(19) Deutsches Patent- und Markenamt

(10) **DE 101 16 640 B4** 2012.09.13

(12) # Patentschrift

(21) Aktenzeichen: **101 16 640.0**
(22) Anmeldetag: **04.04.2001**
(43) Offenlegungstag: **20.12.2001**
(45) Veröffentlichungstag der Patenterteilung: **13.09.2012**

(51) Int Cl.: ***G06F 15/173*** (2006.01)

Innerhalb von drei Monaten nach Veröffentlichung der Patenterteilung kann nach § 59 Patentgesetz gegen das Patent Einspruch erhoben werden. Der Einspruch ist schriftlich zu erklären und zu begründen. Innerhalb der Einspruchsfrist ist eine Einspruchsgebühr in Höhe von 200 Euro zu entrichten(§ 6 Patentkostengesetz in Verbindung mit der Anlage zu § 2 Abs. 1 Patentkostengesetz).

(30) Unionspriorität:
557708 **25.04.2000** **US**

(73) Patentinhaber:
International Business Machines Corp., Armonk, N.Y., US

(74) Vertreter:
Duscher, Reinhard, Dipl.-Phys. Dr.rer.nat., 71139, Ehningen, DE

(72) Erfinder:
Gage, Christopher A.S, Raliegh, N.C., US; Hind, John R., Raleigh, N.C., US; Peters, Marcia L., Raleigh, N.C., US

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:

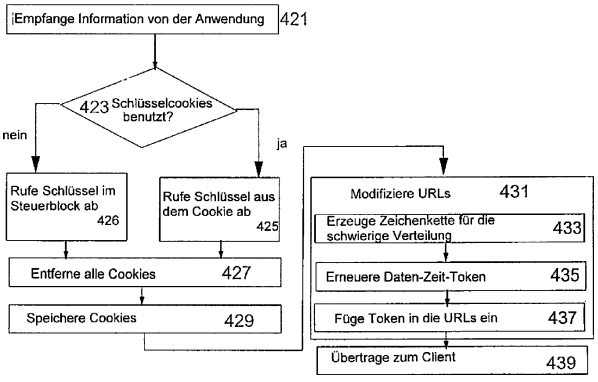
G. APOSTOLOPOULOS (u.a.): Design, Implementation and Performance of a Content-Based Switch, 26.-30. März 2000, IEEE INFOCOM 2000, Vol. 3, S. 1117-1126, ISBN: 0-7803-5880-5, DOI: 10.1109/INFCOM.2000.832470

V. CARDELLINI, M. COLAJANNI, P.S. YU: Dynamic Load Balancing on Web-Server Systems, 1999, IEEE Internet Computing, Vol. 3(3) pp. 28-29, DOI: 10.1109/4236.769420

V. SAMAR: Single sign-on using cookies for Web applications, 1999, IEEE Infrastructure for Collaborative Enterprises (ICE'99), ISBN: 0-7695-0365-9, S. 158-163, DOI: 10.1109/ENABL.1999.805192

(54) Bezeichnung: **Auf URL beruhende Token für schwierige Verteilungen, die einen serverseitigen Cookiebehälter benutzen**

(57) Hauptanspruch: Verfahren zur Herstellung einer dauerhaften Beziehung zwischen einem Client-System und einem Server,
wobei der Server aus einer Vielzahl von Servern stammt, die von einem Dispatcher verwaltet werden, und das Client-System unter Benutzung eines Universellen Ressourcen-Lokalisierers (URL) auf den Server zugreift,
bei dem der Dispatcher eine Informationsanforderung vom Client-System empfängt und bestimmt, welcher aus der Vielzahl von Servern für die Erfüllung der Anforderung auszuwählen ist;
dadurch gekennzeichnet,
– dass der ausgewählte Server ein Token erzeugt, das wenigstens einen Bezeichner für den ausgewählten Server, eine Datums-Zeit-Markierung und einen Schlüssel umfasst, wobei der Schlüssel für den Zugriff auf einen Speicherbereich für Informationen bezüglich der dauerhaften Beziehung zum Client-System verwendet wird;
– dass das Token in den URL eingefügt wird; und
– dass der ausgewählte Server eine Antwort mit dem in den URL eingefügten Token zum Client-System sendet, um das Client-System für die Dauer einer Sitzung an diesen...



Beschreibung

Hintergrund der Erfindung

[0001] In den 90er Jahren sind Computer-Netzwerke exponentiell gewachsen. Das Internet und das World Wide Web (”Weltweites Netz”) haben jedem den Zugriff auf außergewöhnlich große Informationsmengen erlaubt. Sie haben es den Leuten auch ermöglicht, eine Menge ihrer täglichen Geschäfte, zum Beispiel Einkaufen oder Forschung, über das Internet auszuführen. Während der Einkaufssaison 1999 machten verschiedene große Kaufhäuser und Spielzeugläden bessere Geschäfte, wenn ihre Läden eigentlich geschlossen waren, als wenn sie geöffnet waren; dies rührte von ihrer Präsenz im Internet her. Ihre Internet-Webstandorte erlaubten es ihren Kunden, sich die Produkte anzuschauen, die sie anbieten, nach bestimmten Produkten zu suchen, über die Produkte zu lesen, Bestellungen aufzugeben und Lieferinformationen anzusehen, alles das, nachdem die Kinder ruhig zu Bett gebracht wurden. Dies war für einen großen Teil der Industrie eine ziemliche Überraschung. Die Händler mussten die Größe und die Anzahl ihrer Server erhöhen, um den gesamten Verkehr mit den Nutzern zu Hause zu unterstützen. Einige mussten beträchtliche Mengen an neuer Hardware installieren, zum Beispiel Netzwerk-Dispatcher, um Anfragen an ihre Server zu leiten, an redundante oder Sicherungsserver, damit die Antwortzeit für die Nutzer nicht unerträglich lang wurde.

[0002] Dies ist nur eines der vielen Beispiele dafür, wie das Internet wächst und wie der Zugriff auf Informationen über das Internet die heutige Gesellschaft beeinflusst. Die Nutzer gewöhnen sich daran, nahezu unmittelbar eine Antwort von Computer-Netzwerken zu erhalten und tolerieren keine Verzögerungen. Wenn sich die Anzahl der Nutzer des Internets erhöht, muss sich auch die Anzahl der redundanten Server und Netzwerk-Dispatcher erhöhen, die diese Server verwalten, um die Qualität der Dienstleistung aufrechtzuerhalten, die die Kunden erwarten. Netzwerk-Dispatcher werden benutzt, um Anfragen an einen Server zu verwalten und die Arbeit auf mehrere redundante Server zu verteilen, wobei eine vorher festgelegte Belastungsausgleichs-Methodologie (loadbalancing methodology) verwendet wird.

[0003] Ein typisches Netzwerk-Dispatchersystem ist in Fig. 6 dargestellt. Wenn die Endnutzereinheit **601** Informationen von einem Server anfordert, der einen vorgelagerten (front-end) Netzwerk-Dispatcher **603** aufweist, empfängt der Netzwerk-Dispatcher die Anforderung und leitet sie zu einem der redundanten nachgelagerten (back-end) Server **605**, **607**, **609** weiter. Wenn der Netzwerk-Dispatcher **603** anschließend eine weitere Anfrage von der Endnutzereinheit **601** empfängt, durchläuft er wieder seinen Auswahlprozess, um einen kontinuierlichen Ausgleich der Belastung aufrechtzuerhalten. Dies ist die Funktion eines Netzwerk-Dispatchers, so wie sie entworfen wurde.

[0004] Die Verwendung des traditionellen Netzwerk-Dispatchers kann ein Problem werden, wenn ein Nutzer, der wegen einer ersten Anfrage zu einem Server geleitet wurde, wegen wiederholter Anfragen auf Grund der Informationsspeicherung zu dem gleichen Server geleitet werden muss. Dies ist im Falle des Händlers so, wenn ein Nutzer auf den Webstandort des Händlers gelangt und beginnt, eine Bestellung aufzugeben. Nachdem das erste Produkt in einen Einkaufskorb gelegt wurde, müssen die Produkte gespeichert werden, so dass der Nutzer weiter einkaufen und eine erfolgreiche Bestellung aufgeben kann.

[0005] Genauer gesagt wollen wir annehmen, dass eine Menge (cluster) von Webservern gleichartige Dienste zur Verfügung stellen, mit einem vorgelagerten Netzwerkbelastungs-Ausgleichssystem. Die Aufgabe des Belastungs-Ausgleichssystems besteht darin, auf der Grundlage von Entscheidungsmechanismen, die außerhalb des Bereiches dieser Erörterung liegen, einlaufende Pakete zu dem am wenigsten belegten Server zu leiten. Eine vereinfachte Version hiervon ist in Fig. 1 dargestellt, wo ein Mobilfunktelefon **101** auf Informationen in redundanten Servern **109** zugreift, unter Benutzung einer drahtlosen Verbindung **115** zu einer Mobilfunkstation (cellular tower) **105**, danach einer Landverbindung **117** zu einem Dispatcher **107**, danach zu den Informationsservern **109**, während ein Notebook-Computer **103** auch auf Informationen in den gleichen Servern **109** zugreift. Es entsteht das Problem, wie die Belastung auszugleichen ist, unter der Voraussetzung, dass ein Client, zum Beispiel das Mobilfunktelefon **101**, für alle Ströme, die eine Sitzung oder eine Arbeitseinheit ausmachen, zu einem bestimmten Server **109** im Cluster zurückkehren muss. Dies wird als das ”schwierige Verteilungs-”Problem (sticky routing problem) bezeichnet.

