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

(10) **DE 698 37 550 T2** 2007.12.27

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 017 202 B1**
(21) Deutsches Aktenzeichen: **698 37 550.5**
(96) Europäisches Aktenzeichen: **00 200 775.5**
(96) Europäischer Anmeldetag: **14.05.1998**
(97) Erstveröffentlichung durch das EPA: **05.07.2000**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **11.04.2007**
(47) Veröffentlichungstag im Patentblatt: **27.12.2007**

(51) Int Cl.[8]: ***H04L 29/06*** (2006.01)

(30) Unionspriorität:

855902	**14.05.1997**	**US**
855965	**14.05.1997**	**US**
855977	**14.05.1997**	**US**
856051	**14.05.1997**	**US**

(73) Patentinhaber:
Citrix Systems Inc., Fort Lauderdale, Fla., US

(74) Vertreter:
Hössle Kudlek & Partner, Patentanwälte, 70173 Stuttgart

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

(72) Erfinder:
Pedersen, Bradley J., Parkland, FL 33067, US;
Bloomfield, Marc A., Pompano Beach, FL 33062, US

(54) Bezeichnung: **System und Verfahren zur Datenübermittlung von einer Serverapplikation an Klientknoten**

Beschreibung

Gebiet der Erfindung

[0001] Die vorliegende Erfindung betrifft eine Ausführung von Anwendungen in einer verteilten Klienten- und Serverumgebung und insbesondere die entfernte Ausführung bzw. Fernausführung von Anwendungen, die in einer Interpretersprache geschrieben sind, in einer Klienten-Server-Umgebung.

Hintergrund der Erfindung

[0002] Klienten-Server-Computernetzwerke erfordern typischerweise, dass der Klient und der Server Kommunikationen gemäß einem gewissen Satz von vorab eingeführten Regeln einrichten. Diese Regeln werden als Kommunikationsprotokolle bezeichnet. Solche Protokolle können vorab definiert sein, so dass jeder Klientenknoten dasselbe Kommunikationsprotokoll wie der Serverknoten verwendet. Alternativ kann der Server eine Aufzeichnung des Kommunikationsprotokolls halten, das durch jeden Klienten verwendet wird, und dieses Protokoll verwenden, um mit dem Klienten zu kommunizieren, wenn der Klient eine Aufforderung sendet, mit der Anwendung auf dem Server zu kommunizieren.

[0003] Ein Problem, das mit solchen Kommunikationsverfahren in Verbindung steht, besteht darin, dass entweder das Protokoll zu starr für Anwendungen definiert sein kann, die nicht die gesamte Funktionalität erfordern, oder dass das Klientenprotokoll dem Server bekannt sein muss, bevor der Klient in der Lage ist, mit dem Server zu kommunizieren. Die vorliegende Erfindung versucht, sowohl die Starrheit eines vordefinierten Protokolls als auch die Notwendigkeit einer Kenntnis vor dem Kontakt auf Seiten des Servers zu vermeiden.

[0004] Schattenverarbeitung (Shadowing) (Senden von Daten, die für einen Klientenknoten im wesentlichen gleichzeitig zu einem zweiten Klientenknoten bestimmt sind) und Aussenden (Übertragen derselben Daten im wesentlichen gleichzeitig zu mehr als einem Klientenknoten) wurde typischerweise unter Verwendung einer spezialisierten Übertragungsanwendung auf einem Serverknoten und spezialisierter Empfängeranwendungen auf jedem der Klientenknoten durchgeführt. Schattenverarbeitung ist beim Überwachen eines Datenverkehrs und zum Erzeugen einer redundanten Kopie von Informationen, die für eine Datenintegrität und aus Systemsicherheitszwecken übertragen werden, nützlich. Ein Aussenden ist beim Bereitstellen derselben Information für viele Nutzer nützlich, wenn solche Informationen "Echtzeit" ist oder wenn die Informationen an sich keinen Anfang oder kein Ende haben. Beispielsweise überträgt ein Aktienkursnotierungsprogramm einfach die aktuellen Preise von verschiedenen Aktien auf einen gegebenen Austausch und die Liste wiederholt mit den letzten Preisen, sobald die Aktienliste verbraucht ist. Somit ist es für einen Nutzer irrelevant, dass er oder sie nicht dem Notierungsprogramm spezifiziert, wo die Liste zu beginnen ist.

[0005] Solche Programme werden typischerweise unter Berücksichtigung eines Aussendeprogramms geschrieben und erfordern spezialisierte Empfängerprogramme, um die übertragenen Daten zu empfangen. Wenn eine Anwendung nicht als ein Aussendeprogramm geschrieben wurde, können die durch solch eine Anwendung übertragenen Daten nicht typischerweise zu mehreren Klientenknoten ausgesendet werden.

[0006] Die vorliegende Erfindung versucht, dieses Problem zu bewältigen, indem Programmen, die nicht für eine Aussendefunktionalität geschrieben sind, ermöglicht wird, zum Aussenden von Daten über ein Netzwerk verwendet zu werden.

[0007] Ein spezialisiertes Klienten-Server-Netzwerk ist das weltweite Netzwerk von Computern, das allgemein als "Internet" bekannt ist, das in den letzten Jahren ein massives Wachstum erfahren hat. Ein großer Teil dieses Wachstums wurde durch das Anwachsen der Popularität des World Wide Web (WWW) bewirkt. Das WWW ist eine Zusammenstellung von Dateien, die unter Verwendung der HyperText Markup Language (HTML) geschrieben sind, die im allgemeinen als "Webpages" bezeichnet werden. Auf HTML-Dateien kann zugegriffen werden und diese können dargestellt werden unter Verwendung spezialisierter Anwendungen, die als "Webbrowser" bekannt sind, die einem Nutzer einen Zugriff auf HTML-Dateien unter Verwendung einfacher grafischer Nutzerschnittstellen (GUI: Graphical User Interface) erlauben.

[0008] Server, die HTML-Dateien beherbergen bzw. haben, können unter Verwendung des HyperText Transfer Protokolls (HTTP) kommunizieren. HTTP ist ein Anwendungsprotokoll, das Nutzern einen Zugriff auf Dateien (die in verschiedenen Formaten, wie bspw. Text, Grafik, Bilder, Ton, Video usw. sein können) unter Verwendung der HTML-Seitenbeschreibungssprache bereitstellt. HTML stellt eine grundlegende Dokumentenformatierung bereit und ermöglicht dem Entwickler, "Kommunikationsverbindungen" (links) zu anderen Servern und

Dateien zu spezifizieren. Die Verwendung eines HTML-konformen Klientenbrowsers umfasst eine Spezifikation einer Verbindung über eine Internetadresse oder "URL" (Uniform Resource Locator). Bei einer solchen Spezifikation führt der Klient eine TCP/IP-Aufforderung zu dem Server durch, der in der Verbindung identifiziert ist, und empfängt eine Internetseite (Webpage) im Gegenzug. Weiterhin können Unternehmen HTML-Dateien bereitstellen, die von innerhalb des Unternehmens aber nicht von dem WWW zugänglich sind. Diese internen Netzwerke und Zusammenstellungen von HTML-Dateien werden im allgemeinen als "Intranets" bezeichnet.

[0009] Eine Datei, die unter Verwendung von HTML geschrieben ist, umfasst Kennzeichnungen (tags), die einem Browser, der die Datei anzeigt, andeuten, wann spezielle Aktionen durchgeführt werden sollen. Beispielsweise kann eine Kennzeichnung einem Browser anzeigen: (1) dass eine Grafikdatei bei einem bestimmten Punkt in dem Dokument angezeigt werden sollte, (2) dass ein bestimmter Text zentriert, fett oder auf andere Weise formatiert sein sollte, (3) dass der Hintergrund eines Dokuments schattiert oder mit einem bestimmten Muster versehen sein sollte, oder (4) dass ein unterschiedliches HTML anstelle des HTML, das der Browser gegenwärtig anzeigt, geladen werden sollte.

[0010] Die Popularität des World Wide Web und anderer HTML-Anwendungen hat den Absatz und Verkaufsanstrengungen eines breiten Bereichs von Unternehmen angesprochen, die einen weiten Bereich der Industrien repräsentieren. Da eine Unterscheidung bzw. Abgrenzung von anderen Unternehmen immer schwieriger wird, haben viele Unternehmen versucht, die inhärent statische Natur des HTML zu umgehen. Ebenfalls haben Unternehmen, die HTML-Dateien als ein Verfahren zum gemeinsamen Nutzen von Informationen verwenden, bemerkt, dass ein Intranet ein nützliches Verfahren ist, um verschiedenen Nutzern einen Zugriff auf mehr als nur Informationen bereitzustellen.

[0011] Ein Hauptnachteil von HTML-Dateien ist jedoch, dass diese inhärent statisch sind. Das bedeutet, dass HTML eine "Nur-Anzeigesprache" ist, die eine Ausführung von Anwendungen innerhalb einer HTML-Seite nicht einfach erlaubt. Unternehmen, die die Popularität und Allgegenwärtigkeit des WWW nutzvoll einzusetzen versuchen, suchen verstärkt nach Möglichkeiten, Anwendungen in einer HTML-Datei aufzunehmen.

[0012] ActivX™-Objekte sind ein Versuch, HTML-Dateien mit der Fähigkeit auszustatten, ausführende Anwendungen anzuzeigen. Ein ActivX™-Objekt ist ein Datenobjekt, das mit Browsern verwendet werden kann, die eine ActivX™-Schnittstelle haben. Ein offensichtlicher Nachteil dieser Objekte besteht darin, dass, wenn ein Browser eines Nutzers keine ActivX™-Schnittstelle hat, dieser das Ausführungsprogramm nicht anzeigen kann. Dies beschränkt die Nützlichkeit von ActivX™-Objekten, da ein Hauptziel der meisten HTML-Seiten darin besteht, von so vielen Nutzern wie möglich betrachtet zu werden.

[0013] Eine als JAVA™ bezeichnete Computersprache wurde ebenfalls als eine Möglichkeit vorgestellt, einen ausführbaren Code an eine HTML-Datei anfügen zu können. Da JAVA™ eine Sprache ist, erfordert diese nicht eine spezifische Browserschnittstelle und hat einen potentiell größeren Empfängerkreis. Jedoch wird ein JAVA™-Programm, das üblicherweise als Applet bezeichnet wird, zu dem Klienten vor Ausführung heruntergeladen. Dies kann für Klienten problematisch sein, die nicht ausreichend Speicher haben, um das Applet herunterzuladen und selbst wenn der Klient genügend Speicher hat, muss der Klient warten, bis das Applet heruntergeladen ist. Da JAVA™ selbst eine Programmiersprache ist, müssen weiterhin Anwendungen in die JAVA™-Sprache umgeschrieben werden, bevor diese in einer Webseite eingebettet werden können.

[0014] Die Druckschrift US 5 548 726 offenbart ein System zum Aktivieren eines neuen Dienstes in einem Klientenservernetzwerk durch ein dynamisches Rekonfigurieren eines Protrokollstapels innerhalb des Serverknotens. Ein Klient, der einen Zugriff auf einen entfernten Dienst wünscht, fragt das geeignete Dienstobjekt von einem Kommunikationsverzeichnisdienst ab und verwendet das Dienstobjekt, um einen Kommunikationspfad einzurichten.

Zusammenfassung der Erfindung

[0015] Gemäß einem ersten Aspekt stellt die vorliegende Erfindung ein Verfahren zum Kommunizieren zwischen einem Anwendungsprogramm, das auf einem Serverknoten zur Ausführung kommt, und einer Mehrzahl von Klientenknoten bereit, wobei das Verfahren folgende Schritte umfasst:
Ausführen eines Anwendungsprogramms auf dem Serverknoten in Reaktion auf eine Anfrage von einem ersten Klientenknoten, um das Anwendungsprogramm auszuführen,
Einrichten einer ersten Verbindung zwischen dem ersten Klientenknoten und dem Serverknoten in Reaktion auf die Anfrage, wobei ein ersten Klientenprotokollstapel auf dem Serverknoten verwendet wird,
Einrichten einer zweiten Verbindung zwischen einem zweiten Klientenknoten und dem Serverknoten, wobei

ein zweiter Klientenprotokollstapel auf dem Serverknoten verwendet wird, in Reaktion auf eine Anfrage von dem zweiten Klientenknoten, um auf das Anwendungsprogramm zuzugreifen, und
im wesentlichen gleichzeitiges Übertragen von Anwendungsdaten, die dem Anwendungsprogramm zugeordnet sind, durch die erste und zweite Verbindung zu dem ersten bzw. zweiten Klientenknoten.

[0016] Gemäß einem zweiten Aspekt stellt die vorliegende Erfindung ein System zum Übertragen von Daten bereit, die einem Anwendungsprogramm zugeordnet sind, zu einer Mehrzahl von Klientenknoten in einem Klientenservernetzwerk, mit:
einem Serverknoten, der ein Anwendungsprogramm ausführt, in Reaktion auf eine Anfrage von einem ersten Klientenknoten, um das Anwendungsprogramm auszuführen,
einer ersten Verbindung zwischen dem Serverknoten und dem ersten Klientenknoten, der in Reaktion auf die Anfrage eingerichtet ist, wobei die erste Verbindung einen ersten Klientenprotokollstapel auf dem Serverknoten aufweist, um Kommunikationen zwischen dem Anwendungsprogramm und dem ersten Klientenknoten zu leiten,
einer zweiten Verbindung zwischen dem Serverknoten und einem zweiten Klientenknoten, der in Reaktion auf eine Anfrage von dem zweiten Klientenknoten eingerichtet ist, um auf das Anwendungsprogramm zuzugreifen, wobei die zweite Verbindung einen zweiten Klientenprotokollstapel auf dem Serverknoten umfasst, der dem ersten Klientenprotokollstapel zugeordnet ist, und
Mittel zum im wesentlichen gleichzeitigen Übertragen von Anwendungsdaten, die dem Anwendungsprogramm zugeordnet sind, zu dem ersten und zweiten Klientenprotokollstapel.

[0017] In einer Ausführungsform betrifft die Erfindung ein Kommunikationssystem und Verfahren zum Verwalten von Kommunikationen zwischen einem Klientenknoten und einem Anwendungsprogramm, das auf einem Serverknoten zur Ausführung kommt.

[0018] Im folgenden wird Bezug genommen auf Beispiele, die zum besseren Verstehen der Erfindung nützlich sein können. Obwohl diese Beispiele außerhalb des Bereichs der Erfindung liegen, betreffen diese Merkmale eines breiteren Systems, in dem die Erfindung zur Verwendung gebracht werden könnte. Diese Beispiel können somit dem Fachmann beim Einfügen der beanspruchten Erfindung in den Gesamtzusammenhang helfen.

