



(19)
Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 696 35 338 T2** 2006.08.03

(12) **Übersetzung der europäischen Patentschrift**

(97) **EP 0 779 759 B1**

(21) Deutsches Aktenzeichen: **696 35 338.5**

(96) Europäisches Aktenzeichen: **96 309 006.3**

(96) Europäischer Anmeldetag: **11.12.1996**

(97) Erstveröffentlichung durch das EPA: **18.06.1997**

(97) Veröffentlichungstag

der Patenterteilung beim EPA: **26.10.2005**

(47) Veröffentlichungstag im Patentblatt: **03.08.2006**

(51) Int Cl.⁸: **H04Q 7/24 (2006.01)**
H04L 12/28 (2006.01)

(30) Unionspriorität:

570210	11.12.1995	US
570384	11.12.1995	US

(84) Benannte Vertragsstaaten:

**AT, BE, CH, DE, DK, ES, FI, FR, GB, GR, IE, IT, LI,
LU, MC, NL, PT, SE**

(73) Patentinhaber:

Openwave Systems Inc., Redwood City, Calif., US

(72) Erfinder:

Rossmann, Alain, Palo Alto, US

(74) Vertreter:

Berendt und Kollegen, 81667 München

(54) Bezeichnung: **Verfahren und Architektur für ein interaktives Zwei-Wege-Datenübertragungsnetzwerk**

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patent- und Markenamt inhaltlich nicht geprüft.

Beschreibung

[0001] Ein Teil der Offenbarung dieses Patentdokuments enthält Material, das einen Anhang I, einen Anhang II und [Fig. 10A](#) bis [Fig. 10T](#) enthält, welches Gegenstand eines Copyright-Schutzes ist, ist aber nicht darauf beschränkt. Der Copyright-Besitzer hat keine Einwände gegenüber der Faksimile-Wiedergabe des Patentdokuments oder Patentoffenbarung, wie es in den Patentakten oder -dokumenten des Patent- und Markenamts erscheint, durch Irgendjemanden, reserviert aber sonst alle Copyright-Rechte für sich.

[0002] Diese Erfindung betrifft allgemein eine Datenkommunikation und insbesondere ein Verfahren und ein System zum Ermöglichen, dass jeweilige mobile zu Zweiwegedaten fähige drahtlose Kommunikationsvorrichtungen in Taschengröße eine interaktive Zweiwegekommunikation mit einer Anwendung auf einem Server auf einem drahtgebundenen Computernetzwerk haben.

[0003] Für wenigstens die letzten fünf Jahre hat die Industrie für eine drahtlose Kommunikation versucht, die Computeranwendung mit drahtlosen Kommunikationen zu verknüpfen. Diese industrieweite Anstrengung hat das Versprechen gehalten, Software-Intelligenz zu Telekommunikationsvorrichtungen zu bringen, die mobile drahtlose Kommunikationsvorrichtungen, wie beispielsweise Zellulartelefone, enthalten.

[0004] Nach Jahren der Forschung und Entwicklung und einer Investition von hunderten Millionen Dollar durch einige der größten Firmen auf dem Gebiet, wie beispielsweise Motorola, AT&T, Sony, Matsushita, Phillips und IBM, sind die Ergebnisse nichts außer enttäuschend gewesen. Typischerweise enthalten die in intelligenten Kommunikationsvorrichtungen, die aus diesen Anstrengungen resultieren, sowohl die Hardware, die für ein Computermodul nötig ist, als auch die Hardware für ein drahtloses Kommunikationsmodul. Beispiele für solche Produkte sind Simon von IBM und Bell South, MagicLink von Sony und Envoy von Motorola.

[0005] Ein Dokument von Liljeberg M et al. mit dem Titel "Optimizing world-wide web for weakly connected mobile workstations: and An Indirect Approach" – International Workshop von Services in Distributed and Networked Environments, 5. Juni 1995, Seiten 132-193 (XP-000764774) beschreibt eine Kommunikationsarchitektur, bei welcher eine mobile Workstation mit einem HTTP-Agenten mittels einer langsamen Kommunikationsverbindung verbunden ist. Der HTTP-Agent fängt Anforderungen nach Hypertext-Dokumenten von der Workstation ab und leitet sie über eine drahtlose Verbindung weiter zu einem HTTP-Proxy. Der HTTP-Proxy posiert dann als der Client und verbindet mit einem geeigneten Server, um ihn zu fragen, die Anforderung durchzuführen. Die Antwort vom Server wird dann zu dem HTTP-Agenten gesendet, der die Antwort zu der Workstation kommuniziert. Kompressionsalgorithmen können für den Datentransfer zwischen dem Proxy und dem Agenten verwendet werden oder der Agent kann Daten in seinem lokalen Cache speichern, und einige von diesen Daten können hervorgeholt werden. Insbesondere ist eine Kompression von HTTP-Anfangsblöcken oder transferierten Daten zum Bewahren einer Bandbreite offenbart.

[0006] Ein Dokument von Kaasshoek M F et al. mit dem Titel "Dynamic Documents: Mobile Wireless Access to the WW" – Proceedings, Workshop on Mobile Computing Systems and Applications, 8. Dezember 1994, Seiten 179-184 (XP002016896) beschreibt ein Kommunikationssystem für eine mobile Workstation unter Verwendung einer drahtlosen Kommunikation. Wenn eine HTTP-Anforderung durch die Workstation durchgeführt wird, wird das Dokument von einem Server angefordert und wird eine genaue Kopie der Antwort in einem Cache gespeichert. Zufällig herausgenommene Einträge aus dem Dokumenten-Cache können weggeworfen sein oder die Einträge können überaltert sein. Eine Strategie zum Hervorholen von Dokumenten ist basierend auf der Vorgeschichtenliste eines Anwenders offenbart. Dynamische Dokumente werden verwendet, die Programme enthalten, die auf den Workstations laufen und das anzuzeigende Dokument erzeugen.

[0007] Grundsätzliche Entwicklungs- und Kostenprobleme, die direkt aus dem durch die Entwickler von diesen intelligenten Kommunikationsvorrichtungen genommen ist, haben eine weit verbreitete Akzeptanz dieser Vorrichtungen auf dem Markt beschränkt. Die Kombination eines drahtlosen Kommunikationsmoduls mit einem Rechenmodul führt zu einer Vorrichtung, die zu großräumig, zu teuer und zu wenig flexibel ist, um den Marktanforderungen zu genügen bzw. diese zu berücksichtigen.

[0008] Die Kombination aus den zwei Modulen ist zu groß und zu schwer, um in die Tasche eines Anwenders zu passen. Eine Taschengröße ist eine Hauptanforderung des mobilen Kommunikationsmarkts, welche durch diese Vorrichtungen unerfüllt bleibt.

[0009] Zusätzlich sind die Kosten für diese Vorrichtungen nahe der Summe der Kosten des Computermoduls und des Kommunikationsmoduls, welche um einen Endanwenderpreis von eintausend Dollar liegen. Eine

Marktforschung zeigt an, dass der Markt für intelligente drahtlose Kommunikationsvorrichtungen bei Preisen von etwa \$ 300 liegt. Selbst bei einer Senkung der zusammengesetzten Kosten um 20% würde es fünf Jahre dauern, bis die Kombinationseinheiten Preisanforderungen von Kunden von heute erfüllen. Es ist daher unwahrscheinlich, dass durch Kombinieren eines Computers und eines drahtlosen Moduls entwickelte Vorrichtungen, gleichgültig wie miniaturisiert und kostenreduziert sie sind, die Kostenanforderung des Marktes während dieser Dekade erfüllen können.

[0010] Um auf dem Markt Erfolg zu haben, müssen intelligente drahtlose Kommunikationsvorrichtungen eine weite Vielfalt von Anwendungen unterstützen können, die für jedes Marktsegment spezifisch sind. Typischerweise müssen diese Anwendungen zu der Vorrichtung durch den Endanwender nach einem Kauf hinzugefügt werden. Somit muss die Vorrichtung ein Verfahren zum Laden der anfänglichen Anwendung und für ein nachfolgendes Updaten bzw. Aktualisieren der Anwendung zur Verfügung stellen.

[0011] Die Preisempfindlichkeit für intelligente Kommunikationsvorrichtungen und die Größenbeschränkungen haben die Bedeutung, dass eine intelligente Kommunikationsvorrichtung das Ausmaß eines Kernspeichers bzw. Hauptspeichers (RAM), einer Festplatte oder eines nicht löschbaren Speichers oder eines herkömmlichen Diskettenlaufwerks, die an Computern gemeinsam gefunden werden, nicht unterstützen kann. Diese Beschränkungen schließen die herkömmlichen Wege zum Liefern neuer Anwendungen oder Updates zu intelligenten Kommunikationsvorrichtungen.

[0012] Als Ergebnis laufen auf der gegenwärtigen Masse von intelligenten Kommunikationsvorrichtungen nur die wenigen Anwendungen, die bei der Herstellerfirma in ihre ROMs gebrannt wurden oder die in einer ROM-Karte enthalten sind, die in einen Schlitz gesteckt wird, der für diesen Zweck entwickelt ist. Diesem Schema fehlt die Flexibilität, die zum Laufen lassen der Tausende von Anwendungen nötig ist, die zum Adressieren der fragmentierten Anforderungen des Marktes erforderlich sind, und stellt kein einfaches Verfahren zum Aktualisieren bzw. Updaten der Anwendungen zur Verfügung, nachdem die Vorrichtung verkauft worden ist.

[0013] Zwei andere kommunikationsorientierte Versuche beim Bringen einer Intelligenz zu Telefonen sind der Kurznachrichtendienst (SMS = Short Messaging Service) und eine analoge Anzeigendienstschnittstelle (ADSI = Analog Display Service Interface). SMS spezifiziert, wie Nachrichten zu und von einem Zellulartelefon geliefert werden und wie das Zellulartelefon die Nachrichten speichern sollte. SMS definiert auch eine einfache Verarbeitung, die das Zellulartelefon an der Nachricht durchführen kann, wie beispielsweise ein Anrufen einer Telefonnummer, die in der Nachricht eingebettet ist.

[0014] Die SMS-Architektur ist gleich derjenigen von Funkruf- bzw. Paging-Netzwerken, mit dem Unterschied, dass Vorrichtungen, die die SMS-Architektur implementieren, über dem Steuerkanal des Zellulartelefonnetzwerks arbeiten. SMS wird primär in Europa über das GSM-Netzwerk genutzt.

[0015] SMS-Nachrichten werden nicht in Echtzeit geliefert. Die Zeitverzögerungen können von 30 Sekunden bis zu 10 Minuten reichen, was SMS für Echtzeitanwendungen ungeeignet macht. Der Hauptzweck von SMS besteht in der Lieferung von Nachrichten. SMS spezifiziert kein Anwendungsprotokoll oder Zellulartelefon-Anwendungsmodul, was seine Nützlichkeit beim Laufen lassen von Anwendungen auf Zellulartelefonen weiter beschränkt. Nach einigen Jahren der Nutzung in Europa sind SMS-Implementierungen auf Mitteilungsdienste, wie beispielsweise ein Zweiwege-Paging und eine Sprachmailmitteilung, beschränkt worden.

[0016] SMS als Medium ist ungeeignet zum Bilden von Anwendungen, die die Wiedergewinnung, Manipulation und Speicherung von Information zulassen. Dies ist der Grund dafür, warum die Industrieriesen sich bei ihrer Suche nach einem Hinzufügen von Intelligenz zu Zellulartelefonen nicht SMS zugewendet haben, sondern konsistent versucht haben, ein Computermodule mit einem drahtlosen Kommunikationsmodul zu kombinieren.

[0017] ADSI wurde als eine Erweiterung für interaktive Sprach-Antwort-Systeme entwickelt. ADSI lässt zu, dass ein intelligentes Telefon mit einem kleinen Bildschirm Meldungen bzw. Mahnungen anzeigt, um Anwendern beim Auswählen unter verschiedenen Optionen zu helfen. Es wird angenommen, dass ADSI durch Verwenden von visuellen Meldungen anstelle von mühsamen Sprachmeldungen die Verwendung von interaktiven Sprachdiensten einfacher und schneller macht.

[0018] ADSI lässt zu, dass Daten von dem Serviceprovider zu dem Telefon in der Form von Anzeigeschirmen gesendet werden. ADS lässt auch zu, dass das Telefon durch eine Berührungstonsignalgabe mit einer speziellen Codierung antwortet, um die vollständige alphanumerische Zeichengruppe zu beschreiben. Mit ADSI ist

ein Telefon primär eine passive Vorrichtung. Dienste senden Textanzeigeschirme zu dem Telefon und das Telefon sendet Sortierketten zurück, die die Auswahlen des Anwenders anzeigen, die aus dem Textanzeigeschirm gemacht sind.

[0019] ADSI sieht nicht vor, eine Verarbeitung im Telefon durchzuführen. Als Ergebnis erzeugt ADSI eine hohe Verkehrsbelastung für das Telefonnetzwerk, da jede Anwendereingabe zu dem Dienst zur Verarbeitung zurückgesendet wird. Dies macht ADSI ungeeignet für drahtlose Netzwerke, wo die Bandbreite das wichtigste ist und eine "Lufteffizienz" eine der am meisten gesuchten Qualitäten ist. Der Mangel an Verarbeitungsfähigkeit im Telefon und die hohen Bandbreitenanforderungen von ADSI haben verhindert, dass es durch die Industrie zum Implementieren von intelligenten drahtlosen Vorrichtungen berücksichtigt wird.

[0020] Bis heute haben intelligente Kommunikationsvorrichtungen ein Computermodul mit einem drahtlosen Kommunikationsmodul kombiniert. Jedoch wird zum Erhöhen einer weit verbreiteten Akzeptanz eine Zweiwegedaten-Kommunikationsvorrichtung mit einer Verarbeitungsfähigkeit und der Fähigkeit zum Laufen lassen einer weiten Vielfalt von sich unterscheidenden Anwenderanwendungen nötig. Zusätzlich sollte eine Vorrichtung bezüglich der Größe, der Kosten und des Gewichts mit einem Zellulartelefon vergleichbar sein. Gemäß der vorliegenden Erfindung wird ein Verfahren zum Verfügung gestellt, das in einem Netzwerkknoten implementiert ist, der mit einem drahtlosen Telekommunikationsnetzwerk und mit einem drahtgebundenen Computernetzwerk gekoppelt ist, um zu ermöglichen, dass jeweilige mobile zu Zweiwegedaten fähige drahtlose Kommunikationsvorrichtungen in Taschengröße eine interaktive Zweiwegekommunikation mit einer Anwendung haben, wobei das Verfahren folgendes aufweist:

Empfangen einer Anforderung von der mobilen Vorrichtung über das drahtlose Kommunikationsnetzwerk, um eine Anwendung auf einem Server auf dem drahtgebundenen Computernetzwerk auszuführen;

Senden der Anforderung über das drahtgebundene Computernetzwerk zu dem Server, um auf die Anwendung zuzugreifen; gekennzeichnet durch

Empfangen einer Antwort über das drahtgebundene Computernetzwerk gemäß der Ausführung der Anwendung, auf die zugegriffen ist, wobei die Antwort aus Information besteht, die durch die mobile Vorrichtung zu interpretieren ist, und wodurch eine interaktive Anwenderschnittstelle daraus erzeugt wird und wodurch eine weitere Anforderung daraus zu dem oder einem anderen Server gemäß der Anwendung erzeugt wird, die von dieser erzeugten Anwenderschnittstelle durchgeführt ist;

und

Senden der Antwort zu der mobilen Vorrichtung über das drahtlose Telekommunikationsnetzwerk.