[0006] In dem vorliegenden Dokument ist eine ”Sitzung” definiert als eine Folge von zusammengehörigen Transaktionen, die eine Arbeitseinheit ausführen sollen. Eine Sitzung benutzt im Allgemeinen HTTP-(Hyper-text transfer protocol oder HTTPS-(secure hyper-text transfer protocol Ströme, die aus einer oder mehreren TCP/IP-(transmission control protocol/Internet protocol bestehen. Eine einfache elektronische Geschäftstransaktion besteht typischerweise aus einer Folge von zusammengehörigen Aktionen, zum Beispiel Browsen in einem Online-Katalog, Auswahl von einem oder mehreren Handelsprodukten, Aufgeben der Bestellung, Bereitstellung von Bezahlungs- und Versand-Informationen und schließlich Bestätigung oder Abbruch der gesamten Transaktion. Die Informationen über den Stand der Sitzung können mehrere TCP/IP-Verbindungen überdecken, da Infor-

mationen wie zum Beispiel die Identität des Kunden, das gewünschte Produkt, der vereinbarte Preis, Bezahlungsinformationen usw. gespeichert sein müssen, bis die gesamte Transaktion abgeschlossen ist.

[0007] Wenn ein bestimmter Client eine Sitzung mit einem bestimmten Server hat, gibt es Zustandsinformationen über diese Sitzung nur auf diesem bestimmten Server. In diesem Fall muss ein Belastungs-Ausgleichssystem zusätzliche Intelligenz aufwenden, um die Pakete korrekt zu verteilen. Insbesondere muss es den gleichen Server wiederholt als Ziel aller Pakete auswählen, die von einem bestimmten Client für eine bestimmte Sitzung oder Transaktion ankommen. Diese Client-Server-Beziehung wird als ”Bindung” bezeichnet. Um die Belastung zeitlich wirksam auszugleichen, muss das System auch die Bindung des Client an einen bestimmten Server zwischen den Sitzungen oder Transaktionen freigeben.

[0008] Früher war eine Quell-IP-Adresse eindeutig genug, um für diese Art der ”schwierigen” Verteilung zur Unterscheidung benutzt zu werden. Bei der gegenwärtigen Technologie ist die Quell-IP-Adresse auf Grund der weitverbreiteten Verwendung von NAT (Network Address Translation und SSL (Secure Sockets Lager nicht länger als Verteilungs-Token nützlich. Network Address Translation (NAT) wurde in großem Rahmen von ISPs (Internet Service Providers) als Mittel zur Verbindung der großen Anzahl von Nutzern verwendet, die das Internet zu Hause verwenden, ohne dabei eine größere Anzahl registrierter Adressen zu benutzen (da die registrierten Adressen eine begrenzte Ressource und demzufolge teuer sind) und die Vertraulichkeit der IP-Adressen von einzelnen Teilnehmern zu schützen. Die Spezifikationen von NAT sind in dem RFC (Request for Comment) 1631 der IETF (Internet Engineering Task Force) dargestellt. Die NAT-Realisierungen platzieren Netzwerkadress-Übersetzer **503** an die Grenzen der Stub-Bereiche (stub domains), so wie in Fig. 5 dargestellt. Jede NAT-Box weist eine Tabelle auf, die aus Paaren lokaler IP-Adressen und global eindeutiger Adressen besteht. Die IP-Adressen innerhalb des Stub-Bereiches sind nicht global eindeutig. Sie werden auch in anderen Bereichen benutzt. Die NAT kann ohne Änderungen an den Routern (Verteilern) **501** oder den Wirtsrechnern installiert werden, was sie für schnell wachsende ISPs sehr attraktiv macht.

[0009] Die ISPs benutzen auch das DHCP (Dynamic Host Configuration Protocol, RFC-Nummer xxxx) oder das PPP (Point-to-Point Protocol, RFC-Nummer xxxx), um Kundensystemen dynamisch private Adressen zuzuordnen und transparente Proxy-Server zu benutzen (für Dinge wie das World-Wide Web, Nachrichten und Multimedia-Informationen), als eine Möglichkeit, den Backbone-Verkehr zu minimieren. NAT, DHCP/PPP und transparente Proxy-Server lösten die Adressierungsprobleme in sich erweiternden, immer verbundenen Haushaltsnetzwerken, reduzierten die Kosten der Backbones der Anbieter und halfen, die Hacker daran zu hindern, Vorteile aus den offenen Anschlüsse zu den Nutzersystemen zu ziehen, aber diese Schritte führten zum Verlust der eindeutigen IP-Adresse des Nutzers.

[0010] Mit der Einführung der NAT und transparenter Proxyserver kann man nicht länger mit Sicherheit annehmen, dass sich eine einzelne IP-Adresse auf nur einen Client bezieht. Tatsächlich ist es ein Ziel der NAT, die wahre lokale IP-Adresse des Wirtsrechners zu verstecken, indem einige konstante IP-Adressen für die wahre IP-Adresse ersetzt werden. Die NAT-Technologie wird allgemein in einem System benutzt, das eine Vielzahl mobiler Clients mit dem Internet verbindet, zum Beispiel dem Wireless Acess Protocol (WAP) Gateway, und erscheint auch in Haushaltsnetzwerk-Systemen, zum Beispiel einem LAN-Router oder einem intelligenten Hub oder in Modems für Haushalte (der ISDN-LAN-Modem von 3Comm ist ein Beispiel für einen kleinen Router für den Haushalt, der die NAT-Funktion enthält). NAT-Einheiten und transparente Proxy-Server werden auch von ISPs verwendet und bieten Dienste der Art ”immer bereit”, zum Beispiel auf der Grundlage der Kabelmodem-Technologie oder der Asymmetric Digital Subscriber Line(ADSL)-Technologie” sowie den traditionellen anwählbaren ”Point of Presence” (POPs).

[0011] Der SSL-ID (Secure Sockets Layer Identifier) wurde auch als Lösung für das schwierige Verteilungsproblem versucht und war nicht erfolgreich. Verbindungen, die SSL oder TSL benutzen, werden verschlüsselt. Sobald eine SSL-Verbindung zwischen einem gegebenen Client und einem bestimmten Server hergestellt ist, könnte der SSL ID (eine quasi-eindeutige Zahl) von dem Belastungs-Ausgleichssystem geprüft und für die Zwecke der schwierigen Verteilung verwendet werden. Obwohl es der SSL-Standard immer jedem Endpunkt der Verbindung erlaubt, die Schlüsselzustimmung zurückzuweisen und eine Neuverhandlung der SSL-Parameter und nachfolgend die Zuweisung eines neuen SSL-ID zu erzwingen, geschah dies in der Praxis nur bei Servern, was die Vorgehensweise eine Zeit lang lebensfähig machte. Jedoch ist diese Technik mit der unlängst erfolgten Freigabe des Microsoft Internet Explorer 5.0 (geschützte Bezeichnung der Firma Microsoft) nicht länger lebensfähig. Der Internet Explorer 5.0 ist so codiert, dass entweder der Server oder der Client die Schlüsselzustimmung zurückweisen kann, was es für ein Belastungs-Ausgleichssystem unmöglich macht, die frühere SSL-Verbindung mit der jetzigen in Verbindung zu bringen.

[0012] Die nächste Lösung, die für diese beiden Probleme versucht wurde, verwendete ”Cookies”. Ein Cookie ist ein Datenobjekt, das in Feldern variabler Länge innerhalb des HTTP-Headers transpor-

tiert und normalerweise im Client, entweder für die Dauer der Sitzung oder dauerhaft gespeichert wird. Ein Cookie speichert gewisse Daten, an die sich die Serveranwendung für einen bestimmten Client erinnern möchte. Dies könnte die Kennzeichnung des Client, der Sitzungsparameter, der Nutzerpräferenzen, den Stand der Sitzung oder nahezu alles andere einschließen, woran ein Anwendungsschreiber denken kann. Obwohl ein Lastenausgleichssystem mit auf dem Inhalt beruhender Verteilung auch in den HTTP-Header schauen und auf der Grundlage von in den Cookies enthaltenen Daten verteilen könnte, zeigte es sich, dass diese anfänglich verheißungsvolle Lösung auch eine katastrophale Unzulänglichkeit aufweist. Gewisse Kunden können keine Cookies speichern. Unter diesen gewissen Kunden befinden sich Web-Telefon-Kunden, die auf das Internet über ein WAP-Gateway, unter Benutzung des Wireless Session Protocol (WSP), zugreifen. WSP berücksichtigt keine Cookies. Selbst falls WSP Cookies unterstützen würde, können Web-Telefon-Kunden auf Grund ihres außerordentlich begrenzten Speichers keine Cookies speichern. Während es für ein drahtloses Gateway-Produkt möglich ist, Cookies für den drahtlosen Client zu speichern (das IBM eNetwork Wireless Gateway macht das, das Nokia WAP Gateway macht das nicht), können solche Funktionen in dem Gateway nicht vorausgesetzt werden, wie oben nachgewiesen wird. Zusätzlich treffen viele Nutzer die Wahl, Cookies insgesamt zu sperren oder das Nachfragen bei Cookies einzuschalten und Cookies, falls überhaupt, auf Grund zunehmender Vertraulichkeits-Befürchtungen bei der Verwendung von Cookies, dass skrupellose Werber den Surfgewohnheiten eines Internet-Nutzers nachspüren könnten, nur selektiv zu akzeptieren. So kann die Fähigkeit, permanente Sitzungsinformationen in Cookies zu speichern, nicht vorausgesetzt werden.

