(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 600 36 072 T2** 2008.05.21

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 131 919 B1**
(21) Deutsches Aktenzeichen: **600 36 072.5**
(86) PCT-Aktenzeichen: **PCT/EP00/05717**
(96) Europäisches Aktenzeichen: **00 940 392.4**
(87) PCT-Veröffentlichungs-Nr.: **WO 2001/001632**
(86) PCT-Anmeldetag: **21.06.2000**
(87) Veröffentlichungstag der PCT-Anmeldung: **04.01.2001**
(97) Erstveröffentlichung durch das EPA: **12.09.2001**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **22.08.2007**
(47) Veröffentlichungstag im Patentblatt: **21.05.2008**

(51) Int Cl.[8]: ***H04L 12/28*** (2006.01)

(30) Unionspriorität:
340272 **25.06.1999** **US**

(73) Patentinhaber:
Koninklijke Philips Electronics N.V., Eindhoven, NL

(74) Vertreter:
Volmer, G., Dipl.-Ing., Pat.-Anw., 52066 Aachen

(84) Benannte Vertragsstaaten:
DE, ES, FR, GB, IT

(72) Erfinder:
SHTEYN, Yevgeniy E., NL-5656 AA Eindhoven, NL

(54) Bezeichnung: **VERFAHREN ZUR BRÜCKENVERBINDUNG VON MEHREREN HEIMNETZSOFTWAREARCHITEKTUREN**

Beschreibung

ANWENDUNGSBEREICH DER ERFINDUNG

[0001] Die Erfindung bezieht sich auf ein System und ein Verfahren, die es Netzen mit möglicherweise unterschiedlicher Softwarearchitektur, wie einem HAVi-Heimnetz und einem auf Home-API basierenden oder einem auf JINI basierenden Heimnetz, erlauben zusammenzuarbeiten.

STAND DER TECHNIK

[0002] Heimnetze und entsprechende Softwarearchitekturen wie JINI von Sun Microsystems, Inc. (http://www.sun.com/jini), HAVi (http://www.havi.org), Home-API (http://www.homeapi.org) und Universal-Plug-and-Plag (UPnP) von Microsoft (http://www.upnp.org) stehen inzwischen für Anwendungsentwickler, Gerätehersteller und Diensteanbieter (Service Provider) zur Verfügung.

HAVi-Softwarearchitektur

[0003] HAVi betrifft einen Kern von Schnittstellen zur Anwendungsprogrammierung (eng. Application Programming Interfaces, API) für digitale Geräte der Unterhaltungselektronik in einem Heimnetz und stellt einen Standard für die Audio- bzw. Videoelektronik- und Multimedia-Industrie bereit. Eine API spezifiziert das Verfahren, das erforderlich ist, um Anfragen an ein Betriebssystem oder Anwendungsprogramm zu stellen. Das Heimnetz wird als verteilte Computerplattform angesehen. Das primäre Ziel des als HAVi-(Home Audio/Video interoperability)Architektur bezeichneten Standards besteht darin sicherzustellen, dass Produkte von verschiedenen Anbietern interoperieren, d. h. zusammenarbeiten können, um Anwendungsaufgaben auszuführen. Aktuelle Einrichtungen der Unterhaltungselektronik wie Einrichtungen der Heimunterhaltung (DVD-Player, DV-Camcorder, digitale Fernsehgeräte usw.) sind Systeme der digitalen Verarbeitung und digitalen Speicherung. Die Verbindung dieser Einrichtungen in Netzen ermöglicht eine gemeinsame Nutzung der Verarbeitungs- und Speicherressourcen. Dadurch kann die Steuerung von mehreren Einrichtungen der Unterhaltungselektronik gleichzeitig koordiniert werden, um beispielsweise die Benutzerinteraktion zu vereinfachen. Beispielsweise kann eine erste Einrichtung die Aufnahme auf einer zweiten Einrichtung instanzieren, während auf eine elektronische Programmzeitschrift (engl. Electronic Program Guide, EPG) auf einer dritten Einrichtung zugegriffen wird. Das Heimnetz liefert die Grundlage für die Verbindung der Einrichtungen der Unterhaltungselektronik. Es ermöglicht verbundenen Einrichtungen, sowohl Steuer-(wenn eine Einrichtung einen Befehl an eine andere sendet) als auch AV-(Audio/Video) Daten (wenn eine Einrichtung einen Audio- oder Videostrom an eine andere Einrichtung sendet) auszutauschen. Das Netz muss zu diesem Zweck verschiedene Anforderungen erfüllen. Es muss rechtzeitig den Transfer von AV-Datenströmen mit hoher Geschwindigkeit unterstützen. Das Netz muss die Selbstkonfiguration, die Selbstorganisation und das Hot Plug-and-Plag unterstützen und muss mit kostengünstiger Verkabelung und Schnittstellen auskommen.

[0004] Die HAVi-Softwarearchitektur ist plattformunabhängig und basiert auf Java. HAVi nutzt das Hochleistungs-IEEE-1394-Serienbusprotokoll für den Transport von Steuerung und Inhalt zwischen den mit dem Netz verbundenen Einrichtungen. Der IEEE-1394-Standard ist ein dynamisch konfigurierbares kostengünstiges digitales Netz. IEEE 1394 definiert sowohl eine Ausführung als physikalische so genannte Backplane als auch als verkabelter virtueller Punkt-zu-Punkt-Bus. Die Backplaneversion funktioniert mit 12,5, 25 oder 50 Mbit/s. Die Kabelversion unterstützt Transferraten von 100, 200 und 400 Mbit/s. Der Standard spezifiziert die Medien, die Topologie und das Protokoll. Das IEEE-1394-Transportprotokoll ist aufgrund seiner hohen Transferrate besonders nützlich bei der Unterstützung von Audio- und Videokommunikationsprotokollen.

[0005] Die HAVi-Architektur steuert die Einrichtungen der Unterhaltungselektronik in dem Netz durch abstrakte Darstellungen der Einrichtungen der Unterhaltungselektronik. Die abstrakten Darstellungen werden von einem Controller gesteuert und verbergen die Eigenheiten der zugehörigen reellen Einrichtung der Unterhaltungselektronik. Die abstrakte Darstellung bietet somit eine einheitliche Schnittstelle für Software höherer Ebenen. Die abstrakten Darstellungen werden mit ihren Steuereigenschaften registriert, die diejenigen der dargestellten Einrichtung wiedergeben. Die abstrakten Darstellungen machen den Anwendungen ihre so genannten Interoperabilitäts-API zugänglich und bilden gemeinsam einen Satz mit Diensten zum Einrichten von tragbaren verteilten Anwendungen in dem Heimnetz.

[0006] Die Architektur ermöglicht es einer Einrichtung, einen Befehl oder Steuerinformationen zu einer anderen Einrichtung in dem Heimnetz zu senden. Eine HAVi-konforme Einrichtung enthält Daten (die obige abstrakte Darstellung bezeichnet als Device Control Model oder DCM, siehe nachfolgende Erläuterung), die ihre Benutzerschnittstelle (beispielsweise grafische Benutzerschnittstelle, GUI) und ihre Steuerfähigkeiten betreffen. Diese Daten umfassen beispielsweise den HAVi-Bytecode (Java), der hinauf geladen und von anderen Einrichtungen im Netz ausgeführt werden kann. Eine HAVi-konforme Einrichtung verfügt als Minimum über ausreichende Funktionalität, um mit anderen Einrichtungen in dem System zu kommunizieren. Während der Interaktion können Einrichtungen Steuerung und Daten in einem so

genannten P2P-Netz (Peer-to-Peer-Netz) austauschen. Dadurch wird sichergestellt, dass auf der Kommunikationsebene keine der Einrichtungen als Master oder Controller des Systems zu agieren braucht. Andererseits erlaubt es einem logischen Master oder Controller, dem grundlegenden P2P-Kommunikationsmodell eine Steuerungsstruktur aufzuerlegen. HAVi unterscheidet zwischen Controller und gesteuerten Einrichtungen, wie nachfolgend erläutert wird. Ein Controller ist eine Einrichtung, die für eine gesteuerte Einrichtung als Host agiert. Ein Controller hostet die abstrakte Darstellung für die gesteuerte Einrichtung. Die Steuerschnittstelle wird über die API der abstrakten Darstellung zugänglich gemacht. Diese API ist der Zugangspunkt für Anwendungen zur Steuerung der Einrichtung.

[0007] HAVi-konforme Einrichtungen der Unterhaltungselektronik werden folgendermaßen kategorisiert: so genannte FAV-Einrichtungen (Full-AV), IAV-Einrichtungen (Intermediate-AV) und BAV-Einrichtungen (Base-AV).

[0008] Eine FAV-Einrichtung enthält einen vollständigen Satz mit Softwarekomponenten der HAVi-Softwarearchitektur (siehe unten). Eine FAV-Einrichtung ist dadurch gekennzeichnet, dass sie eine Laufzeitumgebung für den HAVi-Bytecode aufweist. Dadurch kann eine FAV-Einrichtung den Bytecode von anderen Einrichtungen hinauf laden, um beispielsweise erweiterte Fähigkeiten für ihre Steuerung bereitzustellen. Eine FAV-Einrichtung kann beispielsweise aus einer HAVi-konformen Set-Top-Box, einem HAVi-konformen digitalen Fernsehempfänger und einem Heim-PC gebildet werden. Ein intelligenter Fernsehempfänger kann beispielsweise der HAVi-Controller von anderen Einrichtungen sein, die in dem Netz verbunden sind. Der Empfänger erhält den Bytecode von einer anderen Einrichtung hinauf geladen, um eine Benutzerschnittstelle für diese Einrichtung und eine externe Steuerung dieser Einrichtung zu schaffen. Es kann veranlasst werden, dass ein diese Einrichtung darstellendes Symbol auf dem Fernsehbildschirm erscheint, und durch die Interaktion des Benutzers mit dem Symbol können Elemente des Steuerprogramms veranlasst werden, die dargestellte Einrichtung auf eine vorher spezifizierte Art zu bedienen.

[0009] Eine IAV-Einrichtung stellt keine Laufzeitumgebung für den HAVi-Bytecode bereit, kann jedoch eine native Unterstützung für die Steuerung bestimmter Einrichtungen in dem Heimnetz bieten. Eine IAV-Einrichtung umfasst integrierte Softwareelemente, die eine Schnittstelle für die Steuerung allgemeiner Funktionen der bestimmten Einrichtungen schaffen. Diese Softwareelemente brauchen nicht der HAVi-Bytecode zu sein und können als native Anwendungen in der IAV-Einrichtung ausgeführt werden, die native Schnittstellen für den Zugang zu anderen Einrichtungen nutzt.

[0010] Eine BAV-Einrichtung kann den hinaufladbaren HAVi-Bytecode bereitstellen, hostet jedoch keines der Softwareelemente der HAVi-Architektur. Eine BAV-Einrichtung ist durch eine FAV-Einrichtung mit Hilfe des hinaufgeladenen Bytecodes der ersteren steuerbar. Eine BAV-Einrichtung ist durch eine IAV-Einrichtung über den nativen Code steuerbar. Die Kommunikation zwischen einer FAV- oder einer IAV-Einrichtung einerseits und einer BAV-Einrichtung andererseits erfordert die Übersetzung des HAVi-Bytecodes zum und vom von der BAV-Einrichtung verwendeten Befehlsprotokoll.

[0011] Die Hauptsoftwareelemente, die in der Kernspezifikation der HAVi-Architektur enthalten sind, sind nachfolgend aufgelistet. Für eine ausführlichere Erläuterung dieser Elemente wird auf die HAVi-Spezifikation verwiesen, die durch Nennung als hierin aufgenommen betrachtet wird.

1) Ein so genannter 1394-Communications Media Manager (CMM) wirkt als Schnittstelle zwischen den anderen Softwareelementen und dem IEEE 1394.

2) Ein so genannter Eventmanager (EM) informiert die verschiedenen Softwareelemente über Ereignisse im Netz, wie Änderungen der Netzkonfiguration, die auftreten, wenn Geräte zum Netz hinzugefügt oder aus ihm entfernt werden.

3) Eine so genannte Registry (Registrierdatenbank) hält Informationen über die mit dem Netz verbundenen Geräte und die von ihnen gebotenen Funktionen aufrecht. Anwendungen können diese Informationen von der Registry erhalten.

4) Ein so genanntes Messaging System (MS) dient als API, die die Kommunikation zwischen den Softwareelementen der verschiedenen Geräte im Netz erleichtert. Das MS stellt für die HAVi-Softwareelemente Kommunikationsfunktionen bereit. Es ist unabhängig vom Netz und den Transportschichten. Ein MS ist in jeglicher FAV- und IAV-Einrichtung integriert. Das MS ist für die Zuordnung von Identifikatoren für die abstrakten Darstellungen in der FAV- oder IAV-Einrichtung zuständig. Diese Identifikatoren werden zuerst von den abstrakten Darstellungen dazu verwendet, sich in der FAV- oder IAV-Einrichtung zu registrieren. Dann werden sie von den abstrakten Darstellungen dazu verwendet, sich untereinander innerhalb des Heimnetzes zu identifizieren. Wenn eine erste abstrakte Darstellung eine Meldung an eine andere abstrakte Darstellung senden möchte, muss sie beim Aufrufen der Messaging-API den Identifikator der letzteren verwenden.