[0021] Gemäß einem weiteren Aspekt der vorliegenden Erfindung wird ein Netzwerkknoten zur Verfügung gestellt, der folgendes aufweist:

ein erste Kommunikationsschnittstelle zum Kommunizieren mit jeweiligen mobilen zu Zweiwegedaten fähigen drahtlosen Kommunikationsvorrichtungen in Taschengröße über ein drahtloses Telekommunikationsnetzwerk;

eine zweite Kommunikationsschnittstelle, die zu einer Kommunikation mit einem Server über ein drahtgebundenes Computernetzwerk fähig ist; und

einem Prozessor, der mit der ersten Kommunikationsschnittstelle und der zweiten Kommunikationsschnittstelle gekoppelt ist, um einen Prozess auszuführen, wobei der Prozess folgendes enthält:

Empfangen einer Anforderung von der mobilen Vorrichtung über das drahtlose Telekommunikationsnetzwerk, um eine Anwendung auf dem Server auf dem drahtgebundenen Computernetzwerk auszuführen;

Senden einer Anforderung über das drahtgebundene Computernetzwerk zu dem Server, um auf die Anwendung zuzugreifen; gekennzeichnet durch

Empfangen einer Antwort über das drahtgebundene Computernetzwerk gemäß der Ausführung der Anwendung, auf die zugegriffen ist, wobei die Antwort aus Information besteht, die durch die mobile Vorrichtung zu interpretieren ist, und wodurch eine interaktive Anwenderschnittstelle daraus erzeugt wird und wodurch eine weitere Anforderung daraus zu dem oder einem anderen Server gemäß der Anwendung erzeugt wird, die von dieser erzeugten Anwenderschnittstelle durchgeführt wird; und

Senden der Antwort zu der mobilen Vorrichtung über das drahtlose Telekommunikationsnetzwerk.

[0022] Zusätzliche vorteilhafte Merkmale sind den untergeordneten Ansprüchen definiert.

[0023] Ein Zweiwegedaten-Kommunikationsvorrichtung dieser Erfindung, wie beispielsweise ein Zellulartelefon oder ein Zweiwege-Pager bzw. eine Zweiwege-Funkrufeinheit kann ein Client-Modul mit leichtgewichtigen Ressourcen enthalten, das mit einem Servercomputer über ein Zweiwegedaten-Kommunikationsnetzwerk kommuniziert. Die Prinzipien dieser Erfindung können mit einer weiten Vielfalt von Zweiwegedaten-Kommunikationsnetzwerken verwendet werden. Beispielsweise enthalten Zweiwegedaten-Kommunikationsnetzwerke für Zellulartelefone, die verwendet werden können, ein zellulares digitales Paketdatennetzwerk sowie TDMA-,

CDMA- und GSM-Schaltkreis geschaltete Datennetzwerke; und das analoge AMPS-Zellularnetzwerk mit einem Modem. Gleichmaßen enthalten für Zweiwege-Funkrufeinheiten Zweiwegedaten-Kommunikationsnetzwerke PACT, den neuen von AT&T eingeführten Zweiwege-Funkrufstandard oder andere Zweiwege-Funkrufnetzwerke einer Priorität mit einer Datentransportfähigkeit.

[0024] Unter Verwendung der Zweiwege-Kommunikationsvorrichtung, die das Client-Modul enthält, kann ein Anwender Information zu dem Servercomputer liefern, Information von dem Servercomputer wiedergewinnen, Daten zu einer Anwendung auf dem Servercomputer liefern, der die Daten verwendet und Information zu der Zweiwege-Kommunikationsvorrichtung liefert oder die Information zu einer anderen Stelle sendet. Die dem Anwender der Zweiwege-Kommunikationsvorrichtung zur Verfügung gestellte Funktionalität ist nur durch die Anwendungen beschränkt, die auf einem Servercomputer verfügbar sind, auf den durch den Anwender über das Zweiwegedaten-Kommunikationsnetzwerk zugegriffen werden kann.

[0025] Diese Erfindung lässt zum ersten Mal zu, dass Zweiwege-Kommunikationsvorrichtungen, wie beispielsweise Zellulartelefone und Zweiwege-Funkrufeinheiten, offene Anwendungsplattformen werden, was wiederum Softwareentwickler dazu befähigt, Anwendungen und Dienste mit Mehrwert zu irgendeiner Zweiwege-Kommunikationsvorrichtung zu liefern, die die Prinzipien dieser Erfindung verwendet. Dies ist eine radikale Verschiebung von der aktuellen Situation, in welcher Telefone und Zweiwege-Funkrufeinheiten abgeschlossene eigene Systeme sind. Folglich wird ein ausgeglichenes Spielfeld für den Markt erzeugt, um neue Anwendungen für Zweiwege-Kommunikationsvorrichtungen und für Zweiwege-Kommunikationsnetzwerke zu erfinden. Jede Einheit von Firmen bis zu Individuen kann neue Anwendungen für die installierte Basis von Zweiwege-Datenkommunikationsvorrichtungen verfügbar machen, die bei dieser Erfindung verwendet werden, ohne eine physikalische Modifikation oder ein Hinzufügen zu der Zweiwege-Kommunikationsvorrichtung. Jahre nach einem Kauf wird eine Zweiwege-Kommunikationsvorrichtung, die zur Verwendung bei dieser Erfindung geeignet ist, alle Anwendungen laufen lassen, die seit ihrem Kauf entwickelt wurden.

[0026] Weiterhin sind alle diese Anwendungen verfügbar, ohne dass der Endbenutzer irgendetwas hinzufügen oder irgendeine Modifikation an der Zweiwege-Kommunikationsvorrichtung durchführen muss. Ebenso sind die Anwendungen unabhängig von dem Zweiwege-Datenkommunikationsnetzwerk. Die Anwendungen hängen nicht von irgendeiner Eigenschaft des Zweiwege-Datenkommunikationsnetzwerks ab. Somit werden die Anwendungen durch eine Änderung bezüglich des Zweiwege-Datenkommunikationsnetzwerks nicht beeinflusst.

[0027] Ebenso sind die Anwendungen auf dem Servercomputer unabhängig von der Zweiwege-Datenkommunikationsvorrichtung, mit welcher der Servercomputer interagiert. Eine Anwendung auf dem Servercomputer kann mit irgendeiner Zweiwege-Datenkommunikationsvorrichtung kommunizieren, die ein geeignetes Client-Modul und ein Netzwerkschnittstellenmodul enthält, um Daten darüber zu senden und Daten von dem Zweiwege-Datenkommunikationsnetzwerk zu empfangen. Diese zwei Eigenschaften bedeuten, dass eine Investition in eine Entwicklung einer Anwendung von entweder Fortschritten in Bezug auf Zweiwege-Datenkommunikationsvorrichtungen oder Fortschritten in Bezug auf eine Zweiwege-Datenkommunikationsnetzwerkstechnologie isoliert ist.

[0028] Die Zweiwege-Datenkommunikationsvorrichtung dieser Erfindung kann ein Client-Modul zum Senden einer Nachricht einschließlich eines durch den Anwender ausgewählten Betriebsmittellokalisierers (Methode zum Auffinden von Ressourcen) über das Zweiwege-Datenkommunikationsnetzwerk zu einem Server auf einem Servercomputer auf dem Computernetzwerk verwenden. Beispielsweise kann das Computernetzwerk ein weiträumiges Firmennetzwerk, ein lokales Firmennetzwerk, das Internet oder irgendeine Kombination von Computernetzwerken sein.

[0029] Der Server verarbeitet die Nachricht, d.h. führt die durch den Betriebsmittellokalisierer adressierte Anwendung aus und sendet eine Antwort über das Zweiwege-Datenkommunikationsnetzwerk zu der Zweiwege-Datenkommunikationsvorrichtung, die die Antwort in einem Speicher speichert. Das Client-Modul interpretiert die Antwort und erzeugt eine Anwenderschnittstelle verwendende Information in der Antwort. Bei einem Ausführungsbeispiel enthält die Anwenderschnittstelle wenigstens eine Anwenderdateneingabeoption, die mit einem Betriebsmittellokalisierer verbunden ist. Bei einem weiteren Ausführungsbeispiel ist die Anwenderschnittstelle eine Anzeige.

[0030] Der mit der wenigstens einen Anwenderdateneingabeoption verbundene Betriebsmittellokalisierer kann irgendeines einer weiten Vielfalt von Objekten adressieren. Bei einem Ausführungsbeispiel adressiert der mit der wenigstens einen Anwenderdateneingabeoption verbundene Betriebsmittellokalisierer ein Objekt auf

dem Servercomputer, der die Antwort sendete. Bei einem weiteren Ausführungsbeispiel adressiert der Betriebsmittellokalisierer ein Objekt auf einem anderen Servercomputer, der mit dem Zweiwege-Datenkommunikationsnetzwerk gekoppelt ist. Bei einem weiteren Ausführungsbeispiel adressiert der Betriebsmittellokalisierer ein Objekt, das in der Zweiwege-Kommunikationsvorrichtung gespeichert ist.

[0031] Wenn der Anwender die wenigstens eine Anwenderdateneingabeoption auswählt, interpretiert das Client-Modul die Auswahl und hängt dann, wenn es erforderlich ist, irgendwelche Eingabedaten an den Betriebsmittellokalisierer an, der mit der wenigstens einen Anwenderdateneingabeoption verbunden ist. Das Client-Modul sendet eine Nachricht einschließlich des Betriebsmittellokalisierers mit irgendwelchen angehängten Eingabedaten zum Servercomputer. Alternativ dazu kann der Betriebsmittellokalisierer mit irgendwelchen angehängten Daten zu einem anderen Servercomputer adressiert werden oder kann ein Objekt adressieren, das in der Zweiwege-Kommunikationsvorrichtung gespeichert ist. Wenn der Betriebsmittellokalisierer ein Objekt auf einem Servercomputer adressiert, liefert das Client-Modul die Nachricht zu dem Netzwerkschnittstellenmodul, das wiederum die Nachricht über das Zweiwege-Datenkommunikationsnetzwerk sendet.

[0032] Somit enthielt bei diesem Ausführungsbeispiel die zu der Zweiwege-Datenkommunikationsvorrichtung ursprünglich gesendete Nachricht all die Information, die für das Client-Modul nötig ist, um die Anwenderschnittstelle zu erzeugen, um die Anwenderauswahl und irgendwelche Daten zu verbinden, die mit einem bestimmten Betriebsmittellokalisierer eingegeben sind, und um den geeigneten Betriebsmittellokalisierer in einer nachfolgenden Nachricht zu senden. Das Client-Modul enthält einen Interpretierer, der die Information in der Nachricht verarbeitet. Da die Nachricht all die Information enthielt, die durch das Client-Modul benötigt wird, hielt der Servercomputer, der die Nachricht sendete, keine Zustandsinformation in Bezug auf die Nachricht zurück. Folglich ist der Servercomputer als zustandsloser Servercomputer definiert.

[0033] Ein wichtiger Aspekt dieser Erfindung besteht darin, dass die Nachricht all die Information enthält, die für das Client-Modul nötig ist, die Anwenderschnittstelle zu erzeugen, und eine bestimmte Anwenderschnittstelle kann von anderen Anwenderschnittstellen unabhängig sein. Ungleichsystemen nach dem Stand der Technik, die dem Anwender ein vorbestimmtes Menü vorgaben, aus welchem Elemente auszuwählen sind, oder die den Anwender auf ein e-Mail-artiges Format beschränkten, sind gemäß den Prinzipien dieser Erfindung die Anwenderschnittstellen und mögliche Interaktionen, die für den Anwender verfügbar sind, nur durch Anwendungen bestimmt, die Entwickler verfügbar machen. Die möglichen Interaktionen und Anwenderschnittstellen für eine Anwendung können völlig unterschiedlich und unabhängig von den möglichen Interaktionen und Anwenderschnittstellen von einer anderen Anwendung sein. Somit werden ein Zellulartelefon und eine Zweiwege-Funkrufeinheit beide wirklich eine offene Plattform.

[0034] Diese Merkmale der Erfindung sind eine signifikante Abweichung von Systemen nach dem Stand der Technik. Typischerweise erforderte beim Stand der Technik eine Anwendung einer bestimmten Anwendung auf einer bestimmten Plattform, dass die Anwendung kompatibel mit dem Betriebssystem auf dieser Plattform ist. Weiterhin war es für den Anwender jedes Mal dann, wenn eine neue Version der Anwendung herausgegeben wurde, erforderlich, Schritte zum Aktualisieren der Anwendung auf der Plattform des Anwenders vorzunehmen. Weiterhin würde dann, wenn der Anwender der Plattform das Betriebssystem nicht modifizierte, wenn neue Versionen des Betriebssystems herausgegeben wurden, die Plattform zu irgendeinem Zeitpunkt nicht mehr eine neue Version einer Anwendung verarbeiten können, die eine aktuelle Version des Betriebssystems erfordert.

[0035] Diese Erfindung eliminiert diese Probleme. Wie es oben erklärt ist, fungiert das Client-Modul in der Zweiwege-Datenkommunikationsvorrichtung als Interpretierer. Die Anwendung auf dem Servercomputer liefert alle Information, die für den Interpretierer nötig ist, um eine Anwenderschnittstelle an der Zweiwege-Datenkommunikationsvorrichtung zu erzeugen, und in Reaktion auf Anwenderauswahlen oder eine Dateneingabe unter Verwendung der Anwenderschnittstelle, um eine Nachricht zu einem geeigneten Server zu führen, d.h. entweder dem Server, der die ursprüngliche Nachricht sendete, oder einem anderen Server.

[0036] Somit interpretiert das Client-Modul nur diese Information und interagiert geeignet mit der Hardware der Zweiwege-Datenkommunikationsvorrichtung. Folglich erfordert ein Aktualisieren einer Anwendung nur Änderungen am Servercomputer und keine Änderungen bei einer jeweiligen Zweiwege-Datenkommunikationsvorrichtung, die mit diesem Servercomputer kommuniziert. Diese Erfindung eliminiert die normale Anforderung für eine Verteilung von Anwendungssoftware und Anwendungssoftwareaktualisierungen zu dem Endanwender der Zweiwege-Datenkommunikationsvorrichtung.

[0037] Bei einem Ausführungsbeispiel enthält ein Zweiwege-Datenkommunikationssystem für eine Kommu-

nikation zwischen einem Servercomputer und einer Zweiwege-Datenkommunikationsvorrichtung, die aus einer Gruppe ausgewählt ist, die aus einem Zellulartelefon und einer Zweiwege-Funkrufeinheit besteht, ein Zweiwege-Datenkommunikationsnetzwerk, einen Servercomputer, der mit dem Zweiwege-Datenkommunikationsnetzwerk gekoppelt ist, und eine Zweiwege-Datenkommunikationsvorrichtung, die mit dem Zweiwege-Datenkommunikationsnetzwerk gekoppelt ist. Der Servercomputer enthält ein Zweiwege-Datenkommunikationsschnittstellenmodul, das mit dem Zweiwege-Datenkommunikationsnetzwerk gekoppelt ist, und einen Server, der mit dem Zweiwege-Datenkommunikationsschnittstellenmodul gekoppelt ist. Der Server empfängt eine Nachricht einschließlich eines Betriebsmittellokalisierers von dem Zweiwege-Datenkommunikationsnetzwerk. Der Betriebsmittellokalisierer enthält eine Adresse des Servercomputers und einer Anwendung auf diesem Servercomputer. Der Server verarbeitet die Nachricht unter Verwendung des Betriebsmittellokalisierers. Bei diesem Ausführungsbeispiel sendet der Server eine Antwort auf die Nachricht über das Zweiwege-Datenkommunikationsnetzwerk.

[0038] Die Zweiwege-Datenkommunikationsvorrichtung, die aus der Gruppe ausgewählt ist, die aus einem Zellulartelefon und einer Zweiwege-Funkrufeinheit besteht, enthält ein Netzwerkschnittstellenmodul, das mit dem Zweiwege-Datenkommunikationsnetzwerk gekoppelt ist, und ein Client-Modul, das mit dem Netzwerkschnittstellenmodul gekoppelt ist. Das Client-Modul sendete die Nachricht einschließlich des Betriebsmittellokalisierers zu dem Server über das Zweiwege-Datenkommunikationsnetzwerk. Das Client-Modul verarbeitet auch die Antwort auf die Nachricht vom Server. Die Antwort enthält Information für eine Anwenderinteraktion über das Zweiwege-Datenkommunikationsnetzwerk.