[0019] In einem Beispiel umfasst die Erfindung den Schritt des Einrichtens einer Verbindung zwischen dem Klientenknoten und einem allgemeinen Kommunikationsanschluss, der auf dem Serverknoten angeordnet ist. Das Verfahren umfasst weiterhin den Schritt des Erzeugens einer Endpunkt-Datenstruktur, die einen Klientenraum mit der Endpunkt-Datenstruktur in Verbindung bringt und den Schritt des Erzeugens eines Protokollstapels für einen spezifischen Kommunikationsanschluss, der mit dem Anwendungsprogramm in Zusammenhang steht. Eine Benachrichtigung wird einem Verbindungsverwalter der Verbindung gegeben und die Verbindung wird von dem allgemeinen Kommunikationsanschluss zu dem Protokollstapel des spezifischen Kommunikationsanschlusses übertragen.

[0020] In einem weiteren Beispiel umfasst der Schritt des Einrichtens einer Verbindung die Schritte des Empfangens einer Anwendungsanfrage von dem Klientenknoten durch einen Hauptnetzwerk-Informationsknoten, des Bereitstellens einer Serveradresse des Servers mit der angefragten Anwendung für den Klientenknoten durch den Hauptnetzwerk-Informationsknoten, des Empfangens einer Anfrage von dem Klientenknoten durch den Server, um den allgemeinen Kommunikationsanschluss zu verbinden, basierend auf den vorgesehenen Adressen, und des Einrichtens einer Verbindung zwischen dem Klientenknoten und dem allgemeinen Kommunikationsanschluss.

[0021] In einem weiteren Beispiel umfasst das Kommunikationssystem einen Serverknoten und einen Klientenknoten. Der Serverknoten hat einen allgemeinen Kommunikationsanschluss. Der Klientenknoten hat eine Kommunikationseinrichtung, die eine Verbindung zwischen dem Klientenknoten und dem allgemeinen Kommunikationsanschluss des Serverknotens einrichtet. Der Serverknoten umfasst ebenfalls einen Protokollstapel, einschl. einer Endpunkt-Datenstruktur, und einen Klientenraum, der in dem Speicher angeordnet ist. Der Klientenraum ist mit dem Protokollstapel verbunden. Ein Kommunikationsverwalter und eine Benachrichtigungseinrichtung sind auf dem Serverknoten angeordnet. Die Benachrichtigungseinrichtung benachrichtigt den Verbindungsverwalter der Verbindung zwischen dem Klientenknoten und dem allgemeinen Kommunikationsanschluss und in Reaktion darauf überträgt der Kommunikationsverwalter die Verbindung zwischen dem allgemeinen Kommunikationsanschluss und dem Klientenknoten zu dem Protokollstapel. In einer Ausführungsform des Systems umfasst diese einen Multiplexer in Verbindung bzw. Kommunikation mit jedem Protokollstapel einer Mehrzahl von Protokollstapeln.

[0022] In noch einem weiteren Beispiel stellt die Erfindung einen Herstellungsartikel mit computerlesbaren Programmitteln zum Kommunizieren mit einem Klientenknoten bereit, der darauf verkörpert ist. Der Artikel umfasst computerlesbare Programmittel zum Einrichten einer Verbindung mit dem Klientenknoten über einen vorbestimmten Anschluss, computerlesbare Programmittel zum Erzeugen einer Endpunkt-Datenstruktur, computerlesbare Programmittel zum Verbinden eines Speicherraums mit der Endpunkt-Datenstruktur, computerlesbare Programmittel zum Erzeugen eines Protokollstapels, der mit dem Speicherraum und der zugeordneten Endpunkt-Datenstruktur in Verbindung steht, computerlesbare Programmittel zum Benachrichtigen eines Verbindungsverwalters der Verbindung zwischen dem vorbestimmten Anschluss und dem Klientenknoten, und computerlesbare Programmittel zum Übertragen der Verbindung zwischen dem vorbestimmten Anschluss und dem Klientenknoten zu dem zugeordneten Protokollstapel.

[0023] In einem weiteren Beispiel ist ein Verfahren zum Anzeigen einer Ausführungsanwendung in einer dargestellten HTML-Datei vorgesehen, ohne dass erforderlich ist, dass die Anwendung in einer speziellen Sprache erneut geschrieben wird und ohne dass der Browser des betrachtenden Nutzers eine spezialisierte Schnittstelle unterstützen muss. Die Anwendung wird auf dem Server zur Ausführung gebracht, was die Zeit zum Herunterladen und die Beschränkungen des Speichers auf Seiten des Klienten verringert. Weiterhin kann ein Klient eine Ausführung der mehrfachen Anwendungen für mehrfache Seiten aufrufen und sich zwischen den HTML-Dokumenten bewegen, ohne irgendeine der Anwendungen abzuschließen.

[0024] In einem weiteren Beispiel ist ein Verfahren zum Anzeigen einer Ausführungsanwendung in einer HTML-Seite bereitgestellt, was beginnt durch Empfangen einer Eingabe von einem Nutzer, die anzeigt, dass der Nutzer eine Ausführung eines Anwendungsprogramms beginnen möchte. Parameter des Fensters, in dem die Anwendung ausgeführt wird, werden bestimmt, und ein Kommunikationskanal zu dem Anwendungsfenster in der HTML-Seite wird erzeugt. Die Ausgabe des Anwendungsprogramms, das auf einem Server zur Ausführung kommt, wird in dem Anwendungsfenster über den Kommunikationskanal angezeigt.

[0025] In noch einem weiteren Beispiel ist eine Vorrichtung zum Anzeigen einer Ausführungsanwendung in einer HTML-Seite bereitgestellt, die einen Parameterhandhaber und eine Netzwerkexekutive umfasst. Der Parameterhandhaber empfängt Parameter, die mit einem Anwendungsausführungsfenster in Verbindung stehen, das in einer HTML-Datei umfasst ist. Der Parameterhandhaber empfängt Parameter von dem Parameterhandhaber und bewirkt, dass eine Ausführung eines Anwendungsprogramms auf einem Server beginnt und zeigt die Ausgabe der ausführenden Anwendung in dem Anwendungsausführungsfenster an, basierend auf den Parametern, die durch die Netzwerkexekutive von dem Parameterhandhaber empfangen werden.

[0026] In einem weiteren Beispiel hat ein Herstellungsartikel computerlesbare Codemittel zum Anzeigen einer Ausführungsanwendung in einer HTML-Seite, die darauf verkörpert ist. Der Herstellungsartikel umfasst computerlesbare Codemittel zum Empfangen einer Eingabe von einem Klienten, die anzeigt, dass eine Ausführung eines Anwendungsprogramms auf einem Server beginnen sollte. Der Herstellungsartikel umfasst ebenfalls computerlesbare Codemittel zum Bestimmen der Parameter des Fensters, in dem die Ausführungsanwendung angezeigt werden wird. Ebenfalls umfasst sind computerlesbare Codemittel zum Erzeugen eines Kommunikationskanals zu der HTML-Seite unter Verwendung der bestimmten Parameter und computerlesbare Codemittel zum Anzeigen der Ausgabe einer Anwendung, die auf einem Server zur Ausführung kommt, in dem Anwendungsfenster über den Kommunikationskanal.

[0027] In noch einem weiteren Beispiel ist ein System zum Einbetten einer Anwendung in einer HTML-Seite bereitgestellt, das einen Server, eine Netzwerkexekutive, einen Parameterhandhaber und eine HTML-Datei umfasst. Der Server speichert und führt Anwendungsprogramme aus. Die Netzwerkexekutive sendet Befehle zu dem Server, die anzeigen, dass eine Ausführung einer spezifischen Anwendung beginnen sollte, und die Netzwerkexekutive empfängt eine Ausgabe von Anwendungen, die auf dem Server ausgeführt werden. Der Parameterhandhaber empfängt Parameter und reicht diese zu der Netzwerkexekutive weiter. Die HTML-Datei umfasst ein Anwendungsfenster. Das Anwendungsfenster reicht Fensterparameter zu dem Parameterhandheber weiter und empfängt ein Anwendungsprogramm, das von dem Netzwerkausführer ausgegeben wird.

[0028] Ein weiterer Aspekt der Erfindung betrifft ein System und ein Verfahren zum Übertragen derselben Daten zu mehr als einem Klientenknoten im wesentlichen gleichzeitig. In einer bevorzugten Ausführungsform betrifft die Erfindung ein Verfahren zum Übertragen derselben Daten im wesentlichen gleichzeitig von einer Anwendung, die auf einem Serverknoten zur Ausführung kommt, zu zumindest zwei Klientenknoten, die ein verallgemeinertes Empfängerprogramm ausführen. Das Verfahren umfasst die Schritte des Einrichtens einer Verbindung zwischen einem ersten Klientenknoten und einem ersten Klientenprotokollstapel auf dem Serverknoten, des Einrichtens einer Verbindung zwischen der Anwendung, die auf dem Serverknoten zur Ausführung

kommt, und dem ersten Klientenprotokollstapel, des Zuordnen eines ersten minimalen Kommunikationsprotokollstapels zu dem ersten Klientenprotokollstapel, des Einrichtens einer Verbindung zwischen der Anwendung, die auf dem Serverknoten zur Ausführung kommt, und dem ersten minimalen Kommunikationsprotokollstapel, des Einrichtens einer Verbindung zwischen einem zweiten Klientenknoten und einem zweiten Klientenprotokollstapel auf dem Serverknoten, des Zuordnens eines zweiten minimalen Kommunikationsprotokollstapels zu dem zweiten Klientenprotokollstapels, des Bereitstellens einer Verbindung zwischen dem ersten minimalen Protokollstapel und dem zweiten minimalen Protokollstapel, des Bereitstellens einer Verbindung zwischen dem zweiten minimalen Protokollstapel und dem zweiten Klientenprotokollstapel und des Übertragens von Daten von dem Anwendungsprogramm zu dem ersten Klientenprotokollstapel und dem ersten minimalen Protokollstapel, im wesentlichen gleichzeitig.

[0029] Eine weitere bevorzugte Ausführungsform der Erfindung betrifft ein Kommunikationssystem einschl. eines Servers und zweier oder mehrerer Klientenknoten. In einer bevorzugten Ausführungsform umfasst der Serverknoten ein Anwendungsprogramm, einen ersten Klientenprotokollstapel in elektrischer Verbindung mit dem Anwendungsprogramm, einen ersten minimalen Protokollstapel in elektrischer Verbindung mit dem Anwendungsprogramm, einen zweiten minimalen Protokollstapel in elektrischer Verbindung zu dem ersten minimalen Protokollstapel und einen zweiten Klientenprotokollstapel in elektrischer Verbindung zu dem zweiten minimalen Protokollstapel. Zusätzlich umfasst das System einen ersten Klientenknoten in elektrischer Verbindung zu dem ersten Klientenprotokollstapel und einen zweiten Klientenknoten in elektrischer Verbindung zu dem zweiten Klientenprotokollstapel. Daten von dem Anwendungsprogramm werden zu dem Klientenprotokollstapel und dem ersten minimalen Protokollstapel im wesentlichen gleichzeitig übertragen.

[0030] In einem weiteren erläuternden Beispiel ist ein Verfahren bereitgestellt, um entfernt interpretierende Sprachen in einer Klient-Server-Umgebung auszuführen. Der Server, mit dem ein Klient verbunden ist, lädt eine Anwendung herunter und führt diese aus, die in einer interpretierenden Sprache geschrieben ist, wie bspw. ein Java-Applet. Der Server akzeptiert eine Eingabe von dem Klienten und stellt Bildschirmdaten diesem bereit. Dies ermöglicht dem Klienten zu erscheinen, als ob dieser die Anwendung auf eine herkömmliche Weise ausführt, ohne dass erforderlich ist, dass der Klient Rechen- und Speicherressourcen erweitert, die die Anwendung aufnehmen und ausführen. Zusätzlich kann der Server in der Lage sein, die Anwendung schneller als der Klient herunterzuladen. Der Server akzeptiert ebenfalls eine Eingabe von dem Klientenknoten, was dem Klientenknoten ermöglicht, eine Eingabe zu der heruntergeladenen Anwendung zu steuern und bereitzustellen.

[0031] In einem weiteren Beispiel ist ein Verfahren zum entfernten Ausführen einer Anwendung bereitgestellt, die in einer interpretierenden Sprache geschrieben ist, was durch Herunterladen der Anwendung zu einem Serverknoten in Reaktion auf eine Anfrage, die durch einen Klientenknoten durchgeführt wird, beginnt. Eine Verbindung wird zwischen dem Klientenknoten und einem vorbestimmten Kommunikationsanschluss eingerichtet, der auf dem Server angeordnet ist, der Server erzeugt eine Endpunkt-Datenstruktur und verbindet einen Klientenraum, der von dem Server aufgenommen ist, mit der Endpunkt-Datenstruktur. Der Server erzeugt einen Protokollstapel, der mit dem Klientenraum und der zugeordneten Endpunkt-Datenstruktur in Verbindung steht, benachrichtigt einen Verbindungsverwalter der Verbindung und überträgt die Verbindung zwischen dem vorbestimmten Kommunikationsanschluss und dem Klientenknoten zu dem zugeordneten Protokollstapel.

[0032] In einem weiteren Beispiel ist ein Herstellungsgegenstand bzw. -artikel mit computerlesbaren Programmitteln bereitgestellt, die darauf verkörpert sind, zum entfernten Ausführen einer Anwendung, die in einer interpretierenden Sprache geschrieben ist. Der Herstellungsgegenstand umfasst: computerlesbare Programmittel zum Herunterladen der Anwendung zu einem Serverknoten in Reaktion auf eine Anfrage, die von einem Klientenknoten gemacht ist, computerlesbare Programmmittel zum Einrichten einer Verbindung zwischen dem Klientenknoten und einem vorbestimmten Kommunikationsanschluss, der auf dem Serverknoten angeordnet ist, computerlesbare Programmittel zum Erzeugen einer Endpunkt-Datenstruktur, computerlesbare Programmittel zum Zuordnen eines Klientenraums, der von dem Server aufgenommen ist, zu der Endpunkt-Datenstruktur, computerlesbare Programmittel zum Erzeugen eines Protokollstapels, der mit dem Klientenraum und der zugeordneten Endpunkt-Datenstruktur in Verbindung steht, computerlesbare Programmittel zum Benachrichtigen eines Verbindungsverwalters der Verbindung und computerlesbare Programmittel zum Übertragen der Verbindung zwischen dem vorbestimmten Kommunikationsanschluss und dem Klientenknoten zu dem zugeordneten Protokollstapel.