5) Ein so genanntes Device Control Module (DCM) stellt ein Gerät im Netzwerk dar. Anwendungsprogramme können direkt mit einem DCM interagieren. Dies schirmt sie vor den Eigenheiten jedes einzelnen Gerätes ab.

6) Ein so genannter DCM-Manager installiert die DCMs. Er reagiert automatisch auf Veränderungen im Netzwerk, indem er neue DCMs für neue Geräte installiert.
7) Ein so genannter Data Driven Interaction (DDI-) Controller gibt im Auftrag eines HAVi-Softwareelementes eine grafische Benutzerschnittstelle (GUI) auf der Anzeige eines Gerätes wieder. Er unterstützt ein großes Spektrum an Anzeigen, von der grafischen bis zur reinen Textanzeige. Ein DCM kann eine Benutzerschnittstelle (UI) bereitstellen. Der DCM kann die Benutzerschnittstelle auf einer oder mehreren Einrichtungen in dem Netz darstellen, die über eine Anzeige verfügen. Ein Mechanismus zum Erreichen dieses Ziels wird DDI (Data Driven Interaction) genannt. Mit Hilfe dieses Mechanismus kann ein DCM eine DDI-Beschreibung seiner Benutzerschnittstelle bieten. Die Beschreibung kann von jeglicher HAVi-konformen Einrichtung mit einer Anzeige abgerufen werden. Der Benutzer kann mit der Benutzerschnittstelle über die lokale Anzeige interagieren. Eine Benutzerinteraktion bewirkt das Senden von Meldungen an das zugeordnete DCM zur Steuerung der physikalischen Einrichtung, die durch das DCM dargestellt wird.
8) Ein so genannter Stream Manager (SMGR) erstellt Verbindungen und leitet Echtzeit-AV-Datenströme zwischen zwei und mehr Geräten im Netz weiter.

[0012] Die HAVi-Architektur spezifiziert mindestens zwei Ebenen der Interoperabilität, bezeichnet als Ebene 1 und Ebene 2.

[0013] Die Interoperabilität der Ebene 1 betrifft den allgemeinen Bedarf daran, existierenden Einrichtungen die Kommunikation auf einer grundlegenden Funktionalitätsebene zu ermöglichen. Zu diesem Zweck definiert und verwendet die Interoperabilität der Ebene 1 einen allgemeinen Satz mit Steuermeldungen (Befehlen), die eine Einrichtung befähigen, mit einer anderen Einrichtung zu kommunizieren, und einen Satz mit Ereignismeldungen, die es eigentlich von einer Einrichtung angesichts seiner Klasse (Fernseher, Videorecorder, DVD-Player usw.) erwarten sollte. Zur Unterstützung dieses Ansatzes ist ein grundlegender Satz mit Mechanismen erforderlich: Geräteerkennung, Kommunikation und ein HAVi-Meldungssatz.

[0014] Bezüglich der Geräteerkennung benötigt jede Einrichtung in dem Heimnetz ein genau definiertes Verfahren, das es ihr ermöglicht, ihre Fähigkeiten anderen mitzuteilen. Der HAVi-Ansatz besteht darin, die so genannten SDD-Daten (Self-Describing Data) zu nutzen. Die SDD-Daten werden in allen Einrichtungen im Netz benötigt. SDD-Daten umfassen als Minimum ausreichend Informationen, um die Instanzierung eines so genannten integrierten DCM zu ermöglichen. Ein integriertes DCM ist ein vorher in einer steuernden IAV- oder FAV-Einrichtung in einem plattformabhängigen Code installierter Teilcode, der native Schnittstellen nutzt, um auf die Ressourcen der IAV- oder FAV-Einrichtungen zuzugreifen. Wie oben erwähnt ist ein DCM für eine Einrichtung ein Softwareelement, das eine Schnittstelle zum Steuern von allgemeinen Funktionen der Einrichtungen schafft. Die Instanzierung eines integrierten DCM bewirkt die Registrierung der Fähigkeiten der Einrichtung in einer Registrierdatenbank. Die Registrierdatenbank stellt einen Verzeichnisdienst bereit und ermöglicht es jeglichem Objekt in dem Netz, ein anderes Objekt in dem Netz zu lokalisieren. Die Registrierung erlaubt es Anwendungen, den grundlegenden Satz mit Befehlsmeldungen abzuleiten, die zu einer bestimmten Einrichtung in dem Netz gesendet werden können.

[0015] Bezüglich der Kommunikation muss eine Anwendung, nachdem sie die Fähigkeiten einer Einrichtung ermittelt hat, auf diese Fähigkeiten zugreifen können. Dies erfordert eine allgemeine Kommunikationsfähigkeit, die es Anwendungen erlaubt, Anfragen an Einrichtungen auszugeben. Dieser Dienst wird von den HAVi-MS und DCMs bereitgestellt. Die Anwendung sendet HAVi-Meldungen an DCMs, und die DCMs führen dann selbst Kommunikation mit den Einrichtungen aus.

[0016] Bezüglich der HAVi-Meldungssätze ist zur Unterstützung der Interoperabilität der Ebene 1 ein genau definierter Satz mit Meldungen erforderlich, der von allen Einrichtungen einer betreffenden bekannten Klasse (beispielsweise der Klasse der Fernsehempfänger, der Klasse der Videorecorder, der Klasse der DVD-Player usw.) unterstützt werden muss. Dadurch wird sichergestellt, dass eine Einrichtung mit existierenden Einrichtungen sowie mit zukünftigen Einrichtungen unabhängig vom Hersteller zusammenarbeiten kann.

[0017] Diese drei grundlegenden Anforderungen unterstützen einen gewissen minimalen Grad der Interoperabilität. Da jegliche Einrichtung die Fähigkeiten einer anderen über die Registrierdatenbank abfragen kann, kann jegliche Einrichtung den von einer anderen Einrichtung unterstützten Meldungssatz ermitteln. Da Anwendungen Zugriff auf das MS haben, kann jegliche Einrichtung mit jeglicher anderen Einrichtung interagieren.

[0018] Die Interoperabilität der Ebene 1 stellt sicher, dass Einrichtungen auf einer grundlegenden Funktionalitätsebene zusammenarbeiten können. Es ist jedoch ein erweiterter Mechanismus erforderlich, der es einer Einrichtung auch erlaubt, mit anderen Einrichtungen mit jeglicher zusätzlichen Funktionalität zu kommunizieren, die nicht in den integrierten DCMs in einer FAV-Einrichtung vorliegt. Beispielsweise un-

terstützen integrierte DCMs eventuell nicht alle Funktionen von existierenden Produkten und werden wahrscheinlich auch nicht die vollständig neuen Funktionen von zukünftigen Produktkategorien unterstützen. Die Interoperabilität der Ebene 2 stellt diesen Mechanismus bereit. Zu diesem Zweck lässt die HAVi-Architektur hinaufladbare DCMs als Alternative zu den oben erwähnten integrierten DCMs zu. Das hinauf geladene DCM kann ein existierendes DCM in einer FAV-Einrichtung ersetzen. Ein hinaufladbares DCM kann von jeglicher geeigneten Quelle bereitgestellt werden, ein gängiges Verfahren besteht jedoch darin, das hinaufladbare DCM in den HAVi-SDD-Daten der BAV-Einrichtung zu platzieren und von der BAV-Einrichtung zur FAV-Einrichtung hinauf zu laden, wenn die BAV-Einrichtung mit dem Heimnetz verbunden ist. Da die HAVi-Architektur anbieterneutral ist, ist es notwendig, dass das hinauf geladene DCM in einer Vielzahl von FAV-Einrichtungen mit jeweils möglicherweise unterschiedlicher Hardwarearchitektur funktioniert. Zu diesem Zweck werden hinauf geladene DCMs im HAVi- (Java) Bytecode ausgeführt. Die Laufzeitumgebung des HAVi-Bytecodes in FAV-Einrichtungen unterstützt die Instanzierung und Ausführung von hinauf geladenen DCMs. Nachdem das DCM erstellt wurde und in einer FAV-Einrichtung läuft, kommuniziert es mit den BAV-Einrichtungen in der gleichen Weise wie oben beschrieben.

[0019] Die Effizienz der Interoperabilität der Ebene 2 wird deutlich, wenn man die Ressourcen betrachtet, die erforderlich sind, um auf die Funktionalität einer bestimmten Einrichtung zuzugreifen. Die Ebene 2 erlaubt es einer Einrichtung, über ein hinauf geladenes DCM gesteuert zu werden, das alle von der Einrichtung gebotenen Fähigkeiten zeigt, während zum Erzielen der gleichen Funktionalität in Ebene 1 dieses DCM irgendwo im Netz integriert sein müsste. Wenn beispielsweise eine neue Einrichtung zum Netz hinzugefügt wird, erfordert es die Ebene 1, dass mindestens eine andere Einrichtung ein integriertes DCM umfasst, das mit der neuen Einrichtung kompatibel ist. Zum Vergleich erfordert es die Ebene 2 lediglich, dass eine Einrichtung eine Laufzeitumgebung für das von der neuen Einrichtung erhaltene hinauf geladene DCM schafft.

[0020] Das Konzept des Hinaufladens und Ausführens des Bytecodes bietet außerdem die Möglichkeit für gerätespezifische Anwendungen, die so genannten Device Control Applications. Durch diese Anwendungen kann ein Gerätehersteller dem Benutzer die Möglichkeit bieten, spezielle Merkmale einer Einrichtung zu steuern, ohne dass alle Merkmale im HAVi genormt sein müssen. Die Anwendung wird von einem DCM im HAVi-Bytecode bereitgestellt und kann von jeder FAV-Einrichtung im Netz hinauf geladen und installiert werden.

[0021] Für weitere Informationen wird auf die frei zugängliche HAVi-Spezifikation und IEEE-1394-Spezifikation verwiesen. Die HAVi-Kernspezifikation wurde beispielsweise unter http://www.sv.philips.com/news/press ins Web gestellt und wird durch Nennung als hierin aufgenommen betrachtet.

Home-API-Architektur

[0022] Ein Home-API-System umfasst Softwaredienste und Schnittstellen zur Anwendungsprogrammierung (Application Programming Interface, API), die es auf einem PC laufenden Softwareanwendungen erlauben, steuerbare Einrichtungen, die in dem System registriert sind, zu erkennen und mit ihnen zu interagieren. Die Heimumgebung kann Einrichtungen wie Fernseher, Videorecorder, Set-Top-Boxen, Haussicherheitssysteme, Beleuchtung usw. einschließen. Home-APIs sind protokollunabhängig und verbergen Unterschiede in den zugrunde liegenden Netzen und den Protokollen der Kommunikation zwischen den Einrichtungen, die von den Softwareanwendungen transparent sind. Die Art des Zugriffs einer Anwendung auf eine Einrichtung ist einheitlich und unabhängig von dem zugrunde liegenden Protokoll, das für die Kommunikation mit der Einrichtung eingesetzt wird.

[0023] Softwareanwendungen interagieren mit den Einrichtungen, indem sie Eigenschaften von zur Darstellung dieser Einrichtungen erstellten Softwareobjekten einstellen oder erhalten. Anwendungen können auch an Ereignissen teilnehmen, die Änderungen der Eigenschaften einer Einrichtung betreffen, damit ihnen diese Änderungen gemeldet werden.

[0024] Die Home-API-Architektur umfasst Komponenten genannt Diensteanbieter (Service Provider). Diese Komponenten sind für entsprechende Einrichtungen und Netze bestimmt und dienen dazu, die Operationen auf hoher Ebene an den Eigenschaften des Softwareobjekts in Befehle umzusetzen, die über das Netz an die zugeordneten Einrichtungen gesendet werden. Die Diensteanbieter führen die Home-API-Softwareobjekte aus. Die Diensteanbieterkomponente ist außerdem für die Erzeugung von Ereignissen bei Eigenschaftsänderungen für die Objekte zuständig, die dem Diensteanbieter zugeordnet sind.

[0025] Im Detail setzt die Home-API ein Rechenmodell ein, das auf der so genannten Component-Object-Model-(COM/DCOM)Technologie von Microsoft basiert. Für nähere Informationen wird beispielsweise auf die Component Object Model Specification, Version 0.9 vom Oktober 1995, von Microsoft verwiesen, die durch Nennung als hierin aufgenommen betrachtet wird. COM ist objektorientiert. Ein Objekt hat Eigenschaften, die Steuerfunktionen einer zugeordneten elektronischen Einrichtung darstellen, wie sie einer Softwareanwendung zugänglich gemacht wer-

den. Eine Zustandsänderung eines Objekts infolge eines externen Ereignisses wird zur Softwareanwendung weitergeleitet. Die Anwendung wirkt auf die Objekte, indem sie ihre Eigenschaften verändert oder einstellt. Verändert die Anwendung eine Eigenschaft eines einer gewissen physikalischen Einrichtung zugeordneten Objekts, wird ein Befehl an die zugeordnete Einrichtung gesendet.