[0039] Das Client-Modul, das mit dieser Erfindung verwendet werden kann, ist leichtgewichtig und erfordert somit nur ein leichtgewichtiges Betriebsmittel in einer Zweiwege-Datenkommunikationsvorrichtung. Folglich kann das Client-Modul existierende Betriebsmittel in einer solchen Vorrichtung verwenden und fügt daher keine Kosten zu der Zweiwege-Datenkommunikationsvorrichtung hinzu.

[0040] Bei einem Ausführungsbeispiel enthält der Interpretierer innerhalb des Client-Moduls eine Vielzahl von Managern einschließlich eines Anwenderschnittstellenmanagers, der mit einer Anzeige der Zweiwege-Datenkommunikationsvorrichtung gekoppelt ist, wobei der Anwenderschnittstellenmanager Interaktionen mit der Anzeige handhabt. Der Anwenderschnittstellenmanager ist auch mit einem Tastenfeld der Zweiwege-Datenkommunikationsvorrichtung gekoppelt und handhabt Interaktionen mit dem Tastenfeld. Hierin kann ein Tastenfeld ein Telefon-Tastenfeld, die auf einer Zweiwege-Funkrufeinheit gefundenen Tasten oder eine andere Dateneingabeschnittstelle einer Zweiwege-Datenkommunikationsvorrichtung sein.

[0041] Bei einem Ausführungsbeispiel enthält die durch den Servercomputer erzeugte Antwort eine Vielzahl von Betriebsmittellokalisierern und wenigstens einer der Vielzahl von Betriebsmittellokalisierern enthält eine Adresse zu einem anderen Server, der mit dem Kommunikationsnetzwerk gekoppelt ist.

[0042] Beispiele der vorliegenden Erfindung werden nun unter Bezugnahme auf die beigefügten Zeichnungen beschrieben werden, wobei:

[0043] [Fig. 1](#) ein Ausführungsbeispiel des drahtlosen Telekommunikations-(oder Luftnetz)-Netzwerks dieser Erfindung darstellt, das die Zweiwege-Datenkommunikationsvorrichtungen dieser Erfindung enthält.

[0044] [Fig. 2A](#) bis [Fig. 2H](#) Darstellungen einer Reihe von Bildschirmanzeigen einer Zweiwege-Datenkommunikationsvorrichtung dieser Erfindung sind, die eine Anwendung der Prinzipien dieser Erfindung darstellen.

[0045] [Fig. 3A](#) bis [Fig. 3F](#) Darstellungen einer Reihe von Bildschirmanzeigen einer Zweiwege-Datenkommunikationsvorrichtung dieser Erfindung sind, die eine zweite Anwendung der Prinzipien dieser Erfindung darstellen.

[0046] [Fig. 4A](#) bis [Fig. 4I](#) Darstellungen einer Reihe von Bildschirmanzeigen einer Zweiwege-Datenkommunikationsvorrichtung dieser Erfindung sind, die eine weitere Anwendung der Prinzipien dieser Erfindung darstellen.

[0047] [Fig. 5](#) ein weiteres Ausführungsbeispiel des Luftnetz-Netzwerks dieser Erfindung darstellt, das die Zweiwege-Datenkommunikationsvorrichtungen dieser Erfindung und eine Luftnetz-Netzwerkumsetzungseinheit enthält.

[0048] [Fig. 6](#) ein Blockdiagramm einer mobilen drahtlosen Kommunikationsvorrichtung ist, die das Client-

und Hilfsmodul enthält, die bei dieser Erfindung verwendet werden können.

[0049] [Fig. 7](#) ein detaillierteres Diagramm der mobilen drahtlosen Kommunikationsvorrichtung und eines Servercomputers innerhalb der Luftnetz-Netzwerkarchitektur dieser Erfindung ist.

[0050] [Fig. 8A](#) bis [Fig. 8D](#) ein Verarbeitungsablaufdiagramm sind, das den durch den Client in der mobilen drahtlosen Kommunikationsvorrichtung und den Server auf dem Servercomputer der [Fig. 7](#) durchgeführten Prozess zeigt.

[0051] [Fig. 9](#) ein Diagramm einer mobilen drahtlosen Kommunikationsvorrichtung dieser Erfindung ist, das ein neues Vorhersagetexteingabesystem enthält.

[0052] [Fig. 10A](#) bis [Fig. 10T](#) ein Ausführungsbeispiel einer Buchstabenhäufigkeits-Tabelle sind.

[0053] [Fig. 11](#) ein Verarbeitungsablaufdiagramm für ein Ausführungsbeispiel eines Dateneingabeprozesses ist, das den neuen Vorhersagedateneingabeprozess der [Fig. 9](#) enthält.

[0054] [Fig. 12](#) ein detailliertes Diagramm der mobilen drahtlosen Kommunikationsvorrichtung und der Luftnetz-Netzwerkumsetzungseinheit innerhalb der Luftnetz-Netzwerkarchitektur eines weiteren Ausführungsbeispiels dieser Erfindung ist.

[0055] [Fig. 13](#) ein Verarbeitungsablaufdiagramm ist, das die verschiedenen Prozesse zeigt, die durch die Luftnetz-Netzwerkumsetzungseinheit der [Fig. 12](#) durchgeführt werden.

[0056] [Fig. 14](#) ein Diagramm ist, das die verschiedenen Modulmanager darstellt, die bei einem Ausführungsbeispiel eines Client-Moduls enthalten sind, das bei dieser Erfindung verwendet werden kann.

[0057] Hierin sind Objekte mit denselben Bezugszeichen dasselbe Objekt. Ebenso zeigt die erste Zahl eines Bezugszeichens die Figur an, in welcher das Objekt zuerst erschien.

DETAILLIERTE BESCHREIBUNG

[0058] Bei dieser Erfindung verbindet ein neues Luftnetz-Netzwerk **150**, d.h. ein Zweiwege-Datenkommunikationsnetzwerk, eine oder beide von Zweiwege-Datenkommunikationsvorrichtungen **100** und **101** beispielsweise mit einer weiten Vielfalt von Computernetzwerken **120**, **130** und **140**. Wie es nachfolgend vollständiger erklärt wird, kann jede Zweiwege-Datenkommunikationsvorrichtung **100** und **101** konfiguriert sein, um Daten zu irgendeiner erwünschten Kombination von Computern auf Computernetzwerken **120**, **130** und **140** zu senden und Daten von diesen zu empfangen. Das Luftnetz-Netzwerk **150** ist der Zweiwege-Datenkommunikationspfad von der Zweiwege-Datenkommunikationsvorrichtung zu dem bestimmten Computer, auf den durch den Anwender dieser Zweiwege-Datenkommunikationsvorrichtung zugegriffen wird.

[0059] Jede drahtlose Kommunikationsvorrichtung **100**, die bei dieser Erfindung verwendet wird, kann über das Luftnetz-Netzwerk **150** mit irgendeinem Servercomputer **121**, **131** und **141** auf dem Luftnetz-Netzwerk **150** kommunizieren, das wenigstens eine Anwendung enthält, die mit den Prozessen kommuniziert und interagiert, die innerhalb der Vorrichtung **100** enthalten sind. Somit kann die Vorrichtung **100** auf Information auf dem Computernetzwerk zugreifen und Information zum Computernetzwerk liefern. Gleichermaßen kann eine Zweiwege-Funkrufeinheit **101**, die bei dieser Erfindung verwendet wird, über das Luftnetz-Netzwerk **150** mit irgendeinem von Servercomputern **121**, **131** und **141** kommunizieren, der wenigstens eine Anwendung enthält, die mit den Prozessen kommuniziert und interagiert, die innerhalb der Vorrichtung **101** enthalten sind.

[0060] Wie es nachfolgend vollständiger erklärt wird, kann auf eine Anwendung auf einem Servercomputer durch irgendeine Zweiwege-Datenkommunikationsvorrichtung zugegriffen werden, die mit diesem Servercomputer kommunizieren kann. Die Anwendung ist unabhängig von dem bestimmten Typ einer Zweiwege-Datenkommunikationsvorrichtung, die zum Zugreifen auf die Anwendung verwendet wird, und unabhängig von dem bestimmten verwendeten Zweiwege-Datenkommunikationsnetzwerk. Dies bedeutet, dass ein Anwender auf eine Anwendung von irgendwo zugreifen kann, solange der Anwender eine Zweiwege-Datenkommunikationsvorrichtung hat, die mit dem Servercomputer kommunizieren kann.

[0061] Bei einem Ausführungsbeispiel ist ein Prozess auf der drahtlosen Kommunikationsvorrichtung **100** als Client-Prozess konfiguriert und sind die Anwendungen auf den Servercomputern **121**, **131** und **141** auf dem

Luftnetz-Netzwerk **150**, die mit dem Client-Prozess kommunizieren, Serverprozesse. Diese Architektur lässt zu, dass einiges der Verarbeitungsbelastung von dem Zellulartelefon **100** über das Luftnetz-Netzwerk **150** zu einem Servermodul auf irgendeinem Computer auf dem Luftnetz-Netzwerk **150** wegbewegt wird.

[0062] Insbesondere kommuniziert eine drahtlose Kommunikationsvorrichtung **100**, wie z.B. ein Zellulartelefon, mit einem telefonartigen Tastenfeld über ein datenfähiges Zellulartelefonnetzwerk **110**, wie z.B. ein zellulares digitales Paketdaten-Telefonnetzwerk, mit einer Anwendung auf einem Servercomputer auf einem Computernetzwerk, das eine Schnittstelle zu dem datenfähigen Zellulartelefonnetzwerk **110** hat. Beispielsweise kann das Computernetzwerk ein weiträumiges Firmennetzwerk **120**, ein lokales Firmennetzwerk **130** oder vielleicht das Internet **140** sein.

[0063] Gleichermaßen kommuniziert eine Zweiwege-Funkrufeinheit **101** über ein Zweiwege-Funkrufeinheitsnetzwerk **111** mit einer Anwendung auf einem Servercomputer an einem Computernetzwerk, das eine Schnittstelle zu dem Zweiwege-Funkrufeinheitsnetzwerk **111** hat. Wiederum kann beispielsweise das Computernetzwerk ein weiträumiges Firmennetzwerk **120**, ein lokales Firmennetzwerk **130** oder vielleicht das Internet **140** sein.

[0064] In beiden Zweiwege-Datenkommunikationsvorrichtungen **100** und **101** wird der Client-Prozess als Client-Modul in der Vorrichtung gespeichert und wird die Ausführung des Client-Moduls auf einer Mikrosteuerung in der Vorrichtung manchmal Client-Prozess genannt. Der Client-Prozess führt wichtige Verarbeitungsfunktionen lokal durch. Dies lässt zu, dass die Kommunikation zwischen dem Client-Prozess, der hierin nachfolgend manchmal einfach Client genannt wird, und dem Serverprozess, der hierin nachfolgend manchmal Server genannt wird, minimiert wird, und die Serverrechenanforderungen langsam anwachsen, wenn die Anzahl von Clients, wie z.B. Anwendern, anwächst.

[0065] Das Client-Modul ist klein, wie z.B. unter 64 KByte, und erfordert nur eine geringe Verarbeitungsleistung, die kongruent zu den Speicherchips und eingebauten Mikrosteuerungen in Zweiwege-Datenkommunikationsvorrichtungen, wie beispielsweise einem Zellulartelefon **100** und einer Zweiwege-Funkrufeinheit **101**, ist. Somit werden ungleich den Versuchen nach dem Stand der Technik bei einem intelligenten Telefon die Kosten, die Größe und die Batteriebensdauer von entweder Zellulartelefonen oder Zweiwege-Funkrufeinheiten, die zur Verwendung bei dieser Erfindung geeignet sind, nicht nachteilig beeinflusst.

[0066] Während Client/Server-Architekturen in Computernetzwerken extensiv verwendet worden sind, ergibt eine Client/Server-Architektur, die unter Verwendung von Zweiwege-Kommunikationsdatenvorrichtungen, wie beispielsweise einem Zellulartelefon **100** oder einer Zweiwege-Funkrufeinheit **101**, implementiert ist, neue und unerwartete Ergebnisse. Diese Erfindung lässt zum ersten Mal zu, dass eine weite Vielfalt von Zweiwege-Datenkommunikationsvorrichtungen, die Zellulartelefone und Zweiwege-Funkrufeinheiten enthalten, aber nicht darauf beschränkt sind, offene Anwendungsplattformen werden, welche infolge Softwareentwickler dazu befähigen, Anwendungen und Dienste mit Mehrwert zu irgendeiner Zweiwege-Datenkommunikationsvorrichtung zu liefern, die zur Verwendung bei dieser Erfindung geeignet ist.

[0067] Dies ist eine radikale Verschiebung von der aktuellen Situation, in welcher Zellulartelefone und Zweiwege-Funkrufeinheiten geschlossene eigene Systeme sind. Folglich wird ein gleichmäßiges Spielfeld für den Markt erzeugt, um neue Anwendungen für Zellulartelefone und datenfähige Zellularnetzwerke zu erfinden, und für Zweiwege-Funkrufeinheiten und Zweiwege-Funkrufeinheitsnetzwerke.

[0068] Jede Einheit von Firmen bis zu Individuen kann neue Anwendungen für die installierte Basis von für Daten bereite Zellulartelefone und Zweiwege-Funkrufeinheiten verfügbar machen, die bei dieser Erfindung verwendet werden, ohne eine physikalische Modifikation oder ein Hinzufügen zu den Vorrichtungen. Jahre nach einem Kauf kann ein Zweiwege-Datenkommunikationsvorrichtung, die zur Verwendung bei dieser Erfindung geeignet ist, alle Anwendungen laufen lassen, die seit ihrem Kauf entwickelt wurden. Weiterhin sind alle diese Anwendungen verfügbar, ohne dass der Anwender irgendetwas zu der Zweiwege-Datenkommunikationsvorrichtung hinzufügen oder irgendeine Modifikation an dieser durchführen muss. Diese Eigenschaften der Erfindung sind eine signifikante Abweichung von Systemen nach dem Stand der Technik. Typischerweise erforderte beim Stand der Technik eine Verwendung einer bestimmten Anwendung auf einer bestimmten Plattform, dass die Anwendung mit dem Betriebssystem auf dieser Plattform kompatibel ist. Weiterhin war es für den Anwender jedes Mal dann, wenn eine neue Version der Anwendung herausgegeben wurde, erforderlich, Schritte zum Aktualisieren der Anwendung auf der Plattform des Anwenders vorzunehmen. Weiterhin würde dann, wenn der Anwender der Plattform das Betriebssystem nicht modifizierte, wenn neue Versionen des Betriebssystems herausgegeben wurden, zu irgendeinem Zeitpunkt die Plattform nicht mehr eine neue Version

einer Anwendung verarbeiten können, die eine aktuelle Version des Betriebssystems erforderte.

[0069] Ebenso haben kleine Vorrichtungen, wie beispielsweise Zellulartelefone oder Funkrufeinheiten, normalerweise keine Kartenschlitze, Disketten- oder Festplattenlaufwerke oder irgendeine andere Einrichtung, die auf Computern gemeinsam gefunden werden, um Anwendungen hinzuzufügen oder zu aktualisieren. Diese Beschränkung hat Versuche nach dem Stand der Technik bei intelligenten Kommunikationsvorrichtungen dazu geführt, geschlossene Systeme mit fester Funktionalität zu entwickeln. Solche Vorrichtungen können sich weder an die sich schnell ändernden Anforderungen des Marktes anpassen noch an diese angepasst werden und haben somit keinen Erfolg auf dem Markt gehabt.

[0070] Diese Erfindung eliminiert diese Probleme. Der Client-Prozess in der Zweiwege-Datenkommunikationsvorrichtung bearbeitet einen Interpretierer bzw. funktioniert als Interpretierer. Die Anwendung auf dem Servercomputer liefert die gesamte Information, die für den Interpretierer nötig ist, eine Anwenderschnittstelle an der Zweiwege-Datenkommunikationsvorrichtung zu erzeugen, und in Reaktion auf Anwenderauswahlen oder eine Dateneingabe unter Verwendung der Schnittstelle, um Nachrichten zu einem geeigneten Server zu führen, d.h. entweder dem Server, der die ursprüngliche Information sendete, oder einem anderen Server.