[0033] In noch einem weiteren Beispiel ist ein System zum entfernten Ausführen einer Anwendung bereitgestellt, die in einer interpretierenden Sprache geschrieben ist. Das System umfasst einen Serverknoten mit einem vorbestimmten Kommunikationsanschluss und einem Klientenknoten mit einer Kommunikationseinrich-

tung, die eine Verbindung zwischen dem Klientenknoten und dem vorbestimmten Kommunikationsanschluss des Serverknotens einrichtet. Ein Protokollstapel ist auf dem Serverknoten angeordnet und der Protokollstapel umfasst eine Endpunkt-Datenstruktur. Ein Klientenraum, der in dem Speicher auf dem Server angeordnet ist, ist mit dem Protokollstapel in Verbindung und stellt eine Ausführungsumgebung für eine Anwendung bereit, die in einer interpretierenden Sprache geschrieben ist. Das System umfasst weiterhin einen Kommunikationsverwalter, der auf dem Serverknoten angeordnet ist, und eine Benachrichtigungseinrichtung, die auf dem Serverknoten angeordnet ist. Die Benachrichtigungseinrichtung benachrichtigt den Verbindungsverwalter der Verbindung zwischen dem Klientenknoten und dem vorbestimmten Kommunikationsanschluss und der Kommunikationsverwalter überträgt die Verbindung zwischen dem vorbestimmten Kommunikationsanschluss und dem Klientenknoten zu dem Protokollstapel.

Kurze Beschreibung der Zeichnung

[0034] Die Erfindung ist insbesondere in den beigefügten Ansprüchen hervorgehoben. Die Vorteile dieser Erfindung, die vorstehend beschrieben sind, sowie weitere Vorteile können durch Bezugnahme auf die folgende Beschreibung besser verstanden werden, die in Verbindung zu der beigefügten Zeichnung zu sehen ist.

[0035] Fig. 1 zeigt ein in hohem Maße schematisches Diagramm einer Ausführungsform eines Kommunikationssystems, das die Erfindung verwendet.

[0036] Fig. 2 zeigt ein Blockdiagramm einer Ausführungsform der Erfindung, das die Verbindungen zwischen verschiedenen Komponenten des Servers aus Fig. 1 zeigt, die während einer Kommunikation zwischen den Klienten und dem Server auftreten.

[0037] Fig. 3 zeigt ein Blockdiagramm einer Ausführungsform der Erfindung, die mehrere Klientenknotenverbindungen erhält und verwaltet.

[0038] Fig. 4 zeigt ein Blockdiagramm einer Ausführungsform des Systems zum Einbetten von Anwendungen in einer HTML-Seite.

[0039] Fig. 5 zeigt eine schematische Ansicht eines Klientenknotens.

[0040] Fig. 6 zeigt ein Blockdiagramm einer Ausführungsform der Erfindung, das die Verwendung eines Multiplexers wiedergibt, um dieselben Daten von einem Anwendungsprogramm zu mehr als einem Klienten zu senden.

[0041] Fig. 7 zeigt ein Blockdiagramm der Ausführungsform der Erfindung, bei dem die Sendefähigkeiten durch Ausgangsfächerung (fan out) erhöht werden.

Ausführliche Beschreibung der Erfindung

[0042] In Fig. 1 umfasst in Kurzübersicht ein typisches Netzwerk **20** zumindest einen Klientenknoten **24**, zumindest einen Serverknoten **34**, **34'** und einen Haupt- bzw. Masternetzwerkinformationsknoten **40**, die über eine Kommunikationsverbindung bzw. eine Nachrichtenverbindung **44** miteinander verbunden sind. Die in Fig. 1 gezeigte Ausführungsform stellt die Kommunikationsverbindung **44** als einen lokalen Netzwerkring oder LAN-Ring dar, aber irgendeine Kommunikationstopologie kann verwendet werden. Zur Erklärung wird angenommen, dass der Serverknoten die Anwendung **30** hat, die durch den Klientenknoten **24** angefordert hat. Ebenfalls zu Erklärungszwecken wird angenommen, dass der Masternetzwerkinformationsknoten **40** ein eindeutiger bzw. verschiedener Serverknoten ist, aber in der Realität kann der Masternetzwerkinformationsknoten **40** ein Anwendungsausführungsserverknoten **34** sein. Es sollte beachtet werden, dass bei einem gegebenen LAN verschiedene Knoten in der Lage sein können, als ein Netzwerkinformationsknoten zu funktionieren, aber zu einem gegebenen Zeitpunkt ist nur einer von solchen Knoten als der Masternetzwerkinformationsknoten **40** für das System **20** vorgesehen und es ist dieser Knoten, zu dem Klientenaufforderungen nach Serverinformationen gerichtet sind.

[0043] Der Masternetzwerkinformationsknoten **40** hält eine Adresstabelle für die Anwendungsausführungsserverknoten **34**, **34'**. Zusätzlich empfängt der Masternetzwerkinformationsknoten **40** Nachrichten von jedem Anwendungsausführungsserverknoten **34**, **34'**, die deren Aktivitätsniveau anzeigen. Das Aktivitätsniveau der Anwendungsausführungsserverknoten **34**, **34'** wird in einer Tabelle zusammen mit der Adresse jedes der Anwendungsausführungsserverknoten (**34**) gehalten und wird durch das Kommunikationssystem **44** für eine Be-

lastungsverteilung bzw. Niveauregulierung verwendet.

[0044] Wenn der Klient **24** wünscht, dass eine Anwendung auf einem Anwendungsausführungsserverknoten **34** ausgeführt wird, sendet der Klientenknoten **24** eine Aufforderung bzw. Abfrage zu dem allgemeinen Kommunikationsanschluss, der vorab durch das Kommunikationsprotokoll definiert wird, oder zu dem "wohlbekannten" Kommunikationsanschluss auf dem Masternetzwerkinformationsknoten **40**. In einer Ausführungsform findet die Kommunikation im Wege eines Datagrammdienstes statt. Der Masternetzwerkinformationsknoten **40** greift auf die Tabelle der Serveradressen zu und gibt eine Nachricht zurück, die die Adresse des Anwendungsausführungsservers oder des Anwendungsservers **34** enthält, der die angeforderte Anwendung hat, und ebenfalls derjenige, der die geringste Belastung hat. Nachfolge Kommunikationen werden automatisch durch den Klienten ebenfalls zu einem "wohlbekannten" oder vordefinierten allgemeinen Kommunikationsanschluss auf dem Serverknoten **34** adressiert. In einer Ausführungsform bestimmt der Typ des Protokolls, mit dem die anfängliche Abfrage zu dem Masternetzwerkinformationsknoten **40** durchgeführt wurde, das Protokoll der Informationen, die durch den Masternetzwerkinformationsknoten **40** zu dem Klientenknoten **24** zurückgegeben werden. Somit würde, wenn die Anfrage unter Verwendung eines TCP/IP-Datagramms durchgeführt würde, der Masternetzwerkinformationsknoten **40** die TCP/IP-Adrresse des Servers **34** zu dem Klientenknoten **24** zurückgeben und der Klientenknoten **24** würde nachfolgend einen Kontakt mit dem Serverknoten **34** unter Verwendung dieses Protokolls einrichten. In einer weiteren Ausführungsform umfasst das Datagramm, dass eine Anwendungsadresse durch einen Klienten **24** abfragt, eine Anfrage bzw. Aufforderung für einen unterschiedlichen Typ eines Protokolls als dasjenige, das verwendet wurde, um die Anfrage zu dem Masternetzwerkinformationsknoten **40** zu senden. Beispielsweise kann der Klient **24** eine Aufforderung zu dem Masternetzwerkinformationsknoten **40** unter Verwendung des IPX-Protokolls durchführen und die Adresse des Anwendungsservers als eine TCP/IP-Protokolladresse abfragen.

[0045] Wenn ein Klientenknoten **24** (tatsächlich ein Klientenprozess **56** auf einem Klientenknoten **24**) wünscht, mit einer Anwendung auf einem Serverknoten **34**, **34'** zu kommunizieren, beginnt der Klientenknoten **24**, indem eine Netzwerkaufforderung herausgegeben wird, den Ort des Servers **34** mit der gewünschten Anwendung zu bestimmen. Diese Aufforderung wird durch den Masternetzwerkinformationsknoten **40** (ebenfalls als ein Netzwerkbrowser **40** bezeichnet) empfangen, der irgendwo auf dem Netzwerk sitzt. In dieser Fig. 1 ist der Netzwerkbrowser **40** zur Vereinfachung als auf einem verschiedenen Server **40** als der Server, der die Anwendung hat, sitzend gezeigt, aber dies muss im allgemeinen nicht der Fall sein.

[0046] Der Netzwerkinformationsknoten **40** gibt die Netzwerkadresse des Serverknotens **34**, der die gewünschte Anwendung **30** hat, zu dem Klientenknoten **24** zurück. Der Klientenknoten **24** verwendet dann die von dem Netzwerkmasterinformationsknoten **40** empfangenen Informationen, um eine Verbindung zu der Anwendung aufzurufen, die auf dem spezifizierten Server **34** ausgeführt wird. Wie vorstehend beschrieben ist, wird eine solche Verbindung zunächst zu einem "wohlbekannten" Kommunikationsanschluss eingerichtet und wird später zu einem spezifischen Kommunikationsanschluss unter Kontrolle eines Verbindungsverwalters übertragen. Der spezifische Kommunikationsanschluss ist der Anwendung zugeordnet, die auf dem Serverknoten **34** ausgeführt wird, der dann mit dem Klientenknoten **24** durch den spezifischen Kommunikationsanschluss kommuniziert.

[0047] Genauer gesagt und unter Bezugnahme auf Fig. 2, der Klientenprozess **56** auf dem Klientenknoten **24** führt eine Aufforderung **54** zu dem Netzwerkmasterinformationsknoten **40** durch, um die Adresse eines Serverknotens **34** zu erhalten, der die gewünschte Anwendung **62** umfasst. Der Netzwerkmasterinformationsknoten **40** gibt eine Nachricht **58** zu dem Klientenknoten **24** zurück, die die Adresse des Serverknotens **34** enthält, der die Serveranwendung **62** umfasst. In einer Ausführungsform ist das bei diesem Punkt der Verbindung verwendete Protokoll ein Datagrammdienst.

[0048] Der Klientenknoten **24** verwendet die zurückgegebene Adresse, um einen Kommunikationskanal **68** mit dem Server **34** einzurichten. Die Anschlusszahl, die durch den Klienten **24** verwendet wird, entspricht dem "wohlbekannten" Anschluss in dem Server **34**, der durch das Netzwerkprotokoll als der Anschluss definiert wird, durch den der Server **34** Kommunikationsverbindungen mit Klienten **24** einrichtet. Der wohlbekannte Anschluss **72** hat einen rudimentären Protokollstapel **76**, der vornehmlich eine Endpunktdatenstruktur **78** umfasst.

[0049] Die Endpunktdatenstruktur **78** zeigt zu dem Kommunikationsprotokollstapel **76** und einer Klientenverbindung, wodurch eine eindeutige Repräsentation oder "Kennung" für den Klienten **24** eingerichtet wird. Die Endpunktdatenstruktur **78** ermöglicht, dass die Verbindung zwischen dem Server **34** und dem Klienten **24** nach Belieben zwischen dem Verbindungsverwalter **80** und verschiedenen Anwendungen **62** auf dem Server **34** be-

wegt wird. Die Endpunktdatenstruktur **78** enthält in einer Ausführungsform nicht nur die Kennung für den Klienten **24**, sondern kann auch andere Informationen enthalten, die die Klientenverbindung betreffen. In der dargestellten Ausführungsform überwacht der Anwendungsserver **34** eine Aktivität auf einem spezifischen Kommunikationssystem (bspw. LAN oder WAN) und hat diesen minimalen Protokollstapel **76** mit lediglich den notwendigen Protokollmodulen initialisiert, die benötigt werden, um einen "TTY"-Kommunikationsmodus zu unterstützen. Der "TTY"-Kommunikationsmodus ist ein einfacher ASCII-Strom ohne Protokollannahmen oberhalb der Transportschicht. Das bedeutet, es gibt keine Protokollschichten für eine Kompression, Verschlüsselung, Zuverlässigkeit, Rahmenbildung oder Präsentation von gesendeten Daten. Somit richtet ein Klientenknoten **24**, der eine Anwendung **62** sucht, die auf dem Server **34** läuft, eine Verbindung zu dem wohlbekannten Kommunikationsanschluss **72** mit dem minimalen Protokollsatz ein, der benötigt wird, um einen TTY-Kommunikationsmodus zu unterstützen.

[0050] Ein Verbindungsverwalter **80**, der auf dem Serverknoten **34** ausgeführt wird, "hört" bei dem wohlbekannten Kommunikationsanschluss **72** für eine Verbindungsaufforderung **68** mit. Wenn eine Verbindungsaufforderung **68** von dem Klientenknoten **24** empfangen wird, wird der Verbindungsverwalter **80** benachrichtigt **84**. Der Verbindungsverwalter **80** weiß, basierend auf der Benachrichtigung **84**, welches Protokoll verwendet wird.

[0051] Mit dieser Information erzeugt der Verbindungsverwalter **80** einen neuen minimalen Protokollverbindungsstapel **104**, beginnt die Ausführungsumgebung **96** und verknüpft den neuen minimalen Protokollstapel **104** mit der Ausführungsumgebung **96**. In einer Ausführungsform umfasst der Server **34** eine Anzahl von Ausführungsumgebungen **96**, die vorab gestartet wurden, aber die nicht einem Kommunikationsanschluss zugeordnet wurden. In dieser Ausführungsform ermöglicht das Starten der Ausführungsumgebungen vor dem Verbinden eine schnellere Antwortzeit, als wenn jede Ausführungsumgebung **96** gestartet wird, wenn die Verbindungsaufforderung von dem Klienten **24** empfangen wird. Wenn die Ausführungsumgebung **96** gestartet wird, wird die Serveranwendung **62**, die durch den Klienten **24** abgerufen wird, ebenfalls gestartet. In einer weiteren Ausführungsform wird, wenn der Klient **24** eine Anwendung nicht spezifiziert, entweder eine Standardanwendung gestartet oder einfach die Ausführungsumgebung **96** mit keiner Anwendung gestartet.

[0052] Der Verbindungsverwalter **80** bewegt dann die Klientenverbindung, einschließlich der eindeutigen Klientenkennzeichnung oder Kennung, von dem wohlbekannten Anschluss **96** zu dem neuen minimalen Protokollstapel **104**. Der Verbindungsverwalter **80**, der den minimalen Protokollstapel verwendet, sendet einen TTY-Datenstrom, der anzeigt, dass ein Dienst verfügbar ist. Somit ist dieses Verfahren zum Erfassen einer Klientenverbindung unabhängig von dem Anschluss, zu dem die Verbindung zunächst eingerichtet wird. Wenn der Klientenknoten **24** nicht innerhalb eines vorgeschriebenen Zeitraums (bspw. 5 Sekunden) auf die Nachricht, dass der Dienst verfügbar ist, antwortet, wird ein erneutes Senden der "Dienst verfügbar"-Nachricht durch den Server **34** durchgeführt.