[0026] COM ist ein allgemeiner Mechanismus, der es Anwendungen erlaubt konsistent zu kommunizieren und ein Rahmen für die Entwicklung und Unterstützung von Programmkomponentenobjekten. Es bietet ähnliche Fähigkeiten wie in CORBA (Common Object Request Broker Architecture) definiert, dem Rahmen für das Zusammenwirken von verteilten Objekten in einem Netz. OLE (Object Linking and Embedding) stellt Dienste für das Verbunddokument bereit, das der Benutzer auf seiner Anzeige sieht, COM liefert die zugrunde liegenden Dienste der Schnittstellenabstimmung und die Ereignisdienste (indem es ein Objekt infolge eines bei einem anderen Objekt geschehenen Ereignisses in Betrieb setzt). Bei dieser Ausführung werden Clients als OLE-Automatisierungsobjekte (abstrakte Darstellungen) modelliert, die Eigenschaften verwenden, um Steuerungen und Ereignisse für Signalzustandsänderungen zugänglich zu machen. Die OLE-Automatisierung ist eine COM-Technologie, die das Scripting und die späte Bindung von Clients an Server erlaubt. Die OLE-Automatisierung schafft die Kommunikation mit anderen Programmen durch das Aufrufen von Merkmalen (Befehle und Anfragen), die die Programme für die externe Verwendung verfügbar gemacht haben. Vor dem Einsatz eines Objekts muss eine Client-Anwendung als Erstes den Schnittstellenzeiger des Objekts erhalten. Der Schnittstellenzeiger ist über das Verzeichnis des Netzes verfügbar, indem der Name des Objekts gebunden wird oder die Einrichtungen aufgezählt werden. Es können Standard-COM-APIs für die Moniker-Bindung eingesetzt werden. Verweise auf Objekte sind durch Aufrufen von GetObject oder CoGetObject mit einem String verfügbar, der den Namen oder die Identität der gewünschten Einrichtung spezifiziert. Die Anwendung kann dann auf das Objekt wirken, indem sie seine Eigenschaften durch Aufrufe von „set property" auf die geeigneten Eigenschaften einstellt oder sie abruft. Wenn eine Anwendung eine Eigenschaft eines mit einer Einrichtung korrespondierenden Objekts einstellt oder verändert, wird die Operation der Einstellung der Eigenschaft oder die Operation der Veränderung in einen Befehl umgewandelt, der über das Netz zu der betreffenden Einrichtung gesendet wird. Die Objekte können sich in der Ausführung unterscheiden, weisen jedoch ein ähnliches, auf Eigenschaften basierendes Modell für Client-Anwendungen auf, die auf einem Controller, beispielsweise einem PC mit einem auf Windows basierenden Betriebssystem, laufen.

JINI-Architektur

[0027] JINI vereinfacht die Verbindung und die gemeinsame Nutzung von Einrichtungen in einem Netz. Bei herkömmlichen Systemen erfordert das Hinzufügen einer Einrichtung zu einem PC oder einem Netz die Installation und das Hochfahren. Bei JINI gibt die Einrichtung ihre Existenz bekannt und liefert Informationen über ihre Funktionen. Danach wird die Einrichtung zugänglich für andere Ressourcen in dem Netz. Diese Technologie ermöglicht ein verteiltes Rechnen – die Fähigkeiten werden von den Ressourcen des Netzes gemeinsam genutzt.

[0028] JINI legt seinen Schwerpunkt auf den Prozess des Hinzufügens einer Einrichtung zu dem Netz und des Verbreitens von Informationen über die Einrichtung an andere Maschinen. Auf diese Weise bietet JINI einen so genannten Suchdienst (Look-Up-Service), der es Anwendungen auf anderen Maschinen erlaubt, die neu hinzugefügte Einrichtung einzusetzen. Der Ansatz von JINI setzt voraus, dass das Netz und das Betriebssystem bereits konfiguriert wurden, so dass jeder Computer bereits von den anderen Computer weiß. Die Funktionalität von JINI erfolgt in einer Schicht oberhalb des Netzes. Es löst beispielsweise nicht die Probleme der automatischen Konfiguration des Netzes nach der Verbindung, der Trennung oder der erneuten Verbindung. Es setzt voraus, dass das Netz unabhängig von JINI verbunden oder unterbrochen ist. JINI setzt die vom Netz gebotenen Dienste wirksam ein, um seine Dienste auszuführen.

[0029] Im Besonderen stellt die JINI-Netzinfrastruktur Ressourcen bereit zum Ausführen von Java-Programmiersprachenobjekten, Kommunikationsfähigkeiten zwischen diesen Objekten und die Fähigkeit, Dienste im Netz zu finden und zu nutzen. Durch den Einsatz der Java Remote Method Invocation (RMITM) stellt JINI die Kommunikation zwischen Objekten über Einrichtungsgrenzen hinweg bereit und ermöglicht es diesen Objekten somit zusammenzuarbeiten. RMI ermöglicht die Aktivierung von Objekten und den Einsatz von Multicast zum Kontaktieren von replizierten Objekten, wodurch Objekte mit hoher Verfügbarkeit und hoher Zuverlässigkeit leicht im JINI-Framework ausgeführt werden können. RMI ist eine Erweiterung für klassische Mechanismen des Prozedur-Fernaufrufs (Remote Procedure Call, RPC). RMI ermöglicht es, nicht nur Daten von Objekt zu Objekt sondern vollständige Objekte einschließlich Code im Netz weiterzuleiten. Ein Großteil der Einfachheit des JINI-Systems ist auf diese Fähigkeit zurückzuführen, Codes in einem Netz in einer wie ein Objekt verkapselten Form weiterzuleiten. JINI bietet einen Suchdienst, der das Auffinden von Diensten ermöglicht, die durch die Kommunikationsinfrastruktur verbunden sind. JINI bietet ferner einen Mechanismus bezeichnet als Discovery/Join für JINI-fähige

Einrichtungen (beispielsweise Festplattenlaufwerke, Drucker und Computer) zum Erkennen des geeigneten Suchdienstes und das Verbinden mit dem Gesamtsystem. Wenn eine Einrichtung mit einem JINI-basierten System verbunden wird, werden seine Dienste zu diesem Suchdienst hinzugefügt. In gleicher Weise werden, wenn eine JINI-fähige Einrichtung das System verlässt, beispielsweise weil sie entfernt oder unzuverlässig wird, ihre Dienste aus dem Suchdienst gelöscht.

[0030] Für mehr Informationen über Heimnetze, insbesondere über HAVi, den Einsatz der COM-Technologie und der OLE-Automatisierungsobjekte und Home-API und über JINI, wird auf die folgenden Dokumente verwiesen, die durch Nennung als hierin aufgenommen betrachtet werden: die frei zugänglichen Spezifikationen zur HAVi-Architektur (beispielsweise Version 0.86), die Spezifikationen der Component Object Model Specification (beispielsweise Version 0.9), das von der Home-API Working Group veröffentlichte Home-API White Paper vom März 1999, der Überblick über die JINI-Architektur von Sun Microsystems Inc. (einschließlich der Java Remote Invocation Specification, der Java Object Serialization Specification, der JavaSpaces Specification usw.).

[0031] Jede der speziellen Softwarearchitekturen von HAVi, Home-API, JINI usw. weist ihre eigenen Vorteile und Nachteile auf.

[0032] Beispielsweise erlaubt es die JINI-Architektur, dass Java-Objekte, die von einer Netzplattformumgebung zu einer anderen übertragen werden, lokal laufen. Andererseits setzt sie plattformspezifische Merkmale nicht immer wirksam ein. Ferner ist JINI, da sie auf Java, einer interpretierten Sprache basiert, nicht so leistungsfähig wie ein kompilierter nativer Code.

[0033] Die HAVi-Architektur ist für den Einsatz von IEEE-1394-Audio- bzw. Videoausrüstungen mit hoher Bandbreite ausgelegt. Sie bietet die Möglichkeit der Erweiterung durch von Einrichtungen herunter ladbare Softwarekomponenten (DDI, Java usw.) und weitere nützliche Merkmale. Andererseits können Einrichtungen ohne IEEE-1394-Verbindung oder -Schnittstelle nicht einfach von einer HAVi-Anwendung gesteuert werden.

[0034] Die Home-API-Architektur setzt wirksam die Windows COM/DCOM-Softwarearchitektur und -Dienste ein, ist jedoch noch nicht im großen Rahmen in anderen Betriebssystemen wie UNIX, LINUX, Apple Mac OS usw. verfügbar.

[0035] Da Rechen- und Steuereinrichtungen wie PCs, Set-Top-Boxen, digitale Fernseher, Videorecorder, X10-Module usw. von Verbrauchern zu verschiedenen Zeitpunkten für verschiedene Zwecke erworben werden, gewinnt es an Bedeutung, Lösungen für die Brückenverbindung von Heimnetzen und Architekturen zu entwickeln. Bei derartigen Softwarelösungen ist es höchst wahrscheinlich, dass sie auf relativ leistungsfähigen Rechenplattformen wie PCs, Set-Top-Boxen, Videospielmaschinen usw. erscheinen. In diesem Zusammenhang dominierten PCs zum Preis von $ 1.000 oder weniger im Laufe von 1998 und in das Jahr 1999 hinein im PC-Einzelhandel immer mehr, wobei die größte Zunahme in dem Preissegment unter $ 600 zu beobachten ist. Die PC-Hersteller Emachines, Packard-Bell und NEC sind aktuell (1999) die Hauptlieferanten für dieses Marktsegment. Emachines bietet beispielsweise PCs unter $ 600 an, die Hochleistungselemente wie DVD-Laufwerke, schnelle 400-Mhz-Intelprozessoren und 32-MB-RAMs aufweisen. Ähnlich Trends sind im Bereich der Videospiele zu beobachten, wo Sega und andere Firmen planen, 64-Bit-Spielkonsolen auf den Markt zu bringen. Die meisten der neuen Einrichtungen enthalten Modems und andere Verbindungsoptionen, die ihnen das Einbinden in ein Netz ermöglichen.

[0036] Jede der oben genannten bekannten Softwarearchitekturen bietet einen Dienst, der die Erkennung von Einrichtungen oder Diensten im Netz abstrakt gesehen auf mehr oder weniger ähnliche Weise ermöglicht. Als Endergebnis eines Erkennungsprozesses wird eine Softwaredarstellung einer Einrichtung oder eines Dienstes in einem Suchdienst (oder einer Registry oder einem Verzeichnis) platziert. Die HAVi-Architektur registriert die erkannten Einrichtungen oder Dienste in der Registry, die JINI-Architektur registriert sie in einem Suchdienst, Home-API bezieht sich auf den Dienst als Verzeichnis. Danach werden die registrierten Einrichtungen oder Dienste Softwareanwendungen zur Verfügung gestellt, die auf dem Host laufen. Eine Anwendung lokalisiert die Softwaredarstellung oder das Objekt und nutzt sie oder es, um auf Schnittstellen und Reservierungsprozeduren der betreffenden Softwarearchitektur zuzugreifen.

AUFGABE DER ERFINDUNG

[0037] Es entsteht ein Problem, wenn eine Steuereinrichtung mehr als eine Netzumgebung hostet. Beispielsweise kann ein PC mit einer IEEE-1394-Verbindung Teil einer HAVi- und einer Home-API-Umgebung sein. Daher existiert ein Potenzial dafür, dass auf HAVi-Audio- bzw. Videoeinrichtungen von Home-API-Anwendungen zugegriffen werden kann und umgekehrt Home-API-steuerbare Einrichtungen von HAVi-Anwendungen gesteuert werden usw. Wenn dieses Potenzial in der Praxis umgesetzt würde, würde der Benutzer diese beiden Heimnetzumgebungen als eine wahrnehmen, die Nutzung würde erleichtert und die Zugangsoptionen verbessert. Es würde außerdem Softwareentwicklern erlauben, Anwendun-

gen für ein größeres Spektrum von steuerbaren Einrichtungen zu entwickeln, als es bisher möglich war.

[0038] Eine weitere Vereinfachung einer derartigen hybriden Funktionalität ist notwendig, damit existierende und zukünftige Heimnetzarchitekturen und -anwendungen berücksichtigt werden können.

[0039] Dementsprechend liegt der Erfindung die Aufgabe zugrunde, ein Verfahren zu schaffen, das die Brückenverbindung zwischen verschiedenen Heimnetzumgebungen und dadurch eine Erhöhung der Funktionalität des Heimnetzes als Ganzes ermöglicht.

ZUSAMMENFASSUNG DER ERFINDUNG

[0040] Das erfindungsgemäße Verfahren nutzt eine Softwarekomponente (Reference Factory) in einer Netzumgebung. Die Komponente erkennt Softwaredarstellungen von Einrichtungen oder Diensten, die in einem ersten Netz der Umgebung zur Verfügung stehen. Dies kann durch Aufzählung und/oder Überwachung eines Home-API-Verzeichnisses, einer HAVi-Registrierdatenbank, einer JINI-Registrierdatenbank oder einem funktionell gleichwertigen bestandsbezogenen Dienst in dem ersten Netz erfolgen. Nach Erkennung einer neuen Softwaredarstellung durch die so genannte Reference Factory erstellt letztere einen Verweis, der der in dem ersten Netz erkannten Softwaredarstellung zuzuordnen ist. Ein derartiger Verweis umfasst Informationen, beispielsweise eine Klassenidentität, eine so genannte URL (Uniform Ressource Locator), einen eindeutigen Objektidentifikator, einen so genannten XML- oder DDI-Descriptor usw., die für die Erstellung einer zumindest teilweise funktionellen Darstellung der Einrichtung oder des Dienstes innerhalb einer anderen Netzsoftwareumgebung erforderlich sind.