[0071] Somit interpretiert der Client-Prozess nur diese Information und interagiert geeignet mit der Hardware der Zweiwege-Datenkommunikationsvorrichtung. Folglich erfordert eine Aktualisierung einer Anwendung nur Änderungen an dem Servercomputer und keine Änderungen in einer jeweiligen Zweiwege-Datenkommunikationsvorrichtung, die mit diesem Servercomputer kommuniziert. Diese Erfindung eliminiert die normale Anforderung für eine Verteilung von Anwendungssoftware und Anwendersoftwareaktualisierungen zu dem Endanwender der Zweiwege-Datenkommunikationsvorrichtung.

[0072] Wenn beispielsweise am Anfang die Zweiwege-Funkrufeinheit **101** eine Antwort auf eine Nachricht von einer Anwendung auf dem Servercomputer **121** an dem weiträumigen Firmennetzwerk **120** empfängt, erzeugt der Interpretierer in der Zweiwege-Funkrufeinheit **101** eine Anwenderschnittstelle auf einem Anzeigebildschirm **106** unter Verwendung von Information in der Nachricht. Wie es nachfolgend vollständiger beschrieben wird, können Optionen, die in der Anwenderschnittstelle präsentiert sind, zulassen, dass der Anwender auf Information zugreift oder Information zu irgendeinem, irgendeiner Kombination von oder zu allen der Netzwerke **120**, **130** und **140** liefert.

[0073] Spezifisch enthielt bei der Antwort auf die Nachricht von der Zweiwege-Funkrufeinheit **101** die Anwendung, auf die Anfangs zugegriffen ist, auf dem Servercomputer **121** Betriebsmittellokalisierer für Anwendungen auf jedem der Netzwerke **120**, **130**, **140**, und zwar typischerweise gemeinsame Gateway-Schnittstellenprogramme, auf die für den Anwender der Funkrufeinheit **101** zugegriffen werden kann, sowie Information, die zum Erzeugen der Anwenderschnittstelle erforderlich ist. Folglich greift der Interpretierer dann, wenn der Anwender eine bestimmte Auswahl durchführt oder Daten eingibt, auf den geeigneten Betriebsmittellokalisierer zu und hängt irgendwelche nötigen Daten an den Betriebsmittellokalisierer an. Der Client sendet eine Nachricht einschließlich des Betriebsmittellokalisierers zu dem geeigneten Server.

[0074] Wie es durch dieses Beispiel gezeigt ist, senden die Anwendungen auf den Netzwerken **120**, **130**, **140** die gesamte Information zu der Zweiwege-Datenkommunikationsvorrichtung, die zum Erzeugen einer Anwenderschnittstelle nötig ist, und zum Verarbeiten einer gesamten Anwendereingabe. Folglich muss nur eine Anwendung zum Aktualisieren der Information geändert werden, die zu der Zweiwege-Datenkommunikationsvorrichtung geliefert ist.

[0075] Zusätzlich behält der Computerserver deshalb, weil die gesamte Information, die durch den Client benötigt wird, um eine Anwenderschnittstelle zu erzeugen, und die gesamte Information, die für den Client-Prozess nötig ist, um auf irgendwelche eingegebenen Daten zu antworten, in der Nachricht enthalten sind, nicht irgendeine Zustandsinformation in Bezug auf die zu dem Client-Prozess gesendete Information zurück. Folglich ist der Computerserver zustandslos.

[0076] Beide Zweiwege-Datenkommunikationsvorrichtungen **100** und **101**, die das Luftnetz-Netzwerk **150** verwenden, enthalten eine Datenkommunikationsfähigkeit, einen Anzeigebildschirm, vorzugsweise einen Mehrfachzeilen-Anzeigebildschirm und eine Speicherfähigkeit für die Prozesse dieser Erfindung in einem Speicher im Gerät und für die Nachricht, die verarbeitet wird. Nahezu jedes datenfähige Zellulartelefon, wie z.B. ein Telefon, das ein zellulares digitales Paketdatenetzwerk verwendet, enthält eine übermäßige Speicherkapazität im Gerät und einen Mehrfachzeilen-Anzeigebildschirm. Diese Hardwarebetriebsmittel sind oft verfügbar, aber in einem datenfähigen Zellulartelefon aufgrund der Unteilbarkeit der Speicherchippackungen nicht ge-

nutzt. Die Eignung für eine Verwendung bei den Prozessen dieser Erfindung von solchen Zellulartelefonen hat daher eine sehr geringe Auswirkung auf die Kosten, die Größe und den Energieverbrauch des Zellulartelefons. Gleichmaßen hat die Eignung zur Verwendung bei den Prozessen dieser Erfindung von Zweige-Funkrufeinheiten, die eine Mikrosteuerung und einen Speicher enthalten, sehr wenig Einfluss auf die Kosten, die Größe und den Energieverbrauch dieser Vorrichtungen.

[0077] Somit verwendet dieses Ausführungsbeispiel der Erfindung ungleich Ansätzen nach dem Stand der Technik, die versuchten, ein Computermodule und ein drahtloses Kommunikationsmodule in einer einzigen Packung zu kombinieren, vorzugsweise die Speicher- und Verarbeitungsleistung, die aktuell im Zellulartelefon **100**, in der Zweige-Funkrufeinheit **101** oder einer anderen Zweige-Datenkommunikationsvorrichtung existiert. Dieser Ansatz begrenzt die Kosten der resultierenden Vorrichtung und überwindet viele der Probleme der Vorrichtungen nach dem Stand der Technik, z.B. die Größe und das Gewicht der Zweige-Datenkommunikationsvorrichtung werden nicht geändert, und, wie es oben erklärt ist, wird ein Aktualisieren von Anwenderanwendungen vom Zellulartelefon **100** und von der Zweige-Funkrufeinheit **101** entfernt.

[0078] Insbesondere sind Zweige-Datenkommunikationsvorrichtungen, die zur Verwendung bei dieser Erfindung geeignet sind, Ungleichvorrichtungen, die durch vorherige industrielle Versuche beim Kombinieren von Rechenmodulen und einem drahtlosen Zellularmodule erzeugt sind, bezüglich der Größe und der Kosten wettbewerbsfähig mit Telefonen für nur Sprache und können zum ersten Mal die Kosten- und Größenanforderungen des Marktes für beispielsweise ein intelligentes Zellulartelefon erfüllen.

[0079] Die inkrementalen Kosten zum Unterstützen von interaktiven Anwendungen auf dem Zellulartelefon **100** und der Zweige-Funkrufeinheit **101** werden auf höchstens einen etwas größeren Bildschirm reduziert, der zum Anzeigen der Anwendung zum Anwender erforderlich ist. Dies ist ein Bruchteil der Kosten für ein Hinzufügen von beispielsweise einem vollständigen Computermodule zu einem Zellulartelefon.

[0080] Der inkrementelle Energieverbrauch, der zum Unterstützen dieser Erfindung erforderlich ist, ist auch sehr gering, da der erforderliche inkrementelle Speicher und Bildschirm im Vergleich mit dem Zellulartelefon selbst kleine Verbraucher von Energie sind. Von intelligenten Zweige-Datenkommunikationsvorrichtungen, die zur Verwendung bei dieser Erfindung aufgebaut sind, wird nicht erwartet, dass sie eine signifikant niedrigere Batteriebensdauer als beispielsweise standardmäßige Zellulartelefone oder Zweige-Funkrufeinheiten haben.

[0081] Die Konfiguration und die Prozesse des Client-Prozesses in Zweige-Datenkommunikationsvorrichtungen **100** und **101** sind ähnlich bzw. gleich, wenn die Unterschiede bezüglich der Vorrichtungen und des Zweige-Datenkommunikationsnetzwerks, über welche die Vorrichtungen kommunizieren, betrachtet werden. Folglich wird in der folgenden Beschreibung der Betrieb des datenbereiten Zellulartelefons **100** betrachtet. Dieselben oder ähnliche Operationen können auf der Zweige-Datenkommunikationsvorrichtung **101** durchgeführt werden. Der Hauptunterschied besteht darin, dass einige vorrichtungsabhängige Eigenschaften innerhalb des Client-Moduls geändert werden müssen, um die bestimmte Hardware, die in der Zweige-Kommunikationsvorrichtung verwendet wird, unterzubringen. Jedoch begrenzt die nachfolgend vollständiger beschriebene Client-Modularchitektur die Anzahl von Änderungen, die durchgeführt werden müssen.

[0082] Wie es oben angezeigt ist, sendet die drahtlose Kommunikationsvorrichtung **100** in Reaktion auf Anwenderaktionen eine Nachricht, typischerweise eine Datenanforderung, zu einem Servercomputer **121** auf dem Computernetzwerk **120** und empfängt eine Antwort auf die Nachricht. Alternativ dazu kann die Anwenderaktion in Anweisungen zum Servercomputer **121** auf dem Computernetzwerk **120** resultieren, um die Antwort auf die Nachricht zu einer anderen Stelle oder zu einem anderen Anwender zu senden. Ebenso kann die drahtlose Kommunikationsvorrichtung **100** eine Nachricht von irgendeinem der Computer empfangen, die mit dem Luftnetz-Netzwerk **150** gekoppelt sind.

[0083] Ein wichtiger Aspekt dieser Erfindung besteht darin, dass der Client-Modulinterpretierer in der drahtlosen Kommunikationsvorrichtung **100** eine Anwenderschnittstelle erzeugt, durch welche der Anwender Nachrichten von einer Vielfalt von Anwendungen sowohl initiieren als auch empfangen kann. Die Interaktionen finden in Echtzeit statt und sind nicht durch den Client-Modulinterpretierer beschränkt. Die Anwendungen der drahtlosen Kommunikationsvorrichtung **100** sind nur durch die Verfügbarkeit von Anwendungen auf Servercomputern beschränkt.

[0084] Die verfügbaren Anwendungen werden durch Anwendungsentwickler bestimmt. Vor einem Betrachten einer Implementierung der Erfindung in weiterem Detail werden mehrere illustrative Beispiele von Anwendun-

gen, die mit dieser Erfindung implementiert werden können, beschrieben. Diese Anwendungen sind nur illustrativ und sollen die Erfindung nicht auf die bestimmten Anwendungen und Merkmale, die beschrieben sind, beschränken.

[0085] Bei einer Anwendung konfiguriert der Anwender das Zellulartelefon **100** zum Zugreifen auf den Servercomputer **121** auf einem weiträumigen XYZ-Firmennetzwerk **120**. In Reaktion auf den Zugriff durch den Anwender sendet der Servercomputer **121** ein Kartenspiel bzw. einen Kartenposten zum Zellulartelefon **100** über das datenfähige Zellulartelefonnetzwerk **110**. Wie es nachfolgend vollständiger erklärt wird, enthält ein Kartenposten eine oder mehrere Karten und wird jede Karte durch das Client-Modul interpretiert, um einen Anwenderschnittstellenbildschirm zu erzeugen.

[0086] Bei dem in [Fig. 2A](#) dargestellten Ausführungsbeispiel enthält der zu dem Zellulartelefon **100** gesendete anfängliche Kartenposten eine einführende Anzeigekarte und eine Auswahlkarte. [Fig. 2A](#) ist ein Beispiel einer einführenden Bildschirmanzeige **200**, die auf dem Anzeigebildschirm **105** durch den Client-Prozess im Zellulartelefon **100** durch Interpretieren der Anzeigekarte erzeugt wird. Wie es hierin verwendet wird, ist ein Anzeigebildschirm eine physikalische Anzeigevorrichtung in einer Zweige-Kommunikationsvorrichtung. Eine Bildschirmanzeige ist das auf dem Anzeigebildschirm präsentierte Bild.

[0087] Bei diesem Ausführungsbeispiel ist der Anzeigebildschirm **105** eine Pixelanzeige, die Graphiken anzeigt. Bei einem weiteren Ausführungsbeispiel zeigt der Anzeigebildschirm **105** nur Text an, und somit würden die Graphiken nicht auf dem Anzeigebildschirm **105** erscheinen. Die Bildschirmanzeige **200** und nachfolgend vollständiger beschriebene andere Bildschirmanzeigen enthalten einen horizontalen Pfeil, d.h. einen Mehrfachkartenpostenindikator, um dem Anwender zu kommunizieren, dass der aktuelle Posten eine weitere Karte enthält. Das Enthaltensein von Bildschirmindikatoren, wie beispielsweise des Mehrfachkartenpostenindikators, um mit dem Anwender zu kommunizieren, ist optional. Die Funktionalität dieser Erfindung ist unabhängig von solchen Bildschirmindikatoren.

[0088] Wenn der Anwender eine vorbestimmte Taste drückt, oder eine Tastensequenz, interpretiert der Client-Prozess im Zellulartelefon **100** die nächste Karte im Kartenposten, d.h. die Auswahlkarte, und erzeugt in Folge ein Menü **201** ([Fig. 2B](#)) von Elementen, auf die durch den Anwender zugegriffen werden kann. Bei diesem Ausführungsbeispiel ist jedes der Menüelemente auf dem Servercomputer **121** für den Anwender verfügbar, der bei diesem Beispiel ein Repräsentant bzw. Vertreter der XYZ-Firma ist, der ABC-Entwicklungen besucht.

[0089] Wie es nachfolgend vollständiger erklärt wird, ist jedem der Menüelemente ein Betriebsmittellokalisierer zugehörig, der eine Adresse des bestimmten Objekts enthält, das mit diesem Menüelement verbunden ist, und zwar typischerweise eine Adresse zu einem gemeinsamen Gateway-Schnittstellenprogramm auf dem Servercomputer **121**. Im Allgemeinen enthält ein Betriebsmittellokalisierer eine Adresse und kann angehängte Daten enthalten. Die Adresse kann zu einem lokalen Objekt innerhalb der Zweige-Datenkommunikationsvorrichtung oder zu einem entfernten Objekt auf einen Servercomputer sein. Wie es Fachleuten auf dem Gebiet bekannt ist, ist die gemeinsame Gateway-Schnittstelle ein Internetstandard, der zum dynamischen Erzeugen von Information, wie z.B. Karten, verwendet wird. Angesichts dieser Offenbarung könnten andere Techniken zum Erzeugen von dynamischen Karten verwendet werden.

[0090] Anfangs ist das Hervorheben der ersten Zeile des Menüs **201** nicht vorhanden. Wenn eine Taste auf dem Tastenfeld des Zellulartelefons **100** gedrückt wird, wird das Menüelement entsprechend dieser Taste auf dem Bildschirm **105** hervorgehoben. Somit zeigt das Menü **201** das erste Element hervorgehoben, um anzuzeigen, dass die eine Taste durch den Anwender gedrückt wurde. Jedoch ist ein Hervorheben eines ausgewählten Elements ein Merkmal, das für dieses Beispiel spezifisch ist, und ist im Allgemeinen nicht zum Implementieren der Erfindung erforderlich. Andere Verfahren können verwendet werden, um die Auswahl eines Anwenders auf dem Anzeigebildschirm **105** anzuzeigen, wie beispielsweise ein Pfeil, der auf die Auswahl zeigt, wenn eine solche Anzeige erwünscht ist.

[0091] Nachdem die eine Taste gedrückt ist, drückt der Anwender eine vorbestimmte Taste, wie z.B. eine Eingabetaste, um die Auswahl zu verifizieren. Alternativ dazu ist bei einem weiteren Ausführungsbeispiel die Verifizierung der Auswahl nicht erforderlich. Bei beiden Ausführungsbeispielen wird der Betriebsmittellokalisierer für die Auswahl durch den Client-Prozess im Zellulartelefon **100** über das datenfähige Zellulartelefonnetzwerk **110** zum Servercomputer **121** gesendet. In Reaktion auf die Auswahl verarbeitet der Servercomputer **121** die Nachricht, die die Auswahl enthält, und sendet bei diesem Ausführungsbeispiel einen weiteren Kartenposten zum Zellulartelefon **100**.