[0053] Wenn der Klient **24** die Nachricht empfängt, sendet der Klient **24** eine TTY-Kette, die anzeigt, dass die "Dienst verfügbar"-Nachricht erfasst wurde. Der Klient **24** wartet darauf, dass der Server **34** antwortet, und wenn die Antwort nicht innerhalb eines vorgegebenen Zeitraums (bspw. 5 Sekunden) liegt, sendet der Klient **24** erneut die Nachricht. Der Verbindungsverwalter **80** fragt dann den Klienten **24** ab **90**, indem er die Standardkommunikationsparameter fordert. Diese Abfrage **90** nimmt die Form einer Nachricht an, die zurück zu dem Klienten **24** gereicht wird und die anzeigt, dass der Klient **24** antworten sollte, mit Details, die berücksichtigen, welches Protokoll der Klient **24** in der Verbindung benutzen möchte.

[0054] In Antwort darauf sendet der Klient **24** einen Satz von Protokollpaketen **92**, wobei jedes Paket davon verwendet wird, um ein erforderliches oder optionales Protokollmodul zu spezifizieren, das von dem Server **34** abgerufen wurde. In einer Ausführungsform ist die Anzahl an Paketen in dem Satz variabel, wobei ein Paket für jedes abgerufene Protokoll gesendet wird. In einer weiteren Ausführungsform ist die Anzahl an Paketen, die gesendet werden, in dem Kopf des ersten Pakets enthalten. In einer dritten Ausführungsform ist die verbleibende Anzahl an Paketen, die gesendet werden, in dem Kopf jedes Paktes enthalten und wird mit jedem nachfolgend gesendeten Paket dekrementiert. Somit kann der Klient **24** auf die Abfrage **90** durch Anzeigen antworten, dass bspw. eine Verschlüsselung und Datenkompression verwendet werden wird. In einem solchen Fall werden zwei Protokolle von den Klienten **24** zu dem Server **34** gesendet, und in einer Ausführungsform wird der Kopf des ersten Pakets die Anzahl an Paketen als zwei anzeigen.

[0055] Sobald die Antworten auf die Abfrage **90** empfangen wurden, bildet der Verbindungsverwalter **80** einen Protokollstapel, der Protokolltreiber **120**, **120'**, **120"** verwendet, die den Protokollen entsprechen, die durch die Klientenknoten **24** abgefragt werden. In einer Ausführungsform plaziert der Verbindungsverwalter **80** jeden der erforderlichen Protokolltreiber **120**, **120'**, **120"**, die den abgerufenen Klientenprotokollen entsprechen (bspw.

ein Verschlüsselungstreiber, wenn eine Verschlüsselung durch den Klienten gewünscht ist), in den Protokollstapel-"Behälter" **112** und verbindet diese miteinander. Dieser dynamische Prozess ermöglicht, dass ein Klientenknoten **24** den Inhalt eines Protokollstapels dynamisch spezifiziert, ohne dass erforderlich ist, dass der Server **34** eine vorherige Beschreibung des Protokollstapels für einen bestimmten Klientenknoten **24** hat. Durch Verwendung dieses Verfahrens können mehrere Klienten **24** durch einen einzelnen Server bedient werden, selbst wenn die einzelnen Klienten **24** erheblich unterschiedliche Anforderungen für den zugeordneten Kommunikationskanal haben. In der gezeigten Ausführungsform ist jeder Klient **24**, **24'**, **24''** einem jeweiligen Kommunikationsprotokollstapel **104**, **104'** und **104''** zugeordnet. Solche dynamisch erweiterbaren Protokollstapel sind detaillierter nachfolgend und in der US-Patentanmeldung Nr. 08/540891, angemeldet am 11. Oktober 1995, beschrieben.

[0056] In der gerade erörterten Ausführungsform ist der "Behälter" **112** ein Nutzerniveau- oder Kernniveauvorrichtungstreiber, wie bspw. ein NT™-Vorrichtungstreiber. Dieser Behältertreiber stellt zusätzliche Unterstützung für die inneren Protokollmodule oder "Treiber" (im allgemeinen **120**) bereit, die den Protokollanforderungen des Klientenknotens **124** entsprechen. Die zusätzliche Unterstützung ist in der Form von Helferroutinen, die bspw. einem Protokolltreiber helfen, Daten zu dem nächsten Treiber zu übertragen. Alternativ ist in einer weiteren Ausführungsform jeder Protokolltreiber ein vollständiger Nutzerniveau- oder Kernniveautreiber in sich selbst.

[0057] Unter Bezugnahme auf die Ausführungsform, die in Fig. 3 dargestellt ist, umfasst der Kommunikationsverwalter **60** zwei Hauptsoftwaremodule: ICASRV.EXE **90** und ICAAPI.DLL **94**. In der dargestellten Ausführungsform ist ICASRV.EXE **90** die Serverseite einer Klienten/Serverschnittstelle. ICASRV.EXE **90** verwaltet alle Kommunikationszustände und ist in einer Ausführungsform als ein Windows NT™-Dienst implementiert. Ein zweiter Teil des Verbindungsverwalters **60** ist ICAAPI.DLL **94**. ICAAPI.DLL **94** richtet die Verbindung mit dem Klienten ein, richtet die Protokolle ein, die zu verwenden sind, und benachrichtigt ICASRV.EXE **90** von der Vervollständigung des Protokollstapels. In einer Ausführungsform ist ein drittes Modul CDMODEM.DLL **96** mit ICAAPI.DLL **94'** verbunden. CDMODEM.DLL **96** ist ein Modul, das ICAAPI.DLL **94'** verwendet, um mit Modemvorrichtungen zu kommunizieren.

[0058] Die Verbindungsmethodik, die vorstehend beschrieben ist, kann für einen Klienten **24** verwendet werden, der auf einem Webbrowserprogramm läuft. Zu Zwecken dieser Spezifikation wird der Nutzer, der das Webbrowserprogramm betätigt, als der "betrachtende Nutzer" bezeichnet. Die Ausdrücke "Server" oder "Serverknoten" werden verwendet, um auf Maschinen zu verweisen, die HTML-Dateien oder Anwendungen enthalten, die ausgeführt werden können. Beispielsweise führt ein betrachtender Nutzer einen Webbrowser auf einem Klientenknoten aus und führt Dateiabfragen über das HTTP-Protokoll zu Servern durch. Die Server antworten durch Übertragen von Dateidaten zu dem Klienten über das HTTP-Protokoll. Der Webbrowser, der auf dem Klienten abläuft, empfängt die übertragenen Daten und stellt die Daten als eine HTML-Seite dem betrachtenden Nutzer dar.

[0059] In einer kurzen Übersicht und unter Bezugnahme auf Fig. 4 umfasst eine HTML-Datei **64**, die auf einem Server **34'** angeordnet und in Übereinstimmung mit einer Ausführungsform der Erfindung aufgebaut ist, eine generische eingebettete Fensterkennzeichnung **66**. Die generische eingebettete Fensterkennzeichnung **66** ist irgendein Datenkonstrukt, das einem Browser **60**, der die HTML-Datei **64** anzeigt, andeutet, dass ein generisches eingebettetes Fenster **66'** bei einem bestimmten Ort in der HTML-Seite **64'**, die durch die HTML-Datei **64** beschrieben ist, angezeigt werden sollte. Die generische eingebettete Fensterkennzeichnung **66** kann zusätzlich Informationen enthalten, wie bspw. Höhe des Fensters, Breite des Fensters, Randgestaltung des Fensters, Hintergrundfarbe oder Muster in dem Fenster, welche Anwendungen in dem Fenster angezeigt werden können, wie oft die Ausgabeanzeige aktualisiert werden sollte oder irgendwelche anderen zusätzlichen Informationen, die nützlich sind, um eine Anzeige der Anwendungsausgabe zu verbessern.

[0060] Einige Beispiele von generischen eingebetteten Fensterkennzeichnungen, die in einer HTML-Datei eingebettet werden können, folgen:

AcitveX™ tag

```
<object classid="clsid:238f6f83-b8b4-11cf-8771-
00a024541ee3"
data="/ica/direct.ica"CODEBASE="/cab/wfica.cab"
     width=436 height=295>
     <param name="Start" value="Auto">
     <param name="Border" value="On">
</object>
```

Netscape Plugin™ tag

```
<embed src=http://www.citrix.com/ica/direct.ica

pluginspage="http://www.citrix.com/plugin.html"
     height=295 width=436 Start=Auto Border=On>
<embed>
```

JAVA™ tag

```
<applet code=JICA.class width=436 height=295>

<param name=Address      value="128.4.1.64">
<param name=InitialProgram value=
     Microsoft Word 7.0>
<param name=Start             value=Auto>
<param name=Border            value=On>
</applet>
```

[0061] In jedem vorstehend Fall zeigt die Kennzeichnung an, dass ein Fenster mit einer Höhe von 295 Bildpunkten bzw. Pixeln und einer Breite von 436 Bildpunkten gezeichnet werden sollte, um eine Anwendungsausgabe zu empfangen. Jede Kennzeichnung spezifiziert ebenfalls, dass die Anwendung automatisch eine Ausführung starten sollte und dass das Fenster, in dem die Anwendungsausgabe angezeigt wird, mit einem Rand gezeichnet werden sollte. Bei den ActivX™- und Netscape Plugin™-Kennzeichnungen sind die entfernten Anwendungsparameter in der Datei "direct.ica" spezifiziert, die in dem Verzeichnis "/ica." angeordnet sind. Die JAVA™-Kennzeichnung spezifiziert die entfernten Anwendungsparameter direkt. In dem vorstehenden Beispiel ist die Adresse des Servers, der die Anwendung hat, sowie der Name der Anwendung, die auszuführen ist, spezifiziert.

[0062] Die Browseranwendung **60** greift auf die HTML-Datei **64** durch Ausgeben einer Aufforderung zu einer spezifischen Internetadresse (URL: Uniform Resource Locator) zu. Der Server **34'**, der die HTML-Datei **64** hat, überträgt die Daten der HTML-Datei **64** zu der Browseranwendung **60**, die Text anzeigt und jede Kennzeichnung übersetzt, die in der HTML-Datei **64** enthalten ist. Die Browseranwendung **60** zeigt die Daten der HTML-Datei **64** als eine HTML-Seite **64'** an. Wenn eine generische eingebettete Fensterkennzeichnung **66** in der HTML-Datei **64** vorliegt, wie bspw. die Kennzeichnungen, die vorstehend beschrieben sind, zeichnet der Browser **60** ein leeres Fenster **66'** in der angezeigten HTML-Seite **64'**.

[0063] Eine Ausführung der gewünschten Anwendung **62'** kann unmittelbar bei Anzeigen der HTML-Seite **64'**

beginnen oder eine Ausführung kann irgendein Signal abwarten, bspw. eine spezifizierte Nutzereingabe, die anzeigt, dass eine Ausführung der Anwendung **62'** beginnen sollte. Sobald die Ausführung der Anwendung **62'** begonnen hat, instanziiert die Browseranwendung **60** einen Parameterhandhaber bzw. -treiber **40**, der dem Anwendungsfenster **66'** zugeordnet ist. Die Instanz des Parameterhandhabers **40** kann als ein abgeleiteter Prozess der Browseranwendung **60** erzeugt werden, als ein gleichrangiger bzw. gleichschichtiger Prozess der Browseranwendung **60** oder als eine dynamisch verbundene Bibliothek (DLL: Dynamically Linked Library), die der Browseranwendung **60** zugeordnet ist.

[0064] Die Browseranordnung **60** reicht jeden spezifischen Parameter, der mit dem Anwendungsfenster **66'** in Verbindung steht, die durch die Kennzeichnung des generischen eingebetteten Fensters **66** bereitgestellt wurden, zu der Instanz des Parameterhandhabers **40**. Zusätzlich kann die Browseranwendung **60** die Kennung für das Anwendungsfenster **66'** zu der Instanz des Parameterhandhabers **40** reichen oder die Instanz des Parameterhandhabers **40** kann die Browseranwendung **60** abfragen, um die Kennung für das Anwendungsfenster **66'** wiederzuerlangen. Die Instanz des Parameterhandhabers **40** erzeugt ebenfalls einen Netzwerkausführer bzw. eine Netzwerkexekutive **50**. Die Netzwerkexekutive **50** kann als ein abgeleiteter Prozess der Instanz des Parameterhandhabers **40** oder als ein gleichrangiger Prozess der Instanz des Parameterhandhabers **40** erzeugt werden.

[0065] Die Instanz des Parameterhandhabers **40** leitet jeden Parameter des spezifizierten Anwendungsfensters **66'** zu der Netzwerkexekutive **50** weiter. Parameter, die nicht durch die Instanz des Parameterhandhabers **40** oder die Kennzeichnung **66** des eingebetteten generischen Fensters spezifiziert sind, können auf Standardwerte gesetzt werden. Die Netzwerkexekutive **50** kann bestimmte Parameterstandards fest programmiert haben, oder die Netzwerkexekutive **50** kann auf eine Datei zugreifen, die die Parameterstandardwerte enthält.

[0066] Die Netzwerkexekutive **50** erzeugt ihr eigenes Anwendungsausgabefenster **66"**. Die Netzwerkexekutive **50** erzeugt ihr Anwendungsausgabefenster **66"** als eine Ableitung des angezeigten Anwendungsfensters **66'** und zeigt ihr Anwendungsausgabefenster **66"** direkt über dem Stammfenster **66'**, das durch die Browseranwendung **60** gezeichnet wird. Da das Anwendungsausgabefenster **66"**, das durch die Netzwerkexekutive **50** gezeichnet wird, eine Ableitung des Anwendungsfensters **66** ist, das durch die Browseranwendung **60** gezeichnet wird, erbt das Anwendungsausgabefenster **66"** verschiedene Größen von seinem Stamm, einschließlich Positionsinformationen. Daher wird das Anwendungsausgabefenster **66"** dem Anwendungsfenster **66'** folgen, wenn der betrachtende Nutzer den Bildschirm der Browseranwendung **60** verschiebt oder andere Aktionen durchführt, die die Position des Anwendungsfensters **66'** verändern.