[0013] APOSTOLOPOULOS, G. Et al: „Design, Implementation and Performance of a Content-Based Switch", 26.–30. März 2000, IEEE INFOCOM 2000, Vol. 3, S. 1117–1126, beschreibt einen "contentbased" Switch, der als Dispatcher für einen Cluster von nicht redundanten Web-Servern eingesetzt wird, wobei der gesamte Datentransfer zwischen dem Client und einem ausgewählten Server über den Switch erfolgt, Dieser Switch wird aber einem Cluster von nicht-redundanten Servern vorgeschaltet.

[0014] SAMAR, V.: "Single sign-on using cookies for Web Applications", 1999, IEEE Infrastructure for Collaborative Enterprises (ICE'99), S. 158–163 beschreibt ein SSO-Verfahren zur Authentisierung von Clients gegenüber Web-Anwendungen, bei dem Cookies, die Client-bezogenen Authentisierungsinformationen entsprechen, als URL-Parameter zwischen einem Web-Server und dem Client übertragen werden. Dieses System umfaßt keinen Server-Cluster mit vorgeschaltetem Dispatcher.

[0015] V. Cardellini, M. Colajanni, P. S. Yu: "Dynamic Load Balancing an Web-Server Systems", IEEE Internet Computing, Vol. 3(3), pp. 28–29, DOI: 10.1109/4236.769420 beschreibt verschiedene Verfahren in verteilten Web-Server-Architekturen zur Verteilung eingehender Requests unter den verschiedenen Web-Servern.

Zusammenfassung der Erfindung

[0016] Die vorliegende Erfindung ermöglicht einzelnen Clients die Dauerhaftigkeit einer schwierigen Sitzung mit einem bestimmten Server, selbst wenn sie identische IP-Adressen besitzen, wie zum Beispiel WAP-Telefone, die auf das Netzwerk über ein NAT-Gateway zugreifen. Sie stellt auch die Fähigkeit von Web-Anwendungen wieder her, auf dem Vorhandensein von Cookies aufzubauen, die sie mit dem Aufkommen von "cookie-freien" WAP-Anrufen verloren hatten. Die vorliegende Erfindung stellt eine bessere Unterstützung für robuste, große, zuverlässige, hochverfügbare Installationen des elektronischen Handels zur Verfügung, die allen Arten von Web-Clients einschließlich Microsoft Internet Explorer 5.0 (geschützte Bezeichnung der Firma Microsoft) dienen und die neueren Web-Telefon-Geräte und die mobilen Geräte, die keine Cookie-Unterstützung besitzen. Die vorliegende Lösung kann als ein auf einer Server-Plattform beruhender Dienst zur Verfügung gestellt werden, ohne irgendwelche Änderungen in vorhandenen Web-Anwendungen zu erfordern.

[0017] Die vorliegende Erfindung verwendet eine Modifikation der Uniform Resource Locators (URLs) in einem Hypertext Transport Protocol-(HTTP) oder Secure HTTP-(HTTPS)Dokument, so dass der URL einen gegebenen Client eindeutig kennzeichnet und ihn für die Dauer einer Sitzung an einen bestimmten Server bindet. Zusätzlich zur Modifikation des URL wird ein "serverseitiger Cookie-Behälter" (cookie jar) benutzt, der ein Datenobjekt zur Verfügung stellt, das einem Web-Anwendungs-Server zugänglich ist. Der Cookie-Behälter wird benutzt, um Cookies für einen besonderen Client zu speichern, so dass sie die Clients oder ihre client-seitigen Proxy-Server nicht speichern müssen. Unter Benutzung des modifizierten client-eindeutigen URL zur Kennzeichnung eines bestimmten Servers, eines Client, einer Sitzung und eines Cookie-Behälters in einer ankommenden Web-Anforderung wird die Anforderung zu dem geeigneten Server geleitet. Die vorliegende Erfindung wird in größerer Ausführlichkeit unter Bezugnahme auf eine nachstehende bevorzugte Ausführungsform beschrieben.

Aufgaben der Erfindung

[0018] Es ist eine Aufgabe der vorliegenden Erfindung, Internet-Nutzer, die keine lokalen Cookies speichern wollen oder können oder die sich hinter ei-

ner NAT oder einem transparenten Proxy-Server befinden, in den Stand zu versetzen, eine Sitzung mit einem bestimmten Server aus einem Cluster von Servern, dem ein Dispatcher vorgelagert wurde, herzustellen.

[0019] Es ist eine weitere Aufgabe der vorliegenden Erfindung, den Server in einer Menge von durch einen Dispatcher verwalteten Servern zu befähigen, Informationen bzgl. des Client zu speichern.

[0020] Es ist eine weitere Aufgabe der vorliegenden Erfindung zu ermöglichen, dass diese Sitzung durch Modifikation der URL und ohne Forderung nach eindeutigen Modifikationen der Clients stattfindet.

[0021] Es ist noch eine weitere Aufgabe der vorliegenden Erfindung zu ermöglichen, dass eine vollständige Transaktion zwischen einem Client und einem Server aus einer Vielzahl von Servern stattfindet, die von einem Netzwerk-Dispatcher verwaltet werden.

[0022] Es ist noch eine weitere Aufgabe der vorliegenden Erfindung, eine differenzierte Service-Qualität durch Anwendung der beschriebenen Methodologie zu ermöglichen.

Kurze Beschreibung der Zeichnungen

[0023] Fig. 1 ist eine bildliche Darstellung eines minimalen Netzwerkes, in dem die vorliegende Erfindung arbeiten kann.

[0024] Fig. 2 ist ein Diagramm der Änderungen des URL für die vorliegende Erfindung.

[0025] Fig. 3 ist ein Flussdiagramm für die Änderungen des URL in der vorliegenden Erfindung.

[0026] Fig. 4A zeigt den logischen Fluss für den Empfang von Informationen beim Netzwerk-Dispatcher der vorliegenden Erfindung.

[0027] Fig. 4B zeigt den logischen Fluss für den Empfang von Informationen von der Anwendung.

[0028] Fig. 5 zeigt die Verwendung des Network Address Translators (NAT).

[0029] Fig. 6 zeigt, wie ein typischer Netzwerk-Dispatcher die Informationen verteilen würde.

Ausführliche Beschreibung der bevorzugten Ausführungsform

[0030] Die oben erwähnten sowie weitere Ziele werden ausführlich beschrieben unter Bezugnahme auf eine bevorzugte Ausführungsform der vorliegenden Erfindung sowie auf die hier dargestellten Abbildungen. Gleiche Zahlen in den Abbildungen stellen die gleichen Elemente dar. Die bevorzugte Ausführungsform wird als Beispiel dargestellt und soll die dargestellte Erfindung oder die Ansprüche in keiner Weise beschränken.

[0031] Die bevorzugte Ausführungsform der vorliegenden Erfindung benutzt eine Modifikation des URL in einem HTTP- oder HTTPS-Dokument, so dass das Dokument einen gegebenen Client eindeutig kennzeichnet und den Client für die Dauer einer Sitzung an einen bestimmten Server bindet. Zusammen mit der URL-Modifikation wird ein serverseitiger Cookie-Behälter eingerichtet, der es dem Server ermöglicht, Cookies für einen bestimmten Client zu speichern. Dadurch werden der Client oder clientseitige Proxy-Server von der Aufgabe befreit, Cookies zu speichern. Der Ansatz des Cookie-Behälters löst das Problem der Speicherung eines Cookies zugunsten des Client. Unter Benutzung des modifizierten client-eindeutigen URL für die Kennzeichnung eines bestimmten Servers, eines Clients, einer Sitzung und eines Cookie-Behälters in einer ankommenden Webanforderung wird die Anforderung zu dem geeigneten Server geleitet, wo ein Stromfilter für ankommende Daten den URL in seinen unmodifizierten Zustand zurückversetzt, das entsprechende oder die entsprechenden Cookies aus dem angegebenen Cookie-Behälter abruft und sie in den HTTP-Strom einfügt, bevor die eingegangene Anforderung an die Anwendung weitergegeben wird (oder an das nächste Filter für einen ankommenden Datenstrom in der Kette, falls Filterverkettung benutzt wird). Das Filter für einen ankommenden Datenstrom muss des erste oder einzige Filter sein, das mit dem ankommenden Datenstrom arbeitet, damit die vorliegende Erfindung erfolgreich ihr Ziel erreicht. Ein zugeordnetes Ausgangsfilter, das sich auch auf dem Webanwendungsserver befindet, erhält die von der Anwendung erzeugten abgehenden Daten und Header, schafft alle in den Headern gefundenen Cookies in einen bestimmten Cookie-Behälter, der der Sitzung des Client zugeordnet ist und konvertiert gewisse URLs in der abgehenden Webseite in die modifizierte Form um, die unten ausführlich beschrieben wird. Das Filter für den abgehenden Datenstrom muss das letzte oder einzige Filter sein, das mit dem abgehenden Datenstrom arbeitet.

