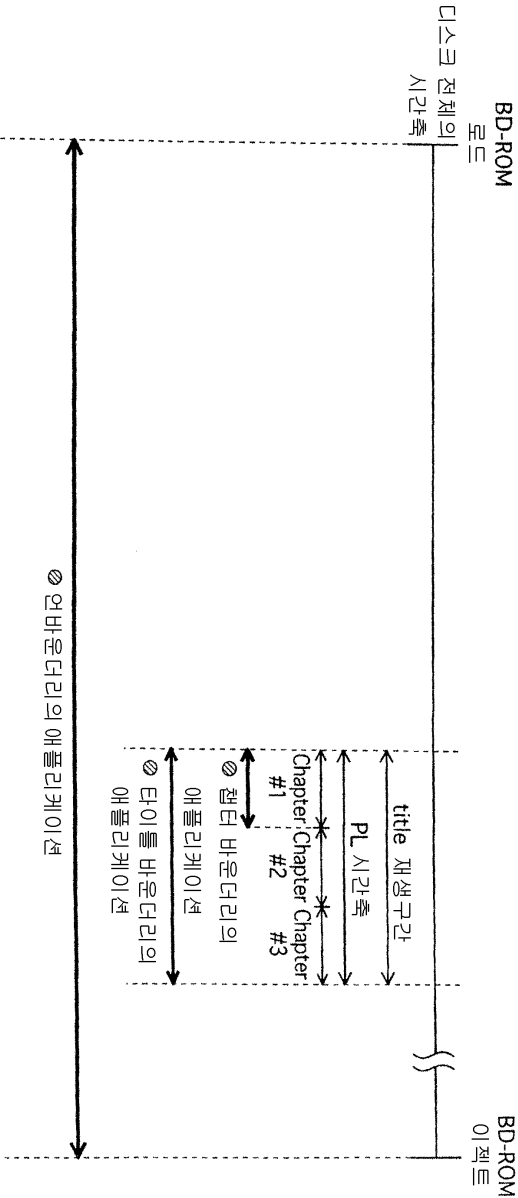
(b) Application Management Table(AMT)

life_cycle	apli_id_ref	run_attribute	run_priority
title,P,L,Chapter를 이용하여 표현된 애플리케이션의 "생존기간"	JAR의 파일명에 부여된 5자릿수의 정수 ZZZZ인 "애플리케이션 ID"의 참조값	• Auto Run • Present(무지정) • Suspend 중 어느 하나인 "기동속성"	0~255의 값을 갖는 "기동 우선도"

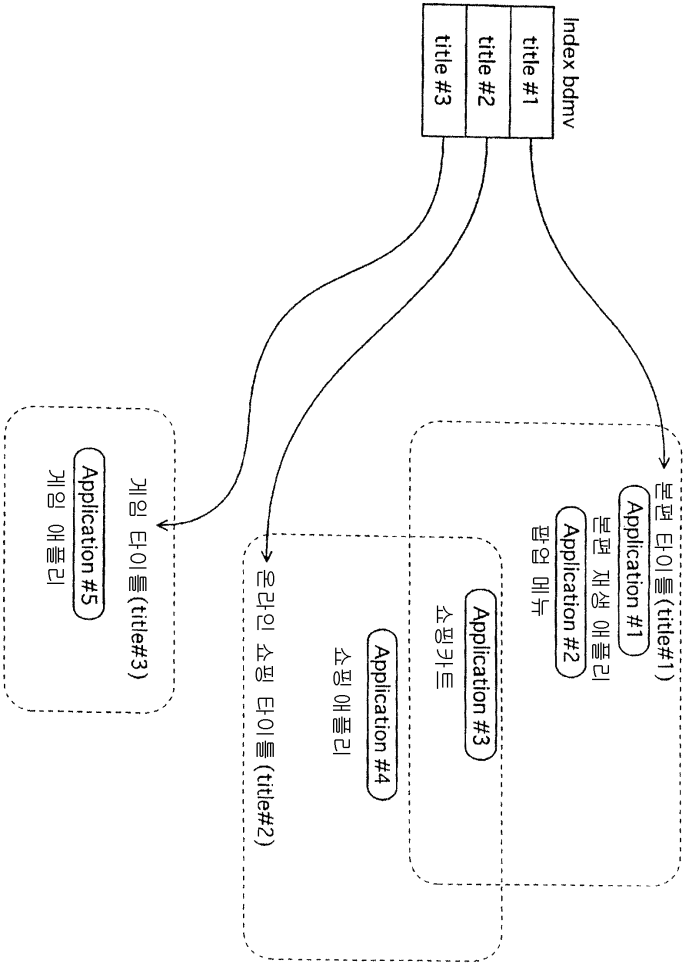


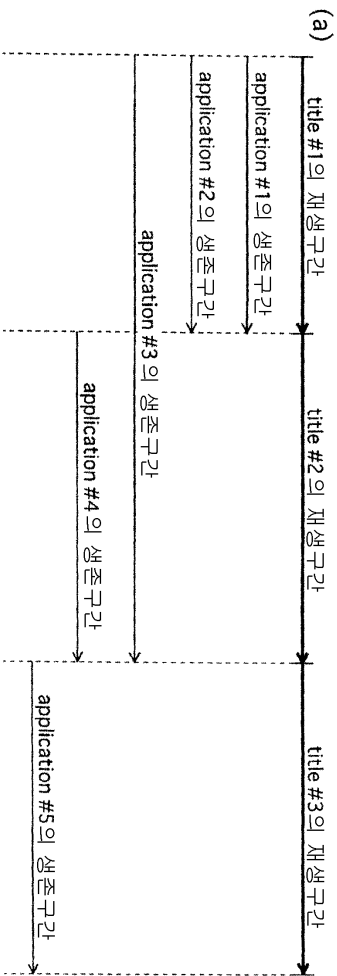


도면23



도면24





(b)

title #1의 애플리케이션 관리 테이블			
생존구간	애플리케이션 ID의 참조값	기동속성	기동우선도
title #1	application #1		
title #1	application #2		
title #1	application #3		

title #2의 애플리케이션 관리 테이블			
생존구간	애플리케이션 ID의 참조값	기동속성	기동우선도
title #2	application #3		
title #2	application #4		

title #3의 애플리케이션 관리 테이블			
생존구간	애플리케이션 ID의 참조값	기동속성	기동우선도
title #3	application #5		

도면26

기동속성에 의한 애플리케이션 상태의 변화				
직전 타이틀 에서의 애플리케이션의 상태	기동속성			
	Present	AutoRun	Suspend	
	비기동	아무것도 하지않고, 상태계속	애플리케이션을 기동함	아무것도 하지않고, 상태계속
	기동중	아무것도 하지않고, 상태계속	아무것도 하지않고, 상태계속	서스펜드한다
	Suspend	재개한다	재개한다	아무것도 하지않고, 상태계속

도면27

(a) Play List Management Table (PLMT)