[0067] Die Netzwerkexekutive **50** richtet ebenfalls einen Kommunikationskanal mit dem Server **60** ein und ruft eine Ausführung der gewünschten Anwendung **62'** durch den Server **34"** unter Verwendung der Verbindungsmethodik, die vorstehend beschrieben ist, auf. Die Netzwerkexekutive **50**, die als der Klient bei der vorstehenden Beschreibung dient, reicht jeden Parameter, den diese empfängt, von der Instanziierung des Parameterhandhabers **40** zu dem Server weiter, zusammen mit allen notwendigen Standardwerten. Wenn ein Parameter nicht zu dem Server weitergeleitet wird, kann der Server den Parameter abfragen, wenn es ein notwendiger Parameter ist, für den es keinen Standardwert gibt, bspw. Nutzer-ID, oder dieser kann einen Standardwert für den Parameter bereitstellen, bspw. Ausführungspriorität. Der Server **34"** beginnt eine Ausführung des gewünschten Anwendungsprogramms **62'** und richtet die Ausgabe zu der Netzwerkexekutive **50**. Die Netzwerkexekutive **50** empfängt Daten von dem Anwendungsprogramm **62'** und zeigt die Ausgabedaten in ihrem Anwendungsausgabefenster **66"** an. Da das Anwendungsausgabefenster **66"** im oberen Bereich von dem Anwendungsfenster **66'** gezeichnet ist, das durch die Browseranwendung **60** gezeichnet wird, werden die Anwendungsausgabedaten in der HTML-Seite **64'** angezeigt. Wie vorstehend bemerkt ist, ist das Anwendungsausgabefenster **66"**, das durch die Netzwerkexekutive **50** gezeichnet wird, von dem Anwendungsfenster **66'** abgeleitet, das durch die Browseranwendung **60** gezeichnet wird. Dies ermöglicht, das Anwendungsausgabefenster **66"** zu verschieben, wie die HTML-Seite **64'** verschoben wird.

[0068] Das Anwendungsausgabefenster **66"** empfängt ebenfalls eine Eingabe von dem betrachtenden Nutzer. Unbearbeitete Eingabedaten, bspw. ein Mausklick, werden in das Anwendungsausgabefenster **66"** durch die Netzwerkexekutive **50** empfangen. Die Netzwerkexekutive **50** leitet die unbearbeiteten Eingabedaten zu der Anwendung **62'** weiter, die auf dem Server **34"** zur Ausführung kommt. Auf diese Weise ist der betrachtende Nutzer in der Lage, mit der Anwendung **62'** über die HTML-Seite **64'** zusammenzuwirken oder zu interagieren.

[0069] Wie in **Fig. 5** gezeigt ist, verwendet der betrachtende Nutzer ein so genanntes "Browserprogramm", um die HTML-Seite **64'** mit einem Anwendungsfenster **66'** auf dem Bildschirm **18** des Computers **14** des Nut-

zers anzuzeigen. Der betrachtende Nutzer kann eine Ausführung eines Anwendungsprogramms **62'** aufrufen. Typischerweise wird dies durch den Nutzer, der eine "Mausklick-Schnittstelle" verwendet, durchgeführt, d.h., der betrachtende Nutzer verwendet eine Maus **16**, um einen Cursor bzw. eine Schreibmarke **12** zu manipulieren, der bzw. die ebenfalls auf dem Bildschirm **18** des Computers **14** des betrachtenden Nutzers angezeigt ist. Sobald der Cursor **12** über einem bestimmten Abschnitt der HTML-Seite **64'** ist, zeigt der betrachtende Nutzer durch "Klicken" einer Taste **15** auf der Maus **16** an. Alternativ kann der betrachtende Nutzer ebenfalls durch Drücken einer Taste auf einer zugeordneten Tastatur **17** signalisieren bzw. anzeigen, wie bspw. die Eingabe- bzw. "Returntaste". In anderen Ausführungsformen muss der Nutzer überhaupt keine Maus **16** benutzen, sondern er kann anstelle dessen ein Berührungsfeld, eine Rollkugel, ein berührungsempfindliches Tablett und Stift oder irgendeinen anderen Eingabemechanismus zum Manipulieren des Cursors **12** verwenden.

[0070] In einer weiteren Ausführungsform kann das Anwendungsfenster **66'** oder ein weiterer Abschnitt der HTML-Seite **64'** einen "Kontrollbereich" (hot zone) definieren. Wenn der betrachtende Nutzer den Cursor **12** in den "Kontrollbereich" bewegt, wird eine Ausführung der Anwendung **62'** auf dem Server **34''** gestartet.

[0071] Sobald der betrachtende Nutzer angezeigt hat, dass eine Ausführung die Anwendung **62'** beginnen sollte, instanziiert die Browseranwendung **60** einen Parameterhandhaber **40** und reicht die Instanziierungsparameter, die mit dem Anwendungsfenster **66'** in Verbindung stehen, durch die generische eingebettete Fensterkennzeichnung **66** weiter. Die Instanz des Parameterhandhabers **40** erzeugt eine Netzwerkexekutive **50** und reicht zu dieser die Parameter des Anwendungsfensters **66'**. Die Netzwerkexekutive **50** bestimmt, welche Anwendung **62'** aufzurufen ist, und auf welchem Server **34''** diese Anwendung **62'** sitzt. Im allgemeinen wird diese Information zu dieser durch die Instanz des Parameterhandhabers **40** gereicht, die diese von der Browseranwendung **60** in der Form der generischen eingebetteten Fensterkennzeichnung **66** bekommt. Aber die Netzwerkexekutive **50** kann es erfordern, einen Masternetzwerkinformationsknoten **40** oder andere verschiedene Server abzufragen, um zu bestimmen, welche Server, wenn überhaupt einer, die gewünschte Anwendung **62'** haben. Die Netzwerkexekutive **50** beginnt dann eine Ausführung der Anwendung und zeigt die Ausgabe des Anwendungsprogramms **62'** in dem Anwendungsfenster **66'** an, wie vorstehend ausführlich beschrieben ist.

[0072] Die Netzwerkexekutive **50** fährt damit fort, eine Anwendungsausgabe in dem Anwendungsausgabefenster **66'** direkt anzuzeigen, bis der betrachtende Nutzer andeutet, dass eine Ausführung der Anwendung **62'** enden sollte, bspw. durch Schließen des Anwendungsfensters **66'**, oder bis der betrachtende Nutzer auf eine Kennzeichnung klickt, die anzeigt, dass eine verschiedene HTML-Seite angezeigt werden sollte. Wenn dies passiert, kann eine Ausführung der Anwendung **62'** beendet werden. Es ist jedoch bevorzugt, die Verbindung in einem Zwischen- bzw. Cachespeicher abzulegen. In der Tat wird die Instanz des ersten Parameterhandhabers **40** nicht unmittelbar beendet. Die Anwendung **62'** fährt jedoch damit fort, mit einem verringerten Prioritätsniveau auszuführen, d.h. in einem "Hintergrundmodus", da der erste Parameterhandhaber **40** nicht länger im "Fokus" steht.

[0073] Im allgemeinen ist es erwünscht, ein in dem Zwischenspeicher Ablegen der Verbindung zu erreichen, indem der Quellcode des Parameterhandhabers **40** mit einer global verfügbaren Datenstruktur zum Registrieren von Instanzen bereitgestellt wird. Beispielsweise kann der Parameterhandhaber **40** mit einer global verfügbaren verbundenen Listendatenstruktur, einem Datenfeld, einer Datentabelle oder einer anderen Datenstruktur versehen sein. Da die Datenstruktur global verfügbar ist, ist jede Instanz des Parameterhandhabers **40** in der Lage, die Datenstruktur zu lesen und zu schreiben. Dies ermöglicht jeder Instanz des Parameterhandhabers **40** mit jeder anderen Instanz zu "registrieren", indem zu der Datenstruktur geschrieben wird, um seine Existenz zu signalisieren.

[0074] Bei Ausführungsformen, bei denen keine weiteren Verbindungsinformationen gespeichert sind, kann eine vorbestimmte Begrenzung der Anzahl von Verbindungen, die in einem Zwischenspeicher zu irgendeiner Zeit abgelegt sein können, gesetzt sein. In diesen Ausführungsformen wird, wenn eine Registrierung einer Instanz zu einer erhöhten Anzahl von abgelegten Verbindungen führen würde, eine der "abgelegten" (cached) Verbindungen entfernt, d.h. die Instanziierung des Parameterhandhabers **40**, die mit der Verbindung in Zusammenhang steht, wird benachrichtigt, dass diese beenden sollte. Vor Beendigung benachrichtigt der Parameterhandhaber **40** seine zugeordnete Netzwerkexekutive **50**, dass diese beenden sollte. Die Netzwerkexekutive **50** schließt wiederum ihre Sitzung mit dem Server, der das Anwendungsprogramm **62'** hat, und bricht dann ab.

[0075] Bei Ausführungsformen, bei denen andere Informationen gespeichert sind, können die zusätzlichen Informationen verwendet werden, um die abgelegten Verbindungen effektiver zu verwalten. Beispielsweise ist, wenn ein Nutzer nicht aktiv eine HTML-Seite **64'** in einer vorbestimmten Anzahl von Minuten betrachtet hat, bspw. 10 Minuten, die Instanziierung des Parameterhandhabers **40** instruiert abzubrechen. Die Sitzung mit

dem beherbergenden Server wird abgebrochen und die Instanz des Parameterhandhabers **40** entfernt deren Eintrag in der Registratur.

[0076] Abgelegte bzw. zwischengespeicherte Verbindungsinformationen können verwaltet werden, indem irgendein bekanntes Zwischenspeicherverwaltungsschema verwendet wird. Verbindungseinträge können auf einer "zuerst herein, zuerst heraus"-Basis verworfen werden, d.h. die ältesten Einträge werden zu jedem Zeitpunkt verworfen, wenn ein neuer Eintrag hinzugefügt werden muss. Alternativ können zwischengespeicherte Verbindungsinformationseinträge auf einer "am wenigsten neulich verwendet"-Basis verworfen werden, was Informationen verwirft, die Verbindungen betreffen, die im gerinsten Umfang durch den Nutzer verwendet wurden. Andere Zwischenspeicherverwaltungstechniken, wie bspw. ein zufälliges Ersetzen, können verwendet werden.

[0077] Wenn der betrachtende Nutzer eine vorherige HTML-Seite **64'** mit einer zwischengespeicherten Verbindung zurückgibt, wird die Netzwerkexekutive **50**, die der HTML-Seite **64'** zugeordnet ist, zu dem Vordergrund zurückgegeben, d.h. diese erlangt wieder "Fokus", und eine Verarbeitung der zugeordneten Anordnung setzt bei einem normalen Prioritätsniveau fort. Wenn es notwendig ist, richtet die Netzwerkexekutive **50** erneut die Verbindung mit der Anwendung **62'** ein. Obwohl keine Ausgabedaten durch die Netzwerkexekutive **50** für zwischengespeicherte Verbindungen abgelegt sind, wird, sobald eine Verbindung für ein Anwendungsfenster **66'** wieder eingerichtet ist, die Verbindung zu der Anwendung **62'** wieder eingerichtet und die Anwendung **10** schreibt wieder direkt zu dem Anwendungsfenster **66'**.

[0078] Auf ähnliche Weise kann die vorstehend beschriebene Verbindungsmethodik verwendet werden, um eine Fernausführung einer Anwendung bereitzustellen, die in einer interpretierenden Sprache geschrieben ist. Nochmals unter Bezugnahme auf Fig. 2 ist ein Klientenknoten **24** mit einem Serverknoten **34** verbunden, der eine Anwendung **62** im Namen des Klientenknotens **24** ausführt. In diesem Beispiel ist die Serveranwendung **62** irgendeine Anwendung, die dem Klientenknoten **24** erlaubt, eine Anwendung abzurufen, die in einer interpretierenden Sprache geschrieben ist. Beispielsweise kann die Anwendung **62** ein Webbrowser sein, der dem Klientenknoten **24** erlaubt, JAVA™-Anwendungen unter Verwendung von Internetadressen herunterzuladen. Wie gerade eben beschreiben wurde, sind der Knoten, von dem die Anwendung heruntergeladen wird, und der Serverknoten **34** getrennte Maschinen, die durch ein Computernetzwerk verbunden sind. In manchen Ausführungsformen können jedoch diese Maschinen ein und dieselbe sein.

[0079] Um zu vermeiden, dass der Klientenknoten **24** erfordert, die heruntergeladene Anwendung zu speichern und auszuführen, was sowohl für den Klientenspeicher als auch für die Prozessornutzung prohibitiv sein kann, stellt die Ausführungsumgebung **96** auf dem Serverknoten **34**, dem der Klientenknoten **24** zugeordnet ist, eine Ausführungsumgebung für die heruntergeladene Anwendung bereit. Die Ausführungsumgebung interpretiert den Bytestrom der heruntergeladenen Anwendung, um eine Reihe von Befehlen zu erzeugen, die die Anwendung repräsentieren. Wenn die Anwendung in der interpretierenden Sprache JAVA geschrieben ist, wird die Ausführungsumgebung manchmal als eine virtuelle JAVA™-Maschine bezeichnet.

[0080] Bei manchen Ausführungsformen umfasst die Ausführungsumgebung einen Kompilierer. Diese Kompilierer konvertieren den Bytestrom der Anwendung in "Maschinencode". Ein Kompiler kann bspw. den Bytestrom einer Anwendung, die in der Sprache JAVA geschrieben ist, in 80486-Maschinencode übertragen. Eine Umwandlung des Bytestroms der interpretierenden Sprache in Maschinencode ermöglicht, dass die Anwendung schneller läuft, als wenn jedes Byte während der Laufzeit interpretiert und ausgeführt werden muss. Manche Kompilierer können jedoch den Datenstrom kompilieren, während die Anwendung ausgeführt wird. Diese Kompilierer werden manchmal als "just in time"-Kompilierer bezeichnet und sehen üblicherweise eine vorbestimmte Anzahl an Bytes vor der gegenwärtig ausgeführten Instruktionsausführung, um einen stetigen Strom kompilierten Codes zu erzeugen.

[0081] Die heruntergeladene Anwendung wird interpretiert und durch den Serverknoten **24** ausgeführt und die Ausgabe der Anwendung wird zu dem Klientenknoten übertragen, wie in Verbindung mit Fig. 2 beschrieben ist. Der Serverknoten **34** akzeptiert auch eine Eingabe von dem Klientenknoten **24**. Dies ermöglicht dem Klientenknoten **24**, die heruntergeladene Anwendung zu steuern oder eine Eingabe für die Anwendung bereitzustellen. Der Serverknoten **34** kann eine getrennte Ausführungsumgebung aufbauen, um die heruntergeladene Anwendung zu interpretieren und auszuführen. Bei diesen Ausführungsformen würde die Ausführungsumgebung, die der heruntergeladenen Anwendung zugeordnet ist, ebenfalls ihre Ausgabe zu einem Multiplexer (mux) **121** richten.

[0082] Unter Bezugnahme auf Fig. 6 sollte bemerkt werden, dass irgendein Klient **24**, **24'**, **24"** oder in der Tat

alle Klienten (im allgemeinen **24**), die dem Server **34** mit der Anwendung **63** beigefügt sind, ein anderer Server **34'**, **34"** sein können. Auf diese Weise werden Daten, die durch die Anwendung **63** übertragen werden, zu anderen Servern gesendet, bevor diese zu Klientenknoten **24** gesendet werden. Auf diese Weise werden Daten, die durch die Anwendung **63** übertragen werden, zu einer stets anwachsenden Anzahl von Klientenknoten übertragen, da dieses Netzwerk auffächert bzw. verzweigt.