[0032] Das Filter für den abgehenden Datenstrom der bevorzugten Ausführungsform fügt eine Zeichenkette, die durch einen Rechtsschrägstrich gekennzeichnet ist, wobei die Zeichenkette vorzugsweise unter Verwendung der modifizierten Base64-Codierung (die Sonderzeichen ”+” und ”/” der Standardcodierung würden entsprechend durch ”–” und ”_” ersetzt) formatiert wird, in den URL zwischen dem Serverteil und dem Pfadteil ein. Die modifizierte Base64-Codierung wird bevorzugt, so dass beliebige binäre Daten in dem eingefügten Teil als eine Reihe von

zulässigen druckbaren URL-Pfad-Zeichen dargestellt werden, obwohl beliebige Mittel akzeptierbar sind, die dies erreichen. Ein Beispiel hierfür ist in Fig. 2 gezeigt. Fig. 2 zeigt den nichtmodifizierten URL **201** sowie den modifizierten URL **203**. Die Zeichenkette, nachstehend als "Token für die schwierige Verteilung" **235** bezeichnet, enthält vier Felder:

1. Ein Verteilungsfeld, das den bestimmten Server **207** kennzeichnet, an den die Client-Sitzung gebunden ist. Vorzugsweise ist dieses Verteilungsfeld die IP-Unternetzadresse des Servers in dem gleichen IP-Unternetz wie das Belastungs-Ausgleichssystem, obwohl auch andere Mittel zur Serverkennzeichnung akzeptierbar sind.
2. Eine Markierung **209** für das Datum und die Zeit. Eine neue Datums-Zeit-Markierung wird von dem Ausgangsfilter in den Ausgangsdatenstrom eingefügt. Sie wird auch vom Belastungs-Ausgleichssystem und vom Eingangsdatenstrom-Filter geprüft, um festzustellen, ob die Bindungsrelation zwischen dem Client und dem Server überaltert ist.
3. Einen "Schlüssel" **211** oder einen Index, der benutzt werden kann, um den richtigen Cookie-Behälter auszuwählen, in dem die Cookies, die zu der bestimmten Client-Server-Bindung gehören, aufbewahrt werden.
4. Eine Prüfsumme oder einen Hashcode **213** (zum Beispiel einen SHA-1 Secure Hash Algorithm (Sicherer Hashcode-Algorithmus)), der geprüft werden kann, um einen gültigen URL für die schwierige Verteilung von einem nichtmodifizierten URL zu unterscheiden.

[0033] Das Ausgangsfilter verzeichnet für die Behandlung nur gewisse MIMF-(Multi-Purpose Internet Mail) Arten (Dokumente, die unter Benutzung einer strukturierten Auszeichnungssprache codiert wurden, einschließlich, aber nicht darauf beschränkt, HTML, WML und XML). Er verzeichnet nicht die Behandlung von "Strommedien", Bildern, oder MIME-Typen mit vom Internet heruntergeladenem Code.

[0034] Wir beziehen uns jetzt auf Fig. 4A – bevor eine Client-Server-Bindung hergestellt wird, sieht das Belastungs-Ausgleichssystem nur einen unmodifizierten Standard-URL in dem Eingangsdatenstrom. Das Belastungs-Ausgleichssystem benutzt eine Standard-Belastungs-Ausgleichstechnologie außerhalb des Bereiches dieser Erörterung, um diese anfängliche eingehende Anforderung zu dem geeignetsten Server **401** zu leiten.

[0035] Wenn diese Anforderung beim Anwendungsserver ankommt und keine Bindung vorhanden ist, erzeugt das Filter für den Eingangsdatenstrom der vorliegenden Erfindung ein Token **403** für die schwierige Verteilung zum späteren Einfügen in den Ausgangsdatenstrom, so wie oben beschrieben. Das dritte Feld der Zeichenkette ist ein neuer Schlüssel, der auf ein Speicherobjekt zugreift, in dem Cookies für die Sitzung gespeichert werden können **405**. Dieser Schlüssel dient auch dazu, die Eingangs- und die Ausgangsströme, die einer gegebenen Bindung zugeordnet sind, miteinander zu verbinden. In einer bevorzugten Ausführungsform erzeugt das Eingangsfilter ein bestimmtes Cookie, in das es den eben erzeugten Schlüssel einfügt, und platziert dieses Cookie in den Header. Dieses "Schlüssel-Cookie" wird dazu benutzt, Zustandsinformationen, nämlich den Schlüssel selbst, während der Datenübertragung vom Eingangsfilter zur Anwendung und zurück zum Ausgangsfilter zu speichern. Alternativ hierzu kann das Eingangsfilter den Schlüssel in einem Verbindungssteuerobjekt speichern, das der TCP-Verbindung zugeordnet ist. Obwohl es für die Ausführung der vorliegenden Erfindung nicht von besonderer Wichtigkeit ist, wo der Schlüssel genau gespeichert wird, ist es wichtig, den Schlüssel an einer Stelle zu speichern, wo er später von dem Filter des Ausgangsdatenstromes so abgerufen werden kann, dass die Verbindung zur Sitzung ermöglicht wird.

[0036] Das Eingangsfilter leitet dann die Daten zum nächsten Filter in der Kette **409**, wenn mehrere Eingangsfilter **407** vorhanden sind, oder direkt zur Anwendung **411**, wenn es nur ein Eingangsfilter gibt.

[0037] Wir beziehen uns jetzt auf Fig. 4B – nachdem die Anwendung die ankommenden Daten verarbeitet und einen abgehenden strukturierten Strom der Auszeichnungssprache erzeugt hat, erhält das Filter der vorliegenden Erfindung für den Ausgangsdatenstrom die Steuerung **421**. An diesem Punkt kann der Header Cookies enthalten, in denen die Anwendung Zustandsinformationen gespeichert hat, die zu dem bestimmten Client gehören. Wenn das Eingangsfilter die in der vorliegenden Erfindung beschriebene "Schlüssel-Cookie-"Technik **423** benutzt hat, um den Schlüssel zu dem Ausgangsfilter zu transportieren, ist dieses Cookie auch in dem Header vorhanden. Das Ausgangsfilter ruft den Schlüssel für die Sitzung **425** ab, entfernt alle Cookies **427** aus dem Header und speichert sie **429** in dem durch den Schlüssel indizierten Cookiebehälter. Es ist wichtig zu beachten, dass das Ausgangsfilter der vorliegenden Erfindung als Teil eines Transcodierungsprozesses realisiert werden könnte und die Pfadlänge der Filterung des Ausgangsdatenstroms nicht wesentlich erhöhen sollte, wenn die Daten bereits analysiert, formatiert und während einer Transcodierungsoperation auf den Ausgangsdatenpfad kopiert wurden.

[0038] Als Nächstes modifiziert das Ausgangsfilter null oder mehr URLs innerhalb des strukturierten Auszeichnungssprachen-Dokumentes **431** (structured markup language document). Alle URLs, die sich auf die gleiche Transaktion beziehen, müssen modifiziert werden. In der Praxis werden alle URLs modifiziert, die sich auf den bestimmten Server oder re-

lativ auf diesen Server beziehen. (Alternativ hierzu könnte ein Software-Entwicklungswerkzeug zur Verfügung gestellt werden, das es dem Anwendungsprogrammierer ermöglicht, gewisse URLs als zu der Transaktion gehörig zu markieren; in diesem Falle würden nur diese von dem Ausgangsfilter modifiziert, jedoch wird diese Ausführung weniger vorgezogen, da sie erfordert, dass Anwendungsprogramme modifiziert werden.) Das Ausgangsfilter der vorliegenden Erfindung erzeugt die oben erwähnte Zeichenkette **433** für die schwierige Verteilung und erneuert die Datums-Zeit-Markierung **435**. Die Modifikation besteht aus der Einfügung dieser Zeichenkette für die schwierige Verteilung in jede der ausgewählten URLs **437** (dieser Prozess wird manchmal als "URL-Zurückschreiben" (URL-rewriting) bezeichnet). In der bevorzugten Ausführungsform kann dies durch Aufruf einer vorhandenen URL-Rückschreibefunktion mit neuen Parametern leicht erreicht werden. (Bezüglich zusätzlicher Information zum Stand der Technik beim URL-Zurückschreiben siehe die Dokumentation zum IBM WebSphere Application Server bei http://web/doc/whatis/icesessta.html. Dieses Dokument erläutert den aktuellen Stand der Technik bei der Sitzungszustands-Korrelation, der Cookie-Verwaltung und dem URL-Zurückschreiben.) Schließlich transportiert das Ausgangsfilter der vorliegenden Erfindung den entstehenden Strom zur Netzwerkschicht zur Übertragung zum Client **439**.

[0039] Wenn der Client den modifizierten HTTP-Strom erhält, kann der Nutzer irgendeine modifizierte URL auswählen, was zu einer Eingangsanforderung führt, die ein Token für die schwierige Verteilung enthält. Die vorliegende Erfindung besitzt einen Vorteil gegenüber dem Stand der Technik, der optionale oder variable Eigenschaften des Protokolls ausnutzte, zum Beispiel die IP-Adresse der Quelle, den SSL-ID oder Cookies, da die URL obligatorischer HTTP-Inhalt ist, der nicht entfernt, verändert oder wählbar gemacht werden kann.