[0041] Auf die Verweisinformationen wird von einer anderen Softwarekomponente (Object Factory) zugegriffen, die in der Lage ist, derartige Softwaredarstellungen (Objekte) zu erstellen und für das andere Netz verfügbar zu machen. Die so genannte Objekt Factory erstellt falls erforderlich ein Objekt oder ruft ein vorher erstelltes Objekt von beispielsweise einer durch ihre URL gegebenen Website ab oder leitet den Verweis entsprechend den Regeln und/oder Präferenzen der Netzumgebung, mit der sie interagiert, weiter. Diese Präferenzen spiegeln beispielsweise allgemeine Architekturrichtlinien oder bestimmte Benutzerinteressen wieder. Beispielsweise ist lediglich eine gewisse Art von Einrichtungen (Beleuchtung) von Interesse für den Benutzer. Ein weiteres Beispiel ist eine HAVi-Umgebung, in der basierend auf der Netzkonfiguration lediglich DDI-Darstellungen zulässig sind.

[0042] In einer betreffenden Netzsoftwarearchitektur können mehrere Reference Factories existieren. Jede von ihnen kann für eine bestimmte Art von Verweisen wie für andere Netzsoftwareumgebungen (JINI, HAVi, Home-API, UPnP) oder andere Objektdarstellungen in einem derartigen Netz (beispielsweise für HAVi: Java DCM, DDI-Daten oder native DCM) oder ein Klasse von Einrichtungen bzw. Diensten (Datenspeicherung, Heimautomatisierung, A/V-Befehlssatz usw.) zuständig sein.

[0043] Die Erfindung basiert auf der Erkenntnis, dass zwei Prozesse getrennt werden sollten – die Analyse der Netzsoftwarekonfiguration einerseits und die Erstellung von für die Steuerung und/oder Interaktion erforderlichen Softwaredarstellungen (Objekten) andererseits. Factory-Verfahren der Objekterstellung sind in der Technik bekannt, siehe beispielsweise „Design Patterns: Elements of Reusable Object-Oriented Software", Addison-Wesley Professional Computing, von Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides, erschienen im Oktober 1994 bei Addison-Wesley Pub. Co., ISBN 0201633612, insbesondere die Seiten 81–116.

[0044] Im Besonderen betrifft die Erfindung ein Verfahren, das es einem ersten Netz mit einer ersten Softwarearchitektur ermöglicht, mit einem zweiten Netz mit einer zweiten Softwarearchitektur zu interagieren. Das Verfahren umfasst folgende Schritte: Verwenden eines Suchdienstes des ersten Netzes zum Erkennen einer ersten Softwaredarstellung einer Einrichtung oder eines Dienstes in dem ersten Netz; Erstellen eines Verweises für die erste erkannte Softwaredarstellung; Speichern des Verweises in einem Repositorium und Zugreifen auf das Repositorium zum Bestimmen einer zumindest teilweise funktionell gleichwertigen zweiten Softwaredarstellung basierend auf dem Verweis, wobei die zweite Softwaredarstellung von dem zweiten Netz aus zugänglich ist. Das erste und das zweite Netz können identische Softwarearchitekturen aufweisen, beispielsweise auf HAVi basieren, oder das erste und das zweite Netz können verschiedene Softwarearchitekturen aufweisen: beispielsweise basiert eines der ersten und zweiten Netze auf einer HAVi-Architektur und das andere auf einer Home-API-Architektur, eines der ersten und zweiten Netze auf einer HAVi-Architektur und das andere auf einer JINI-Architektur, eines der ersten und zweiten Netze basiert auf einer HAVi-Architektur und das andere auf einer UPnP-Architektur, eines der ersten und zweiten Netze basiert auf einer JINI-Architektur und das andere auf einer Home-API-Architektur, eines der ersten und zweiten Netze basiert auf einer JINI-Architektur und das andere auf einer UPnP-Architektur, eines der ersten und zweiten Netze basiert auf einer UPnP-Architektur und das andere Netz basiert auf einer Home-API-Architektur usw.

[0045] Jedes der oben genannten Beispiele für Soft-

warearchitekturen verfügt über einen Erkennungsdienst, der es ermöglicht, einen Bestand von in dem betreffenden Netz registrierten Einrichtungen und/oder Diensten zu erstellen. Das erfindungsgemäße Verfahren nutzt diesen Erkennungsdienst und die Funktionalität der Registrierdatenbank bzw. des Verzeichnisses bzw. des Suchdienstes, um die Verfügbarkeiten erkennen zu können, indem es diesen Bestand abfragt. Danach umfasst das Erstellen das Extrahieren von Informationen über die erste Softwaredarstellung, die aus Sicht des zweiten Netzes von Bedeutung für seine Funktionalität ist, und das Bestimmen umfasst das Erstellen einer zweiten Softwaredarstellung basierend auf den extrahierten Informationen.

[0046] Auf die folgenden Patentdokumente wird verwiesen, um die Erfindung in die richtige Perspektive zu rücken:
In dem Dokument EP-A 0 849 913 wird ein Datenkommunikationssystem dargelegt, das Folgendes umfasst: einen ersten Bus zum bidirektionalen Übertragen von digitalen Daten und Befehlen, einen zweiten Bus zum Übertragen von Befehlen zum Steuern von elektronischen Geräten hauptsächlich im Hausgebrauch und eine Verbindungseinrichtung zum Verbinden des ersten und des zweiten Busses. Die Verbindungseinrichtung wandelt ein Paketformat eines asynchronen Übertragungsmodus des ersten Busses in ein Paketformat des zweiten Busses um und wandelt das Paketformat des zweiten Busses in das Paketformat des asynchronen Übertragungsmodus des ersten Busses um.

[0047] In dem Dokument WO 96/27357 wird Folgendes dargelegt: In einem lokalen Kommunikationssystem mit einer Anzahl von durch einen ersten Datenbus, der einen ersten Satz mit Kommunikationsprotokollen unterstützt, verbundenen Einrichtungen und mindestens einer weiteren mit einem zweiten Bus, der diese Protokolle nicht unterstützt, verbundenen Einrichtung wird, eine Gateway-Einrichtung geschaffen, die den ersten und den zweiten Datenbus verbinden und dadurch eine Kommunikation zwischen ihnen ermöglicht. Der erste Satz mit Protokollen spezifiziert eine maximale Zeitspanne für die Antwort einer ersten Einrichtung auf eine von einer zweiten Einrichtung gesendete Anfrage. Wenn eine Anfrage von einer Einrichtung auf dem ersten Bus zu der weiteren Einrichtung gesendet wird, nimmt der Gateway die Zeitsteuerung für die Anfrage vor und erzeugt und sendet eine temporäre Antwort an die anfragende Einrichtung, wenn von der weiteren Einrichtung keine Antwort innerhalb der spezifizierten maximalen Antwortzeitspanne empfangen wird. Das System kann zwei oder mehr Cluster mit Einrichtungen umfassen, die jeweils durch entsprechende Gateway-Einrichtungen mit dem weiteren Bus verbunden sind.

[0048] In dem am 12.06.2000 veröffentlichten Dokument EP-A 1 058 422 werden Verfahren zur Brückenverbindung von HAVi- und UPnP-Teilnetzen mit Hilfe einer Brückeneinrichtung dargelegt, die sowohl HAVi- als auch UPnP-fähig ist.

[0049] Keines dieser Dokumente behandelt oder schlägt die Brückenbildung von Netzen mit unterschiedlichen Softwarearchitekturen wie oben angegeben, basierend auf dem Speichern eines Verweises einer Softwaredarstellung für eine Einrichtung oder einen Dienste in einem der Netze in einem Repositorium und dem Bestimmen einer zweiten Softwaredarstellung beim Zugriff auf das Repositorium von dem anderen Netz aus vor, die funktionell zumindest teilweise gleichwertig mit der ersten Darstellung ist und in dem anderen Netz eingesetzt werden kann.

[0050] Die folgenden Patentdokumente bieten eventuell weitere Informationen in diesem Zusammenhang:

– Die US-amerikanische Patentschrift 6.159.136 (Aktenzeichen des Patentanwalts PHA 23.942), eingereicht am 02.09.1998 für Yevgeniy Shteyn, mit dem Titel „Low Data-Rate Network Represented an High Data-Rate HAVi-Network". Dieses Dokument betrifft ein auf einem PC basierendes Heimautomatisierungssystem, das eine Transportschicht mit geringer Transferrate und auf COM basierende Softwarekomponenten wie Home-API für die Steuerung von Einrichtungen in einem Heimautomatisierungsnetz nutzt. Das Heimautomatisierungssystem wird mit einem auf Messaging basierenden HAVi-Netz gemischt, das IEEE 1394 als Transportschicht mit hoher Transferrate einsetzt. Das HAVi-Netz steuert die Audio- bzw. Videoausstattung in einem Heimunterhaltungssystem. Die Heimautomatisierungsdienste und -einrichtungen werden als HAVi-konforme Elemente in den FAV- oder IAV-Controllern des HAVi-Netzes registriert. Die Heimautomatisierungsressourcen (Einrichtungen und Dienste) verfügen sowohl über COMOLE-Automatisierungsschnittstellen als auch über HAVi-konforme Schnittstellen, die die Steuerung des Heimautomatisierungssystems von dem HAVi-Netz aus erlauben.

– Die US-amerikanische Patentschrift 6.163.817 (Aktenzeichen des Patentanwalts PHA 23.438), eingereicht am 30.06.1998 für Yevgeniy Shteyn und Gregory Gewickey, mit dem Titel „Dynamic De-registering of Devices in System with Multiple Communication Protocols". Dieses Dokument betrifft ein Informationsverarbeitungssystem mit einem ersten und einem zweiten elektronischen Teilsystem und Steuermitteln zum Steuern der Teilsysteme. Mindestens das erste Teilsystem verfügt über eine Softwaredarstellung, die in den Steuermitteln registriert ist. Die Steuermittel ändern den Zustand des ersten Teilsystems durch

Interaktion mit der Softwaredarstellung. Das erste und das zweite Teilsystem sind auch in der Lage, direkt miteinander zu interagieren, ohne dass die Steuermittel beteiligt sind. Zur Vermeidung von Konflikten ist zumindest das erste Teilsystem in der Lage, sich aus der Registrierung in den Steuermitteln auszutragen und so funktionell seine Softwaredarstellung in den Steuermitteln zu deaktivieren.

– Die US-amerikanische Patentschrift 5.959.536 (Aktenzeichen des Patentanwalts PHA 23.169), eingereicht am 15.10.1996 für Paul Chambers und Saurabh Srivastava, mit dem Titel „Task-driven Distributed Multimedia Consumer System". Dieses Dokument betrifft ein Steuersystem mit mehreren Einrichtungen der Unterhaltungselektronik und aufgabengesteuerten, mit den Einrichtungen verbundenen Steuermitteln zum Steuern der Interaktion zwischen den Einrichtungen. Die Steuermittel wirken auf entsprechende Softwaredarstellungen jeder entsprechenden Unterhaltungseinrichtung. Durch Verkapseln der variablen Komplexität der Aufgabe (Task) in einer Softwaredarstellung kann sie so einfach oder so hoch entwickelt wie nötig gemacht werden, um die Fähigkeiten auf eine gemeinsame Ebene zu bringen. Da die Schnittstellenebene für alle Einrichtungen die gleiche ist, können Anwendungen Einrichtungen einheitlich handhaben, die sehr verschiedene Ebenen der Entwicklung aufweisen.

– Die US-amerikanische Patentschrift 6.480.473 (Aktenzeichen des Patentanwalts PHA 23.405), eingereicht am 29.12.1998 für Paul Chambers und Steven Curry, mit dem Titel „Verification of Active Nodes in Open Networks". Dieses Dokument betrifft ein Netzabrufprotokoll (Network Polling Protocol), das das Netz als logischen Ring oder lineare Sequenz von miteinander verbundenen Knoten behandelt. Eine Poll-Abfrage wird einfach einen Knoten nach dem anderen nach unten oder in dem Netz weitergeleitet, bis ein vollständiger Bestand aktiver Knoten ermittelt wurde. Die Protokolle umfassen auch Prozeduren zum Kurieren oder Reparieren von Unterbrechungen im Verbindungsprotokoll und zum Hinzufügen neuer Knoten zum Verbindungsprotokoll. Das Verbindungsprotokoll kann auch dazu verwendet werden, hierarchisch verknüpfte Netze einzurichten, in denen Hierarchien der höchsten Ebene Adressen eines dauerhaften Gliedes eines verknüpften Netzes umfassen und Hierarchien der untersten Ebene ein gegebenes verknüpftes Netz sind.