[0083] Wenn jeder Klientenknoten **24** seine Verbindung zu dem Server **34** abbricht, wird jeder Klientenprotokollstapel (im allgemeinen **104**) und dessen zugeordneter minimaler Stapel (im allgemeinen **107**) zerstört bzw. gelöscht. Auf ähnliche Weise wird der minimale Protokollstapel (im allgemeinen **106**), der dem ersten Klientenprotokollstapel **104** zugeordnet ist, ebenfalls zerstört. Wenn der letzte des minimalen **107** und zweiten (und nachfolgend) Klientenprotokollstapels **104** beendet wurde, ist die Konfiguration diejenige, wie sie mit nur einem ersten Klientenkommunikationsprotokollstapel **104** war, der der Ausführungsumgebung **96** zugeordnet ist. Es sei bemerkt, dass, bis alle die zweiten und nachfolgenden Klientenprotokollstapel beendet sind, der erste Klientenprotokollstapel **104** nicht gelöscht werden kann, selbst wenn der erste Klient **24** nicht länger vorliegt.

[0084] Wie in Fig. 2 gezeigt ist, kommuniziert jede Ausführungsumgebung **96** mit jedem Protokollstapel **104** durch einen Multiplexer **121**, **121'**, **12"**. Wie ebenfalls anhand Fig. 6 deutlich wird, ist es mit der vorliegenden Erfindung für mehr als einen Klienten möglich, Daten zu empfangen, die zu dem ersten Klienten übertragen werden, bspw. um die Übertragung von Daten von einem Server **34** zu beschatten oder zu überwachen oder um Daten von einer spezialisierten Sendeanwendung auszusenden, wie bspw. eine Aktienkursnotierungsanwendung, von der dieselben Daten im wesentlichen gleichzeitig zu einer Anzahl von Klienten (im allgemeinen **24**) gesendet oder übermittelt werden.

[0085] In einem solchen Fall bewirkt der erste Klientenknoten **24**, dass die spezialisierte Anwendung **63** ihre Daten ausführt und zu dem Klienten **24** überträgt, wie vorstehend erörtert ist. Wenn ein zweiter Klient **24'** einen Zugriff auf die Sendeanwendung **63** aufruft, beginnt der Verbindungsverwalter **80** damit, den Protokollstapel **104'** für den zweiten Klienten **24'** aufzubauen, wie vorstehend erörtert ist, unter Berücksichtigung des ersten Klienten **24**. Da jedoch die Anwendung **63** eine Sendeanwendung ist, erkennt der Verbindungsverwalter **80**, dass es nicht erforderlich ist, eine zusätzliche Ausführungsumgebung **96** zu starten und nimmt anstelle dessen die notwendigen Schritte auf sich, um die Daten von der Sendeanwendung **63** zu dem zweiten Klienten **24'** und zu allen zusätzlichen Klienten **24"** zu senden.

[0086] Zunächst erzeugt der Verbindungsverwalter **80** einen ersten minimalen Kommunikationsprotokollstapel **106**, den dieser mit einem Kommunikationsprotokollstapel **104** des ersten Klienten **24** in Verbindung bringt. Der Verbindungsverwalter **80** erzeugt als nächstes einen zweiten minimalen Protokollstapel **107** und verbindet diesen mit dem Kommunikationsprotokollstapel **104'** des zweiten Klienten **24'**. Da jeder zusätzliche Klient **24"** einen Zugriff auf die Senderanwendung **63** anfordert, wird ein weiterer minimaler Protokollstapel **106** erzeugt und dem ersten Klientenprotokollstapel **104** und einem weiteren minimalen Protokollstapel **107'** zugeordnet, und der Klientenprotokollstapel **104"** wird für jeden neuen Klienten **24"** erzeugt. Der erste Klientenprotokollstapel **104** und alle die minimalen Protokollstapel **106**, **106'**, die dem ersten Klientenprotokollstapel **104** zugeordnet sind, und jedes Paar von Klientenprotokollstapeln **104'**, **104"** und minimale Protokollstapel **107**, **107'**, die jedem zusätzlichen Klienten **24'**, **24"** zugeordnet sind, sind in Verbindung über einen Multiplexer **121**.

[0087] Wenn der Multiplexer **121** Daten zu nur einem Klienten **24** richtet und von diesem empfängt, dient der Multiplexer **121** als eine einfache Durchreichvorrichtung. Wenn es jedoch mehr als einen Klienten **24**, **24'**, **24"** gibt, die Daten von einer einzelnen Anwendung **63** empfangen oder zu dieser übertragen, nimmt jeder Multiplexer (im allgemeinen **121**) zwei zusätzlich Konfigurationen ein. Bei einer Konfiguration ist der Multiplexer **121'** konfiguriert, um Anwendungsdaten von sowohl dem ersten Klientenprotokollstapel **104** als auch jedem der minimalen Kommunikationsprotokollstapel **106**, **106'**, die diesem zugeordnet sind, zu senden als auch von diesen zu empfangen. Bei der zweiten Konfiguration ist der Multiplexer **121"** konfiguriert, Daten, die durch den minimalen Protokollstapel **107**, **107'** empfangen werden zu dem Klientenprotokollstapel **104'** bzw. **104"** zu senden, die diesem zugeordnet sind. In dieser Ausführungsform kann der Multiplexer **121** Eingabedaten direkt von jedem Klientenprotokollstapel **104**, **104'**, **104"** empfangen.

[0088] Der Verbindungsverwalter **80** verbindet die minimalen Protokollstapel **106**, **106'**, die dem ersten Klienten **24** zugeordnet sind, mit dem minimalen Protokollstapeln **107** bzw. **107'** des zweiten **24'** und nachfolgenden Klienten **24"** und instruiert den Multiplexer **121**, eine Ausgabe von der Anwendung **63** zu dem Kommunkationsprotokollstapel **104** des ersten Klienten **24** und dessen zugeordneten minimalen Protokollstapeln **106**, **106'** zu richten. Der Multiplexer **121** ist ebenfalls durch den Verbindungsverwalter **80** instruiert, jeden zweiten und nachfolgenden minimalen Klientenprotokollstapel **107**, **107'** mit dessen zugeordnetem Klientenprotokollstapel

104 bzw. **104'** zu verbinden. Daten, die zu dem ersten Klienten **24** mittels des ersten Klientenprotokollstapels **104** übertragen werden, werden daher ebenfalls zu den minimalen Protokollstapeln **106**, **106'** übertragen, die dem ersten Klienten **24** zugeordnet sind, und daher zu den zweiten **24'** und nachfolgenden Klienten **24''** mit dessen zugeordnetem Protokollstapel **104'** bzw. **104''** und dessen zugeordnetem minimalen Protokollstapeln **107** bzw. **107'**. Bei einer Ausführungsform umfasst der Protokollstapelbehälter eine Datenstruktur, um der Anzahl und dem Typ der Protokolle nachzugehen, die einer gegebenen Anwendung **63** zugeordnet sind.

[0089] Unter Bezugnahme auf Fig. 7, wie vorstehend erörtert wurde, ist es möglich, dass die "Klienten" eines Servers **34** andere Server **34'** und **34''** sein können (zur Vereinfachung sind lediglich zwei gezeigt). Die zweiten Server **34'** und **34''** übertragen dann die Daten zu Klienten (im allgemeinen **24**) oder zu zusätzlichen Servern. Bei dieser Ausführungsform ist die Ausgabe des Serverprotokollstapels (im allgemeinen **104**) mit den Protokollstapeln **107'** der sekundären Server **34'**, **34''** verbunden. Dann, wie vorstehend beschrieben ist, werden die Daten zwischen den Protokollstapeln und heraus zu den Klienten (im allgemeinen **24**) übertragen. Auf diese Weise fächern die Daten auf und werden zu viel mehr Klienten verteilt, als dies vernünftigerweise durch einen Server zu unterstützen ist.

[0090] Obwohl die Erfindung insbesondere unter Bezugnahme auf spezifische bevorzugte Ausführungsformen gezeigt und beschrieben wurde, ist zu verstehen, dass von Fachleuten verschiedene Änderungen in Form und Detail durchgeführt werden können, ohne den Bereich der Erfindung zu verlassen, wie dieser durch die beigefügten Ansprüche definiert ist.

Patentansprüche

1. Verfahren zum Kommunizieren zwischen einem Anwendungsprogramm, das auf einem Serverknoten (**34**) zur Ausführung kommt, und einer Mehrzahl von Klientenknoten (**24**), wobei das Verfahren folgende Schritte umfasst:
Ausführen eines Anwendungsprogramms auf dem Serverknoten in Reaktion auf eine Anfrage von einem ersten Klientenknoten (**24**), um das Anwendungsprogramm auszuführen,
Einrichten einer ersten Verbindung zwischen dem ersten Klientenknoten und dem Serverknoten in Reaktion auf die Anfrage, wobei ein erster Klientenprotokollstapel (**104**) auf dem Serverknoten verwendet wird,
Einrichten einer zweiten Verbindung zwischen einem zweiten Klientenknoten (**24'**) und dem Serverknoten, wobei ein zweiter Klientenprotokollstapel (**104'**) auf dem Serverknoten verwendet wird, in Reaktion auf eine Anfrage von dem zweiten Klientenknoten, um auf das Anwendungsprogramm zuzugreifen, und
im wesentlichen gleichzeitiges Übertragen von Anwendungsdaten, die dem Anwendungsprogramm zugeordnet sind, durch die erste und zweite Verbindung zu dem ersten bzw. zweiten Klientenknoten.

2. Verfahren nach Anspruch 1, das weiterhin die Schritte umfasst:
Übertragen von Eingabedaten durch eine der Verbindungen von einem der Klientenknoten (**24**) zu dem Anwendungsprogramm, das auf dem Serverknoten (**34**) zur Ausführung kommt, und
Übertragen der Eingabedaten zu dem anderen Klientenknoten durch die andere Verbindung.

3. Verfahren nach Anspruch 1 oder 2 bei dem die Anwendungsdaten, die durch die zweite Verbindung übertragen werden, eine Kopie der Anwendungsdaten aufweisen, die zu der ersten Verbindung übertragen werden.

4. Verfahren nach Anspruch 1, 2 oder 3, das weiterhin die Schritte umfasst:
Erzeugen eines dritten Klientenprotokollstapels (**104''**) auf dem auf dem Serverknoten (**34**) in Kommunikation mit dem zweiten Klientenprotokollstapel (**104'**), und
Erzeugen eines vierten Klientenprotokollstapels auf dem Serverknoten in Kommunikation zu dem dritten Klientenprotokollstapel und dem zweiten Klientenknoten (**24'**),
wobei die zweite Verbindung den dritten und vierten Klientenprotokollstapel aufweist.

5. Verfahren nach einem der Ansprüche 1 bis 4, das weiterhin die Schritte umfasst:
Bereitstellen einer Ausführungsumgebung (**96**) auf dem Serverknoten (**34**), innerhalb der das Anwendungsprogramm auszuführen ist, und
Zuordnen der Ausführungsumgebung zu dem ersten und zweiten Klientenprotokollstapel (**104**, **104'**).

6. Verfahren nach einem der vorstehenden Ansprüche, bei dem das Anwendungsprogramm ein erstes Anwendungsprogramm ist und die zweite Verbindung einen dritten Klientenprotokollstapel (**104'**) auf dem Serverknoten (**34**) umfasst, und das weiterhin die Schritte umfasst:

Ausführen eines zweiten Anwendungsprogramms auf dem Serverknoten in Reaktion auf eine Anfrage von dem zweiten Klientenknoten (**24'**), um das zweite Anwendungsprogramm auszuführen,
Zuordnen einer Ausführungsumgebung zum Ausführen des zweiten Anwendungsprogramms mit dem dritten Klientenprotokollstapel,
Übertragen von Anwendungsdaten, die einer Ausführung des ersten Anwendungsprogramms zugeordnet sind, durch den zweiten und dritten Klientenprotokollstapel der zweiten Verbindung, und
Übertragen von Anwendungsdaten, die einer Ausführung des zweiten Anwendungsprogramms zugeordnet sind, durch den dritten Klientenprotokollstapel.

7. Verfahren nach einer der vorstehenden Ansprüche, das weiterhin folgende Schritte umfasst:
Erzeugen eines neuen Klientenprotokollstapels für jeden zusätzlichen Klientenknoten, der einen Zugriff auf das Anwendungsprogramm erfragt, das auf dem Serverknoten zur Ausführung kommt,
Zuordnen jedes neuen Klientenprotokollstapels zu dem ersten Klientenprotokollstapel (**104**),
Einrichten für jeden neuen Klientenprotokollstapel einer neuen Verbindung zwischen dem Anwendungsprogramm, das auf dem Serverknoten zur Ausführung kommt, und dem zusätzlichen Klientenknoten, der dem neuen Klientenprotokollstapel zugeordnet ist, und
im wesentlichen gleichzeitiges Übertragen von Anwendungsdaten, die dem Anwendungsprogramm zugeordnet sind, durch jede neue Verbindung zu jedem zusätzlichen Klientenknoten.

8. Verfahren nach einem der vorstehenden Ansprüche zum Übertragen der selben bzw. gleichen Daten im wesentlichen gleichzeitig von einem Anwendungsprogramm, das auf einem Serverknoten (**34**) zur Ausführung kommt, zu zumindest zwei Klientenknoten (**24**), wobei jeder Klientenknoten ein verallgemeinertes Empfängerprogramm ausführt, wobei der Schritt des Einrichtens der ersten Verbindung umfasst:
Bereitstellen einer Verbindung zwischen dem ersten Klientenknoten (**24**) und dem ersten Klientenprotokollstapel (**104**) auf dem Serverknoten,
Bereitstellen einer Verbindung zwischen der Anwendung, die auf dem Serverknoten zur Ausführung kommt, und dem ersten Klientenprotokollstapel, und
Bereitstellen einer Verbindung zwischen der Anwendung, die auf dem Serverknoten zur Ausführung kommt, und einem ersten minimalen Kommunikationsprotokollstapel (**106**),
wobei der Schritt des Einrichtens der zweiten Verbindung umfasst:
Bereitstellen einer Verbindung zwischen dem zweiten Klientenknoten (**24'**) und dem zweiten Klientenprotokollstapel (**104'**) auf dem Serverknoten,
Bereitstellen einer Verbindung zwischen dem ersten minimalen Protokollstapel und einem zweiten minimalen Protokollstapel (**107**), und
Bereitstellen einer Verbindung zwischen dem zweiten minimalen Protokollstapel und dem zweiten Klientenprotokollstapel, und
wobei der Schritt des im wesentlichen gleichzeitigen Übertragens von Anwendungsdaten des ersten und zweiten Klientenknotens den Schritt des Übertragens von Daten von dem Anwendungsprogramm zu dem ersten Klientenprotokollstapel und dem ersten minimalen Protokollstapel im wesentlichen gleichzeitig umfasst.