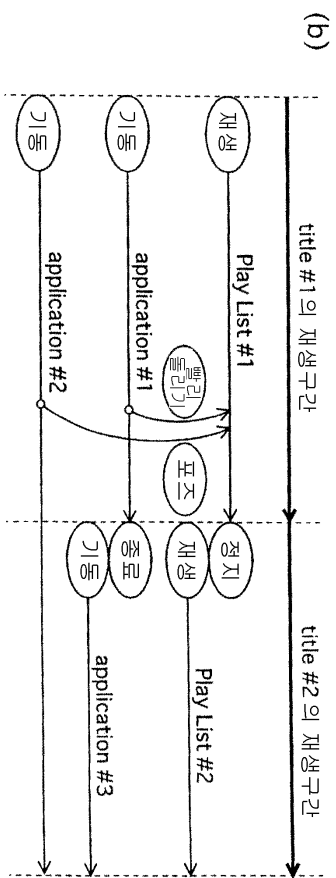
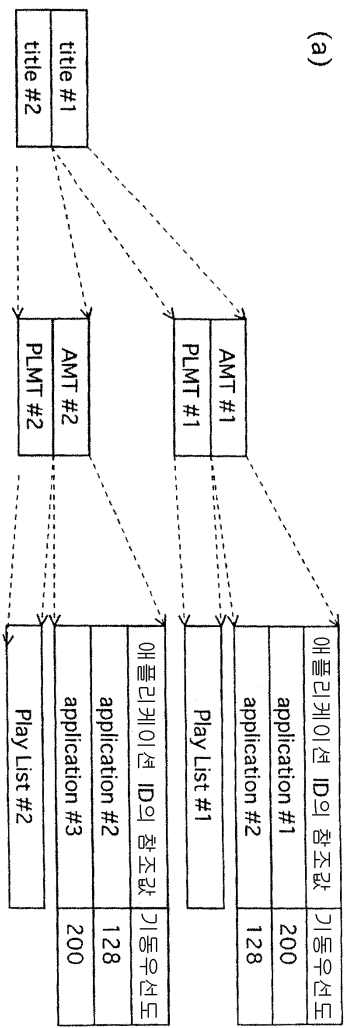
PL_id_ref
Playback_Attribute

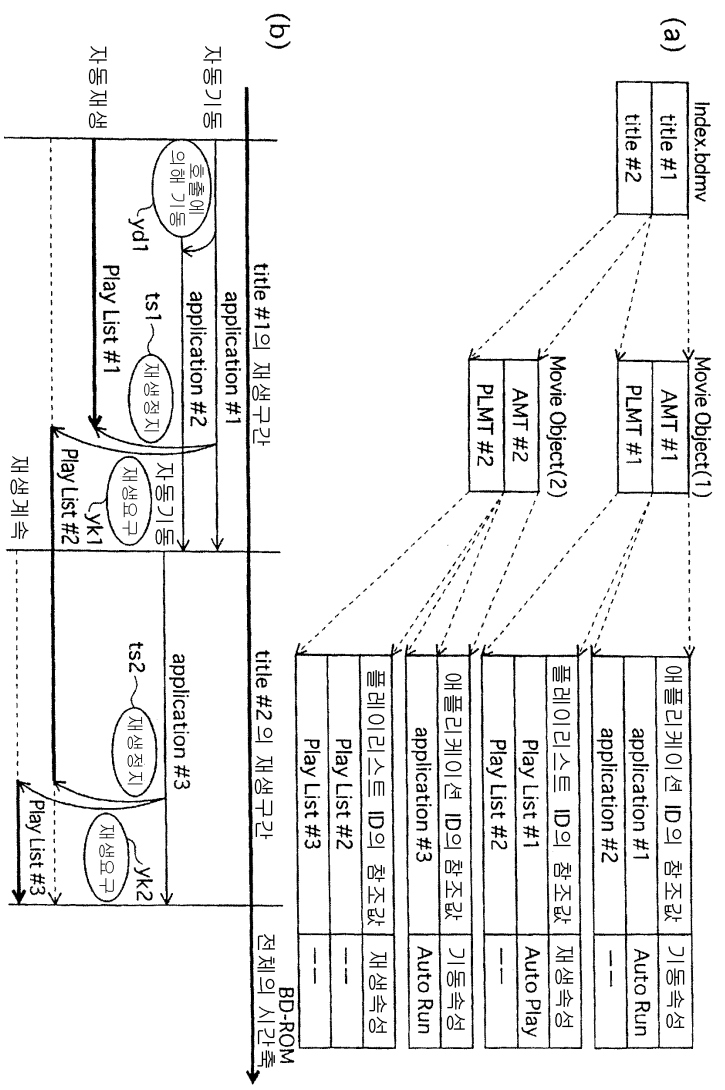
(b) Play List Management Table (PLMT)

PL_id_ref	Playback_Attribute
MPLS 파일의 파일명에서의 5 자릿수의 수치인 “Play List ID”	• Auto Play • Present(무지정) 중 어느 하나인 “재생속성”

도면28

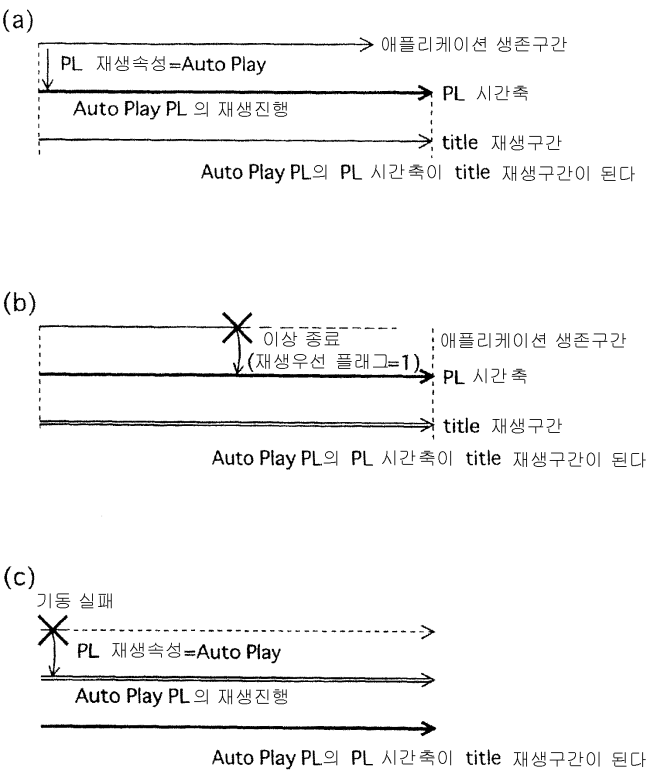
	분기처 title 에 PLMT 있음	분기처 title에 PLMT 있음	
		재생속성 : Auto Play	재생속성 : Present
분기처 title은 재생중 상태	재생정지	상태계속	상태계속
분기처 title은 비재생 상태	상태계속	자동재생 개시	상태계속