– Die US-amerikanische Patentschrift 6.314.459 (Aktenzeichen des Patentanwalts PHA 23.488), eingereicht am 13.08.1998 für Lawrence Freeman, mit dem Titel "Home-Network Autoconfiguration". Dieses Dokument betrifft die automatische Konfigurierung von zwei PCs in einem Netz zur gemeinsamen Nutzung von Ressourcen, die in den einzelnen PCs registriert sind. Lokale Dienste und Ressourcen in einem PC werden bei dem anderen PC registriert und umgekehrt. Die Registrierdatenbank verbirgt, ob ein Dienst oder eine Ressource entfernt oder lokal ist. Beim Betrieb des Netzes ist eine lokale Ressource oder ein lokaler Dienst eines PCs von dem entfernten PC so adressierbar, als wäre sie/er lokal bei letzterem vorhanden. Ein Heimnetz mit PCs wird auf diese Weise automatisch konfiguriert.

– Die US-amerikanische Patentschrift 6.918.123 (Aktenzeichen des Patentanwalts PHA 23.483), eingereicht am 02.10.1998 für Yevgeniy Shteyn, mit dem Titel „Calls Identify Scenario for Control of Software Objects via Property Routes". Dieses Dokument betrifft im Besonderen, jedoch nicht ausschließlich, Home-API und ein Informationsverarbeitungssystem mit ersten und zweiten physikalischen Komponenten, die durch erste und zweite Softwareobjekte dargestellt werden. Beide Objekte weisen Eigenschaften auf, die durch Aufrufe an die Objekte verändert werden können. Das System ermöglicht die Registrierung einer Eigenschaftsroute, die eine erste Eigenschaft des ersten Objekts mit einer zweiten Eigenschaft des zweiten Objekts verbindet, so dass eine Änderung der ersten Eigenschaft bewirkt, dass der zweite Aufruf an das zweite Objekt ausgegeben wird, nachdem die Eigenschaftsroute aufgerufen wurde. Der Eingabeaufruf an das erste Objekt umfasst einen Identifikator, der das bedingte Aufrufen der Route ermöglicht. Auf diese Weise bleiben Routen, die zu anderen Szenarien gehören, unabhängig, so dass das System zuverlässiger arbeitet als ohne Identifikatoren des Szenarios.

– Die US-amerikanische Patentschrift 6.434.447 (Aktenzeichen des Patentanwalts PH 23.484), eingereicht am 02.10.1998 für Yevgeniy Shteyn, mit dem Titel „Control Property is Mapped Onto Modally Compatible GUI Element". Dieses Dokument betrifft im Besonderen, jedoch nicht ausschließlich, Home-API und ein Informationsverarbeitungssystem, das eine elektronische Einrichtung und einen Controller zur Steuerung einer Funktionalität der Einrichtung umfasst. Eine abstrakte Darstellung der Funktionalität wird dem Controller bereitgestellt. Die abstrakte Darstellung macht eine Modalität zum Steuern der Funktionalität zugänglich. Der Controller ermöglicht die Steuerung der Funktionalität durch die Interaktion mit der abstrakten Darstellung. Die Modalität steuert die Zuordnung der Steuerung der Funktionalität zu einer modal kompatiblen Steuerungsfähigkeit des Controllers. Die zugänglich gemachte Modalität kann beispielsweise Boolescher Natur, veränderlicher Natur oder eine ganzzahlige Matrix sein.

– Die US-amerikanische Patentschrift 6.970.081 (Aktenzeichen des Patentanwalts PHA 23.503), eingereicht am 21.10.1998 für Doreen Cheng, mit dem Titel „Distributed Software Controlled Theft

Detection". Dieses Dokument betrifft das Schaffen eines Rahmens für ein Sicherheitssystem, das eigentumspezifisch ist. Der Rahmen umfasst ein verteiltes softwaregesteuertes Sicherheitssystem, das den Zustand von einzelnen Eigentumseinrichtungen überwacht und bewertet. Die Aktivierung eines Alarms und die als Reaktion auf den Alarm ergriffene Maßnahme werden durch den Zustand der gesicherten Eigentumseinrichtungen und die jedem Zustand der Kombination von Zuständen zugeordneten Regeln bestimmt. In Abhängigkeit von den Fähigkeiten der einzelnen Eigentumseinrichtungen werden die Sicherheitsfunktionen zwischen den Einrichtungen verteilt. Bei einer bevorzugten Ausführungsform erfolgt die Übertragung von Meldungen zwischen den Komponenten des Systems in Übereinstimmung mit HAVi- oder Home-API-Standards, wodurch die Interoperabilität zwischen den Komponenten verschiedener Hersteller möglich wird.

– Das Dokument WO 0017789 (Aktenzeichen des Patentanwalts PHA 23.500), eingereicht für Adrian Turner et al, mit dem Titel „Customized Upgrading of Internet-enabled Devices Based an User-Profile". Dieses Dokument betrifft ein Serversystem, das ein Benutzerprofil eines bestimmten Endbenutzers einer netzfähigen Einrichtung der Unterhaltungselektronik aufrechterhält, und eine Datenbank mit neuen technischen Merkmalen für diese Art der Einrichtung, beispielsweise ein Heimnetz. Besteht eine Übereinstimmung zwischen dem Benutzerprofil und einem neuen technischen Merkmal und gibt der Benutzer an, dass er Informationen über Updates oder Verkaufsangebote empfangen möchte, wird der Benutzer über das Netz über die Option informiert, dieses Merkmal zu erhalten.

– Das Dokument WO 0028436 (Aktenzeichen des Patentanwalts PHA 23.527), eingereicht für Yevgeniy Shteyn, mit dem Titel „Upgrading of Synergetic Aspects of Home Networks". Dieses Dokument betrifft ein System mit einem Server, der Zugriff auf einen Bestand an Einrichtungen und Fähigkeiten in dem Heimnetz eines Benutzers hat. Der Bestand ist beispielsweise ein Suchdienst, wie er von HAVi-, JINI- und Home-API-Architekturen geboten wird. Der Server hat ebenfalls Zugriff auf eine Datenbank mit Informationen über Merkmale für ein Netz. Der Server ermittelt, ob die Synergie der in dem Netz des Benutzers vorliegenden Geräte basierend auf der Auflistung des Bestands und dem Profil des Benutzers verbessert werden kann. Existieren basierend auf diesen Kriterien für die Synergie relevante Merkmale, wird der Benutzer informiert.

– Das Dokument WO 0028398 (Aktenzeichen des Patentanwalts PHA 23.528), eingereicht für Yevgeniy Shteyn, mit dem Titel „Content Supplied as Software Objekts for Copyright Protection". Dieses Dokument betrifft die Lieferung von Inhaltsinformationen beispielsweise einen Film, eine Audiodatei oder eine Textmeldung an einen Endbenutzer. Die Inhaltsinformationen sind in einem Softwareobjekt enthalten, das eine verkapselte Prozedur für den Zugriff auf Inhaltsinformationen durch den Endbenutzer in einer Laufzeitumgebung aufweist. Das Objekt kann einen Zeitrahmen für die Inhaltsinformationen und eine Art des Zugriffs auf diese Informationen spezifizieren. Da die Prozedur in dem Objekt zusammen mit den Inhaltsdaten verkapselt ist und die Übertragung des Objekts über das Internet nach der seriellen Anordnung erfolgt, wird ein angemessener Grad der Sicherheit gegen unerlaubtes Abspielen oder Kopieren geschaffen.

– Die US-amerikanische Patentschrift 6.499.062 (Aktenzeichen des Patentanwalts PHA 23.529), eingereicht am 17.12.1998 für Yevgeniy Shteyn, mit dem Titel „Synchronizing Property Changes to Enable Multiple Control Options". Dieses Dokument betrifft im Besonderen, jedoch nicht ausschließlich, Home-API und behandelt ein Informationsverarbeitungssystem wie ein Heimnetz. Komponenten in dem Netz werden durch Softwareobjekte dargestellt, deren Eigenschaften durch Funktionsaufrufe (siehe oben genanntes COM) verändert werden können. Das Einstellen einer Eigenschaft eines Objekts steuert die zugeordnete Komponente. Die Eigenschaften sind durch Routen miteinander verbunden, die Zustandsänderungen im ganzen System weiterleiten, ohne dass eine Client-Anwendung laufen muss. Zweiweg-Eigenschaftsrouten werden dazu verwendet, die Konsistenz zwischen einem gesteuerten Objekt und mehreren steuernden Objekten zu erhalten, ohne dass die Gefahr endloser Schleifen besteht. Zu diesem Zweck wird die Zweiwegroute zur Änderung des Zustandes einer der Eigenschaften nach der Änderung des Zustandes einer anderen Eigenschaft ausgeführt, wenn die Änderung des Zustands der anderen Eigenschaft durch einen anderen Effekt als die Route selbst verursacht wurde.

[0051] Dieser Mechanismus ermöglicht die automatische Synchronisierung von Komponenten von mehreren Steuereingängen aus.

[0052] Die Erfindung erlaubt es Softwareentwicklern auch, Anwendungen mit einem größeren Spektrum an steuerbaren Einrichtungen zu schaffen, als es vorher möglich war, beispielsweise in einer Anwendung zur Netzpersonalisierung, wie sie in dem Dokument WO 0017789 (Aktenzeichen des Patentanwalts PHA 23.500), eingereicht für Adrian Turner et al, und dem oben erwähnten Dokument WO 0028436 (Aktenzeichen des Patentanwalts PHA 23.527), eingereicht für Yevgeniy Eugene Shteyn, mit dem Titel „Upgrading of Synergetic Aspects of Home Networks" dargelegt wird. Zur Personalisierung von

Diensten wird auch auf die US-amerikanische Patentschrift 6.611.654 (Aktenzeichen des Patentanwalts 23.633), eingereicht am 01.04.1999 für Yevgeniy Eugene Shteyn, mit dem Titel „Time- and Location-driven Personalized TV" verwiesen. Dieses Dokument betrifft ein Serversystem und ein Verfahren, das es einem Teilnehmer ermöglicht, ein bestimmtes Rundfunkprogramm zum Aufzeichnen und einen bestimmten Ort und Zeitrahmen zum Abspielen des aufgezeichneten Programms auszuwählen.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

[0053] Ausführungsbeispiele der Erfindung sind in den Zeichnungen dargestellt und werden im Folgenden näher beschrieben. Es zeigen:

[0054] Fig. 1 ein Blockschaltbild eines die Erfindung darstellenden Systems;

[0055] Fig. 2 ein Blockschaltbild einer HAVi- bzw. Home-API-Hardwarekonfiguration; und

[0056] die Fig. 3 und Fig. 4 Blockschaltbilder, die Softwarekonfigurationen für ein HAVi-Home-API-System darstellen.

[0057] Ähnliche oder entsprechende Merkmale haben in allen Figuren die gleichen Bezugszeichen.

BEVORZUGTE AUSFÜHRUNGSFORMEN

[0058] Die Erfindung ermöglicht es, Heimnetze mit unterschiedlichen Softwarearchitekturen zu integrieren. Verweise auf Softwaredarstellungen von Einrichtungen und Diensten in einem ersten Netz werden automatisch erstellt. Die Verweise reichen semantisch für die automatische Erstellung von zumindest teilweise funktionell gleichwertigen Softwaredarstellungen für ein zweites Netz aus, so dass Einrichtungen und Dienste des ersten Netzes von dem zweiten Netz aus zugänglich sind.

Konzepte der Blockschaltbilder

[0059] Fig. 1 ist ein Blockschaltbild eines Systems **100** zur Darstellung der Erfindung. Das System **100** umfasst ein erstes und ein zweites Netz **102** und **104** mit unterschiedlicher Softwarearchitektur. Beispielsweise hat das Netz **102** eine auf HAVi basierende Architektur und das Netz **104** eine auf Home-API basierende Architektur oder das Netz **102** hat eine auf JINI basierende Architektur und das Netz **104** eine auf HAVi basierende Architektur usw.

[0060] Das erste Netz **102** verfügt über einen Dienst **106**, der das Abfragen der Softwaredarstellungen **108**, **110**, ..., **112** von Ressourcen (Einrichtungen und Diensten) ermöglicht, die in dem Netz **102** registriert wurden. Beim Abfragen werden die Attribute der Darstellungen **108**–**112** durchsucht. Der Dienst **106** ermöglicht die Registrierung, die Austragung aus der Registrierung und die Abfrage von Ressourcen, die gesteuert werden können und mit denen interagiert werden kann. Die Registrierung einer Ressource erfordert einen Satz mit Attributen und einen Verweis für die Darstellung. Die Austragung aus der Registrierung erfordert das Liefern desselben Verweises und das Deaktivieren des Zugriffs auf den Verweis durch Softwareanwendungen oder andere Softwareobjekte. Die Abfrage erfordert das Schaffen eines vollständigen oder Teilsatzes mit Attributen, mit denen die registrierten Eingaben abgeglichen werden. In gleicher Weise verfügt das Netz **104** über einen Dienst **114**, der die Erkennung und Steuerung von Ressourcen ermöglicht, deren Softwaredarstellungen **116**, **118**, **120** in ihm registriert sind.

[0061] Das System **100** verfügt über Mittel **122**, die eine Brückenverbindung zwischen den Netzen **102** und **104** herstellen und dazu dienen, dem Netz **102** die Steuerung von einer oder mehreren im Netz **104** registrierten Ressourcen zu übertragen.