[0040] Wir beziehen uns noch einmal auf Fig. 3 – die ankommende Anforderung erreicht das Belastungs-Ausgleichssystem **301**. Das Belastungs-Ausgleichssystem prüft die Daten, um festzustellen, ob ein gültiges Token für die schwierige Verteilung vorhanden ist **303**. In der bevorzugten Ausführungsform besteht die Prüfung aus dem Vergleich der eingebetteten Prüfsumme oder digitalen Signatur (Hashcode, zum Beispiel ein SHA-1 – Hashcode) mit einer berechneten Prüfsumme oder einer digitalen Signatur. Wenn kein gültiges Token für die schwierige Verteilung vorhanden ist, wird das Paket wie üblich verarbeitet **305**. Das Belastungs-Ausgleichssystem testet dann die Datums-Zeit-Kennzeichnung, um festzustellen, ob die Sitzungsbindung überaltert ist. Wenn die Differenz zwischen Datums-Zeit-Markierung und der aktuellen Datums-Zeit-Markierung eine gewisse Konstante überschreitet, wird die Bindung als überaltert angesehen, und die Daten werden verarbeitet, als ob kein Feld für die schwierige Verteilung vorhanden wäre, d. h. vorzugsweise unter Anwendung vorhandener Belastungs-Ausgleichstechniken, um einen Server zur Bearbeitung der Anforderung auszuwählen und danach eine neue Sitzungsbindung und ein entsprechendes Feld für die schwierige Verteilung zu erzeugen. In jedem Fall wird das Verteilungsfeld in dem Token benutzt, um das Paket zu dem gekennzeichneten Server **313** zu leiten. Nach der Ankunft des Paketes beim Anwendungsserver erhält der Filter für den Eingangsdatenstrom die Steuerung.

[0041] Das Filter für den Eingangsdatenstrom prüft das Token für die schwierige Verteilung und das Datums-Zeit-Token auf die gleiche Art und Weise, wie sie oben erklärt wurde. Wenn das Token für die schwierige Verteilung gültig und nicht überaltert ist, benutzt das Eingangsfilter den Schlüssel, um auf den Cookie-Behälter zuzugreifen, in dem die Cookies für die bestimmte Client-Server-Bindung gespeichert werden. Es entfernt das Verteilungstoken und speichert den Schlüssel entweder in einem Schlüsselcookie oder in dem Steuerblock der TCP-Verbindung, so wie oben beschrieben. Es fügt alle die ausgewählten Cookies in den Header ein und sendet die Daten entweder zu dem nächsten Eingangsfilter, falls vorhanden, oder direkt an die Anwendung.

[0042] Das Token für die schwierige Verteilung kann auch als Grundlage für das Belastungs-Ausgleichssystem dienen, um eine differenzierte Qualität der Dienstleistung zu bieten. Da man imstande ist, eine bestimmte Client-Sitzung zu erkennen oder eine Transaktion, die gerade bearbeitet wird, von einer zu unterscheiden, die noch nicht begonnen hat, können geeignete Entscheidungen getroffen werden, um gewisse Pakete gegenüber anderen zu priorisieren.

[0043] Die vorliegende Idee kann auch so erweitert werden, dass eine Ausführung mit hoher Verfügbarkeit bereitgestellt wird, ohne die zugrunde liegende Theorie wesentlich zu ändern. Hierfür ist anstelle eines Cookie-Behälters oder Objektspeichers pro Anwendungsserver ein gemeinsam benutzter Objektspeicher (Object Store) mit geringer Zusatzbelastung vorhanden, der allen Servern im Cluster (und wahlweise dem Belastungs-Ausgleichssystem) zugänglich ist.

[0044] Anstelle des Speicherns der Cookies in einem Cookie-Behälter werden die Cookies in dem gemeinsam benutzten Objektspeicher gespeichert. Zusätzlich zu den Cookies würde die Anwendung neu geschrieben, um alle ihre relevante Zustandsinformationen bezüglich einer stattfindenden Sitzung in dem Objektspeicher zu speichern. Die Anwendung würde beim Empfang von irgendwelchen ankommenden Daten alle ihre Zustandsinformationen aus dem Objektspeicher herausziehen müssen.

[0045] Wenn mehrere Instanzen der Anwendung auf verschiedenen Servern in einem Cluster laufen und ein Belastungs-Ausgleichssystem ankommende Anforderungen an verschiedene Server leitet, ist jeder Server imstande, allen anderen Servern einen identischen Dienst zur Verfügung zu stellen. Somit kann im Falle des Zusammenbruches einer Anwendung oder eines Serverausfalls die Transaktion ohne Unterbrechung fortgesetzt werden, wenn die gespeicherte Zustandsinformationen intakt und zugänglich sind.

Patentansprüche

1. Verfahren zur Herstellung einer dauerhaften Beziehung zwischen einem Client-System und einem Server,
wobei der Server aus einer Vielzahl von Servern stammt, die von einem Dispatcher verwaltet werden, und das Client-System unter Benutzung eines Universellen Ressourcen-Lokalisierers (URL) auf den Server zugreift,
bei dem der Dispatcher eine Informationsanforderung vom Client-System empfängt und bestimmt, welcher aus der Vielzahl von Servern für die Erfüllung der Anforderung auszuwählen ist;
dadurch gekennzeichnet,
– dass der ausgewählte Server ein Token erzeugt, das wenigstens einen Bezeichner für den ausgewählten Server, eine Datums-Zeit-Markierung und einen Schlüssel umfasst, wobei der Schlüssel für den Zugriff auf einen Speicherbereich für Informationen bezüglich der dauerhaften Beziehung zum Client-System verwendet wird;
– dass das Token in den URL eingefügt wird; und
– dass der ausgewählte Server eine Antwort mit dem in den URL eingefügten Token zum Client-System sendet, um das Client-System für die Dauer einer Sitzung an diesen Server zu binden.

2. Verfahren nach Anspruch 1, wobei das Token unter Benutzung einer modifizierten Base64-Codierung codiert wird.

3. Verfahren nach Anspruch 1, wobei das Token zusätzlich ein Prüfsummen- oder Hashcode-Überprüfungsfeld umfasst.

4. Verfahren nach Anspruch 3, wobei der Hashcode ein SHA-1-Hashcode ist, der über den Bezeichner für den ausgewählten Server, das Datums-Zeit-Token und den Schlüssel berechnet wird.

5. Verfahren nach Anspruch 3, wobei die Prüfsumme oder der Hashcode unter Benutzung einer modifizierten Base64-Codierung codiert wird.

6. Verfahren nach Anspruch 1, wobei die Informationen bezüglich der andauernden Beziehung als ein Cookie auf dem Server gespeichert sind.

7. Verfahren zum Verteilen einer Anforderung eines Client-Systems zu einem bestimmten Server aus einer Vielzahl von redundanten Servern, die sich hinter einem Dispatcher befinden,
bei dem der Dispatcher aufgrund eines Universellen Ressourcen-Lokalisierers (URL) eine Anforderung empfängt;
dadurch gekennzeichnet,
– dass der Dispatcher feststellt, ob dieser URL ein gültiges Verteilungstoken enthält, wobei das Verteilungstoken wenigstens einen Bezeichner für den bestimmten Server an dem das Client-System für die Dauer einer Sitzung gebunden ist, eine Datums-Zeit-Markierung und einen Schlüssel umfasst, wobei der Schlüssel für den Zugriff auf einen Speicherbereich für Informationen bezüglich einer Sitzung zwischen dem bestimmten Server und dem Client-System verwendet wird;
– und, falls dieser URL ein gültiges Verteilungstoken enthält, der Dispatcher feststellt, ob eine Sitzungsbindung, die durch dieses Verteilungstoken angezeigt wird, alt ist;
– und falls dieses Verteilungstoken nicht alt ist, der Dispatcher die Anforderung einschließlich des URL an den bestimmten Server sendet, der durch das gültige Verteilungstoken angezeigt wird;
– dass der bestimmte Server das gültige Verteilungstoken aus dem URL entfernt;
– dass der bestimmte Server den Schlüssel des gültigen Verteilungstokens speichert, so dass ein Filter für den abgehenden Datenstrom während der Verarbeitung einer abgehenden, mit dieser Anforderung zusammenhängenden Antwort nachfolgend auf die entsprechenden Sitzungsinformationen zugreifen kann;
– dass der bestimmte Server auf diese Sitzungsinformationen zugreift und sie in die Anforderung einfügt.

8. Verfahren nach Anspruch 7, wobei ein zusätzliches Filtern des URL vor dem Schritt des Weiterleitens erfolgt.

9. Verfahren nach Anspruch 1 bis 8, wobei alle Filterschritte innerhalb des Dispatchers ausgeführt werden.

10. Verfahren zum Senden von Informationen über eine Sitzung von einer Anwendung an ein anforderndes Client-System, wobei sich diese Anwendung auf einem Server aus einer Vielzahl von redundanten Servern befindet, die sich hinter einem Dispatcher befinden, wobei dieses Verfahren die folgenden Schritte umfasst:
Empfang der Antwortinformationen von der Anwendung, wobei die Antwortinformationen einen URL (Universellen Ressourcen-Lokalisierers) enthalten;
Feststellen, ob im URL ein Schlüsselcookie für die Speicherung der Sitzungsinformationen zwischen dem Client-System und der Anwendung benutzt wurde;

Abrufen eines Sitzungsschlüssels aus dem Schlüsselcookie, falls ein Schlüsselcookie für die Speicherung der Sitzungsinformationen verwendet wurde;
Abrufen des Sitzungsschlüssels aus einem Steuerblock der TCP-Verbindung, falls kein Schlüsselcookie benutzt wurde;
Entfernen aller Cookies aus den Antwortinformationen;
Speichern der entfernten Cookies in einem vorher festgelegten Speicherbereich;
Aktualisieren des URL, um das Entfernen dieser Cookies anzuzeigen;
Erzeugen einer Zeichenkette für das schwierige Verteilen, die wenigstens einen Bezeichner für den ausgewählten Server, eine Datums-Zeit-Markierung und einen Sitzungsschlüssel umfasst, wobei der Sitzungsschlüssel für den Zugriff auf den Cookie im festgelegten Speicherbereich verwendet werden kann;
Aktualisieren eines Datums-Zeit-Tokens in dieser Zeichenkette für das schwierige Verteilen;
Einfügen der Zeichenkette für das schwierige Verteilen in den URL; und
Übertragen der Antwortinformationen einschließlich des URL zu dem Client-System.