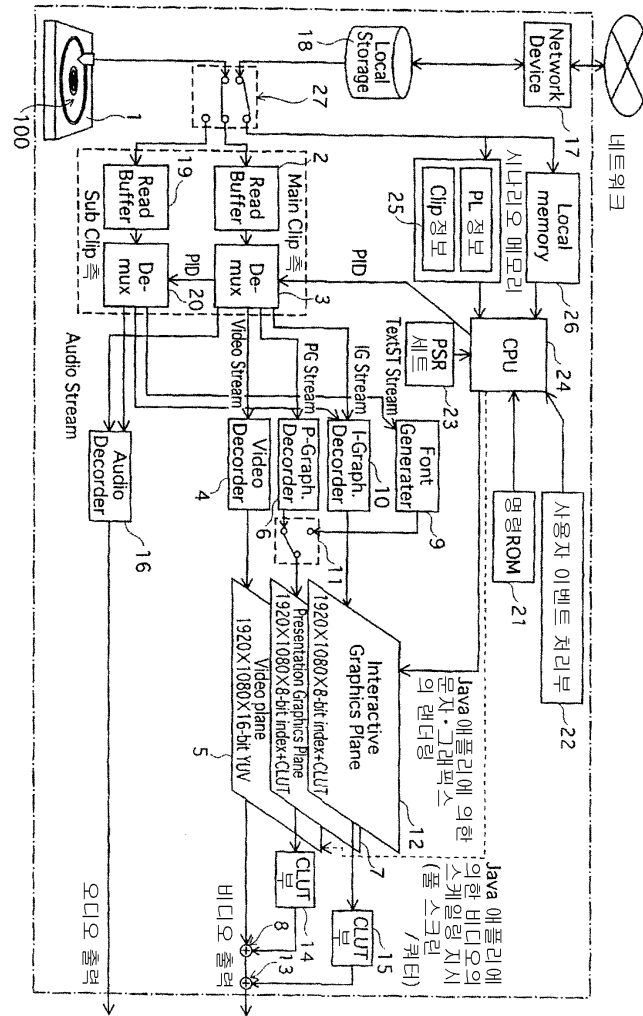


도면30

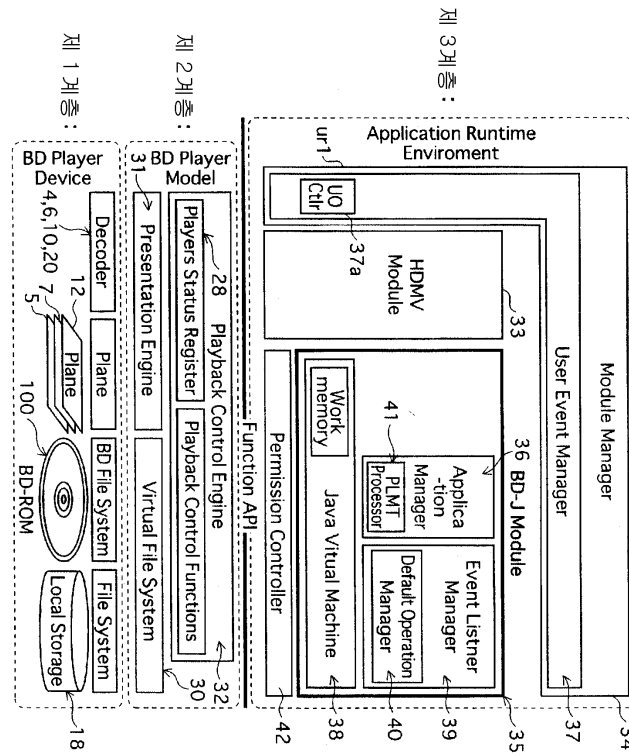
도면31



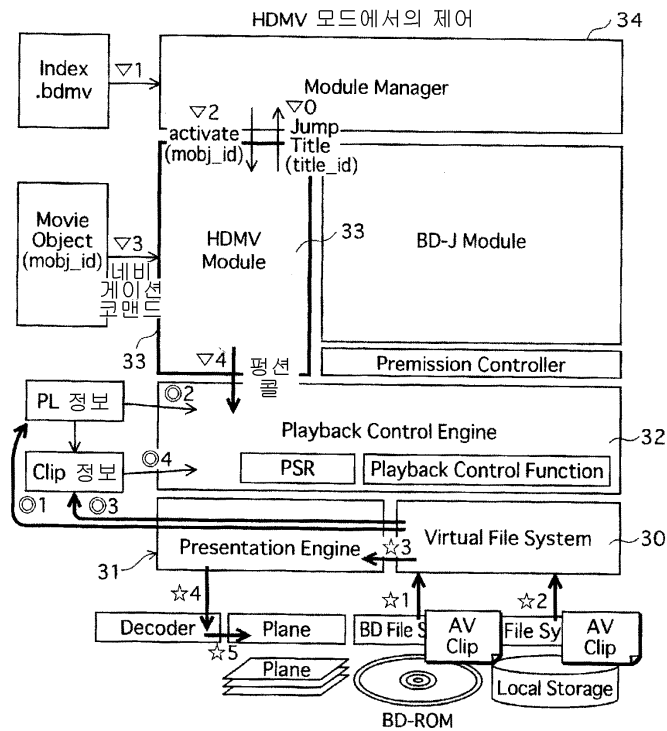
도면32



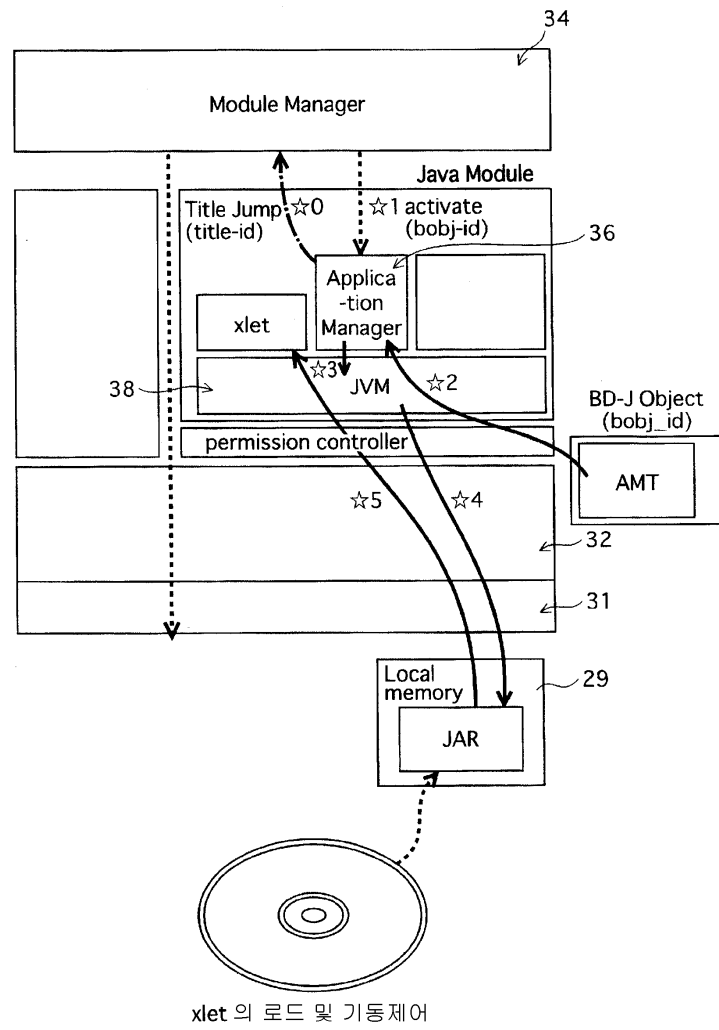
도면33

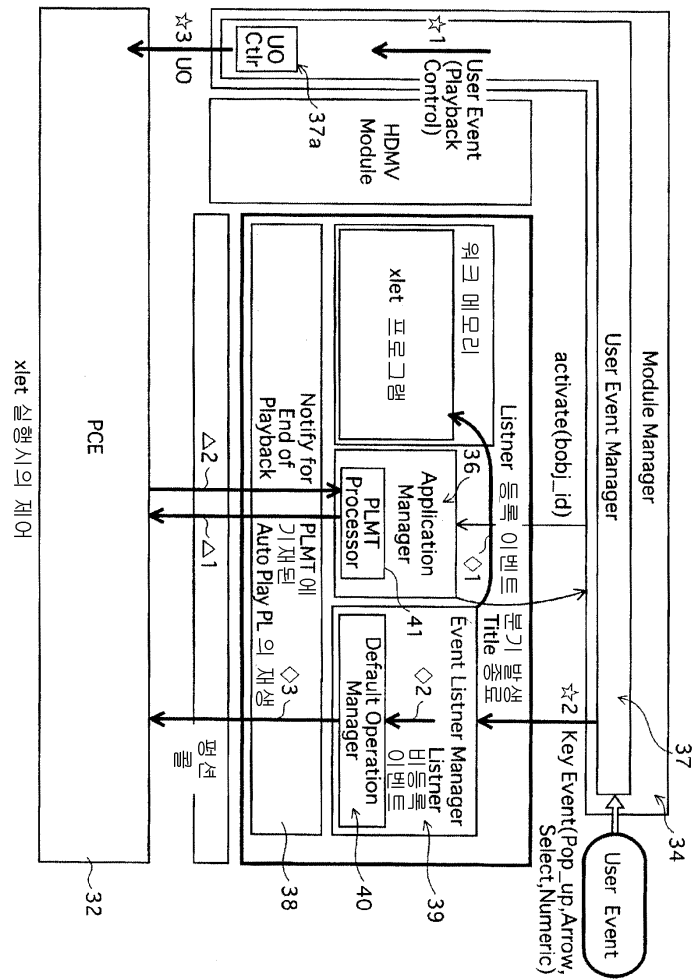