[0062] Die Mittel **122** umfassen ein Softwaremodul (ein Objekt oder eine Anwendung) **124** genannt Reference Factory. Die Reference Factory **124** ist in dem Netz **104** installiert und hat über den Dienst **114** Zugriff auf jegliche der Softdarstellungen **116**–**120**. Die Reference Factory **124** ist in der Lage, den Bestand des Dienstes **114** abzufragen oder über eine neue Softwaredarstellung gemäß den Verfahren der Softwarearchitektur des Netzes **104** informiert zu werden. In diesem Sinne ist die Factory **124** spezifisch für das Netz **104**. Hat die Reference Factory **124** ein interessierendes Objekt, beispielsweise die Softwaredarstellung **116**, gefunden, erstellt die Factory **124** einen Verweis oder einen Satz mit Verweisen auf die Darstellung **116**.

[0063] Die Mittel **122** verfügen ferner über einen so genannten Associations Container **126**, eine Softwaredarstellung für ein Repositorium für die von der Factory **122** erstellten Verweise, damit diese allen Clients des Netzes **104** zur Verfügung gestellt werden.

[0064] Die Mittel **122** verfügen ebenfalls über eine so genannte Software Element Factory **128**, die in dem Netz **102** installiert ist. Die Factory **128** wählt Verweise aus dem Container **126** aus und erstellt vorgefertigte Softwaredarstellungen oder ruft sie ab basierend auf den in den ausgewählten und für eine Installation im Netz **102** geeigneten Verweisen enthaltenen Informationen, indem sie sie beispielsweise bei dem Dienst **106** registriert oder dort gemäß den durch die Softwarearchitektur des Netzes **102** vorgegebenen Regeln platziert. Nach der Registrierung im Dienst **106** werden die Ressourcen im Netz **104** von dem Netz **102** aus zugänglich und steuerbar.

[0065] Die Factory **124** ist in der Lage, die Regeln der Softwarearchitektur des Netzes **104** einzusetzen, um auf den Dienst **114** zuzugreifen und Informationen zum Erstellen der Verweise zu extrahieren. Die Factory **128** ist in der Lage, die Regeln der Softwarearchitektur des Netzes **102** einzusetzen, um zum Registrieren neu erstellter Softwaredarstellungen basierend auf den Verweisen im Repositorium **126** auf den Dienst **106** zuzugreifen. Die Factory **124** und die Factory **128** haben eine Schnittstelle über das Repositorium **126**. Das Repositorium **126** sollte daher in der Lage sein, den Informationsinhalt der von der Factory **124** für die Factory **128** erstellten Verweise zu übertragen. Zu diesem Zweck besteht ein möglicher Mechanismus darin, dass die Factory **124** und die Factory **128** eine Schnittstelle zum Repositorium **126** aufweisen, die auf derselben Sprache, beispielsweise XML, basiert. Dies bedeutet, dass die als Ausgangsdaten von der Factory **124** gelieferten Verweise direkt als Eingangsdaten für die Factory **128** verwendet werden können. Ein weiterer Mechanismus besteht darin, es dem Repositorium **126** zu ermöglichen, die von der Factory **124** empfangenen Informationen mit Hilfe eines Konvertierungsprotokolls in auf geeignete Weise formatierte Ausgangsdaten für die Factory **128** zu übersetzen oder zu konvertieren. In diesem Zusammenhang wird beispielsweise auf die oben erwähnte US-amerikanische Patentschrift 6.434.447 (Aktenzeichen des Patentanwalts PHA 23.484) verwiesen, die eine allgemeine Art des Mapping-Protokolls darlegt. Als Alternative können bestimmte Konvertierungsstufen zwischen dem Repositorium **126** einerseits und den Netzen **102** und **104** andererseits eingefügt werden.

[0066] Mit den nötigen Abänderungen können ähnliche Mittel **130** hinzugefügt werden, die eine Reference Factory **132**, einen Container **134** und eine Software Element Factory **136** umfassen, um die Ressourcen im Netz **102** von dem Netz **104** aus steuerbar zu machen.

[0067] In dem Netz **104** können mehrere Reference Factories installiert werden, beispielsweise eine weitere Reference Factory **136** zum Schaffen bestimmter Verweisinformationen, damit ein drittes (nicht dargestelltes) Netz mit noch einer anderen Softwarearchitektur eine oder mehrere Funktionen steuern kann, die in dem Netz **104** registriert sind, ähnlich wie die Steuerung vom Netz **102** aus. Jede dieser mehreren Reference Factories kann für gewisse Arten von Verweisen, beispielsweise eine weitere Softwareumgebung (HAVi, JINI, Home-API) oder eine weitere Objektdarstellung innerhalb eines derartigen Netzwerkes (Java DCM für FAV-Einrichtungen, DDI-Daten für grafische Benutzerschnittstellen (GUI), native DCM für IAV-Einrichtungen usw.) oder eine Klasse mit Ressourcen (Einrichtungen bzw. Dienste) zuständig sein, beispielsweise die Datenspeicherung, die Heimautomatisierung, den Audio-/Video-Befehlssatz usw.

[0068] Das Repositorium **126** kann für eine weitere Software Element Factory **138** für ein viertes Netz zugänglich sein, wobei die weitere Factory in der Lage ist, direkt oder indirekt die Verweise im Repositorium **126** zu bearbeiten.

Blockschaltbild Hardwarekonfiguration HAVi – Home-API

[0069] Fig. 2 zeigt die physikalische Konfiguration eines integrierten Heimnetzsystems **200** mit einem auf HAVi basierenden Netz **202** und einem auf Home-API basierenden Netz **204**.

[0070] Das Netz **202** umfasst eine HAVi-Set-Top-Box **206** mit FAV-Fähigkeiten (siehe obige Erläuterung zu HAVi), einen HAVi-konformen Fernseher **208**, der als BAV-Einrichtung dient (siehe obige Erläuterung zu HAVi) und einen HAVi-konformen digitalen Videorecorder **210**, der ebenfalls als BAV-Einrichtung dient. Die Einrichtungen **206**, **208** und **210** sind über einen IEEE-1394-Bus miteinander verbunden, so dass die Set-Top-Box **206** den Fernseher **208** und den digitalen Videorecorder **210** steuern kann.

[0071] Das Netz **204** umfasst einen PC **212**, der so konfiguriert ist, dass er die Beleuchtung **214** über eine X-10-Verbindung, Rasensprenger **216** über eine IR-Verbindung und andere (nicht dargestellte) Geräte über zugeordnete Verbindungen steuert.

[0072] Die Set-Top-Box **206** und der PC **212** sind über einen IEEE-1394-Bus miteinander verbunden. Die IEEE-1394-Verbindung ermöglicht es dem PC **212**, auf IEEE-1394-konforme Einrichtungen zuzugreifen. Zur Nutzung dieser Fähigkeit ist ein Home-API-Diensteanbieter **218** (siehe oben „Hintergrund der Erfindung" oder die Home-API-Spezifikation) für HAVi auf dem PC **204** installiert. Der HAVi-spezifische Diensteanbieter **218** ist in der Lage, auf eine auf einem PC basierende HAVi-FAV-Anwendung oder eine auf einem PC basierende HAVi-IAV-Anwendungsumgebung zuzugreifen. Er kann auch selbst eine FAV- oder IAV-Umgebung ausführen oder installieren, wenn keine verfügbar ist. Der Diensteanbieter **218** bestückt das Home-API-Verzeichnis **220** dann mit COM-Objekten, die HAVi-Einrichtungen, beispielsweise die Einrichtungen **206**–**210**, darstellen. Dadurch können Home-API-Awendungen auf dem PC **212** die HAVi-Einrichtungen **206**–**210** steuern.

[0073] Diese Konfiguration erlaubt es jedoch auf HAVi basierenden Anwendungen in der FAV-Einrichtung **206** nicht, auf Home-API-Softwareobjekte wie diejenigen für die Beleuchtung **214**, die Rasensprenger **216** oder andere in dem Verzeichnis **220** registrierte Einrichtungen, zuzugreifen. Damit vom HA-

Vi-Netz **202** ausgehend eine Steuerung möglich wird, muss ein Satz mit HAVi-konformen Softwareelementen (beispielsweise DCMs, FCMs) in einer HAVi-Registrierdatenbank **222** installiert werden. Die Erfindung schlägt nun vor, die Konzepte der Reference Factory und der Software Element Factory einzuführen, um diese Installation wie in Bezug auf Fig. 1 erläutert zu ermöglichen. Diese Steuerfähigkeit erlaubt es dem Benutzer beispielsweise, die Beleuchtung **214** über seine HAVi-Set-Top-Box **206** ein- und auszuschalten und die Sprenger **216** anzuhalten oder zeitlich neu zu steuern, wenn er ungestört einen interessanten Film ansehen möchte, usw.

Blockschaltbilder für die Erstellung der Softwarekonfiguration HAVi – Home-API

[0074] Fig. 3 ist ein erstes Blockschaltbild, das eine Softwarekonfiguration **300** einer Brückenverbindung zwischen HAVi und Home-API zeigt. Die Konfiguration **300** umfasst einen PC **302** mit einer Home-API-Steuerumgebung **304** und eine HAVi-IAV/FAV-Softwareausführung **306**. Die Konfiguration **300** umfasst ebenfalls ein HAVi-Netz **308** und ein Home-API-Netz **310**. Mit der Home-API-Umgebung **304** interagierende HAVi-spezifische Softwareelemente ermöglichen es, dass Einrichtungen oder Dienste des Netzes **310**, die in der Home-API-Umgebung **304** registriert sind, von dem HAVi-Netz **308** aus gesteuert werden.

[0075] Wie oben erläutert schafft die Erfindung ein Verfahren, das die Brückenverbindung zwischen verschiedenen Heimnetzumgebungen, hier dem HAVi-Netz **308** und dem Home-API-Netz **310**, ermöglicht. Zu diesem Zweck umfasst die Home-API-Umgebung **304** eine Softwarekomponente **312**, eine Anwendung oder ein Softwareobjekt und Reference Factory genannt, die Softwaredarstellungen von Einrichtungen und Diensten erkennt, die in der Umgebung **304** zur Verfügung stehen, beispielsweise ein Objekt **314**. Diese Erkennung kann durch Aufzählen und/oder Überwachen des Home-API-Verzeichnisses **316** oder durch Zugreifen auf einen Home-API-Stamm oder einen anderen Container erfolgen. Basierend auf den Fähigkeiten der Reference Factory **312** und/oder den Benutzerpräferenzen wird ein Verweis oder ein Satz mit Verweisen erstellt, der der Softwaredarstellung der erkannten Einrichtung oder des erkannten Dienstes zugeordnet ist. Ein derartiger Verweis umfasst Informationen über die Softwaredarstellung, beispielsweise die Objektart, die Eigenschaftsbeschreibung, eine Klassenidentität, eine URL, einen eindeutigen Objektidentifikator, ein XML-Tag, eine DDI-Beschreibung usw., die für die Erstellung einer gleichwertigen Softwaredarstellung derselben Einrichtung oder desselben Dienstes, jetzt jedoch für den Einsatz innerhalb eines anderen Netzes, hier der HAVi-Umgebung **304** und dem Netz **310**, erforderlich sind. Die Factory **312** nutzt Home-API-Meldungsmechanismen, wie Ereignisteilnahme oder Eigenschaftsrouten, um zur Umgebung **304** hinzugefügte Softwareobjekte zu erkennen. Wenn ein Softwareobjekt zu der Home-API-Umgebung **304** hinzugefügt wird, wird die Reference Factory **312** benachrichtigt, und es wird/werden, falls erforderlich, (ein) Verweis(e) erstellt und in einem so genannten Reference Container **318** platziert, der dem hinzugefügten Objekt zugeordnet ist.

[0076] In diesem Beispiel stellt der Container **318** die Verweise allen Home-API-Clients zur Verfügung. Beispielsweise greift die Reference Factory **312** auf interessierende Objekte, beispielsweise wie vom Benutzer angegeben, über das Home-API-Verzeichnis **316** zu. Der Benutzer hat beispielsweise angegeben, dass er an den Objekten „Beleuchtung" **214** interessiert ist. Die Benutzerpräferenzen können durch eine Konfigurationsanwendung bzw. -führung erzielt werden. Die Anwendung oder Führung listet beispielsweise alle verfügbaren Einrichtungen oder Arten von Einrichtungen in einer Netzumgebung auf. Sie sammelt Benutzereingaben bezüglich derjenigen von ihnen, die über ein anderes Netz zugänglich sein sollen. Die Reference Factory **312** identifiziert die interessierenden Softwareobjekte in dem Verzeichnis **310** und erstellt einen Satz mit Verweisen für HAVi-Softwareelemente wie jene, die DDI-Daten, einen Java-DCM-Verweis oder einen nativen (binären) DCM umfassen, jedoch nicht hierauf beschränkt. Diese Verweise werden in dem Reference Container **318** für „Beleuchtung" **214** platziert. Nachdem die Reference Factory **312** installiert ist, wird sie beispielsweise durch den Home-API-Ereignismechanismus zum Home-API-Stamm hinzugefügt. Das Installationsprogramm kann die Factory **312** zu einem bestimmten Container oder zu einem Teil des Home-API-Verzeichnisses **316**, wie „Wohnzimmer", hinzufügen, damit lediglich die Beleuchtung im Wohnzimmer gesteuert werden kann. Wenn ein neues Beleuchtungsobjekt zum Netz **310** hinzugefügt wird, erstellt die Factory **312** einen neuen Verweis, beispielsweise eine DDI-Beschreibung und eine URL für ein Java-Paket, das die neue Beleuchtung darstellt. Diese Verweise werden dann zum Container **318** der neuen Beleuchtung hinzugefügt.