[0092] Der Client-Prozess im Zellulartelefon **100** interpretiert die erste Karte im Posten, der vom Servercomputer **121** empfangen ist, welche eine Auswahlkarte ist, und erzeugt eine Bildschirmanzeige **202**, die ein zweites Menü enthält, wie es in [Fig. 2C](#) dargestellt ist, auf dem Anzeigebildschirm **105**. Anfangs sind keine der Elemente im zweiten Menü hervorgehoben.

[0093] Es ist zu beachten, dass die Bildschirmanzeige **202** einen Anfangsblock enthält, der die durch den Anwender durchgeführte Auswahl auf der Bildschirmanzeige **201** beschreibt, zusätzlich zu dem zweiten Menü von Auswahlen, das für den Anwender verfügbar ist. Ein Mehrfachanzeigenbildschirmkartenindikator **203**, wie z.B. bei diesem Ausführungsbeispiel ein Handsymbol mit einem Finger, der nach unten zeigt, zeigt, dass der zu der aktuellen Auswahlkarte gehörende Bildschirm zusätzliche Elemente enthält, die nicht auf dem Anzeigebildschirm **105** gezeigt sind. Hierin kann ein Bildschirm größer als die Anzahl von Zeilen sein, die auf dem Anzeigebildschirm **105** verfügbar sind, und somit muss der Anwender die Bildschirmanzeige rollen, um den vollständigen Bildschirm anzuschauen.

[0094] Somit drückt der Anwender zum Anschauen der zusätzlichen Elemente eine erste Bildschirmrolltaste, wie z.B. eine nächste Taste, auf dem Zellulartelefon **100**. Bei diesem Ausführungsbeispiel wird dann, wenn die erste Bildschirmrolltaste gedrückt wird, jede Zeile der Anzeige um eine Zeile nach oben gerollt. Die resultierende Anzeige hat ein Symbol mit einem Finger, der nach oben zeigt (nicht gezeigt), wenn das Menü nur zwei Bildschirmanzeigen erfordert. Wenn das Menü mehr als zwei Bildschirmanzeigen erfordert, würde die zweite Bildschirmanzeige des Menüs zwei Symbole bzw. Icons haben, und zwar eines mit einem Finger, der nach oben zeigt, und ein weiteres mit einem Finger, der nach unten zeigt. Um zwischen den verschiedenen Zeilen im zweiten Menü zu rollen, verwendet der Anwender die erste Bildschirmrolltaste und eine zweite Bildschirmrolltaste.

[0095] Wenn der Anwender die letzte Zeile einer Karte anzeigt, wie z.B. die letzte Zeile im zweiten Menü, und die erste Bildschirmrolltaste drückt, passiert nichts. Bei diesem Ausführungsbeispiel muss der Anwender eine Auswahl treffen, bevor die nächste Karte verfügbar ist.

[0096] Die Bildschirmanzeige **202** enthält auch Darstellungen von zwei weichen Tasten bzw. Softkeys, einer Heimtaste **204** und einer Info-Taste **205**. Bei diesem Beispiel sind diese weichen Tasten nur für die Karte definiert, die zum Erzeugen der Bildschirmanzeige **202** verwendet wird. Wenn der Anwender eine vorbestimmte Tastensequenz drückt, wird die Heimtaste hervorgehoben, um die Auswahl anzuzeigen. Bei diesem Ausführungsbeispiel wird der Anwender dann, wenn die Heimtaste ausgewählt wird, zur Bildschirmanzeige **200** zurückgebracht. Bei einem weiteren Ausführungsbeispiel könnte der Anwender beispielsweise zu einer Heimbildschirmanzeige zurückgebracht werden, die jedes Mal dann angezeigt wird, wenn der Anwender das Zellulartelefon **100** zur Verwendung auf dem Luftnetz-Netzwerk **150** aktiviert.

[0097] Die Heimtaste ist mit einem Zeiger verbunden, der bei einem Ausführungsbeispiel ein Betriebsmittel-lokalisierer ist, und die durch den Zeiger adressierte Karte wird durch den Client-Prozess angezeigt, wenn die Heimtaste durch den Anwender ausgewählt ist. Insbesondere dann, wenn der Zeiger zu einer Karte im aktuellen Posten ist, zeigt der Client-Prozess einfach diese Karte an. Wenn der Zeiger zu einer anderen als einer Karte im aktuellen Posten ist, liest der Client-Prozess im Zellulartelefon **100** den Posten aus, der die Karte bei der Stelle enthält, die durch den Zeiger identifiziert ist. Die Stelle könnte beispielsweise entweder ein Speicher im Zellulartelefon **100** oder ein Speicher im Computer **121** sein.

[0098] Gleichermaßen wird dann, wenn der Anwender eine weitere vorbestimmte Tastensequenz drückt, die Info-Taste hervorgehoben, um die Auswahl anzuzeigen. Bei diesem Ausführungsbeispiel wird dann, wenn die Info-Taste ausgewählt wird, ein Hilfsbildschirm für den Anwender angezeigt, der die möglichen Auswahlen beschreibt. Die bestimmten Inhalte des Hilfsbildschirms werden durch den Provider des Dienstes bzw. den Dienstversorger bzw. den Dienstlieferer bzw. Service Provider bestimmt. Insbesondere ist ein Zeiger mit der Info-Taste verbunden, und dann, wenn die Info-Taste durch den Anwender gedrückt wird, wird die bei der durch den Zeiger identifizierten Stelle gespeicherte Information ausgelesen und durch den Client-Prozess im Zellulartelefon interpretiert.

[0099] Kehrt man zurück zu dem Menü in [Fig. 2C](#), drückt der Anwender deshalb, weil der Anwender wünscht, den Zustand einer Reihenfolge zu bestimmen, die Taste zwei auf dem Tastenfeld des Zellulartelefons **100**. In Reaktion auf das Drücken der Taste wird die zweite Auswahl im Menü hervorgehoben, wie es in [Fig. 2C](#) gezeigt ist. In Reaktion auf eine Verifizierung des Drückens der Taste, z.B. drückt der Anwender eine vorbestimmte Tastensequenz, sendet das Zellulartelefon **100** eine Prüfungsöffnungsreihenfolgenanforderung zum Computer **121**, d.h. der Client-Prozess sendet eine Nachricht, die einen Betriebsmittellokalisierer enthält, der zu

dem Menüelement gehört, das durch Drücken der Taste zwei ausgewählt ist.

[0100] In Reaktion auf die Prüfungsöffnungsreihenfolgenanforderung sendet der Computer **121** noch einen weiteren Kartenposten zum Zellulartelefon **100**. Der Client-Prozess im Zellulartelefon **100** interpretiert diesen Posten, der eine Eingabekarte ist, und erzeugt infolge davon eine Kaufauftragsnummerneingabe-Bildschirmanzeige **206** ([Fig. 2D](#)) auf dem Anzeigebildschirm **105**. Es ist zu beachten, dass die Bildschirmanzeige **206** eine vorherige weiche Taste **207** und eine weiche Fax-Taste **208** hat. Wiederum hat jede dieser weichen Tasten einen zugehörigen Zeiger und die Information, die bei der Stelle gespeichert ist, die durch den Zeiger identifiziert ist, wird durch den Client-Prozess ausgelesen und interpretiert, wenn der Anwender die weiche Taste auswählt.

[0101] Bei diesem Beispiel wählt der Anwender keine weiche Taste aus, sondern gibt der Anwender vielmehr die Kaufauftragsnummer, wie es in [Fig. 2E](#) gezeigt ist, unter Verwendung des Tastenfelds des Zellulartelefons **100** ein. Der Anwender gibt nur die verschiedenen Nummern ein. Der Client-Prozess formatiert die Nummer und fügt die Striche ein, wie es in [Fig. 2E](#) gezeigt ist.

[0102] Nachdem der Kaufauftrag eingegeben ist, drückt der Anwender eine vorbestimmte Tastensequenz, um dem Client-Prozess anzuzeigen, dass eine Eingabe der Kaufauftragsnummer vollständig ist. Es ist zu beachten, dass der Anwender Daten eingibt und nicht einfach ein Menüelement auswählt. Der Anwender verwendet das Zellulartelefon **100** als ob das Zellulartelefon **100** ein mit dem Netzwerk **120** verbundener Computer wäre, aber, wie es nachfolgend vollständiger erklärt wird, das Zellulartelefon **100** ist gleich einem standardmäßigen digitalen datenfähigen Zellulartelefon, das über das datenfähige Zellulartelefonnetzwerk **110** kommuniziert. Insbesondere ist das Zellulartelefon **100** keine Kombination aus einem Computermodule und einem drahtlosen Kommunikationsmodule, wie bei Versuchen nach dem Stand der Technik zum Erzeugen eines intelligenten Telefons.

[0103] Zusätzlich gibt der Anwender Daten unter Verwendung von nur dem standardmäßigen Zellulartelefon-Tastenfeld ein. Somit eliminiert das Zellulartelefon **100** die Notwendigkeit für eine Computertastatur oder für einen ausgeklügelten Berührungsbildschirm, der eine Bewegung eines zeigenden Objekts erkennt. Dies ist wichtig zum Beibehalten der Größe, des Gewichts und der Energieanforderungen des Zellulartelefons **100** gleich denjenigen eines Zellulartelefons für nur Sprache. Bei einem Ausführungsbeispiel enthält das Zellulartelefon **100**, um eine Dateneingabe zu erleichtern, wie es vollständiger nachfolgend erklärt wird, einen Textvorhersageprozess, der die Anzahl von Tastenhüben reduziert, die zum Eingeben von Textdaten erforderlich sind. Bei diesem Ausführungsbeispiel wird der Textvorhersageprozess für jede Eingabekarte ein- oder ausgeschaltet.

[0104] In Reaktion auf eine Eingabe der Kaufauftragsnummer sendet der Client-Prozess eine Anforderung zum Servercomputer **121** für den bestimmten Kaufauftrag. Insbesondere hängt der Client-Prozess die eingegebenen Daten an einen Betriebsmittellokalisierer an und sendet eine Nachricht, die den Betriebsmittellokalisierer enthält, zum Servercomputer **121**. Der Servercomputer **121** liest in Reaktion auf die Nachricht den geeigneten Kaufauftrag aus und sendet den Kaufauftrag als Kartenposten zu dem Client-Prozess im Zellulartelefon **100** über das Luftnetz-Netzwerk **150**.

[0105] Der Client-Prozess interpretiert den Kartenposten und erzeugt eine Bildschirmanzeige **209** ([Fig. 2F](#)). Anfangs ist eine Fax-Taste **208** in der Bildschirmanzeige **209** nicht hervorgehoben.

[0106] Es ist zu beachten, dass die Bildschirmanzeige **209** einen Mehrfachanzeigebildschirmkartenindikator **103** enthält, um dem Anwender zu zeigen, dass der Kaufauftragsbildschirm mehr Information enthält, die auf einmal auf dem Anzeigebildschirm **105** angezeigt werden kann.

[0107] Nachdem der Anwender den Kaufauftrag durchschaut, drückt der Anwender die Tastensequenz für die Fax-Taste **208**, und in Reaktion darauf wird die Fax-Taste **208** hervorgehoben, wie es in [Fig. 2F](#) dargestellt ist.

[0108] In Reaktion auf eine Auswahl der Fax-Taste **208** liest der Client-Prozess den Kartenposten bei der Stelle, die durch den Zeiger identifiziert ist, der zu der Fax-Taste **208** gehört, aus. Wenn die Stelle auf dem Servercomputer **121** ist, sendet der Client-Prozess eine Nachricht, die einen Betriebsmittellokalisierer enthält, zu dem Servercomputer **121**, und in Reaktion auf die Nachricht sendet der Servercomputer **121** einen weiteren Kartenposten zurück. Wenn die Stelle auf einem Servercomputer ist, der ein anderer als der Servercomputer **121** ist, sendet der Client-Prozess eine Nachricht, die einen Betriebsmittellokalisierer enthält, zu diesem Servercomputer, und in Reaktion auf die Nachricht sendet der Servercomputer einen weiteren Kartenposten zu-

rück. Wenn die durch den Zeiger identifizierte Stelle innerhalb des Zellulartelefons **100** ist, liest der Client-Prozess einfach den Posten aus. In jedem Fall wird eine Fax-Form **210** ([Fig. 2G](#)), die eine Eingabekarte ist, auf dem Anzeigebildschirm **105** durch das Zellulartelefon **100** angezeigt. Dieses Beispiel demonstriert, dass die Information, auf die durch den Client-Prozess zugegriffen wird, bei irgendeiner Anzahl von Stellen angeordnet sein kann. Der zu der Fax-Taste gehörende Betriebsmittellokalisierer identifiziert die geeignete Stelle.

[0109] Wenn die Faxform **210** angezeigt wird, gibt der Anwender die Faxnummer bei ABC-Designs ein, wie es in [Fig. 2H](#) gezeigt ist, indem er das Zellulartelefon-Tastenfeld verwendet. Bei diesem Ausführungsbeispiel wird die Telefonnummer durch den Client-Prozess automatisch formatiert. Nachdem die Telefonnummer eingegeben ist, hängt der Client-Prozess die Telefonnummer an einen Betriebsmittellokalisierer an und sendet die Information zum Servercomputer **121**.

[0110] Wenn der Servercomputer **121** die Information empfängt, führt der Servercomputer **121** eine Anwendung für eine gemeinsame Gateway-Schnittstelle (CGI) aus, auf die durch den Betriebsmittellokalisierer gezeigt wird. Die CGI-Anwendung greift die nötige Information und sendet die Information per e-Mail zu einem Fax-Gateway. Der Fax-Gateway wandelt auf einen Empfang der e-Mail hin die Information in ein Fax um und sendet die Information zu der spezifizierten Telefonnummer. Somit erfordert das Zellulartelefon **100** weder eine Druckerverbindung noch einen Drucktreiber, kann aber dennoch unter Verwendung des Faxgeräts bei ABC-Designs drucken.

[0111] Wie es bei diesem Beispiel dargestellt ist, sendete das Zellulartelefon **100** eine Anforderung für einen bestimmten Kaufauftrag und plante die Zeit für ein Senden von Daten in Reaktion auf die Anforderung zu einer lokalen Maschine, die die Daten drucken kann. Somit lassen die Prozesse dieser Erfindung, wie sie nachfolgend vollständiger beschrieben werden, im Zellulartelefon **100** in Kombination mit einem datenfähigen Zellulartelefonnetzwerk **110** und dem Servercomputer **121** zu, dass das Zellulartelefon **100** effektiv eine Anwendung auf den Servercomputer **121** am Netzwerk **120** verwendet, selbst wenn das Zellulartelefon **100** nur eine im Telefon **100** gefundene Mikrosteuerung verwendet und kein separates Computermodul erfordert, wie beim Stand der Technik.

[0112] Zusätzlich erzeugt der Client-Prozess unter Verwendung der Information, die vom Servercomputer **121** gesendet ist, d.h. der Karten, eine weite Vielfalt von Anwenderschnittstellen, wie es in den [Fig. 2A](#) bis [Fig. 2H](#) gezeigt ist. Die besondere Konfiguration der verschiedenen Anwenderschnittstellen wird durch die Karten definiert, die in einem Kartenposten gesendet werden. Folglich ist die Anwenderschnittstelle nicht auf ein bestimmtes Format, wie beispielsweise ein Format vom e-Mail-Typ festgelegt, sondern ist das Format vielmehr variabel und kann durch jede Karte neu definiert werden, die durch den Client-Prozess interpretiert wird. Ebenso ist die Anwenderschnittstelle für eine Anwendung auf einem Servercomputer im Allgemeinen unabhängig von der Anwenderschnittstelle für eine andere Anwendung auf diesem Servercomputer.