도면34

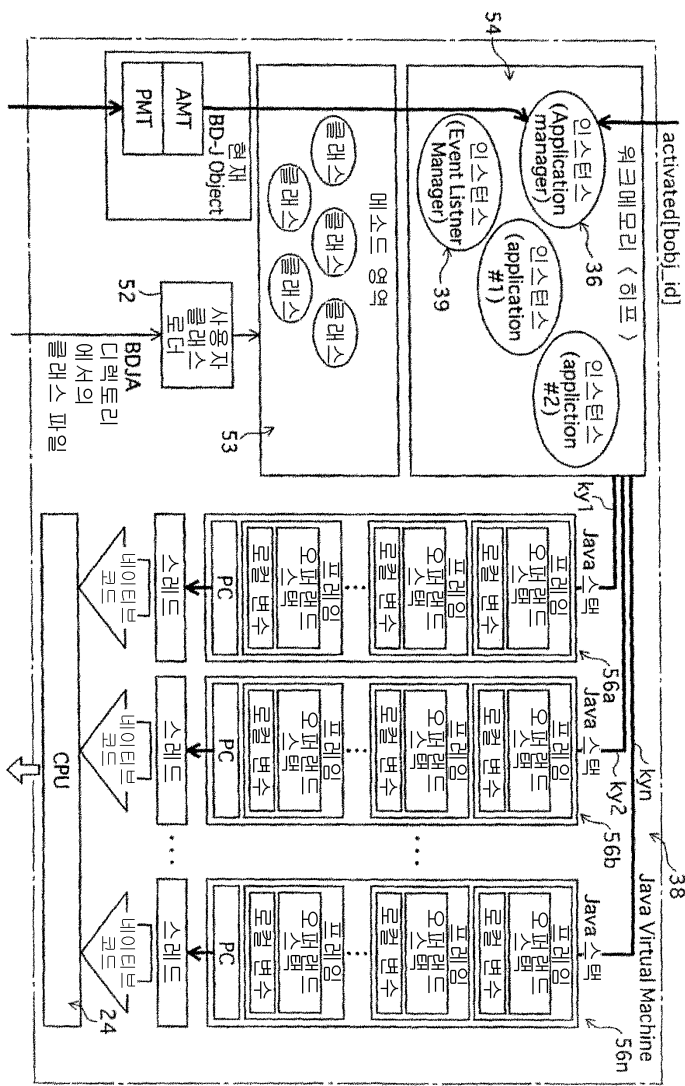


도면35

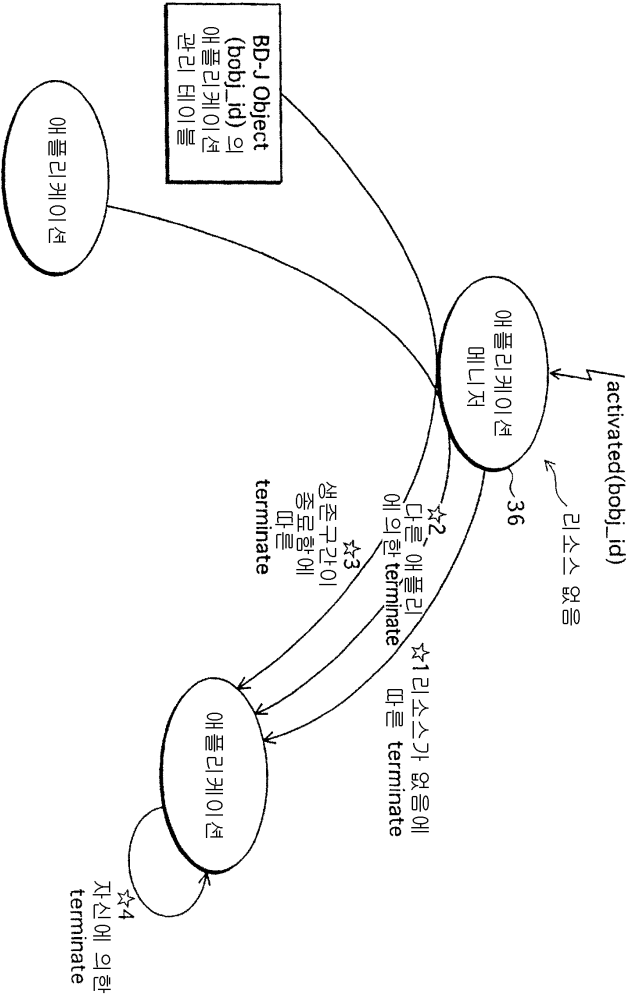




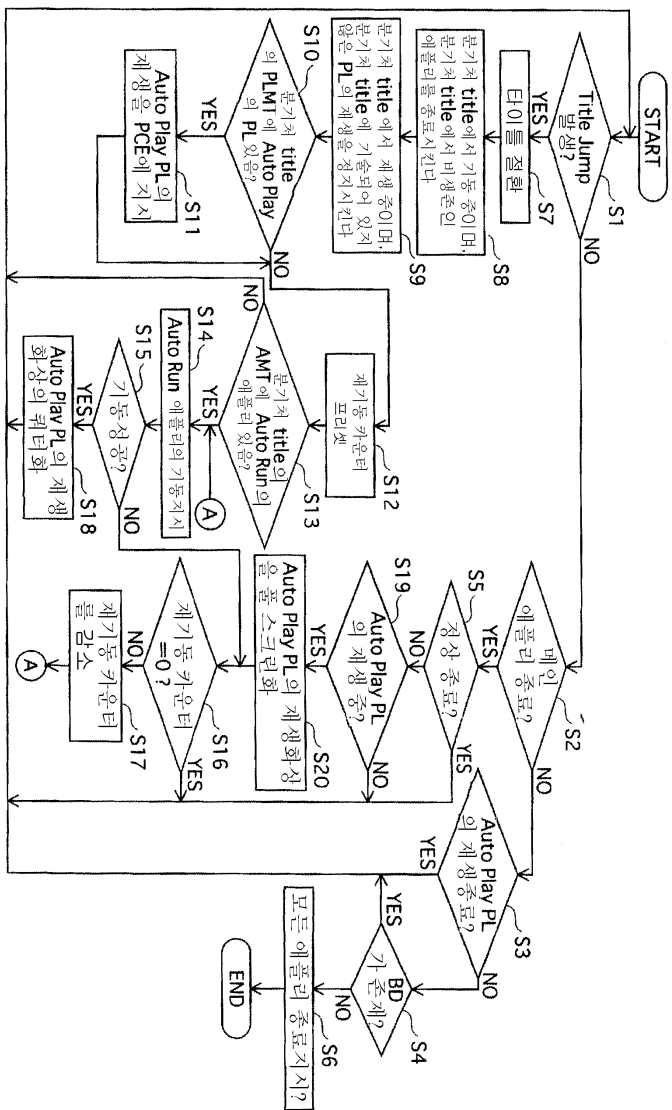
도면37

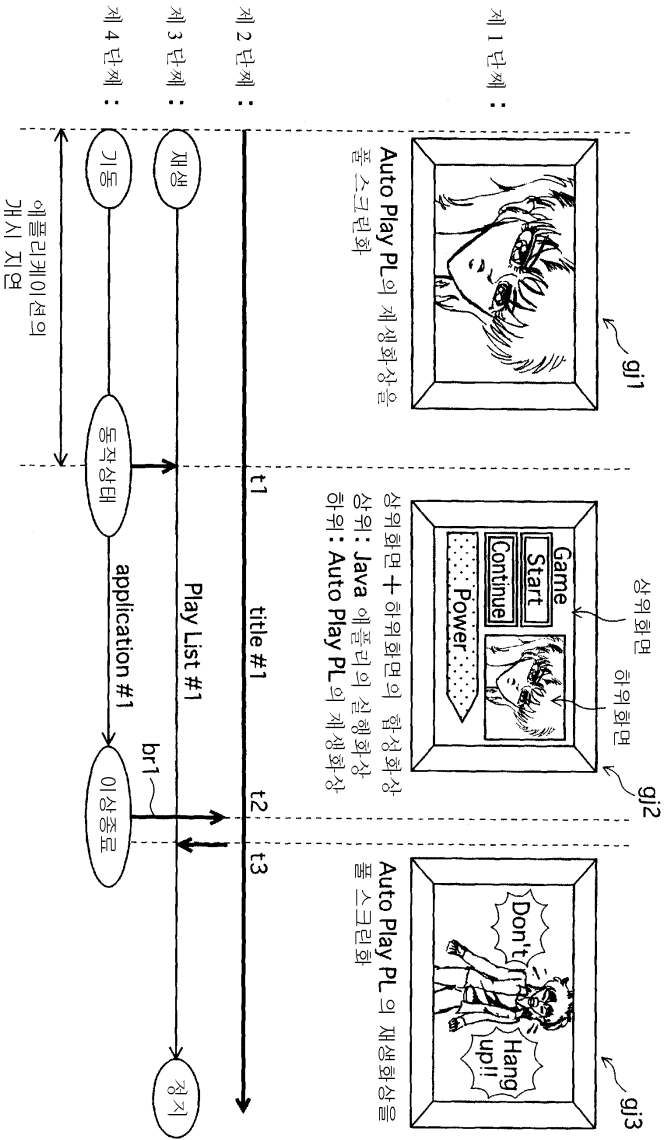


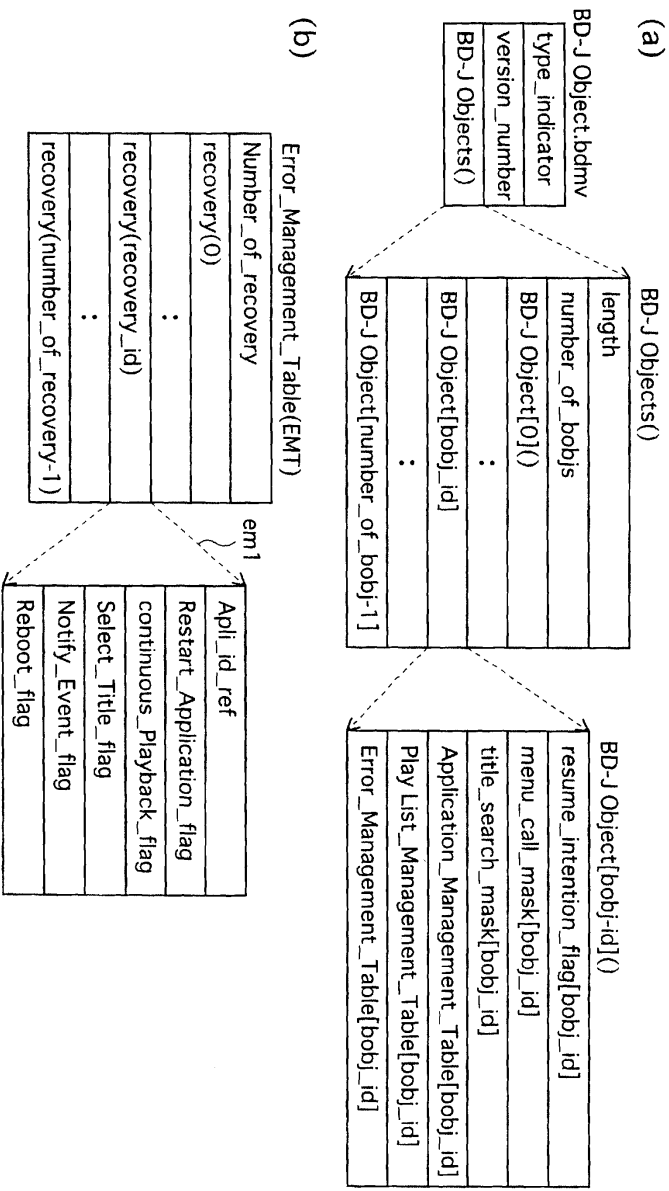