11. Verfahren nach Anspruch 10, wobei die Antwortinformationen vor dem Feststellungsschritt von der Anwendung durch ein oder mehrere Filter übertragen werden.

12. Computerprogrammprodukt zur Realisierung eines Verfahrens zur Herstellung einer dauerhaften Beziehung zwischen einem Client-System und einem Server, nach einem der Ansprüche 1 bis 6.

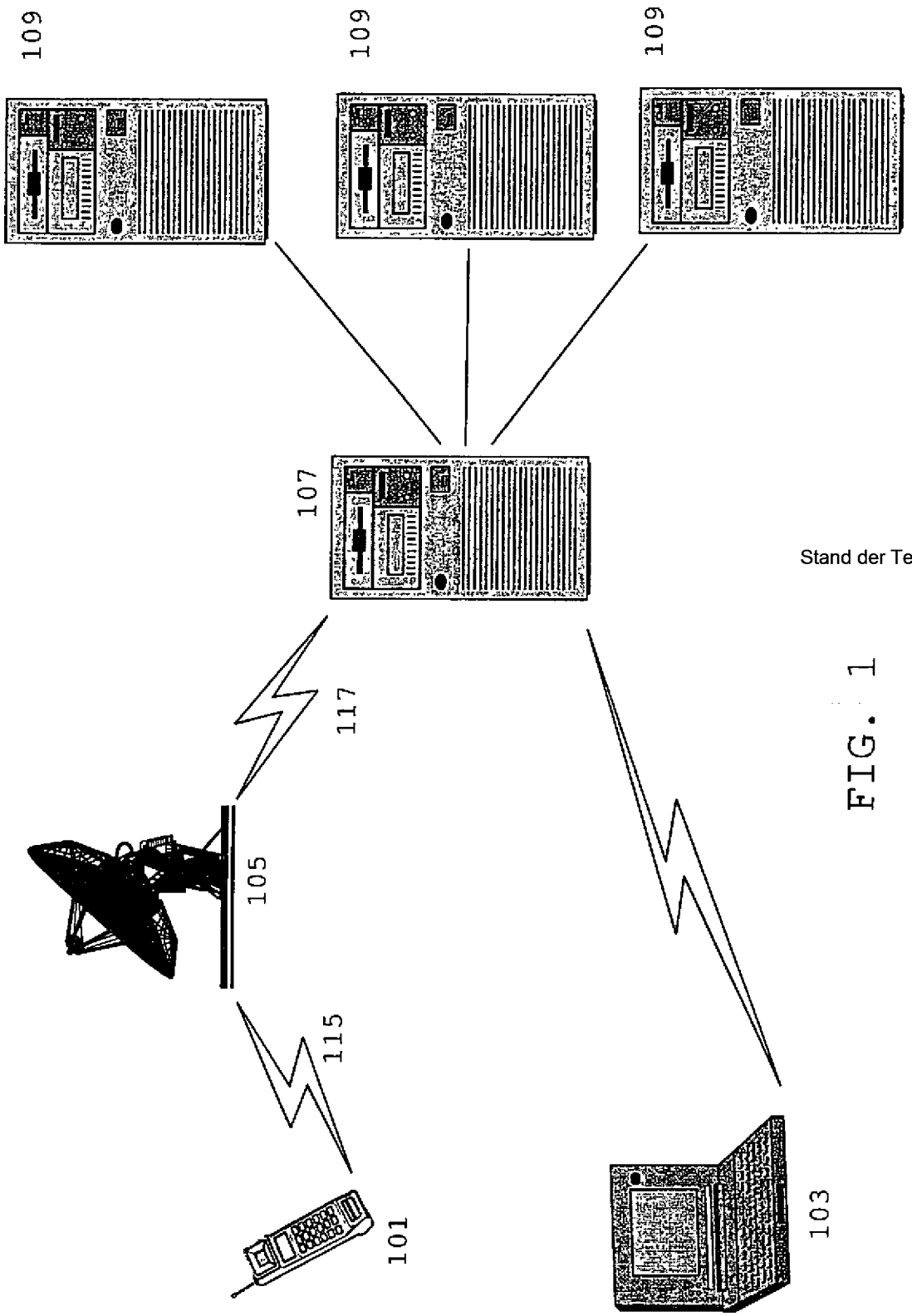
13. Computerprogrammprodukt zur Realisierung eines Verfahrens zum Verteilen einer Anforderung eines Client-Systems zu einem bestimmten Server aus einer Vielzahl von redundanten Servern, nach einem der Ansprüche 7 bis 9.

14. Computerprogrammprodukt zur Realisierung eines Verfahrens zum Senden von Informationen über eine Sitzung von einer Anwendung an das anfordernde Client-System, nach einem der Ansprüche 10 oder 11.

15. Netzwerk-Dispatcher zur Durchführung eines Verfahrens nach einem der Ansprüche 1 bis 6.

Es folgen 7 Blatt Zeichnungen

Anhängende Zeichnungen



Stand der Technik

FIG. 1

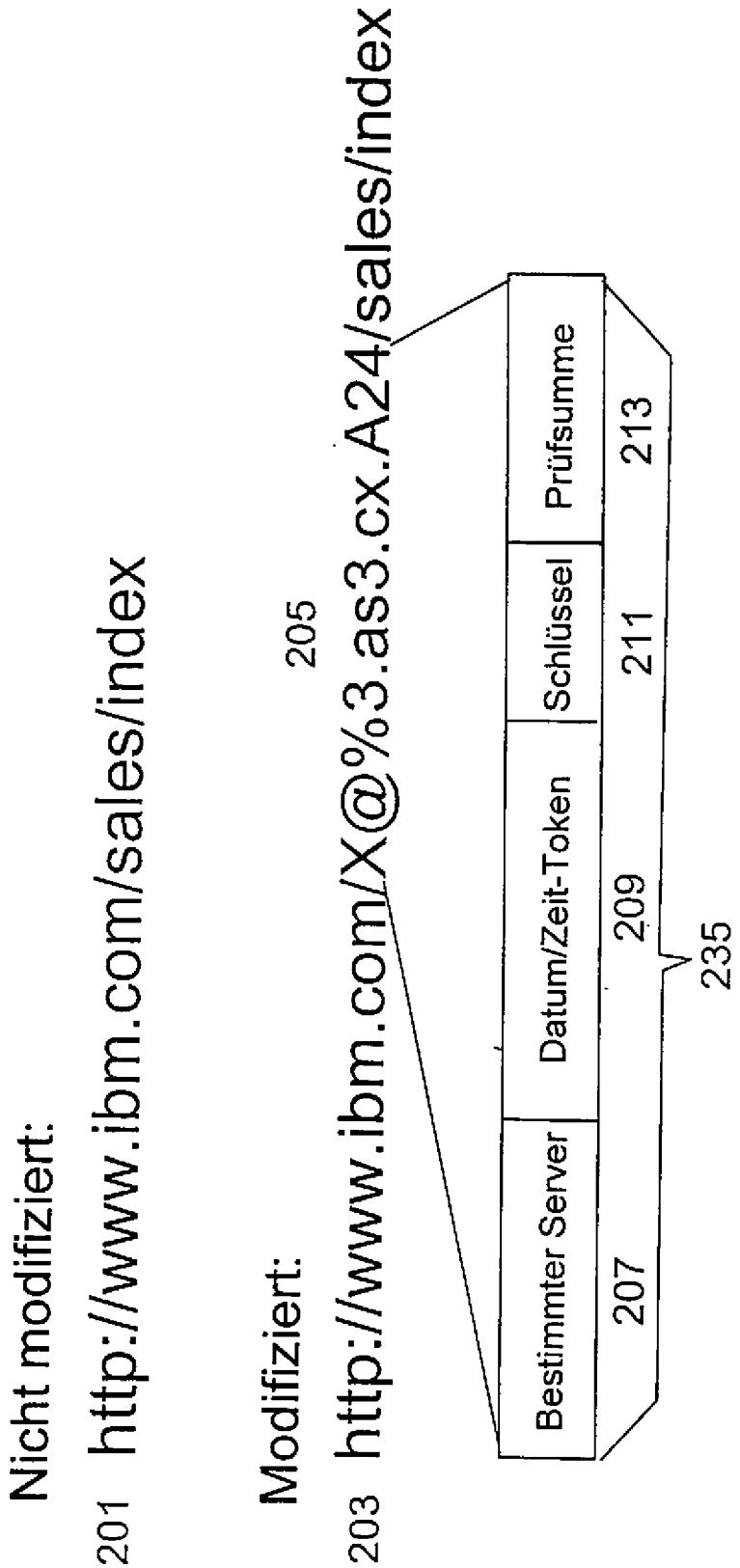

Nicht modifiziert:
201 http://www.ibm.com/sales/index
Modifiziert:
205
203 http://www.ibm.com/X@%3.as3.cx.A24/sales/index
Bestimmter Server
Datum/Zeit-Token
Schlüssel
Prüfsumme
207
209
211
213
235