9. Verfahren nach einem der vorstehenden Ansprüche, bei dem die Verbindung zwischen dem ersten Klientenprotokollstapel (**104**) und dem Anwendungsprogramm durch einen Multiplexer (**121**) erfolgt.

10. Verfahren nach Anspruch 8 oder 9, bei dem die Verbindung zwischen dem ersten minimalen Protokollstapel (**106**) und dem Anwendungsprogramm durch einen Multiplexer (**121**) erfolgt.

11. Verfahren nach Anspruch 8, 9 oder 10, bei dem die Verbindung zwischen dem zweiten Klientenprotokollstapel (**104'**) und dem zweiten minimalen Protokollstapel (**107**) durch einen Multiplexer (**121**) erfolgt.

12. Verfahren nach einem der Ansprüche 8 bis 11, das weiterhin den Schritt des Zuordnens des ersten minimalen Protokollstapels (**106**) zu dem ersten Klientenprotokollstapel (**104'**) umfasst.

13. Verfahren nach einem der Ansprüche 8 bis 12, das weiterhin den Schritt des Zuordnens des zweiten minimalen Kommunikationsprotokollstapels zu dem zweiten Klientenprotokollstapel umfasst.

14. Verfahren nach einem der vorstehenden Ansprüche, das weiterhin den Schritt des Bestimmens umfasst, ob das Anwendungsprogramm für eine Übertragung bzw. Rundfunk geeignet ist.

15. System zum Übertragen von Daten, die einem Anwendungsprogramm zugeordnet sind, zu einer Mehrzahl von Klientenknoten (**24**) in einem Klientenservernetzwerk, mit:

einem Serverknoten (**34**), der ein Anwendungsprogramm ausführt, in Reaktion auf eine Anfrage von einem ersten Klientenknoten (**24**), um das Anwendungsprogramm auszuführen,
einer ersten Verbindung zwischen dem Serverknoten und dem ersten Klientenknoten, der in Reaktion auf die Anfrage eingerichtet ist, wobei die erste Verbindung einen ersten Klientenprotokollstapel (**104**) auf dem Serverknoten aufweist, um Kommunikationen zwischen dem Anwendungsprogramm und dem ersten Klientenknoten zu leiten,
einer zweiten Verbindung zwischen dem Serverknoten und einem zweiten Klientenknoten (**24'**), der in Reaktion auf eine Anfrage von dem zweiten Klientenknoten eingerichtet ist, um auf das Anwendungsprogramm zuzugreifen, wobei die zweite Verbindung einen zweiten Klientenprotokollstapel (**104'**) auf dem Serverknoten umfasst, der dem ersten Klientenprotokollstapel zugeordnet ist, und
Mittel zum im wesentlichen gleichzeitigen Übertragen von Anwendungsdaten, die dem Anwendungsprogramm zugeordnet sind, zu dem ersten und zweiten Klientenprotokollstapel.

16. System nach Anspruch 15, bei dem die erste Verbindung aufweist:
den ersten Klientenprotokollstapel (**104**), der in elektrischer Kommunikation zu dem Anwendungsprogramm ist,
den ersten Klienten (**24**), der in elektrischer Kommunikation zu dem ersten Klientenprotokollstapel (**104**) ist, und
einen ersten minimalen Protokollstapel (**106**) in elektrischer Kommunikation zu dem Anwendungsprogramm,
wobei die zweite Verbindung aufweist:
einen zweiten minimalen Protokollstapel (**107**) in elektrischer Kommunikation zu dem ersten minimalen Protokollstapel,
den zweiten Klienten- (**24''**) Protokollstapel (**104'**), der in elektrischer Kommunikation zu dem zweiten minimalen Protokollstapel ist, und
den zweiten Klienten, der in elektrischer Kommunikation zu dem zweiten Klientenprotokollstapel ist, und
wobei die Daten von dem Anwendungsprogramm zu dem ersten Klientenprotokollstapel und dem ersten minimalen Protokollstapel im wesentlichen gleichzeitig übertragen sind.

17. System nach Anspruch 15 oder 16, bei dem das Mittel zum wesentlichen gleichzeitigen Übertragen von Anwendungsdaten einen Multiplexer (**121**) in Kommunikation zu jeder Verbindung und dem Anwendungsprogramm, das auf dem Serverknoten (**34**) zur Ausführung kommt, aufweist.

18. System nach einem der Ansprüche 15 bis 17, bei dem die Anwendungsdaten, die zu dem zweiten Klientenprotokollstapel (**104'**) übertragen werden, eine Kopie der Anwendungsdaten aufweisen, die zu dem ersten Klientenprotokollstapel (**104**) übertragen werden.

19. System nach Anspruch 15 oder 18, bei dem die zweite Verbindung weiterhin einen dritten Klientenprotokollstapel (**104''**) auf dem Serverknoten (**34**) in Kommunikation zu dem zweiten Klientenprotokollstapel (**104'**) aufweist, und einen vierten Klientenprotokollstapel auf dem Serverknoten in Kommunikation zu dem dritte Klientenprotokollstapel.

20. System nach einem der Ansprüche 15 bis 19, bei dem der Serverknoten (**34**) weiterhin eine Ausführungsumgebung (**96**) aufweist, in der das Anwendungsprogramm auszuführen ist, wobei die Ausführungsumgebung gleichzeitig mit dem ersten Klientenprotokollstapel (**104**) und jedem Klientenprotokollstapel kommuniziert, der dem ersten Klientenprotokollstapel zugeordnet ist.

21. System, nach einem der Ansprüche 15 bis 20, bei dem jeder Klientenprotokollstapel einen Satz von Protokollmodulen aufweist und der Satz von Protokollmodulen in dem zweiten Klientenprotokollstapel (**104'**) sich von dem Satz von Protokollmodulen in dem ersten Klientenprotokollstapel (**104**) unterscheidet.

22. System nach einem der Ansprüche 15 bis 21, bei dem der Serverknoten ein erster Serverknoten (**34**) ist und das weiterhin einen zweiten Serverknoten (**34**) in Kommunikation zu dem ersten Serverknoten und dem ersten Klientenknoten (**24**) aufweist, und wobei die erste Verbindung zwischen dem ersten Klientenknoten und dem ersten Serverknoten durch den zweiten Serverknoten ist.

Es folgen 5 Blatt Zeichnungen

Anhängende Zeichnungen

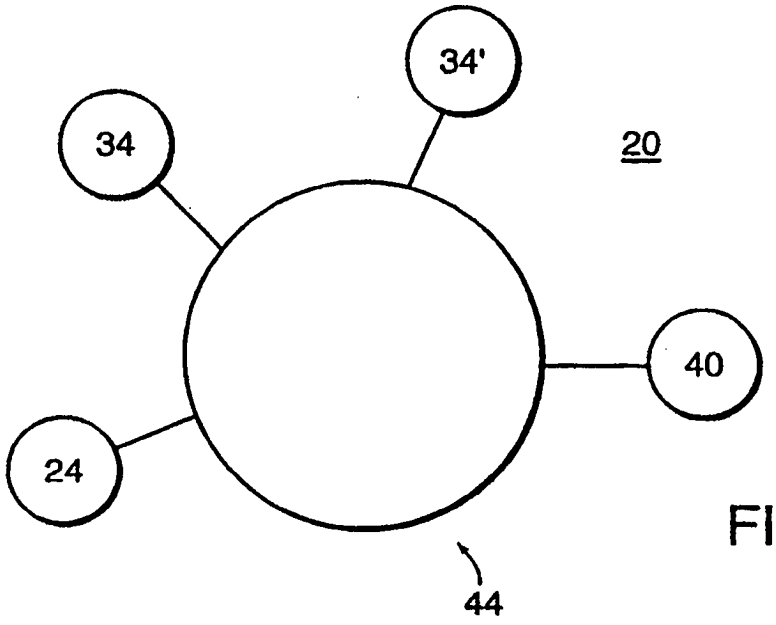

FIG. 1

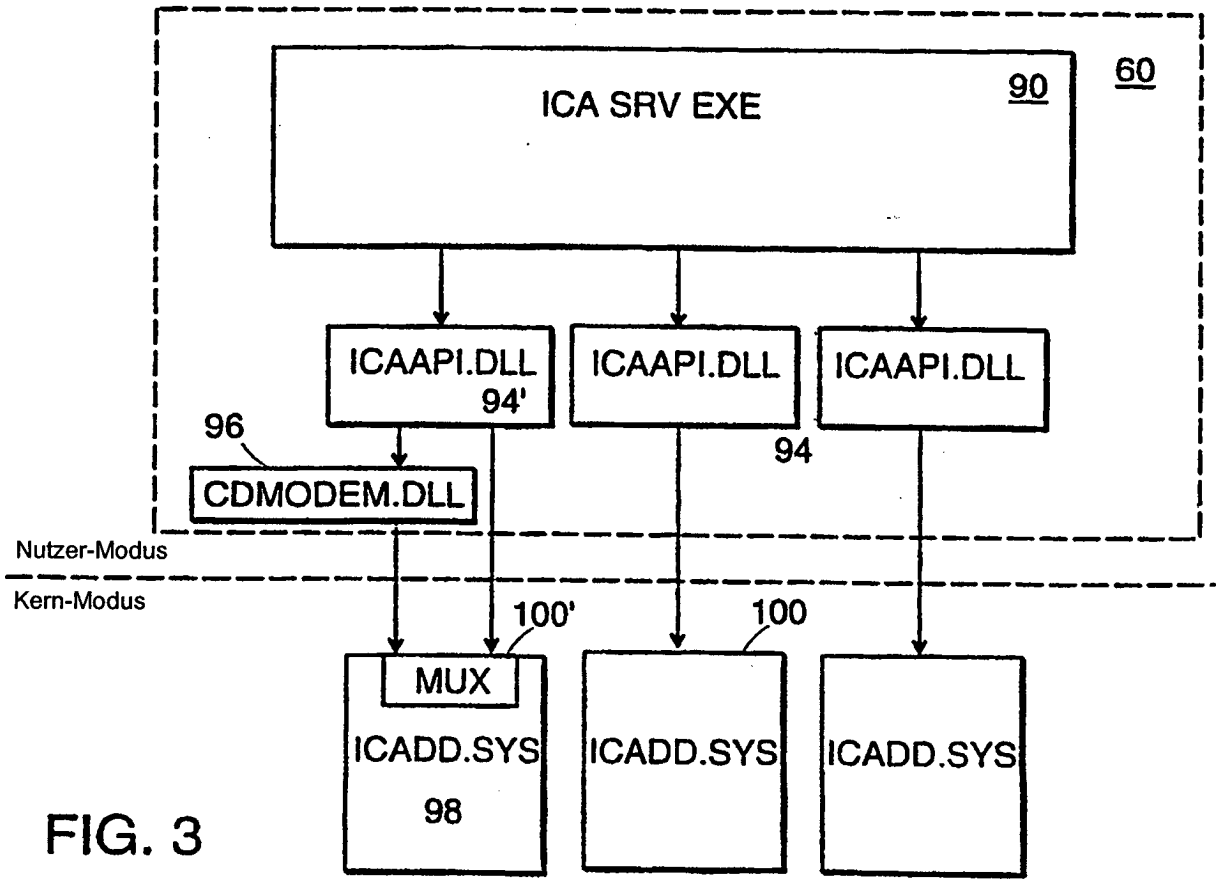

FIG. 3

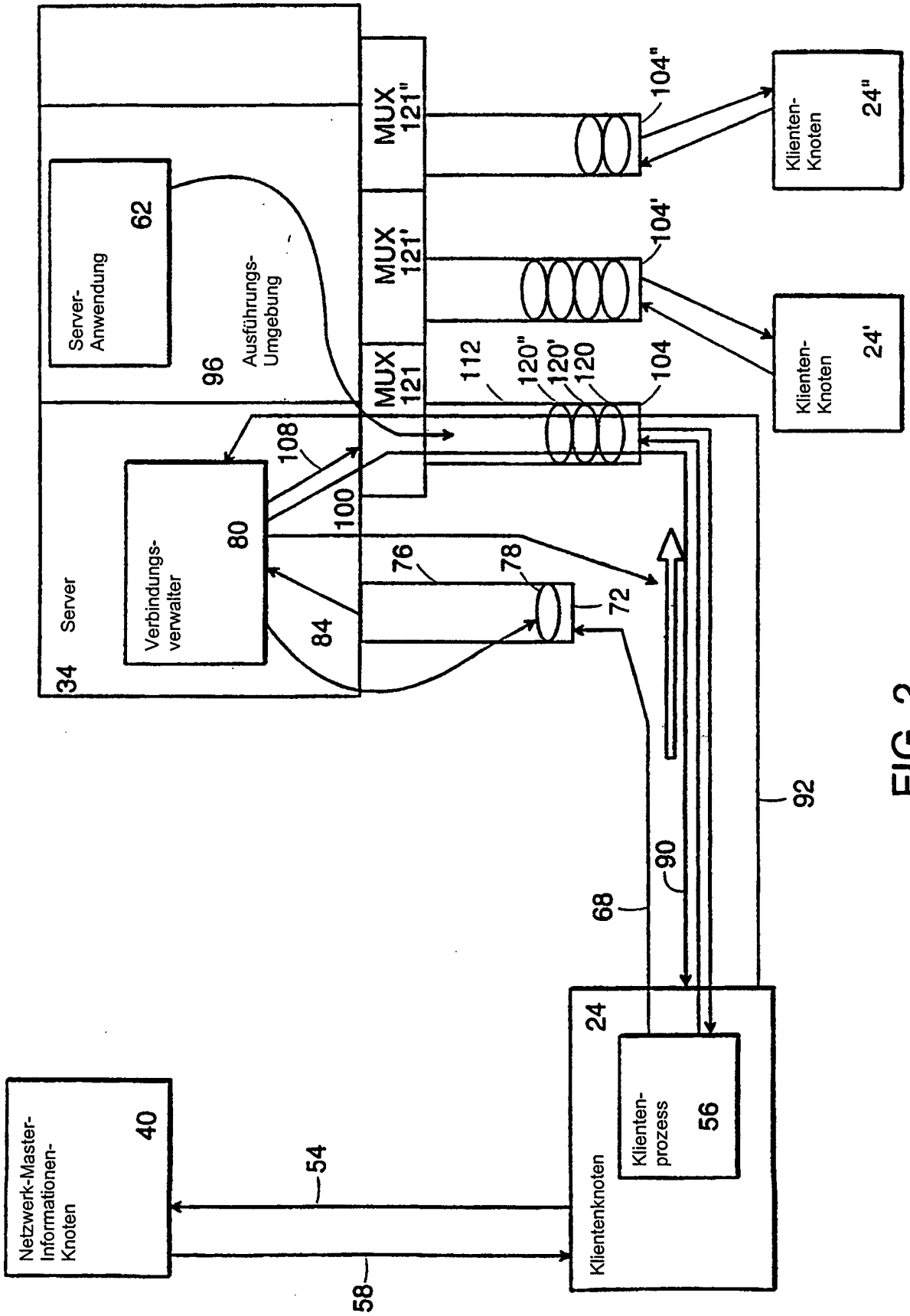
Server
34
Server-Anwendung
62
Ausführungs-Umgebung
96
Verbindungs-verwalter
80
MUX 121
MUX 121'
MUX 121"
112
120
120'
120"
104
104'
104"
108
100
76
78
72
84
Klienten-Knoten
24'
Klienten-Knoten
24"
92
90
68
24
Klientenknoten
Klienten-prozess
56
54
58
Netzwerk-Master-Informationen-Knoten
40