[0077] Eine HAVi-Software Element Factory **320** ist die Softwarekomponente, die für den Zugriff auf interessierende Home-API-Objekte zuständig ist. In diesem Beispiel wählt sie geeignete Verweise aus dem Container **318** für das Objekt „Beleuchtung" aus. Die Factory **320** erstellt HAVi-Softwareelemente basierend auf den abgerufenen Verweisen und mit Hilfe von internen oder externen Ressourcen (beispielsweise dem Internet). Es wird beispielsweise auf die oben erwähnte Infrastruktur verwiesen, wie sie in den Dokumenten WO 00017789 und WO 0028436 dargelegt ist. Der PC **302** verfügt über einen DCM-Manager **322** und eine Registry **324** als Teile der HA-

Vi-IAV/FAV-Ausführung auf dem PC **302**. Die Factory **320** interagiert mit dem HAVi-DCM-Manager **322** und der HAVi-Registry **324** gemäß den HAVi-Architekturregeln und/oder den Präferenzen der Element Factory. Die Factory **320** erstellt beispielsweise DDI-Daten, um die Funktionalität des Horne-API-Objekts „Beleuchtung" über eine Anzeige und HAVi-Anwendungen zugänglich zu machen, die vom HAVi-Netz **308** gehostet werden. Als Alternative oder Ersatz erstellt oder ruft die Factory **320** aus dem Internet (über eine URL) ein Java-DCM ab und registriert es in der Registry **324**, um es den HAVi-Anwendungen oder anderen DCMs zu ermöglichen, mit der Softwaredarstellung der „Beleuchtung" **214** zu interagieren.

[0078] Hinsichtlich der Installation der Reference Factory **302** kann diese beispielsweise durch einen Gerätehersteller während der Installation einer Einrichtung im Netz aktiviert werden. Beispielsweise können Beleuchtungen von Philips im Paket mit PC-Software verkauft werden, die das Softwareobjekt Reference Factory **302** umfassen und den Zugriff auf HAVi erlauben. Die Installation kann von einem Diensteanbieter, von dem Benutzer selbst oder von Dritten als Teil eines Upgrades der PC-Softwarefunktionalität vorgenommen werden.

[0079] Hinsichtlich der Installation der Software Element Factory **320** kann diese von einem HAVi-Diensteanbieter oder von einem Dritten, dem Benutzer selbst oder einem Diensteanbieter installiert werden, um eine existierende Element Factory aufzurüsten.

[0080] Fig. 4 ist ein zweites Blockschaltbild, das eine Softwarekonfiguration **400** einer Brückenverbindung zwischen HAVi und Home-API darstellt. Die Konfiguration umfasst einen PC **302** mit einer Home-API-Steuerumgebung **304** und einem Home-API-Netz **310**. Die Konfiguration **400** umfasst ebenfalls ein HAVi-Netz **308**, das eine HAVi-FAV-Einrichtung **402** enthält. Der PC **302** verfügt nun über ein Softwaremodul **404** zur Darstellung der Home-API-Steuerumgebung **304** in dem HAVi-Netz **308**. Das Modul **404** ist eine HAVi-BAV-Einrichtung, die die Software Element Factory **320** umfasst und mit anderen BAV-Softwarekomponenten interagiert, um das Home-API-Objekt **314** als ein DCM **406** (oder ein FCM) in dem HAVi-Netz **308** darzustellen. Einer oder mehrere (nicht dargestellte) DCM-Manager in HAVi-FAV- oder IAV-Einrichtungen interagieren mit der BAV-Software **404**, um das Home-API-Objekt **314** über das DCM **406** im Netz **308** verfügbar zu machen.

[0081] Es wird auch auf die oben für diese Konfiguration aufgelistete US-amerikanische Patentschrift mit der Seriennummer 09/146.020 (Aktenzeichen des Patentanwalts PHA 23.492) verwiesen.

[0082] Im Zusammenhang mit der Installation von Softwareelementen wird auf die oben erwähnten Dokumente WO 0017789 (Aktenzeichen des Patentanwalts PHA 23.500), eingereicht für Adrian Turner et al, mit dem Titel „Customized Upgrading of Internet-Enabled Devices Based an User-Profile" und WO 0028436 (Aktenzeichen des Patentanwalts PHA 23.527), eingereicht für Yevgeniy Eugene Shteyn, mit dem Titel „Upgrading Of Synergetic Aspects of Home Networks" verwiesen.

[0083] Als ein weiteres Beispiel wird eine Brückenverbindung zwischen HAVi und UPnP gebildet, indem HAVi-Softwareelemente dem UPnP-Netz zugänglich gemacht werden und umgekehrt. XML ist die Basis für die Darstellung von Einrichtungen bei UPnP. Damit HAVi-Softwareelemente an einem UPnP-Netz beteiligt sein können, müssen sie eine ihnen zugeordnete XML-Darstellung aufweisen. Eine derartige Darstellung kann automatisch, d. h. bei jeder Verbindungs- bzw. Aufzählungsabfrage, oder vorzugsweise einmal erstellt und in der Registrierdatenbank als ein Attribut des Softwareelements oder ein getrenntes Softwareelement platziert werden, das dem ersteren durch eine eindeutige Softwareelementidentität zugeordnet ist. Wenn eine direkte Übersetzung in XML eines bestimmten HAVi-Softwareelementes funktionell nicht möglich ist (beispielsweise wenn ein Softwareelement durch ein Java-Objekt von Drittherstellern dargestellt wird usw.), kann eine allgemeine XML-Darstellung erstellt werden. Der Hersteller des Softwareelements (DCM, FCM) kann eine bevorzugte XML-Darstellung der Komponente liefern, wenn die Beteiligung an dem UPnP-Netz geplant ist. In gleicher Weise kann die HAVi-DDI-Schnittstelle basierend auf den frei zugänglichen DDI-Benutzerschnittstellenelementen (für weitere Details siehe beispielsweise Abschnitt 4 der HAVi-Spezifikation) in XML übersetzt werden. Der Reflexionsmechanismus von Java kann dazu eingesetzt werden, Schnittstellen eines bestimmten Java-Objekts abzufragen, und ein gleichwertiges XML-Modell kann erstellt werden, wenn die Schnittstelle als bekannt erkannt wird.

Patentansprüche

1. Verfahren, das es einem ersten Heimnetz (**104**) mit einer ersten Softwarearchitektur ermöglicht, mit einem zweiten Heimnetz (**102**) mit einer zweiten Softwarearchitektur zu interagieren, wobei das Verfahren Folgendes umfasst:
– Verwenden eines Suchdienstes (**114**) des ersten Netzes zum Erkennen einer ersten Softwaredarstellung (**116**, **118**, **120**) einer Einrichtung oder eines Dienstes in dem ersten Netz,
– Erstellen (**124**) eines Verweises auf die erkannte erste Softwaredarstellung,
– Speichern des Verweises in einem Repositorium (**126**) und
– Zugreifen auf das Repositorium zum Bestimmen

(**128**) einer zumindest teilweise funktionell gleichwertigen zweiten Softwaredarstellung basierend auf dem Verweis, wobei die zweite Softwaredarstellung von dem zweiten Netz aus zugänglich ist.

2. Verfahren nach Anspruch 1, wobei das erste und das zweite Netz (**202**, **204**) verschiedene Softwarearchitekturen aufweisen.

3. Verfahren nach Anspruch 2, wobei
– eines der ersten und zweiten Netze auf einer JINI-Architektur basiert und
– das andere der ersten und zweiten Netze auf einer HAVi-Architektur basiert.

4. Verfahren nach Anspruch 2, wobei
– eines der ersten und zweiten Netze auf einer JINI-Architektur basiert und
– das andere der ersten und zweiten Netze auf einer UPnP-Architektur basiert.

5. Verfahren nach Anspruch 2, wobei
– eines der ersten und zweiten Netze auf einer UPnP-Architektur basiert und
– das andere Netz auf einer HAVi-Architektur basiert.

6. Verfahren nach Anspruch 1, wobei
– das erste Netz einen Bestand von Einrichtungen und/oder Diensten umfasst, die in dem ersten Netz registriert sind,
– das Erkennen das Abfragen des Bestandes umfasst,
– das Erstellen das Extrahieren von Informationen über die erste Softwaredarstellung umfasst, die für das zweite Netz relevant sind, und
– das Bestimmen das Schaffen einer zweiten Softwaredarstellung basierend auf den extrahierten Informationen umfasst.

7. Informationsverarbeitungssystem (**100**), das Folgendes umfasst:
– ein erstes Heimnetz (**104**) mit einer ersten Softwarearchitektur und
– ein zweites Heimnetz (**102**) mit einer zweiten, von der ersten Softwarearchitektur verschiedenen Softwarearchitektur, wobei
– das erste Netz einen Suchdienst (**114**) umfasst, der es ermöglicht, eine erste Softwaredarstellung (**116**, **118**, **120**) einer ersten Einrichtung oder eines ersten Dienstes in dem ersten Netz zu erkennen,
– das System einen Verweisgenerator (**124**) zum Erstellen eines Verweises auf die erste Softwaredarstellung durch Interaktion mit dem Suchdienst umfasst,
– das System ein Repositorium zum Speichern des Verweises umfasst und
– das System einen Softwareelementgenerator (**128**) zum Zugreifen auf das Repositorium und zum Ermöglichen der Bestimmung einer zumindest teilweise funktionell gleichwertigen zweiten Softwaredarstellung basierend auf dem Verweis umfasst, wobei die zweite Softwaredarstellung vom zweiten Netz aus zugänglich ist.

8. Softwarekomponente (**122**) zum Einsatz in einem Informationsverarbeitungssystem (**100**), wobei
– das System ein erstes Netz (**104**) mit einer ersten Softwarearchitektur und ein zweites Heimnetz (**102**) mit einer zweiten, von der ersten Softwarearchitektur verschiedenen Softwarearchitektur umfasst,
– das erste Netz einen Suchdienst (**114**) umfasst, der es ermöglicht, eine erste Softwaredarstellung (**116**, **118**, **120**) einer ersten Einrichtung oder eines ersten Dienstes in dem ersten Netz zu erkennen,
– die Komponente erste Codemittel zum Ausführen eines Verweisgenerators (**124**) zum Erstellen eines Verweises auf die erste Softwaredarstellung durch Interaktion mit dem Suchdienst umfasst,
– die Komponente zweite Codemittel zum Ausführen eines Repositoriums (**126**) zum Speichern des Verweises umfasst und
– die Komponente dritte Codemittel zum Ausführen eines Softwareelementgenerators (**128**) zum Zugreifen auf das Repositorium und zum Ermöglichen der Bestimmung einer zumindest teilweise funktionell gleichwertigen zweiten Softwaredarstellung basierend auf dem Verweis umfasst, wobei die zweite Softwaredarstellung vom zweiten Netz aus zugänglich ist.