[0113] Insbesondere erzeugt die Anwendung, auf die auf dem Servercomputer **121** zugegriffen wird, den Kartenposten und definiert somit infolge davon jede der verschiedenen Anwenderschnittstellen. Jede Anwenderschnittstelle lässt zu, dass der Anwender eine bestimmte Auswahl identifiziert. Jede bestimmte Auswahl könnte in einer Erzeugung einer anderen Anwenderschnittstelle mit anderen Auswahlen resultieren. Somit sind die Anwenderschnittstellen nur durch die Anwendungen beschränkt, auf die für die Zweizeige-Datenkommunikationsvorrichtung zugegriffen werden kann.

[0114] Wie es nachfolgend gezeigt ist, kann eine weite Vielfalt von Anwendungen auf einem Servercomputer vorgesehen sein. Trotz der Robustheit des Client-Moduls beim Interpretieren einer weiten Vielfalt von Anwendungen, ist der Client-Prozess typischerweise von leichtem Gewicht und erfordert somit nur Betriebsmittel von leichtem Gewicht, wie z.B. 60 KBytes eines Nurlesespeichers (ROM) für das Client-Modul, 10 KBytes eines Direktzugriffsspeichers (RAM) und weniger als eine Million Anweisungen pro Sekunde (MIPS) an Verarbeitungsleistung. Da der Client-Prozess nur diese leichtgewichtigen Betriebsmittel in einer Zweizeige-Datenkommunikationsvorrichtung benötigt, kann der Client existierende Betriebsmittel in einer solchen Vorrichtung verwenden und fügt daher keine Kosten zu der Zweizeige-Datenkommunikationsvorrichtung hinzu, wie beispielsweise einem datenfähigen Zellulartelefon **100**.

[0115] Bei einem weiteren Ausführungsbeispiel kann der Anwender das Zellulartelefon **100** konfigurieren, um auf den Servercomputer **131** auf das lokale Firmennetzwerk **130** zuzugreifen. In Reaktion auf den Zugriff durch den Anwender sendet der Computer **131** eine Heimkarte (nicht gezeigt) zum Zellulartelefon **100**, welches infolge davon eine Heimbildschirmanzeige auf dem Anzeigebildschirm **105** erzeugt.

[0116] Wenn der Anwender persönliche Information auf der Heimbildschirmanzeige oder auf einer nachfolgenden Bildschirmanzeige, die zu der Heimkarte gehört, auswählt, wird eine Nachricht, die einen Betriebsmittellokalisierer für einen persönlichen Informationsposten enthält, vom Zellulartelefon **100** zum Computer **131** gesendet. In Reaktion auf die Nachricht sendet der Computer **131** einen Kartenposten, der eine Anzeigekarte und eine Auswahlkarte enthält, zum Zellulartelefon **100**. Bei diesen Beispielen ist der Kartenposten derart beschrieben, dass er eine von drei Karten enthält, nämlich eine Anzeigekarte, eine Auswahlkarte und eine Eingabekarte. Jedoch sind diese Beispiele nur illustrativ und sollen die Erfindung nicht auf diese bestimmten Ausführungsbeispiele von Karten beschränken. Angesichts dieser Offenbarung werden Fachleute auf dem Gebiet dazu fähig sein, Kombinationen aus diesen Typen von Karten zu bilden und andere Typen von Karten zu definieren, wenn solche Karten für die besondere Anwendung geeignet sind.

[0117] Der Client-Prozess im Zellulartelefon **100** interpretiert die Anzeigekarte, die Bild- und Textdaten enthält, und erzeugt eine Bildschirmanzeige **300** auf dem Anzeigebildschirm **105** ([Fig. 3A](#)). Die Bildschirmanzeige **300** enthält eine Heimtaste **301** und eine Info-Taste **302**. Wenn der Anwender die Heimtaste **301** auswählt, wird der Anwender zu dem Heimbildschirm zurückgebracht. Die Info-Taste **302** funktioniert auf eine Art, die gleich derjenigen ist, die oben für die Info-Taste **205** beschrieben ist.

[0118] Wenn der Anwender eine vorbestimmte Taste drückt, interpretiert der Client-Prozess die Auswahlkarte und wird eine zweite Bildschirmanzeige **304** ([Fig. 3B](#)) auf dem Anzeigebildschirm **105** angetrieben. Die Bildschirmanzeige **304** ist ein Menü aus der persönlichen Information, die auf dem Servercomputer **131** gespeichert ist, zur Verwendung durch den Anwender des Zellulartelefons **100**. Ein Mehrfachanzeigebildschirmkartenindikator **203**, wie z.B. die Hand mit einem Finger, der nach unten zeigt, stellt dem Anwender dar, dass die Liste zusätzliche Elemente hat, die auf der nächsten Bildschirmanzeige erscheinen. Die Bildschirmanzeige **304** zeigt auch die Anzahl von e-Mail-Nachrichten, Faxen und Sprachnachrichten an, die auf den Anwender warten.

[0119] Der Anwender rollt die Bildschirmanzeige Zeile für Zeile, bis die Bildschirmanzeige **305** auf dem Anzeigebildschirm **105** ist. Anfangs ist das vierte Element im Menü nicht hervorgehoben. Bei diesem Beispiel drückt der Anwender die vier Tasten auf dem Tastenfeld des Zellulartelefons **100**, um den Zeitplan des Anwenders anzuschauen. In Reaktion auf das Drücken der Taste sendet das Client-Modul im Zellulartelefon **100** eine Nachricht, die einen Betriebsmittellokalisierer enthält, der dem Menüelement zugeordnet ist, das durch Drücken der vier Tasten bzw. der Taste vier ausgewählt ist, zum Servercomputer **131** unter Verwendung des datenfähigen Zellulartelefonnetzwerks **110** und des lokalen Firmennetzwerks **130**.

[0120] In Reaktion auf die Nachricht führt der Servercomputer **131** die Anwendung aus, die im Betriebsmittellokalisierer identifiziert ist. Auf eine Beendigung der Ausführung hin, sendet der Servercomputer **131** über das lokale Firmennetzwerk **130** und das datenfähige Zellulartelefonnetzwerk **110** zum Zellulartelefon **100** einen Kartenposten, der eine Auswahlkarte enthält, die den Zeitplan des Anwenders für diesen Tag beschreibt.

[0121] Bei diesem Ausführungsbeispiel hat der Servercomputer **131** dann, wenn der Servercomputer **131** das Senden beendet, die Antwort auf die Nachricht fertig gestellt und hat die gesamte nötige Information zum Zellulartelefon **100** gesendet. Daher behält der Servercomputer **131** keinerlei Zustandsinformation in Bezug auf die gesendete Information zurück und wird somit als zustandsloser Servercomputer **131** bezeichnet. Bei diesem Ausführungsbeispiel kann der Client-Prozess nur einen Kartenposten anfordern. Wie es jedoch hierin demonstriert ist, versorgen die Kartenposten und das interaktive Zweizeige-Datenkommunikationssystem dieser Erfindung den Anwender mit einer neuen Fähigkeitsebene.

[0122] Wenn das Zellulartelefon **100** den Kartenposten empfängt, interpretiert der Client-Prozess im Zellulartelefon **100** die Auswahlkarte und treibt die Bildschirmanzeige **306** ([Fig. 3D](#)) auf dem Anzeigebildschirm **105** an. Anfangs ist das erste Element im Menü der Bildschirmanzeige **306** nicht hervorgehoben. Wenn der Anwender die eine Taste auf dem Tastenfeld des Zellulartelefons **100** drückt, hebt das Zellulartelefon **100** das erste Element im Menü hervor. Das Zellulartelefon **100** erzeugt eine Bildschirmanzeige **308** ([Fig. 3E](#)) daraufhin, dass der Anwender nachfolgend eine vorbestimmte Taste drückt. Die Bildschirmanzeige **308** enthält eine Zeitplanungstaste **309**, die dann, wenn sie ausgewählt ist, den Anwender zur Bildschirmanzeige **306** ([Fig. 3D](#)) zurückbringt. Die Bildschirmanzeige **308** enthält auch eine detaillierte Beschreibung des Treffens um 10:00 vormittags.

[0123] Während die Bildschirmanzeige **308** aktiv ist, werden dem Anwender dann, wenn der Anwender eine vorbestimmte Taste drückt, die Optionen in der Bildschirmanzeige **310** ([Fig. 3F](#)) präsentiert. Anfangs ist ein Element 2 in der Bildschirmanzeige **310** nicht hervorgehoben.

[0124] Bei diesem Beispiel drückt der Anwender die Taste 2 auf dem Tastenfeld des Zellulartelefons **100** und somit sendet das Zellulartelefon **100** eine Nachricht, die einen Betriebsmittellokalisierer enthält, zu dem Servercomputer **131**, um eine e-Mail-Nachricht zu Bill Smith zu senden, die das Treffen um 10:00 vormittags bestätigt. Wenn der Servercomputer **131** die durch den Betriebsmittellokalisierer adressierte Anwendung ausführt, wird eine e-Mail-Nachricht gesendet.

[0125] Bei einem weiteren Beispiel verbindet der Anwender des Zellulartelefons mit einem Internet-Service-provider-Computer **141** im Internet **140** unter Verwendung des datenfähigen Zellulartelefonnetzwerks **110**. Auf eine Verbindung des Zellulartelefons **100** hin sendet der Serviceprovider **141** zum Zellulartelefon **100** einen Kartenposten, um die [Fig. 4A](#) bis [Fig. 4C](#) zu erzeugen.

[0126] Der Client-Prozess im Zellulartelefon **100** interpretiert die erste Karte im Kartenposten vom Computer **141** und erzeugt eine Bildschirmanzeige **400** ([Fig. 4A](#)). Wenn der Anwender eine vorbestimmte Taste drückt, zeigt das Zellulartelefon **100** eine Bildschirmanzeige **401** ([Fig. 4B](#)) an. Die Bildschirmanzeige **401** stellt dem Anwender eine Reihe von Auswahlen zur Verfügung, die Dienste alphabetisch gruppieren.

[0127] Wenn der Anwender die Taste 7 auf dem Tastenfeld des Zellulartelefons **100** drückt, zeigt das Zellulartelefon **100** eine Liste der Dienste an, die die Buchstaben P, R oder S als den ersten Buchstaben im Dienstnamen haben. Bei diesem Ausführungsbeispiel sind die Bildschirmanzeigen **401** und **402** eine einzige Karte, wie z.B. ein einzelner Bildschirm. Jeder der verschiedenen Dienste, die einer Taste zugehörig sind, hat einen Index, und dann, wenn eine bestimmte Auswahl durch den Anwender durchgeführt wird, definiert die Auswahl einen Index. Der Client-Prozess zeigt dann alle Dienste mit dem Index an, der dem Index entspricht, der durch die Auswahl des Anwenders definiert ist.

[0128] In der Bildschirmanzeige **402** wird dem Anwender eine Reihe von Auswahlen von Diensten, die dem Anwender verfügbar sind, unter Tabelle 7 gegeben. Anfangs ist ein Element 3 in der Bildschirmanzeige **402** nicht hervorgehoben. Bei diesem Beispiel drückt der Anwender die Taste 3 auf dem Tastenfeld des Zellulartelefons **100** zum Auswählen der Aktienkurse und wird das Element Drei in der Bildschirmanzeige **402** hervorgehoben.

[0129] In Reaktion auf diese Auswahl sendet das Zellulartelefon **100** eine Anforderung für einen Aktienkurs, d.h. eine Nachricht, die einen Betriebsmittellokalisierer enthält, über das Zellulartelefonnetzwerk **100** und das Internet **140** zu dem Serviceprovider **141**. In Reaktion auf die Anforderung hin führt der Serviceprovider-Computer **141** die durch den Betriebsmittellokalisierer adressierte Anwendung aus. Die Anwendung liest einen Kartenposten aus, der infolge davon zum Zellulartelefon **100** gesendet wird. Der Kartenposten enthält eine Anzeigekarte und eine Eingabekarte.

[0130] Auf ein Empfangen des Kartenpostens hin interpretiert der Client-Prozess im Zellulartelefon **100** die Anzeigekarte und erzeugt eine Bildschirmanzeige **403** ([Fig. 4D](#)). Wenn der Anwender eine vorbestimmte Taste drückt, wird eine Eingabebildschirmanzeige **406** ([Fig. 4E](#)) auf dem Anzeigebildschirm **105** des Zellulartelefons **100** erzeugt.

[0131] Anfangs ist der Kasten mit den Buchstaben SUNW in der Bildschirmanzeige **406** leer. Die Buchstaben SUNW werden durch den Anwender in den Kasten eingegeben, um das Fernschreibersymbol der Aktie anzuzeigen, für welche der Anwender Information wünscht. Nachdem der Anwender das Aktien-Fernschreibersymbol eingegeben hat, drückt der Anwender die vorbestimmte Taste, um anzuzeigen, dass die Eingabe beendet ist.

[0132] In Reaktion auf die Eingabe durch den Anwender hängt das Client-Modul das Aktien- bzw. Börsen-Fernschreibersymbol an den Betriebsmittellokalisierer an und sendet den Betriebsmittellokalisierer zu dem Serviceprovider-Computer **141**, der infolge davon eine durch den Betriebsmittellokalisierer adressierte Anwendung ausführt, um die letzte Aktienmarktinformation für das Aktien-Fernschreibersymbol auszulesen. Der Serviceprovider **141** verwendet die ausgelesene Information zum Erzeugen eines Kartenpostens, der die Information enthält, und sendet dann den Kartenposten zum Zellulartelefon **100**.

[0133] Der Client-Prozess im Zellulartelefon **100** interpretiert die erste Karte im Posten und erzeugt eine Bildschirmanzeige **409** ([Fig. 4F](#)). Der Annehmlichkeit halber sind die [Fig. 4E](#) bis [Fig. 4I](#) zusammengruppiert und durch eine gestrichelte Linie getrennt. Jedoch kann bei diesem Ausführungsbeispiel der Anzeigebildschirm **105** zu jeder gegebenen Zeit irgendwelche vier benachbarten Linien anzeigen, und somit dient die Gruppierung der Linien in den [Fig. 4E](#) bis [Fig. 4I](#) der Annehmlichkeit halber nur zum Demonstrieren der Ebene von

Information, die durch den Client-Prozess ausgelesen und angezeigt werden kann. Die Verwendung eines Anzeigebildschirms mit vier Linien ist nur illustrativ. Der Client-Prozess kann mit einem Anzeigebildschirm von irgendeiner Größe arbeiten, und zwar selbst mit einem Anzeigebildschirm mit einer Linie. Jedoch ist ein Anzeigebildschirm mit mehreren Linien bevorzugt.

[0134] In den oben diskutierten Figuren ist der Anzeigebildschirm eine Pixelanzeige und kann somit Bilder anzeigen. Bei einem weiteren Ausführungsbeispiel zeigt die Bildschirmanzeige nur Text an und ist bezüglich der Größe kleiner. Für ein derartiges Ausführungsbeispiel sind die verschiedenen Einträge abgekürzt und wird nur Text angezeigt, aber die allgemeine Operation ist identisch zu derjenigen, die gerade beschrieben ist. Ebenso können die verschiedenen Computernetzwerke miteinander so verbunden sein, dass ein Anwender mit einem Zugriff auf ein Computernetzwerk Information über ein weiteres Computernetzwerk erhalten kann. Darüber hinaus sind die oben beschriebenen Ausführungsbeispiele lediglich illustrativ. Ein wichtiger Aspekt dieser Erfindung besteht darin, dass das Zellulartelefon **100** mit irgendeinem Typ von Serveranwendung interagieren kann, die konfiguriert ist, um mit dem Client-Prozess im Zellulartelefon **100** zu kommunizieren und mit diesem zu interagieren. Somit ist der Anwender nicht mehr auf nur einige Dienste beschränkt, die durch einen Telefonnetzwerk-Provider angeboten werden.