도면38



도면39



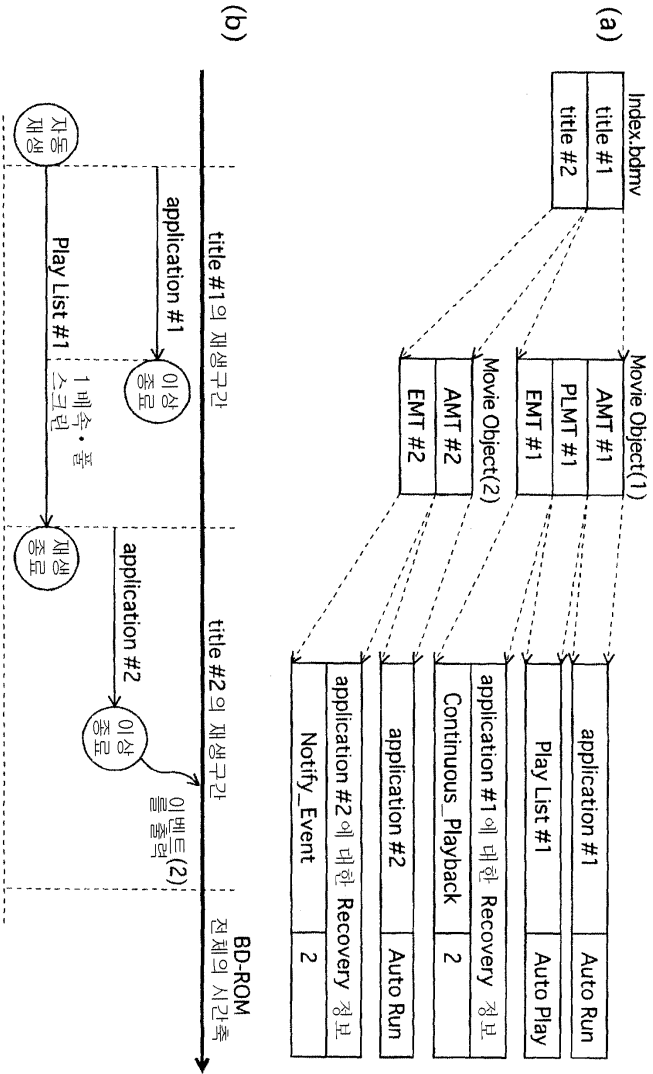




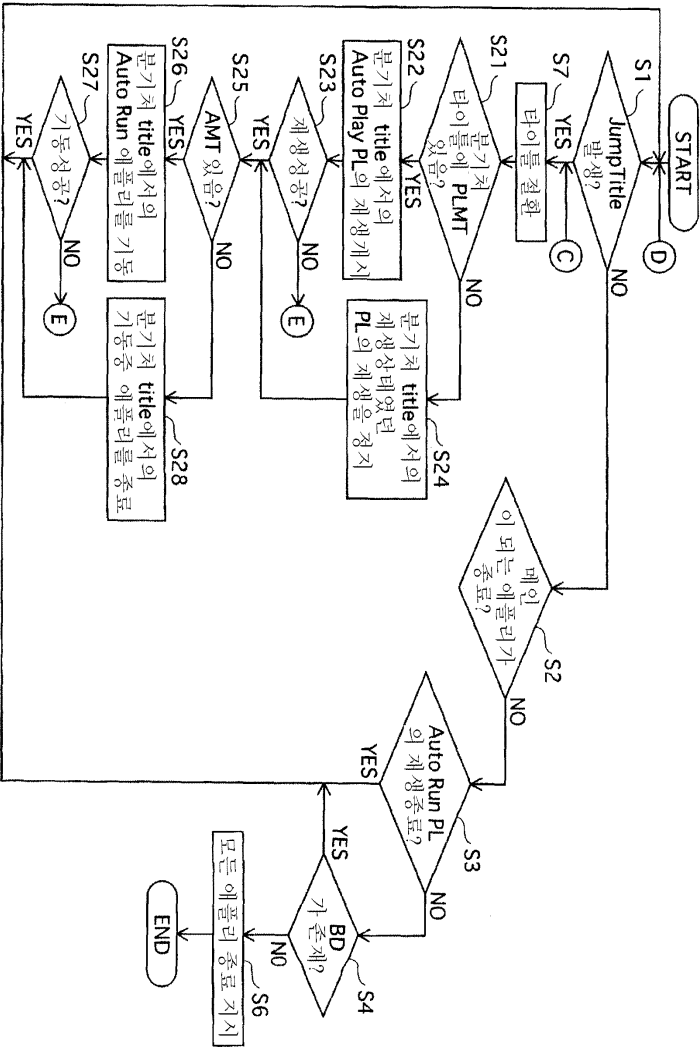
Recovery()

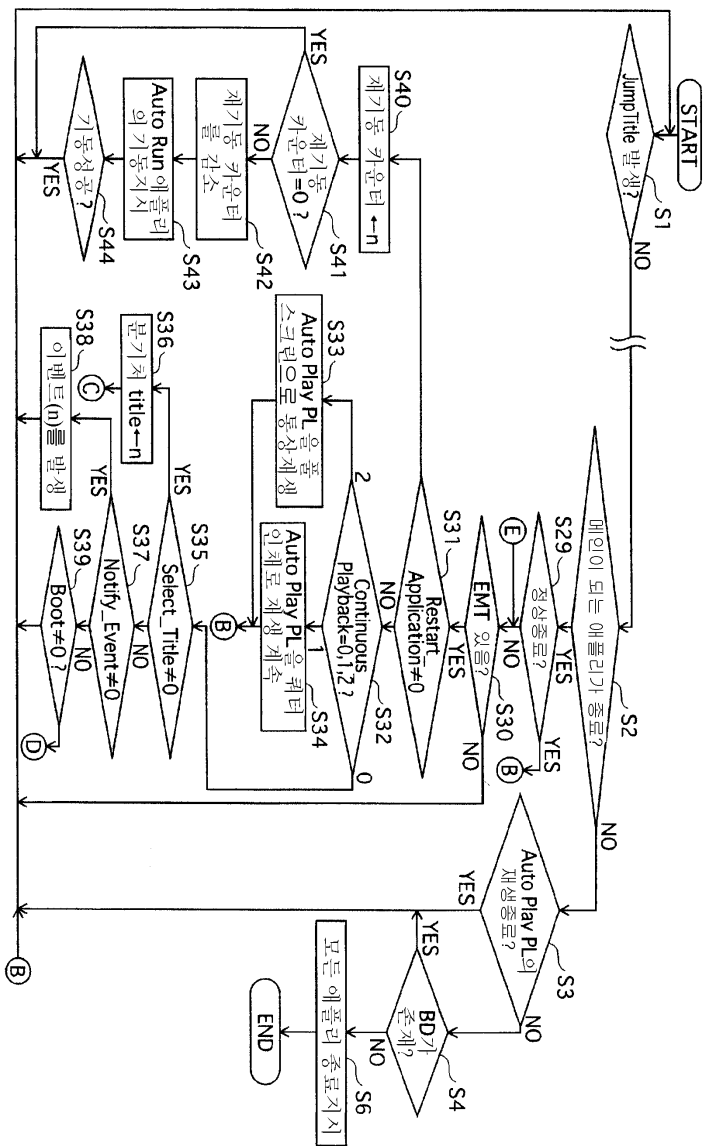
	값	디폴트
Restart_Application_Flag (애플리 재기동)	0 : No n : 재기동 횟수	0
Continuous_Playback_Flag (플레이어리스트 재생계속)	0 : No 1 : Yes 2 : 폴스크린 / 1 배속에 의한 재생	0
Select_Title_Flag (타이틀 선택)	0 : No n : 타이틀 번호	0
Notify_Event_Flag (이벤트 통지)	0 : No n : 이벤트 번호	1
Reboot_Flag (시스템 재기동)	0 : No 1 : Yes	0