Es folgen 4 Blatt Zeichnungen

Anhängende Zeichnungen

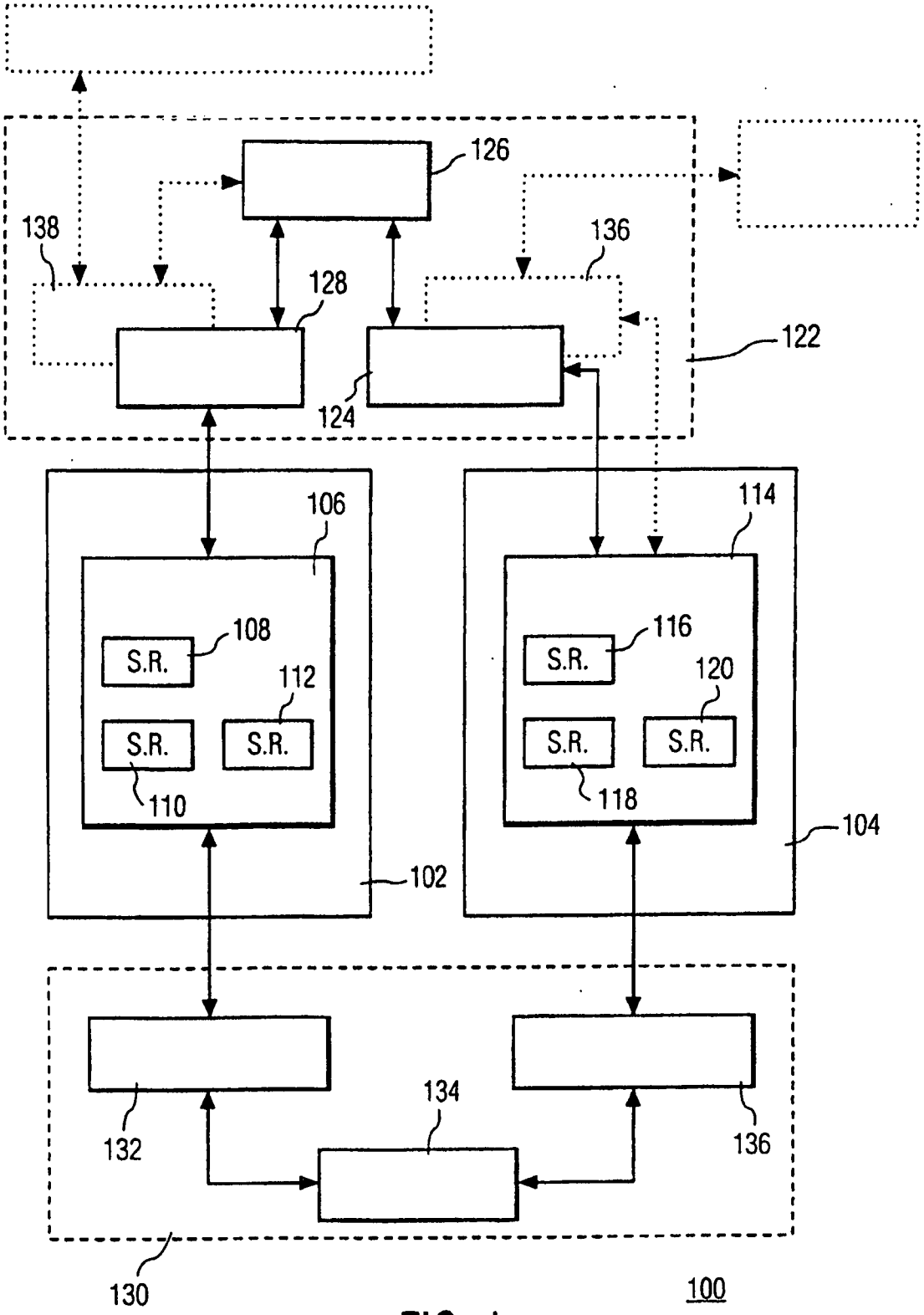

FIG. 1

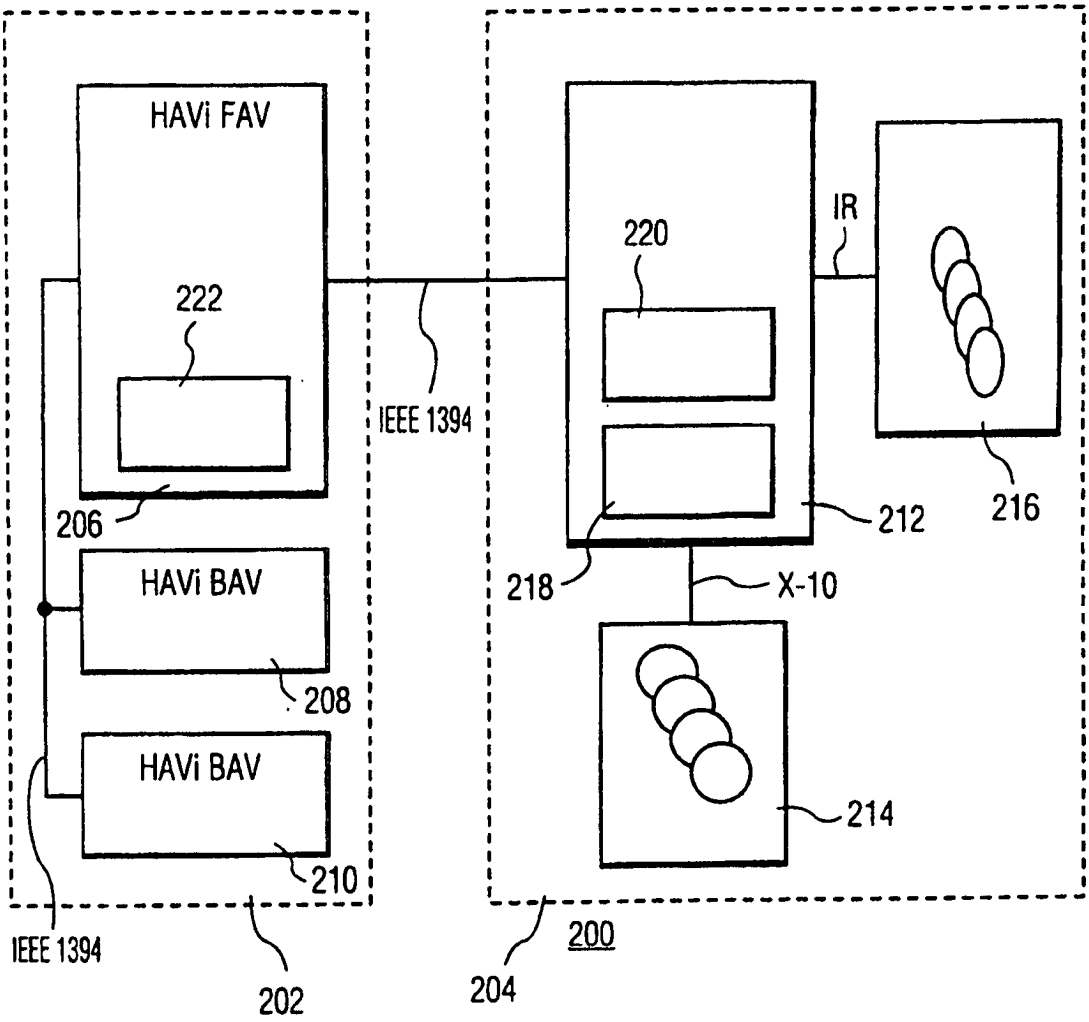
HAVi FAV
222
206
HAVi BAV
208
HAVi BAV
210
IEEE 1394
202
IEEE 1394
220
IR
212
216
218
X-10
214
200
204

FIG. 2

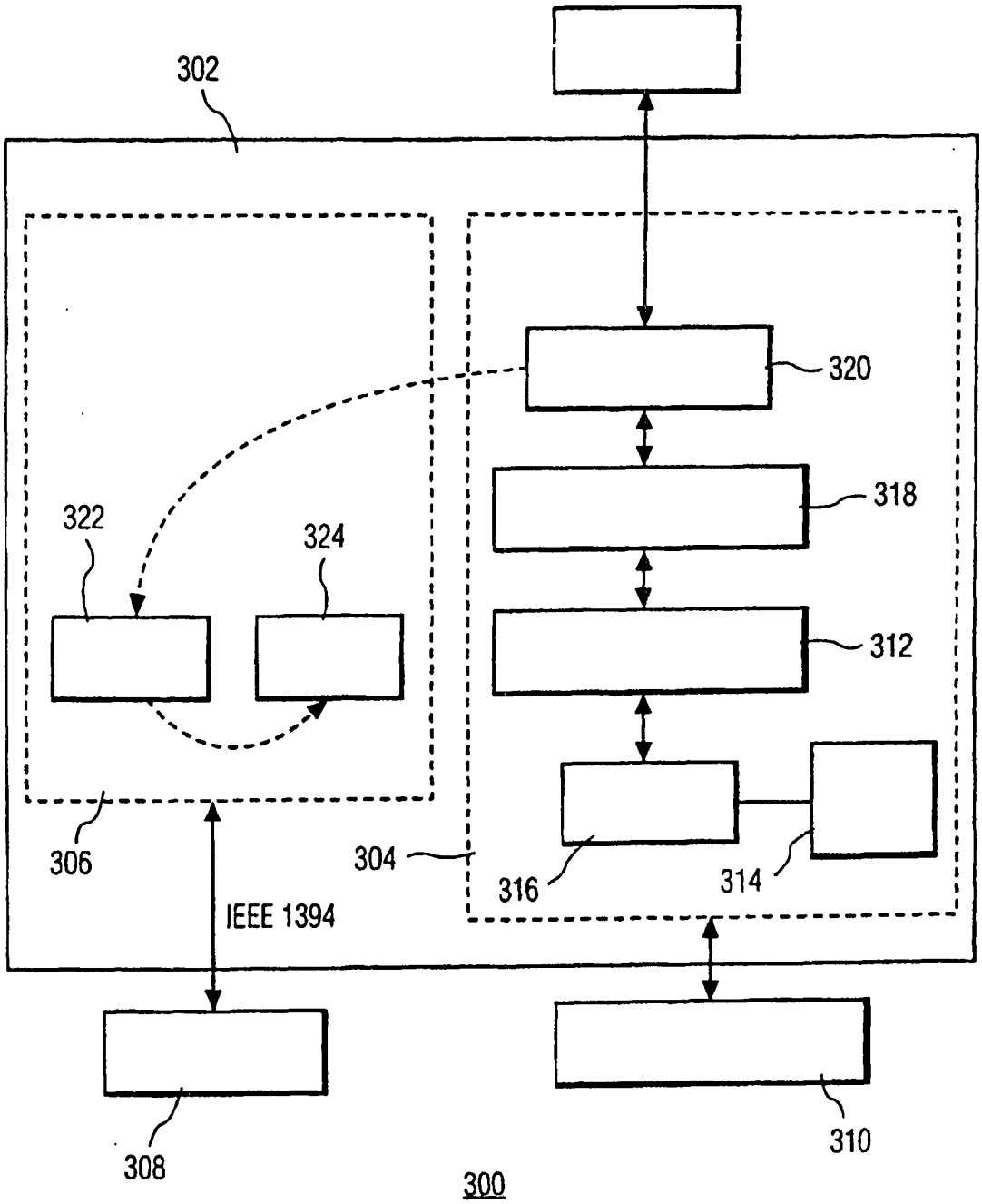

FIG. 3

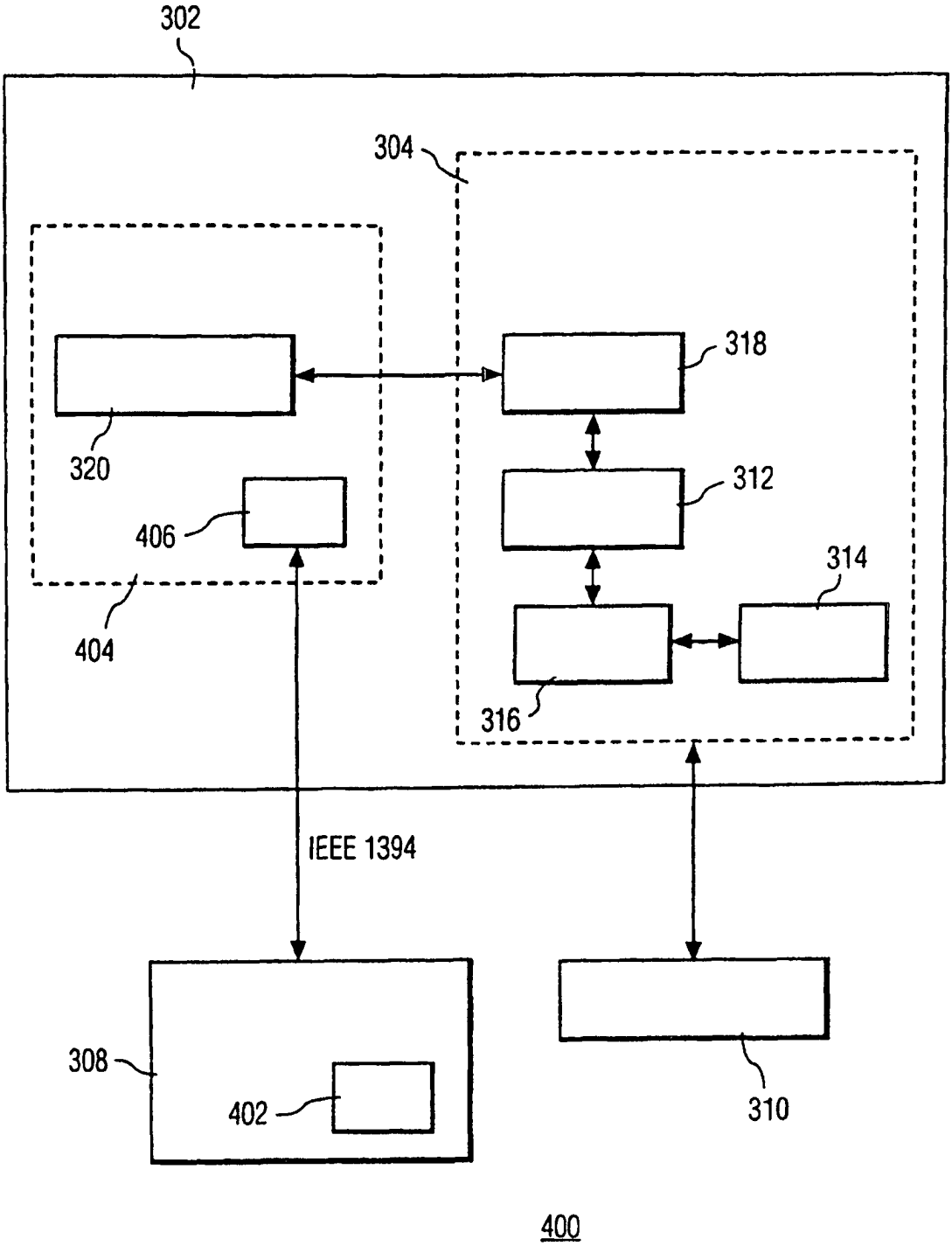

FIG. 4