FIG. 2

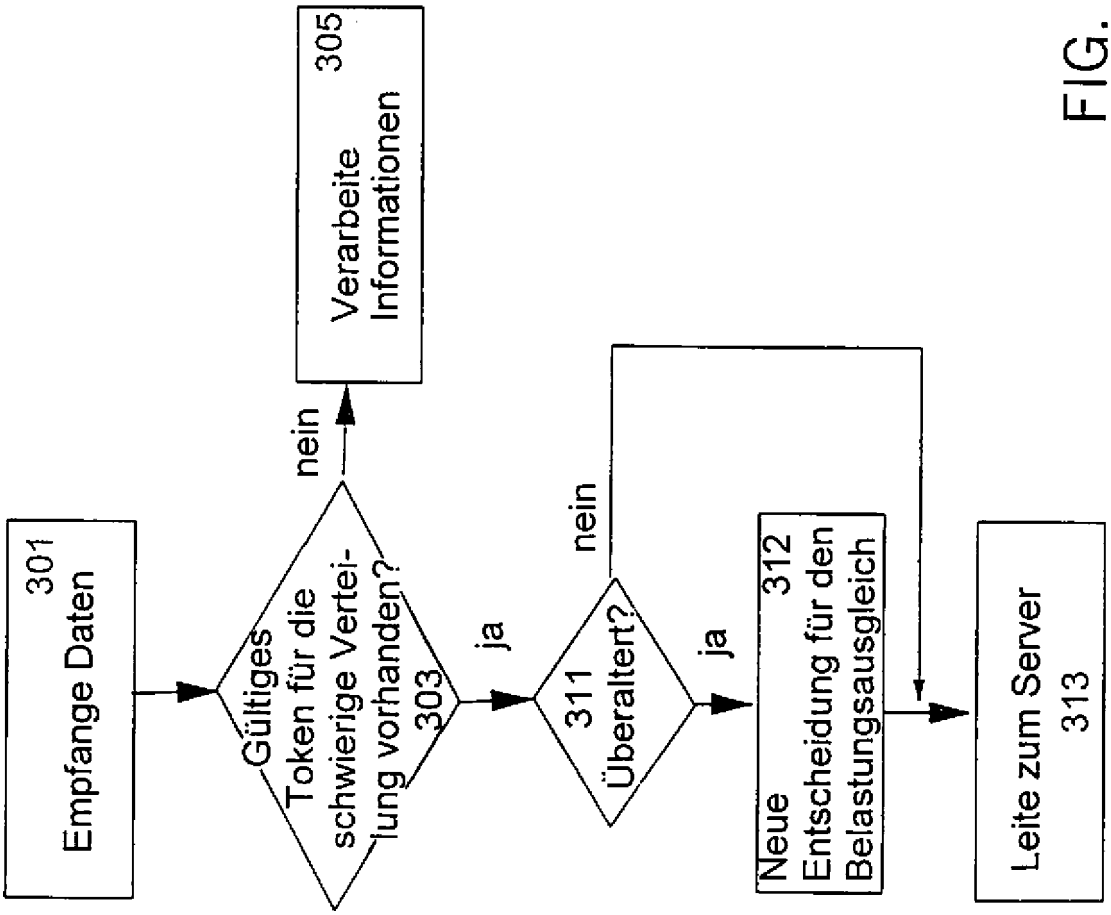
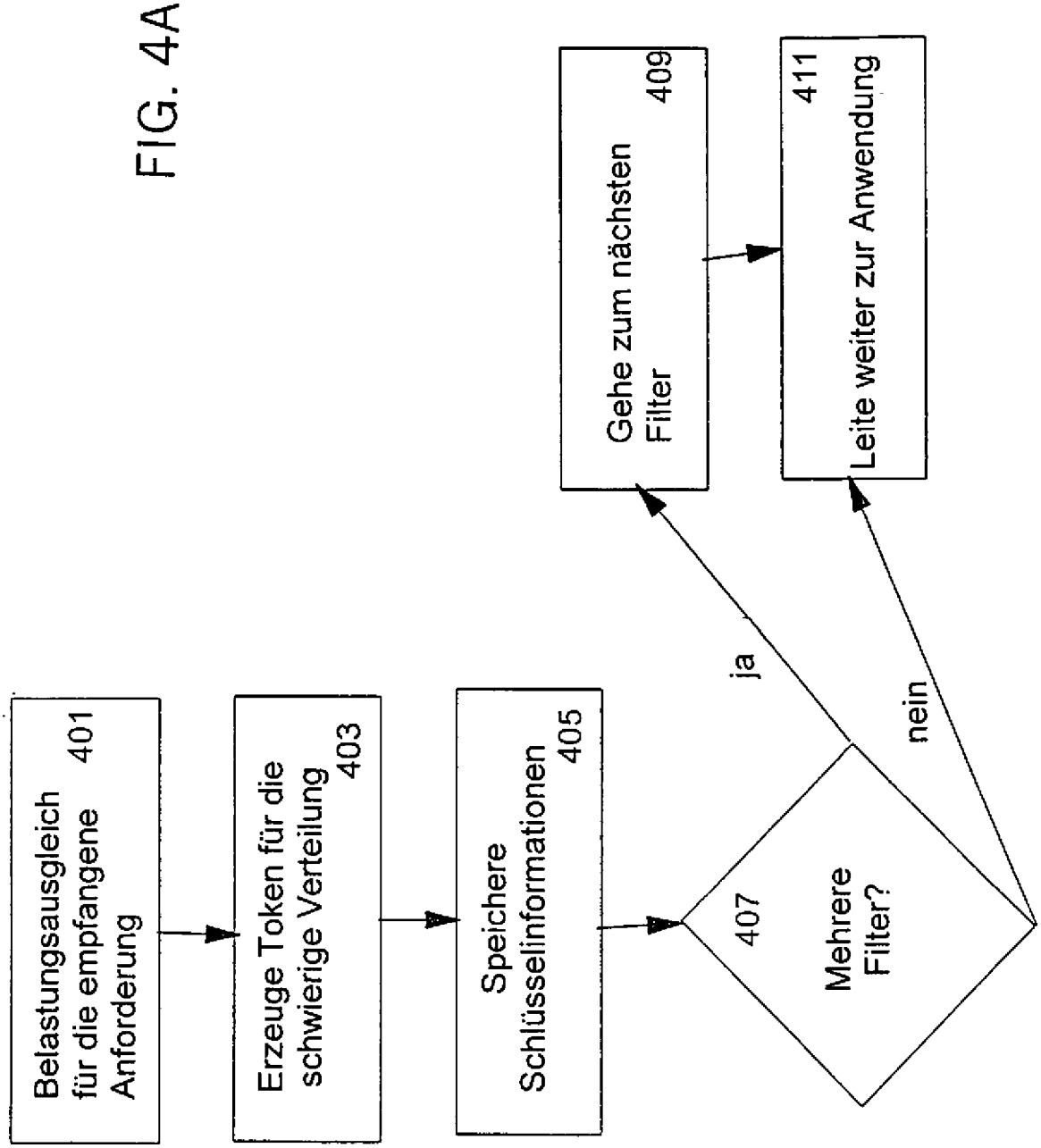

301
Empfange Daten
Gültiges Token für die schwierige Verteilung vorhanden? 303
nein
305
Verarbeite Informationen
ja
311
Überaltert?
nein
ja
312
Neue Entscheidung für den Belastungsausgleich
Leite zum Server 313

FIG. 3

FIG. 4A


Belastungsausgleich für die empfangene Anforderung
401
Erzeuge Token für die schwierige Verteilung
403
Speichere Schlüsselinformationen
405
Mehrere Filter?
407
ja
nein
Gehe zum nächsten Filter
409
Leite weiter zur Anwendung
411