[0135] In [Fig. 1](#) muss der Anwender eines Zellulartelefons jeden Computer von Interesse adressieren, d.h. mit diesem verbinden, um auf die unterschiedlichen Dienste zuzugreifen. Folglich erfordert jeder Computer die Information, die zum Kommunizieren mit dem Zellulartelefon **100** nötig ist. Bei einem weiteren Ausführungsbeispiel, das nicht dargestellt ist, kontaktiert das Zellulartelefon **100** einen einzelnen Zentralcomputer über das datenfähige Zellulartelefonnetzwerk **110**. Dieser Computer ist mit jedem der anderen Netzwerke verbunden, die in [Fig. 1](#) dargestellt sind. Folglich sendet der Anwender des Zellulartelefons **100** eine Nachricht, die einen Betriebsmittellokalisierer enthält, zum Zentralcomputer, verarbeitet der Zentralcomputer die Nachricht und liest die Information, die durch den Betriebsmittellokalisierer adressiert ist, aus dem geeigneten Netzwerk aus, das in [Fig. 1](#) gezeigt ist. Nachdem die angeforderte Information ausgelesen ist, erzeugt der Zentralcomputer einen Kartenposten und sendet den Kartenposten zum Zellulartelefon **100**. Bei diesem Ausführungsbeispiel muss nur ein Computer konfiguriert sein, um mit dem Zellulartelefon **100** zu kommunizieren. Jedoch muss derselbe Computer konfiguriert sein, um mit allen anderen Computernetzwerken zu kommunizieren, die für den Anwender des Zellulartelefons **100** von Interesse sind.

[0136] Somit kann der Client-Prozess auf einer Zweiwege-Datenkommunikationsvorrichtung bei dieser Erfindung eine Interaktion mit einem bestimmten Servercomputer initiieren. Der Servercomputer sendet (i) Information zu dem Client-Prozess, um eine Anwenderschnittstelle zu erzeugen, und (ii) einen Betriebsmittellokalisierer für jede mögliche Auswahl durch den Anwender von der Anwenderschnittstelle. Die Betriebsmittellokalisierer können Anwendungen auf dem Servercomputer, Anwendungen auf anderen Servercomputern oder eine Anwendung auf dem Servercomputer, der infolge davon auf andere Servercomputer zugreift, adressieren. Folglich ist der Anwender einer Zweiwege-Datenkommunikationsvorrichtung nur durch die Anwendungen beschränkt, die auf den Servercomputern zur Verfügung gestellt sind.

[0137] Weiterhin kann der Anwender durch Modifizieren der Anwendungen auf den Servercomputern mit neuen und/oder aktualisierten Fähigkeiten versehen werden. Es gibt kein Erfordernis dafür, dass der Client-Prozess für eine neue oder aktualisierte Anwendung geändert wird. Der Client-Prozess muss nur die von einer Anwendung empfangene Information interpretieren und eine Nachricht für zusätzliche Information senden. Diese Operationen sind durch eine neue oder aktualisierte Anwendung nicht beeinflusst. Folglich erfordert diese Erfindung, wie es oben angegeben ist, keine Verteilung von Anwenderupdates oder neuen Anwendungen zu dem Endanwender der Zweiwege-Datenkommunikationsvorrichtung.

[0138] [Fig. 5](#) ist eine Darstellung eines weiteren Ausführungsbeispiels des Luftnetz-Netzwerks **150**. Bei diesem Ausführungsbeispiel werden die Nachrichten von einer Zweiwege-Datenkommunikationsvorrichtung, z.B. den Vorrichtungen **100** und **101**, zu einer Luftnetz-Netzwerkumsetzungseinheit **500** geführt. Die Luftnetz-Netzwerkumsetzungseinheit **500** und eine bestimmte Zweiwege-Datenkommunikationsvorrichtung, z.B. irgendeine der Vorrichtungen **100** und **101**, kommunizieren unter Verwendung des Protokolls für eine Punkt-zu-Punkt-Kommunikation auf dem bestimmten Netzwerk, das die Luftnetz-Netzwerkumsetzungseinheit **500** und diese Zweiwege-Datenkommunikationsvorrichtung verbindet. Wenn beispielsweise das datenfähige Zellulartelefonnetzwerk **110** ein zellulares digitales Paketdatenetzwerk ist, kann entweder das Sendesteuerprotokoll (TCP = Transmission Control Protocol) oder das Anwenderdatagrammprotokoll (UDP = User Datagram Protocol) verwendet werden.

[0139] Die Luftnetz-Netzwerkumsetzungseinheit **500** transferiert Daten zwischen der Zweiwege-Datenkom-

munikationsvorrichtung und dem ausgewählten Computernetzwerk, nachdem die Umsetzungseinheit **500** den Kommunikationspfad für gültig erklärt, wie es nachfolgend vollständiger erklärt wird, und verschlüsselt die zu dem Computernetzwerk transferierte Nachricht, wenn es nötig ist. Zusätzlich sammelt die Luftnetz-Netzwerkumsetzungseinheit **500** Transaktions- und Buchungsinformation in Bezug auf die Kommunikation zwischen der Zweiwege-Datenkommunikationsvorrichtung und dem bestimmten Computernetzwerk. Insbesondere stellt die Luftnetz-Netzwerkumsetzungseinheit **500** eine Zugriffssteuerung zum Bezahlen von Diensten und einen Protokollierungsmechanismus zur Abbuchung zur Verfügung. Die Luftnetz-Netzwerkumsetzungseinheit **500** kann auch einen Führungsdienst für Anwender zur Verfügung stellen.

[0140] [Fig. 6](#) ist ein Blockdiagramm eines typischen digitalen zellularen GSM-Telefons. Jede der Hardware-Komponenten im Zellulartelefon **600** ist Fachleuten auf dem Gebiet bekannt, und somit werden die Hardware-Komponenten hierin nicht detailliert beschrieben. Die kompilierten und gebundenen Prozesse sind in einem ROM **601** als Client-Modul **602** und als Unterstützungsmodule **603** gespeichert. Auf eine Aktivierung einer vorbestimmten Tastensequenz unter Verwendung des Tastenfelds hin, initiiert ein Prozessor **610** einer physikalischen Schicht, der hierin manchmal als Mikrosteuerung bezeichnet wird, einen Client-Prozess unter Verwendung des Client-Moduls **602** im ROM **601**.

[0141] Bei diesem Ausführungsbeispiel enthält das Client-Modul **602** eine Vielzahl von Managermodulen, wie es nachfolgend vollständiger erklärt wird. Die bestimmten Managermodule, die verwendet werden, werden durch die Eigenschaften des bestimmten Zellulartelefons **100** bestimmt, in welchem das Client-Modul **602** implementiert ist. Das Client-Modul **602** muss Managermodule enthalten, um eine Schnittstelle zu Modulen zu bilden, die die bestimmte Hardware im Zellulartelefon **100** steuern, ein Managermodul, um eine Schnittstelle zu dem bestimmten Zellulartelefonnetzwerk-Protokoll zu bilden, das durch das Zellulartelefon **100** verwendet wird, und ein Managermodul, um die empfangenen Kartenposten zu interpretieren. Daher sind die hierin beschriebenen bestimmten Managermodule nur illustrativ für die Prinzipien dieser Erfindung und sollen die Erfindung nicht auf die spezifischen Module beschränken, die nachfolgend vollständiger beschrieben sind.

[0142] Bei diesem Ausführungsbeispiel steuert der Client-Prozess die Operationen einer Vielzahl von Zellulartelefon abhängigen Unterstützungsprozessen bzw. Hilfsprozessen, die im ROM **601** gespeichert sind, wie beispielsweise ein Anzeigemodul, ein Tastenfeldmodul und ein Netzwerk- und Endgeräte-Steuermodul, auf die oben gemeinsam als Hilfsmodule **603** Bezug genommen wurde. Die Kombination aus dem Client-Prozess, dem Anzeigeprozess, dem Tastenfeldprozess und dem Netzwerk- und Endgeräte-Steuerprozess wird als vordergründige Aufgaben durch den Mikro-Kernel m Zellulartelefon **600** angesehen. Ebenso werden hierin Modul und Prozess austauschbar verwendet, aber Fachleute dem Gebiet werden erkennen, dass das Modul die Computersoftware ist, wie sie in einem Speicher gespeichert ist, und zwar vorzugsweise einem ROM, des Zellulartelefons **600**, und der entsprechende Prozess die Ausführung des Moduls durch die Mikrosteuerung im Zellulartelefon **600** ist. Wiederum ist zu beachten, dass diese Erfindung keinen separaten Prozessor erfordert und stattdessen die Verarbeitungsleistung verwenden kann, die bereits im Zellulartelefon **600** existiert, weil, wie es oben beschrieben ist, der Client-Prozess so leichtgewichtig ist.

[0143] Die Anwenderschnittstelle für das Zellulartelefon **600** bestimmt die Version des Anwenderschnittstellen-Managermoduls, das im ROM **601** gespeichert ist. Bei einem Ausführungsbeispiel sind die Parameter, die zum Definieren der Anwenderschnittstellenebene bzw. -schicht verwendet werden, die Anzeigeauflösung, der Pixelzugriff der Anzeige und die Unterstützung von weichen Tasten. Eine Definition der Anwenderschnittstellenebenen ist in Tabelle 1 angegeben.

TABELLE 1

ANWENDERSCHNITTSTELLENEBENEN-DEFINITIONEN

Ebene 1	Nur Text; 1 oder mehr Zeilen; 12 bis 15 Zeichen pro Zeile; und keine weichen Tasten.
Ebene 2	Nur Text; 4 oder mehr Zeilen; 20 bis 25 Zeichen pro Zeile; und weiche Tasten.
Ebene 3	Pixelzugriff; 150 mal 75 Pixel oder darüber; und weiche Tasten.

[0144] Das Anwenderschnittstellen-Managermodul präsentiert Daten zum Anzeigemodul, das infolge davon einen Anzeigebildschirm **605** antreibt; und fängt Daten, die durch den Anwender eingegeben sind, auf dem

Anzeigebildschirm **605**. In Reaktion auf diese Information bereitet der Client-Prozess eine Nachricht zum Senden durch ein Netzwerk-Managermodul vor.

[0145] Um die Operationen vollständiger zu erklären, die über das Luftnetz-Netzwerk **150** durchgeführt werden, ist [Fig. 7](#) ein Blockdiagramm, das die verschiedenen Komponenten bei einem Ausführungsbeispiel dieser Erfindung eines Zellulartelefons **700** darstellt. Fachleute auf dem Gebiet werden erkennen, dass das Zellulartelefon **700** eine Schaltung und eine Software enthält, die gleich denjenigen ist, die im Zellulartelefon **600** für Sprach- und Datenoperationen dargestellt sind, die durch das Zellulartelefon **700** zusätzlich zu den Modulen zur Operation auf dem Luftnetz-Netzwerk **750** unterstützt werden. Gleichermäßen enthält ein Servercomputer **743** andere Software und Hardware, die Fachleuten auf dem Gebiet bekannt sind, und somit der Klarheit halber in [Fig. 7](#) nicht dargestellt sind.

[0146] Bei diesem Ausführungsbeispiel kommuniziert ein Client-Modul **702** im digitalen Zellulartelefon **700**, dass auf der Mikrosteuerung des Telefons **700** ausführt, mit dem Servercomputer **743** über ein zellulares digitales Paketdaten-(CDPD = cellular digital packet data)-Netzwerk **710**. Das zellulare digitale Paketdatennetzwerk **710** wird zum Darstellen eines Ausführungsbeispiels dieser Erfindung auf einem Zweizege-Datenkommunikationsnetzwerk verwendet. Die Prinzipien dieser Erfindung können mit einer weiten Vielfalt von Zweizege-Datenkommunikationsnetzwerken verwendet werden. Beispielsweise enthalten andere Zweizege-Datenkommunikationsnetzwerke für Zellulartelefone, die verwendet werden können, TDMA-, CDMA- und GSM-Schaltkreis geschaltete Datennetzwerke; und das analoge zellulare AMPS-Netzwerk mit einem Modem. Gleichermäßen enthalten Zweizege-Datenkommunikationsnetzwerke für Zweizege-Funkrufnetzeinheiten PACT oder andere Zweizege-Funkrufnetzeinheiten einer Priorität mit einer Datentransportfähigkeit.

[0147] Vor einem Betrachten der Operation dieser Konfiguration des Luftnetz-Netzwerks **750** in größerem Detail ist ein weiterer Aspekt dieser Erfindung erforderlich. Insbesondere ist eine Technik zum Befördern von Anweisungen vom digitalen Zellulartelefon **700** zu einer Serveranwendung auf dem Servercomputer **743** und umgekehrt erforderlich.

[0148] Eine Telefoninteraktionsbeschreibungssprache (PIDL = telephone interaction description language) ist zur Verwendung durch Dienstentwickler definiert. Eine Endgeräteinteraktionssprache (TIL = terminal interaction language) ist eine Destillation der Telefoninteraktionsbeschreibungssprache und beschreibt dieselbe Interaktion zum digitalen Zellulartelefon **700** wie die Telefoninteraktionsbeschreibungssprache zum Computer **743** beschreibt.

[0149] Mit den Ausnahmen, die nachfolgend vollständiger beschrieben werden, ist ein Prozess in der Endgeräteinteraktionssprache eine komprimierte Version desselben Prozesses, der in der Telefoninteraktionsbeschreibungssprache geschrieben ist. Die Endgeräteinteraktionssprache lässt ein einfaches Zerlegen auf der Zweizege-Datenkommunikationsvorrichtung zu, was infolge davon den Client kleiner als einen Client für die Telefoninteraktionsbeschreibungssprache macht, die durch Menschen lesbar ist, aber nicht zum Zerlegen durch eine Maschine optimiert ist.

[0150] Die Kompression von der Telefoninteraktionsbeschreibungssprache zu der Endgeräteinteraktionsbeschreibungssprache wird typischerweise zur Laufzeit durchgeführt, weil einige Karten berechnete Karten sind und somit nicht vorkompiliert werden können. Eine weite Vielfalt von Techniken kann zum Umwandeln der Telefoninteraktionsbeschreibungssprache in die Endgeräteinteraktionssprache verwendet werden. Der wichtige Aspekt besteht darin, dass dann, wenn eine Bandbreite über dem Zellulartelefonnetzwerk beschränkt ist, eine komprimierte Form der Telefoninteraktionsbeschreibungssprache verwendet wird.

[0151] Vorzugsweise wird jeder Datentyp komprimiert, um einen optimalen Transfer über das Zweizege-Datenkommunikationsnetzwerk zu ermöglichen. Beispielsweise werden die Verben in der Telefoninteraktionsbeschreibungssprache unter Verwendung einer binären Tokenisierung komprimiert. Graphiken werden unter Verwendung einer laulängenbeschränkten Kompression komprimiert und Text wird unter Verwendung von irgendeiner der wohlbekannten Techniken für eine Textkompression komprimiert. Während eine Kompression der Telefoninteraktionsbeschreibungssprache nicht erforderlich ist, um diese Erfindung zu implementieren, macht eine Kompression die Erfindung durch Verwenden der Bandbreite des Netzwerkes auf effektivere Weise effizienter.

[0152] Anweisungen in der Telefoninteraktionsbeschreibungssprache und in der Endgeräteinteraktionssprache werden in einen Posten und eine Karte gruppiert. Jeder Posten enthält eine oder mehrere Karten. Eine Karte enthält die Information, d.h. eine Gruppe einer Telefoninteraktionsbeschreibungssprache, die zum Er-

zeugen eines Bildschirms erforderlich ist. Wie es oben angezeigt ist, kann ein Bildschirm größer als die Anzahl von Zeilen in einem Anzeigebildschirm sein. Andere äquivalente Ausdrücke für eine Karte enthalten eine Seite und eine atomare Interaktion. Somit ist ein Kartenposten einfach eine Gruppe aus Bildschirmen. Die Anzahl von Karten in einem Kartenposten ist ausgewählt, um eine effiziente Verwendung der Betriebsmittel in der Zweiwege-Datenkommunikationsvorrichtung und im Luftnetz-Netzwerk zu ermöglichen.