FIG. 2

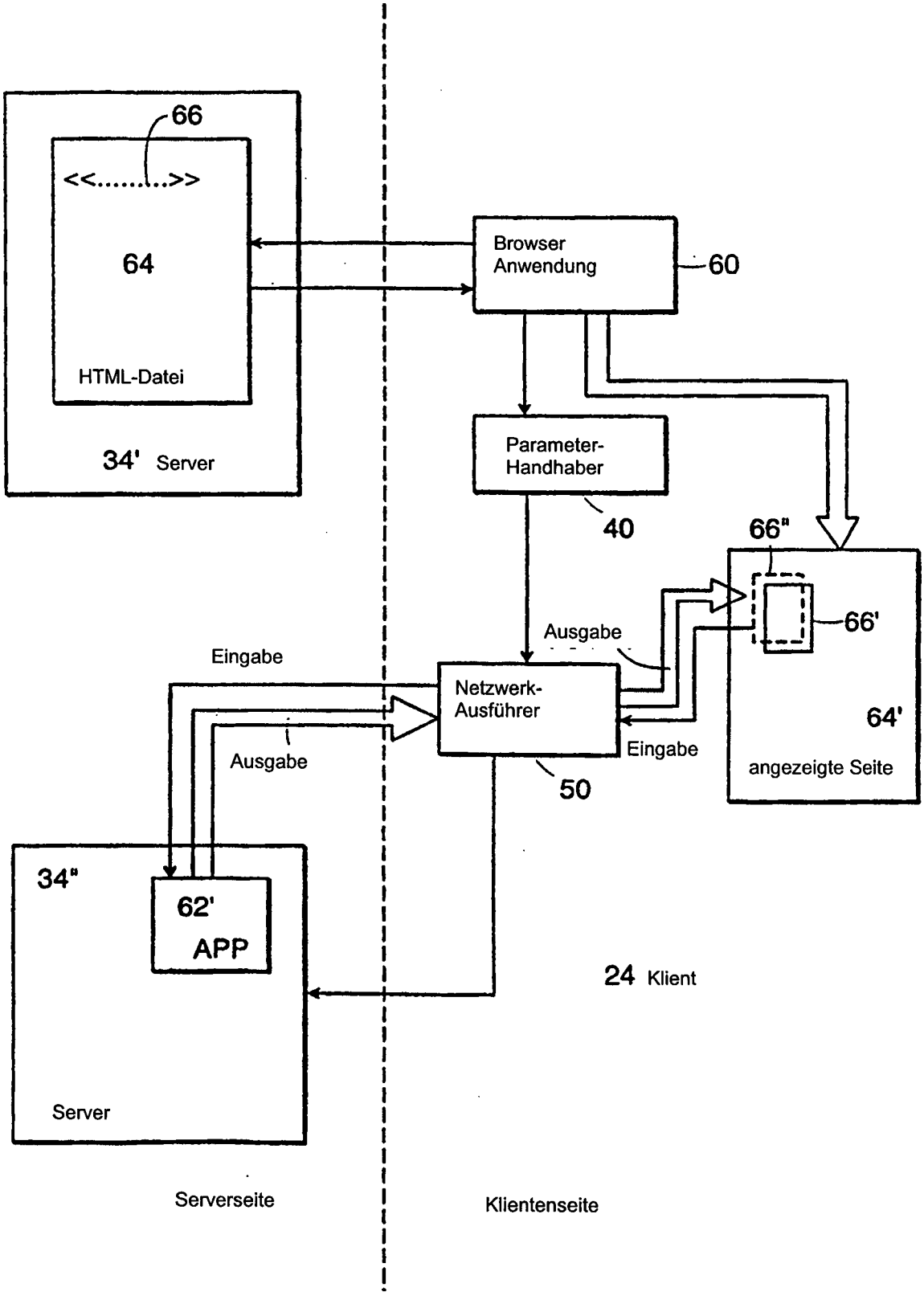
66
<<.....>>
64
HTML-Datei
34' Server
Browser
Anwendung
60
Parameter-
Handhaber
40
66"
66'
64'
angezeigte Seite
Ausgabe
Eingabe
Netzwerk-
Ausführer
50
Eingabe
Ausgabe
34"
62'
APP
24 Klient
Server
Serverseite
Klientenseite

FIG. 4

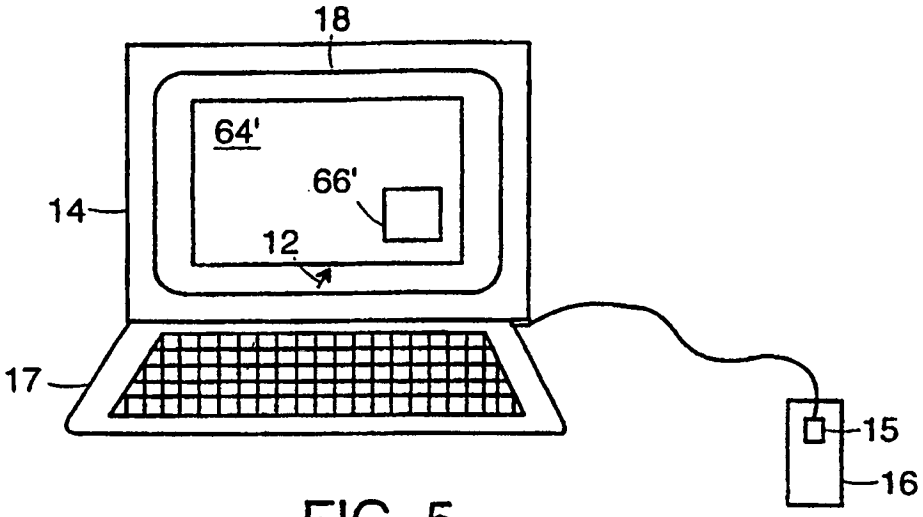
18
64'
66'
14
12
17
15
16

FIG. 5

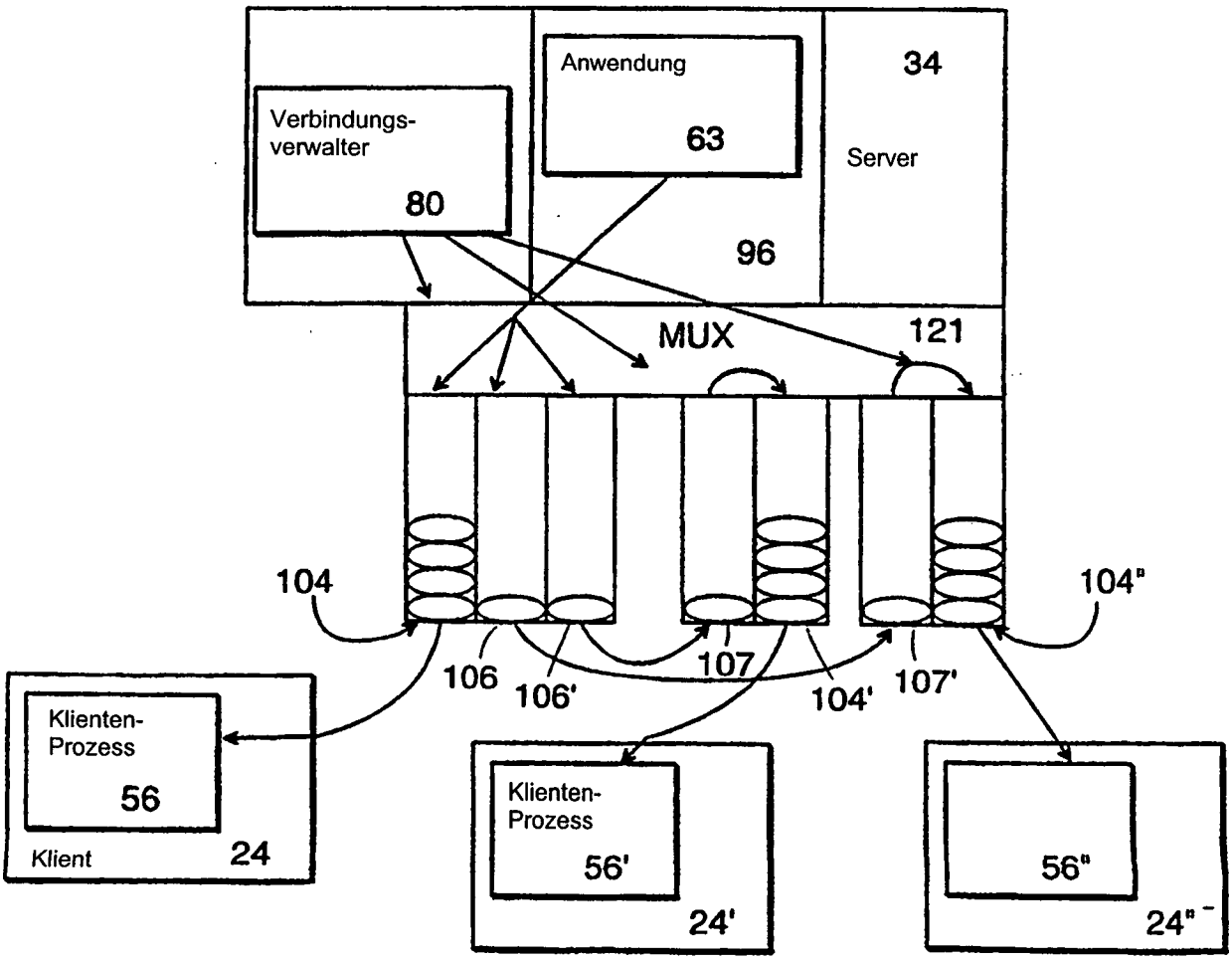
Anwendung
63
34
Server
Verbindungs-
verwalter
80
96
MUX
121
104
104"
106
106'
107
104'
107'
Klienten-
Prozess
56
Klient
24
Klienten-
Prozess
56'
24'
56"
24"

FIG. 6

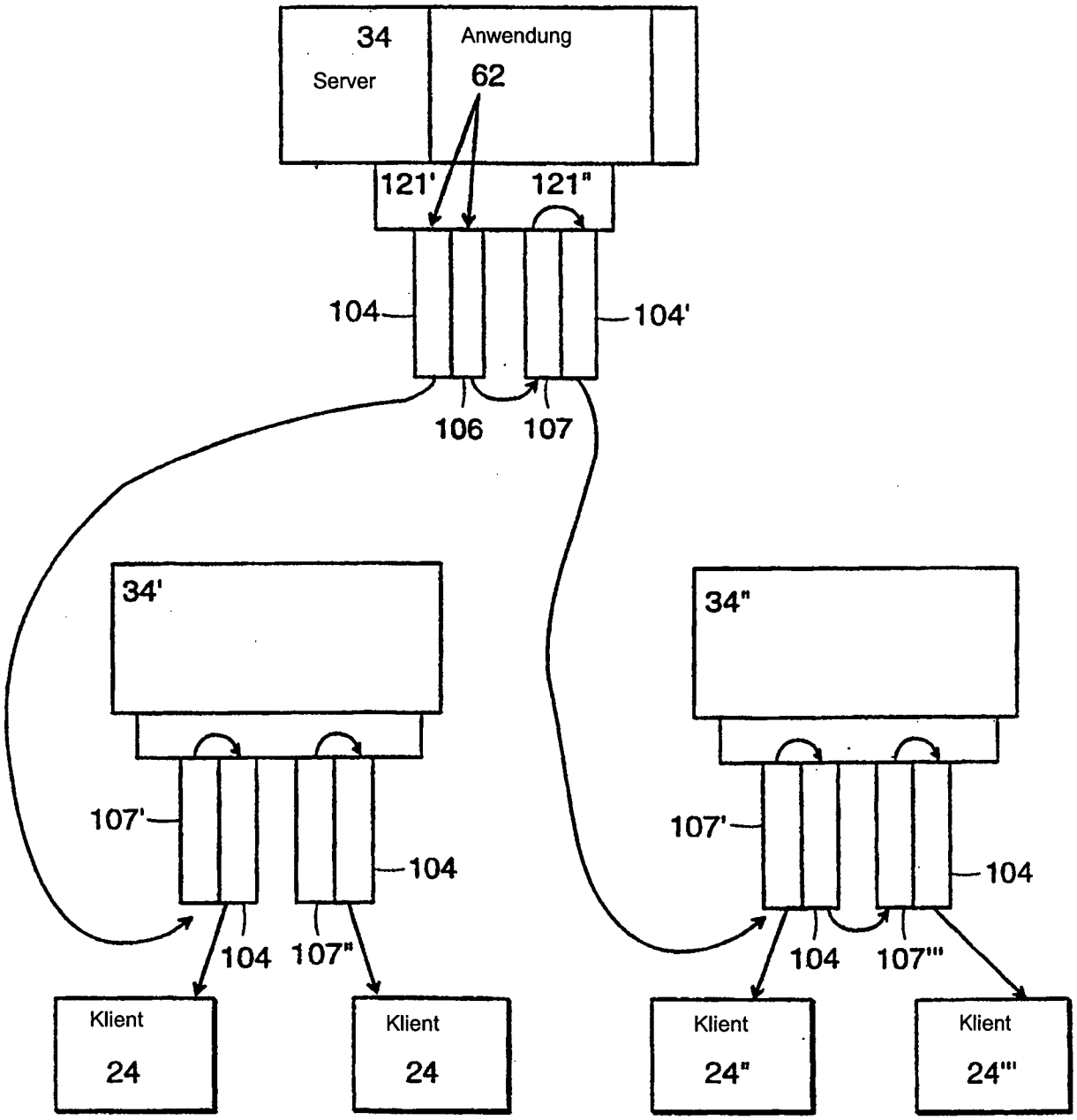
34
Server
Anwendung
62
121'
121"
104
104'
106
107
34'
34"
107'
104
104
107"
107'
104
104
107'"
Klient
24
Klient
24
Klient
24"
Klient
24'"

FIG. 7