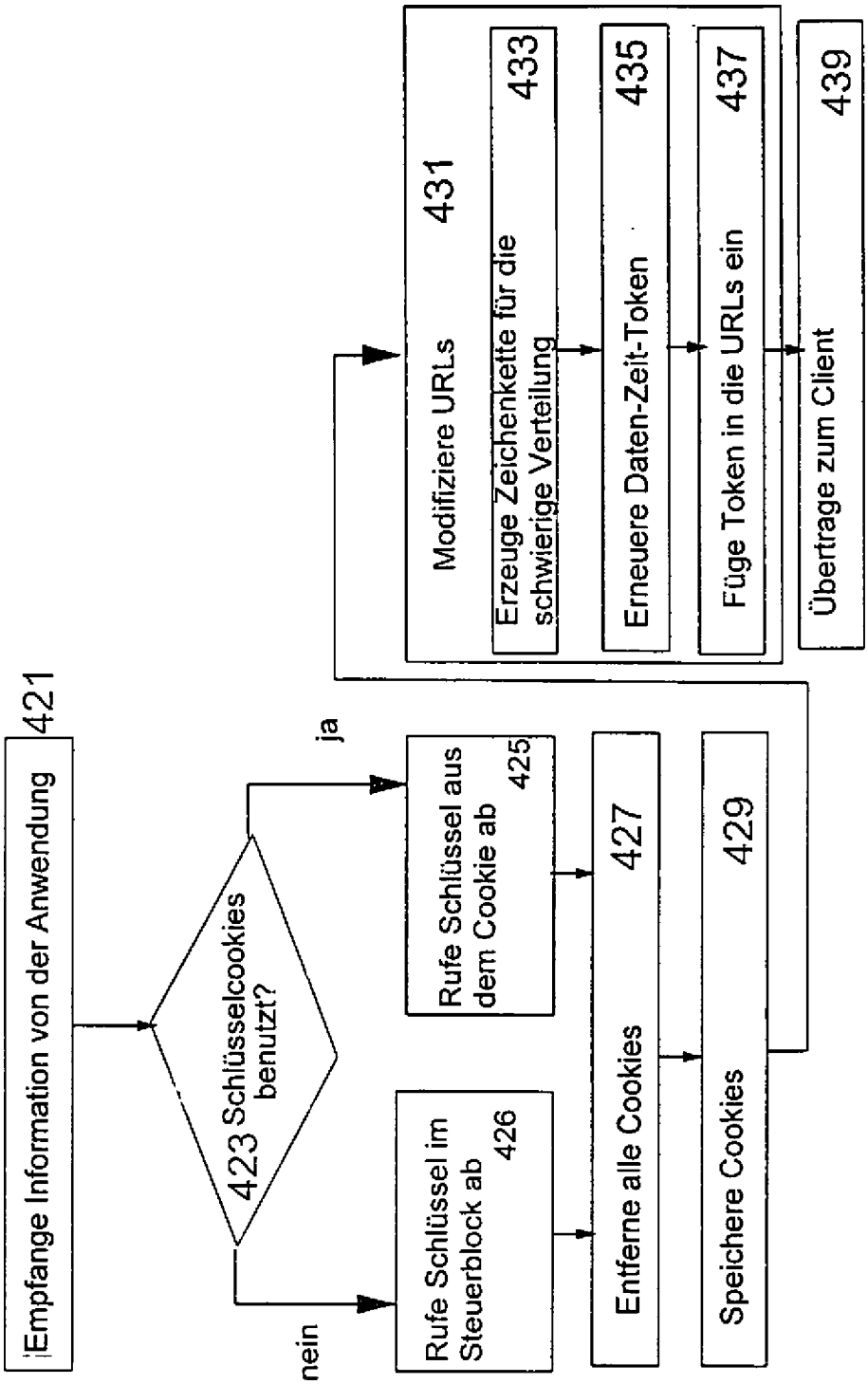

Empfange Information von der Anwendung
421
Schlüsselcookies benutzt?
423
ja
nein
Rufe Schlüssel aus dem Cookie ab
425
Rufe Schlüssel im Steuerblock ab
426
Entferne alle Cookies
427
Speichere Cookies
429
Modifiziere URLs
431
Erzeuge Zeichenkette für die schwierige Verteilung
433
Erneuere Daten-Zeit-Token
435
Füge Token in die URLs ein
437
Übertrage zum Client
439

FIG. 4B

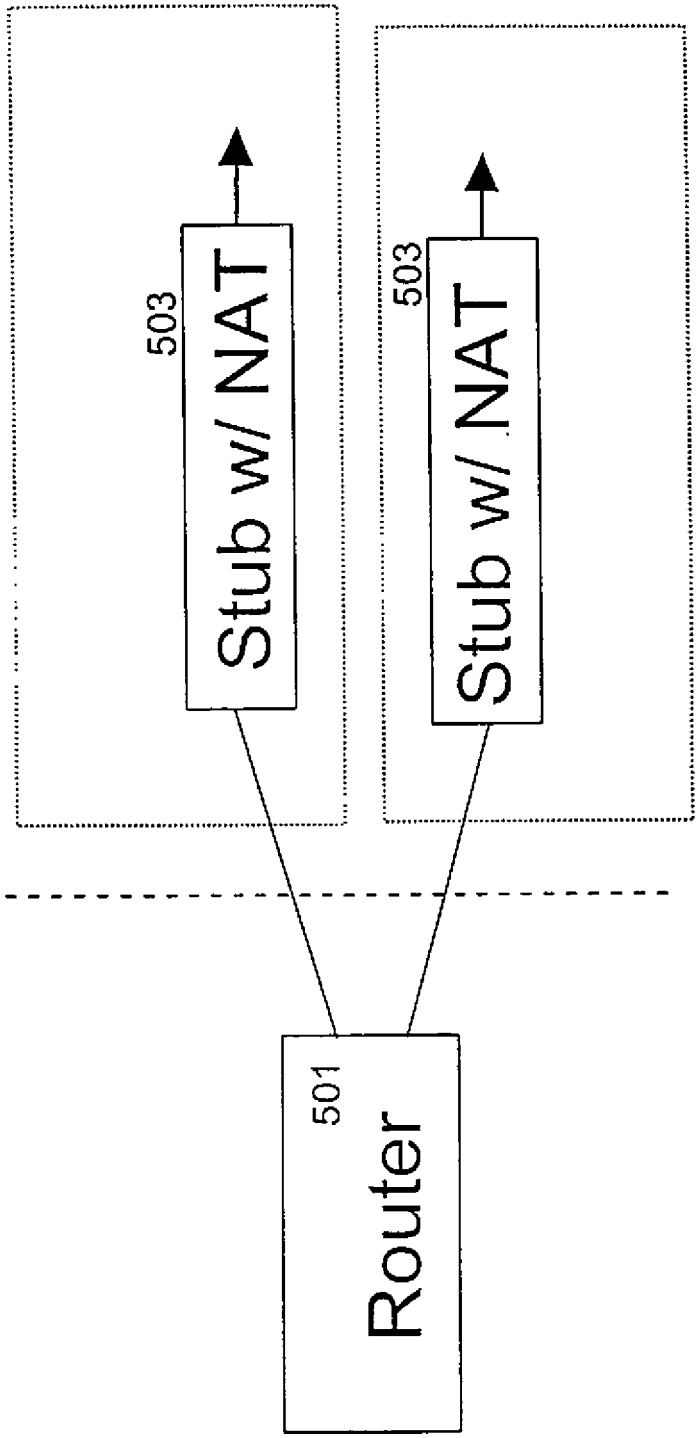
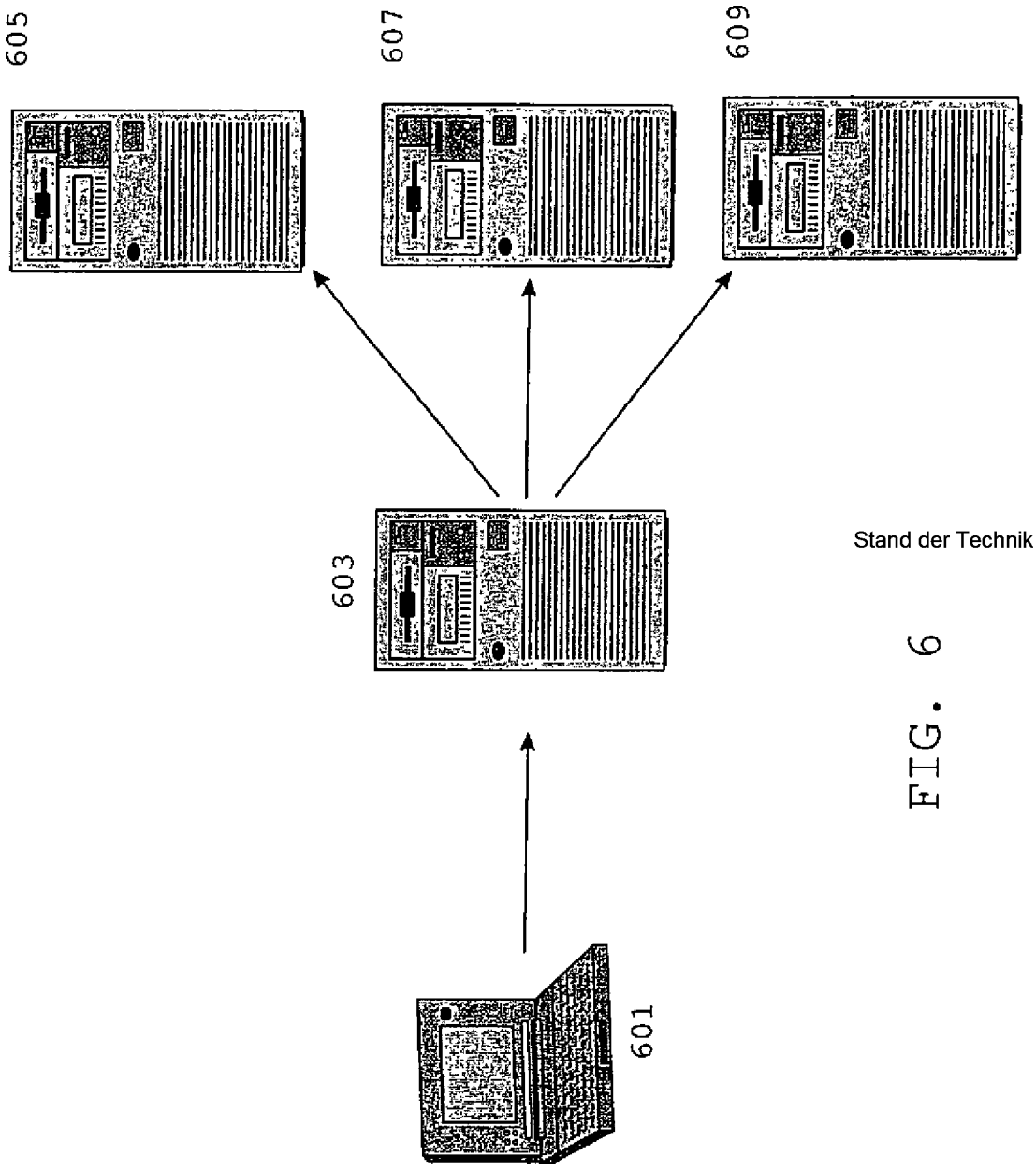


FIG. 5



Stand der Technik

FIG. 6