도면43

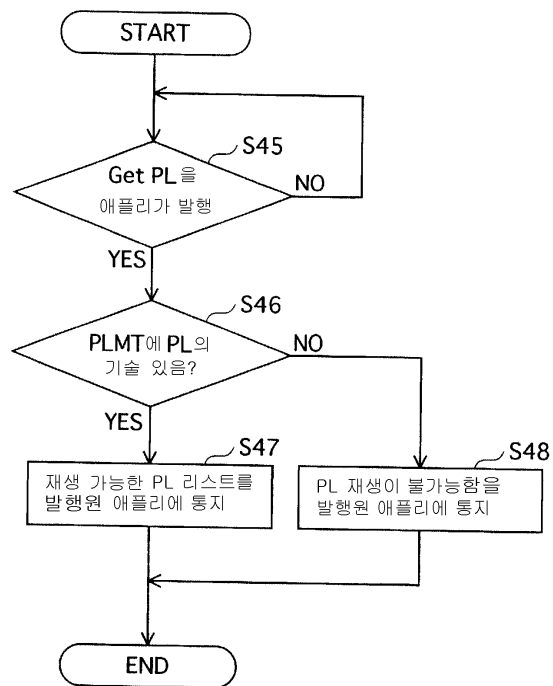


도면44

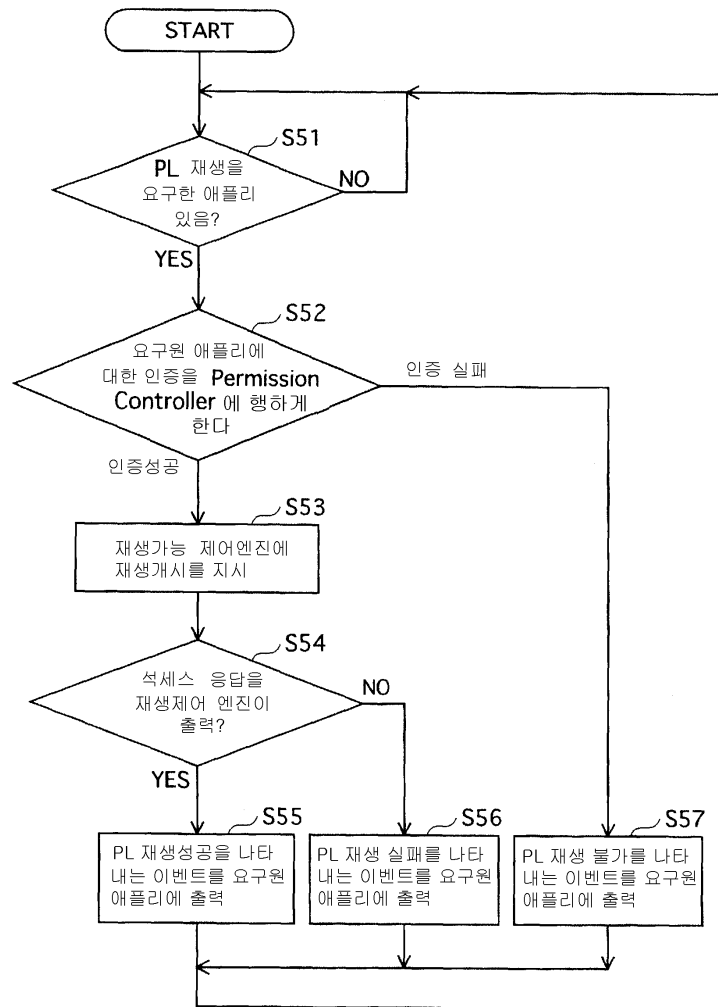




도면46



도면47



(a)

Parental Level에 의한 선택 알고리즘

PSR range	Play List_id_ref
$PSR(13) < 14$	Play List #1
$14 \leq PSR(13) < 18$	Play List #2
$18 \leq PSR(13)$	Play List #3

(b)

Language Code for Audio 에 의한 선택 알고리즘

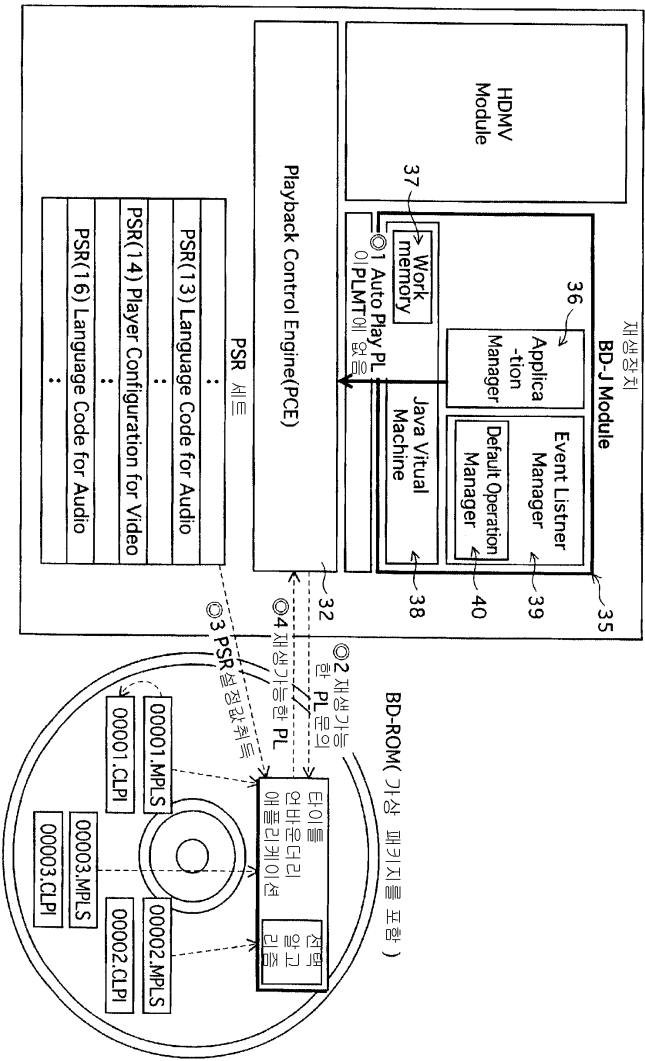
PSR range	Play List_id_ref
$PSR(16) = \text{English}$	Play List #1
$PSR(16) = \text{Japanese}$	Play List #2
$PSR(16) = \text{others}$	Play List #3

(c)

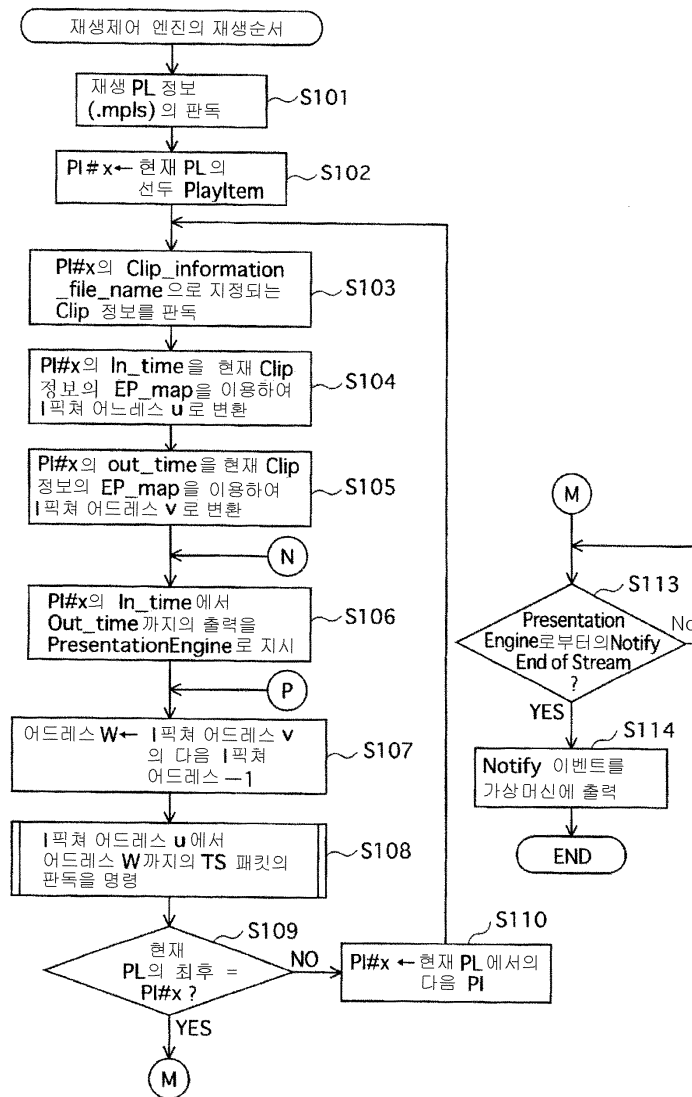
Player Configuration for Video 에 의한 선택 알고리즘

PSR range	Play List_id_ref
$PSR(14) = 525 \times 60 \text{ TV system } 4:3 \text{ Letter Box}$	Play List #1
$PSR(14) = 525 \times 60 \text{ TV system } 16:9$	Play List #2
$PSR(14) = 1920 \times 1080 \text{ TV system}$	Play List #4

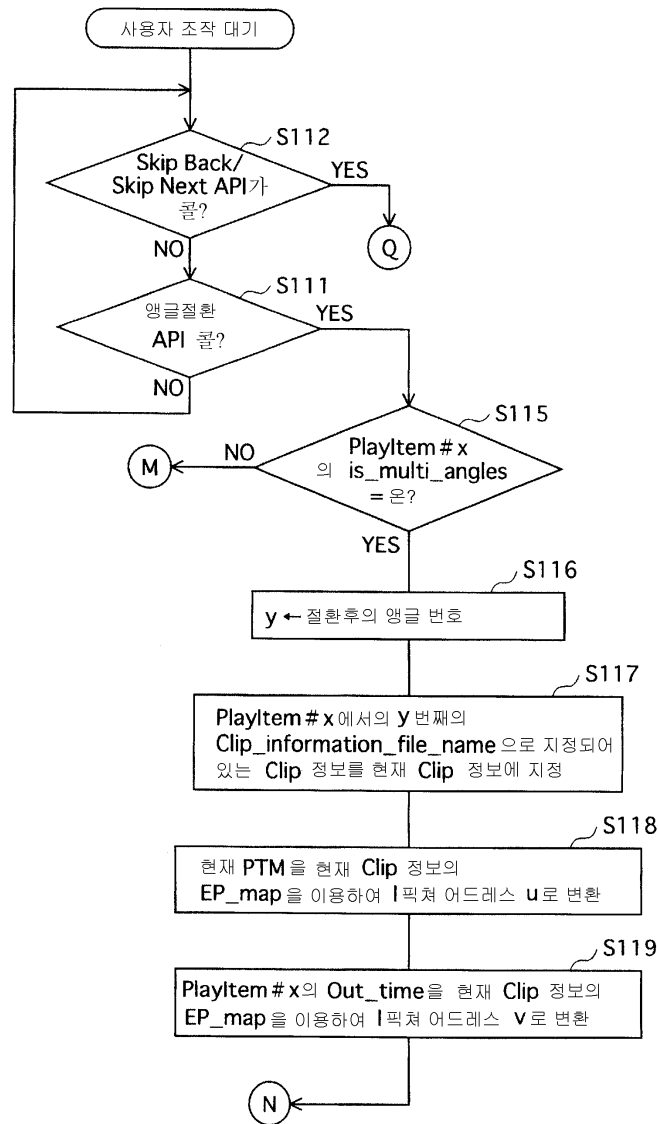
도면49



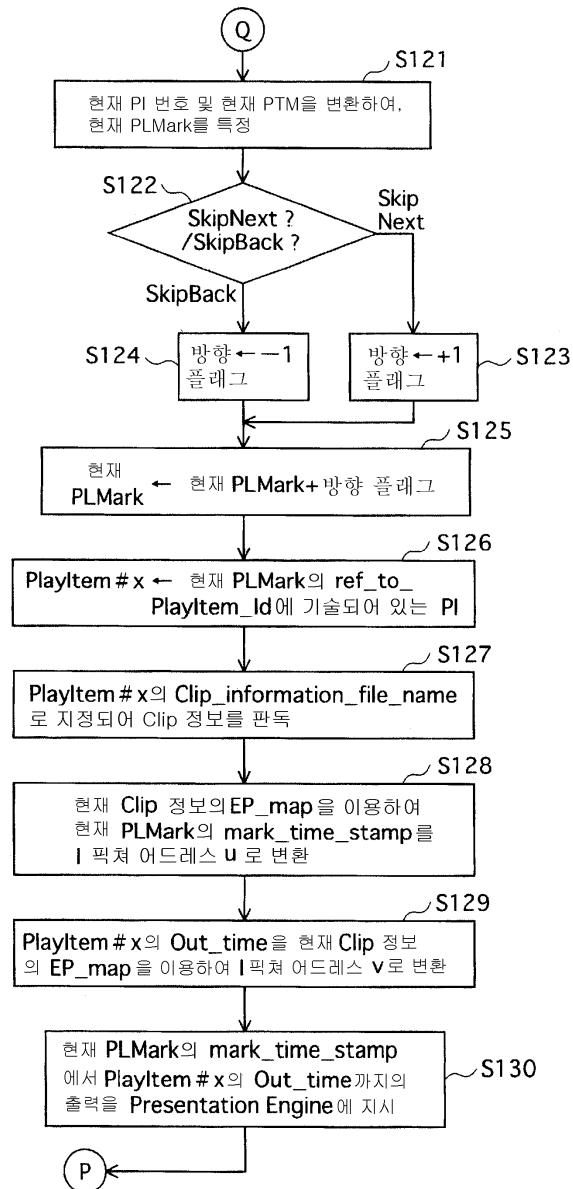
도면50



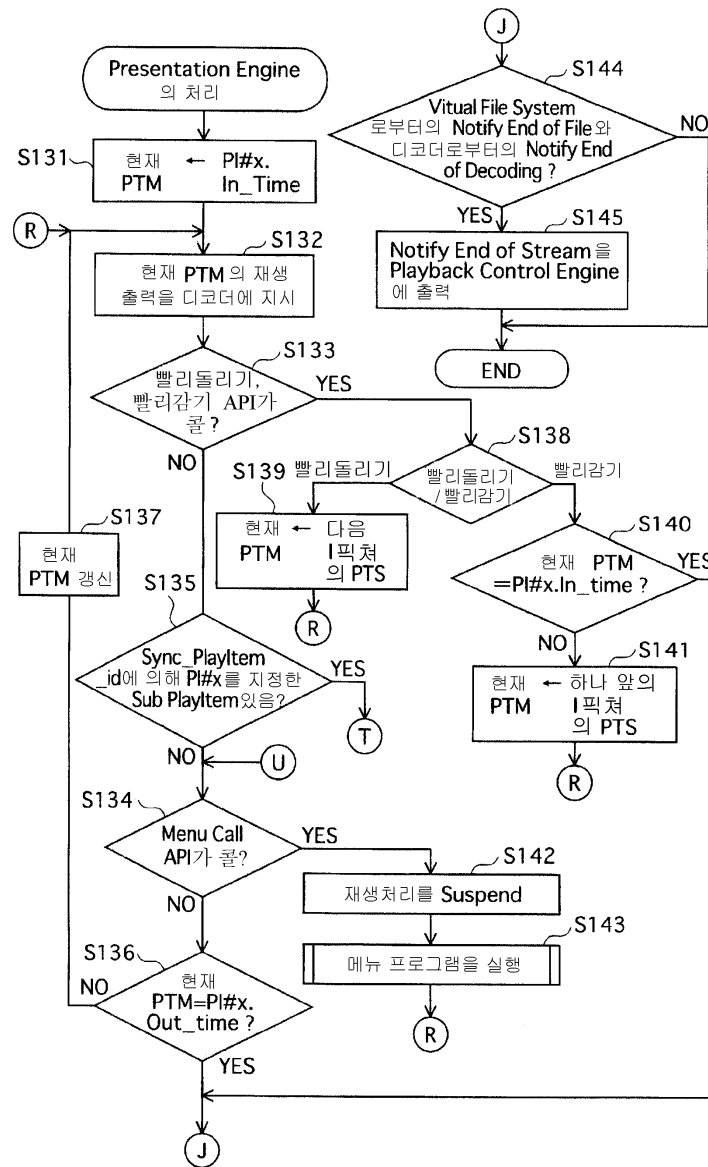
도면51



도면52



도면53



도면54