[0153] Der Einfachheit halber ist bei diesem Ausführungsbeispiel jede Karte eine einzelne Operation. Hierin ist eine Operation als eine bezogene Gruppe von Aktionen so definiert, dass der Anwender keiner nicht vorhergesagten Verzögerung beim Bewegen von einer Aktion zur nächsten begegnet, d.h. der Anwender nicht darauf warten muss, dass das Client-Modul **702** einen weiteren Kartenposten vom Computer **743** ausliest. Ebenso kann ein Posten Definitionen von weichen Tasten enthalten, die in Kraft bleiben, während der Posten aktiv ist, d.h. durch die Mikrosteuerung des Zellulartelefons ausgeführt werden.

[0154] Der Computer **743** kann gespeicherte statische Telefoninteraktionsbeschreibungssprachen-Posten enthalten. Der Computer **743** erzeugt auch Telefoninteraktionsbeschreibungssprachen-Posten in Reaktion auf Daten von dem Anwender des Zellulartelefons oder auf Auswahlen, die durch diesen durchgeführt sind.

[0155] Bei dem in [Fig. 7](#) gezeigten Ausführungsbeispiel wandelt der Computer **743** einen Telefoninteraktionsbeschreibungssprachen-Posten in einen Endgeräteinteraktionssprachen-Posten um, der infolge davon zum Zellulartelefon **700** gesendet wird. Die Endgeräteinteraktionssprache ist so entwickelt, dass Posten unverändert in einem Speicher **716** des Zellulartelefons **700** gespeichert und auf diese direkt mit wenig oder keiner Zerlegung Bezug genommen werden kann. Während Telefoninteraktionsbeschreibungssprachen-Posten auf dem Computer **743** Referenzen zu Bildern enthalten können, enthält ein Endgeräteinteraktionssprachen-Posten die Bilder am Ende des Postens. Somit werden dann, wenn eine bestimmte Zweiwege-Datenkommunikationsvorrichtung eine Anzeige von Bildern nicht unterstützt, die Bilder einfach von dem Endgeräteinteraktionssprachen-Posten abgestreift, bevor der Posten zu dieser bestimmten Zweiwege-Datenkommunikationsvorrichtung gesendet wird.

[0156] Wie es oben angezeigt ist, ist jede Interaktion mit dem Anwender des Zellulartelefons **700** durch einen Posten oder eine Reihe von Posten beschrieben. Logischerweise liest der Anwender einen Endgeräteinteraktionssprachen-Posten, der in einem Speicher **716** des Zellulartelefons gespeichert ist, nach einem Empfang vom Computer **743** über das CDPD-Netzwerk **710** aus. Der Anwender prüft die durch Karten im Posten angezeigte Information und führt Auswahlen durch und/oder gibt angeforderte Information ein und fordert dann einen weiteren Posten an, wie es oben beispielsweise in Bezug auf die [Fig. 2A](#) bis [Fig. 2H](#) beschrieben ist.

[0157] Wenn der Anwender einen Posten empfängt, wird die erste Karte von Information auf dem Anzeigebildschirm **705** angezeigt. Typischerweise ist, wie es oben gezeigt ist, die erste Karte ein Text, ein Bild oder eine Kombination aus einem Bild und Text. Nachdem der Anwender die erste Karte geprüft hat, schlägt der Anwender auf eine Taste NEXT bzw. NÄCHSTE, um die nächste Karte in dem Posten anzuschauen. Gleichermaßen kann ein Anwender zu einer vorherigen Karte in dem Posten zurückkehren, indem er eine Taste PREV bzw. VORH. verwendet. Somit kann der Anwender unter Verwendung der Tasten NEXT und PREV rückwärts und vorwärts durch den Posten navigieren. Innerhalb einer Karte verwendet der Anwender eine Rolltaste oder Rolltasten zum Bewegen des Teils der angezeigten Karte nach oben und nach unten. Diese Beschreibung eines bestimmten Verfahrens, das zum Navigieren durch einen Posten und innerhalb einer Karte verwendet wird, soll die Erfindung nicht auf dieses bestimmte Verfahren beschränken. Angesichts dieser Offenbarung werden Fachleute auf dem Gebiet dazu fähig sein, eine weite Vielfalt von Arten zum Navigieren durch einen Posten und innerhalb einer Karte zu verwenden.

[0158] Karten sind bei diesem Ausführungsbeispiel einer von drei Typen, nämlich eine Anzeigekarte, eine Auswahlkarte und eine Eingabekarte. Unabhängig vom Typ der Karte kann die Karte Text und Bilder enthalten. Zusätzlich ist die Erfindung nicht auf diese drei bestimmten Typen von Karten beschränkt. Die Definition der drei besonderen Typen von Karten wird dazu verwendet, eine Beschreibung der Erfindung zu erleichtern und den Entwickler beim Organisieren von Anwendungen zu unterstützen.

[0159] Eine Anzeigekarte gibt Information zum Anwender zum Lesen. Der Anzeigeeinhalt kann irgendetwas von oder irgendeine Kombination von Text, einem Bild und einer weichen Taste enthalten. Die weiche Taste ist in Wirklichkeit nur dann da, während die Anzeigekarte aktiv ist.

[0160] Eine Auswahlkarte zeigt eine Liste von Auswahlen für den Anwender an. Die Auswahlen werden automatisch in einem Format präsentiert, das auf der Auswahlkarte spezifiziert ist. Siehe Anhang I, welcher ein

Teil der vorliegenden Offenbarung ist und hierin durch Bezugnahme in seiner Gesamtheit enthalten ist. Wie es oben erklärt ist, führt der Anwender eine Auswahl durch Drücken der Taste entsprechend der Auswahl durch.

[0161] Eine Eingabekarte wird dazu verwendet, Eingangsdaten vom Anwender zu erhalten. Eine Eingabekarte zeigt eine oder mehrere Eingabelinien an. Typischerweise enthält jede Eingabelinie bzw. Eingabezeile eine Anzeige, der eine Eingabelinie folgt. Die Eingabelinie kann bei diesem Ausführungsbeispiel für entweder numerische oder Textdaten sein.

[0162] Bei diesem Ausführungsbeispiel verhindern Auswahl- und Eingabekarten, dass sich der Anwender zur nächsten Karte bewegt, bis der Anwender die angeforderte Information eingegeben hat. Wenn der Anwender die letzte Karte in einem Posten erreicht und auf die Taste NEXT schlägt, wird eine Anforderung für einen neuen Posten initiiert. Der angeforderte Posten wird durch entweder den Posten bestimmt, den der Anwender beendet hat, oder durch die Auswahlen, die durch den Anwender durchgeführt sind. Ebenso werden dann, wenn der Posten beendet ist, die Auswahlen und/oder Daten, die durch den Anwender eingegeben sind, typischerweise zusammen mit der Anforderung für den neuen Posten zum Computer **743** gesendet.

[0163] Der Anhang I ist ein Ausführungsbeispiel einer Syntax für die Telefoninteraktionsbeschreibungssprache und die Endgeräteinteraktionssprache, die bei dieser Erfindung verwendet wird. Bei einem Ausführungsbeispiel wird die Telefoninteraktionsbeschreibungssprachen unter Verwendung einer Untergruppe der standardmäßigen verallgemeinerten Markierungssprache beschrieben. Nur eine Untergruppe der standardmäßigen verallgemeinerten Markierungssprache wird verwendet, so dass Telefoninteraktionsbeschreibungssprachen-Zerlegungseinheiten auch auf einfache Weise unter Verwendung von einfachen Werkzeugen, wie lex und yacc, geschrieben werden können.

[0164] Kehrt man zurück zur Operation über das Luftnetz-Netzwerk **750**, enthält das Zellulartelefon **700** ein Anzeigemodul **712**, ein Tastaturmodul **711**, ein Client-Modul **702** und ein UDP-Schnittstellenmodul **714**. Bei diesem Ausführungsbeispiel ist das Modul **702** in einem nichtflüchtigen Speicher (nicht gezeigt) des Telefons **700** gespeichert und wird durch die Mikrosteuerung (nicht gezeigt) im Telefon **700** ausgeführt. Die Module **711**, **712** und **714** arbeiten unter der Steuerung des Client-Moduls **702**.

[0165] Das Client-Modul **702** enthält Anweisungen, die die Mikrosteuerung im Zellulartelefon **700** lenken, um die Operationen durchzuführen, die nachfolgend in Bezug auf die [Fig. 8A](#) bis [Fig. 8D](#) vollständiger beschrieben sind. Die Operationen enthalten ein Senden von Anforderungen für einen einheitlichen Betriebsmittellokalisierer bzw. eine genormte Methode zum Auffinden von Ressourcen (URL) zu einem Server **749** mit einem Hypertext-Übertragungsprotokoll (HTTP = HyperText Transfer Protocol), ein Zerlegen und Anzeigen eines TIL-Postens oder von Posten, die durch den HTTP-Server **749** zurückgebracht sind, und ein Erzeugen von neuen URLs basierend auf den Betätigungen einer Taste durch einen Anwender. Für eine Beschreibung der HTTP-Server-Software und der Plattformen, die die HTTP-Server-Software laufen lassen können, siehe beispielsweise Ian S. Graham, The HTML Sourcebook, John Wiley & Sons, Inc., New York, Chapt. 8 (1995).

[0166] Ein Anwenderdatagrammprotokoll-(UDP-)Schnittstellenmodul **714** koppelt das CDPD-Netzwerk **710** mit dem Client-Modul **702** und lässt zu, dass das Client-Modul **702** unter Verwendung von UDP über das CDPD-Netzwerk **710** kommuniziert. Das Anwenderdatagrammprotokoll ist Fachleuten auf dem Gebiet wohl bekannt und ist äußerst gut dokumentiert. Das UDP-Schnittstellenmodul **714** unterstützt ein Senden von einfachen alleinstehenden Nachrichten zwischen den Verbindungspartnern.

[0167] Das Anzeigemodul **712** ist ein Anzeigetreiber, der das Client-Modul **702** mit dem Anzeigebildschirm **705** koppelt und somit zulässt, dass das Client-Modul **702** die auf dem Anzeigebildschirm **705** präsentierte Information spezifiziert. Das Anwenderschnittstellen-Managermodul innerhalb des Client-Moduls **702** wandelt die Anzeigeinformation in einer Karte in Anweisungen für das Anzeigemodul **704** um, welches infolge davon Signale liefert, die die Hardware antreiben, die die Operation des Anzeigebildschirms **705** steuert. Wenn beispielsweise der TIL-Posten ein Bild enthält, bestimmt das Anwenderschnittstellen-Managermodul, ob die aktive Karte eine Anzeige des Bildes aufruft. Wenn die aktive Karte das Anwenderschnittstellen-Managermodul dazu führt, das Bild anzuzeigen, führt das Anwenderschnittstellen-Managermodul das Bild im Speicher **716** zum Anzeigemodul **712**, welches infolge davon das Bild auf dem Anzeigebildschirm **705** anzeigt.

[0168] Das Tastaturmodul **705** koppelt ein Tastenfeld **715** mit dem Client-Modul **702** und speichert Daten, die Tasten darstellen, die durch den Anwender auf dem physikalischen Tastenfeld **715** gedrückt sind, im Speicher **716**. Das Tastaturmodul **705** teilt dem Client-Modul **702** mit, wenn der Anwender eine Taste gedrückt hat.

[0169] Wenn das Client-Modul **702** über ein Drücken einer Taste benachrichtigt ist, führt das Anwenderschnittstellen-Managermodul innerhalb des Client-Moduls **702** Information über das Drücken der Taste zum Anzeigemodul **712**, das infolge davon das geeignete Zeichen auf dem Anzeigebildschirm **705** anzeigt, wenn eine Eingabekarte aktiv ist. Wenn das Anwenderschnittstellen-Managermodul bestimmt, dass eine Auswahlkarte aktiv ist, und das Drücken der Taste einer der Auswahlen entspricht, sendet das Anwenderschnittstellen-Managermodul Anweisungen zum Anzeigemodul **712**, die darin resultieren, dass die Auswahl für den Anwender identifiziert wird, z.B. hervorgehoben wird, wie es oben beschrieben ist.

[0170] Zusätzlich zum HTTP-Server **749** enthält der Hostcomputer **743** ein UDP-Schnittstellenmodul **748**, CGI-Programme **761**, die in einem Speicher **755** des Hostcomputers **743** gespeichert sind, und TIL-Posten **760**, die im Speicher **755** gespeichert sind.

[0171] Der HTTP-Server **749** verwendet das UDP-Schnittstellenmodul **748** zum Senden von Daten zum CDPD-Netzwerk **710** und zum Empfangen von Daten von diesen. Die TIL-Posten **760** sind TIL-Posten, auf die durch den HTTP-Server **749** zugegriffen werden können. Statische Dateien, die PIDL-Posten enthalten, werden nur einmal am HTTP-Server **749** in TIL-Posten umgewandelt. Die CGI-Programme **761** sind gemeinsame Gateway-Schnittstellenprogramme, die PIDL-Posten erzeugen, die durch den HTTP-Server **749** verwendet werden, um TIL-Posten zu erzeugen, die infolge davon über die UDP-Schnittstellenmodule **748** und **714** und das Zellulartelefonnetzwerk **710** zum Client-Modul **702** gesendet werden. Bei diesem Ausführungsbeispiel sind die Dienste, die über das Luftnetz-Netzwerk **750** verfügbar sind, Anwendungen, auf die durch den HTTP-Server **749** auf dem Internet **140** zugegriffen werden kann, für welche ein Dienstentwickler einen PIDL-Posten geschrieben hat, oder ein CGI-Script, das infolge davon einen PIDL-Posten erzeugt, und wird auf dem Computer **743** gespeichert.

[0172] Die Architektur in [Fig. 7](#) demonstriert einige wichtige Aspekte dieser Erfindung. Als erstes sind die Anwendungen, die PIDL-Posten und die CGI-Scripten bei diesem Ausführungsbeispiel von dem bestimmten Zweizege-Datenkommunikationsnetzwerk unabhängig. Für den HTTP-Server **749** muss zum Kommunizieren über ein anderes Zweizege-Datenkommunikationsnetzwerk, das UDP nicht unterstützt, nur das UDP-Schnittstellenmodul **748** geändert werden. Die Anwendungen werden durch eine solche Änderung nicht beeinflusst.

[0173] Als zweites sind die Anwendungen vom HTTP-Server **749** unabhängig von der Zweizege-Datenkommunikationsvorrichtung, mit welcher der HTTP-Server **749** interagiert. Eine Anwendung auf den HTTP-Server **749** kann mit irgendeiner Zweizege-Datenkommunikationsvorrichtung kommunizieren, die den geeigneten Client und ein Modul zum Senden und Empfangen von Daten über das Zweizege-Datenkommunikationsnetzwerk enthält. Diese zwei Tatsachen bedeuten, dass eine Investition in eine Entwicklung einer Anwendung von entweder Fortschritten in Bezug auf Zweizege-Datenkommunikationsvorrichtungen oder Fortschritten in Bezug auf eine Zweizege-Datenkommunikationsnetzwerktechnologie isoliert ist.

[0174] Die [Fig. 8A](#) bis [Fig. 8D](#) sind ein Verfahrensablaufdiagramm für ein Ausführungsbeispiel dieser Erfindung. Anfangs initialisiert das Client-Modul **702** dann, wenn der Anwender eine Kommunikation über das Luftnetz-Netzwerk **750** initiiert, einen Arbeitsraum bzw. Arbeitsbereich im Speicher **716** des Zellulartelefons **700** und speichert dann beim Bekommen eines Heim-URL-Prozesses **801** einen URL im Arbeitsbereich. Bei dieser Erfindung hat bei einem Ausführungsbeispiel jedes Zellulartelefon, das das Luftnetz-Netzwerk verwendet, eine Heim-URL in einem nichtflüchtigen Speicher gespeichert, der zum Auslesen eines Heim-Kartenpostens für das Zellulartelefon verwendet wird. Bei einem weiteren Ausführungsbeispiel erhält das Zellulartelefon die Heim-URL vom Server **749**. Somit erhält das Client-Modul **702** beim Bekommen des Heim-URL-Prozesses **801** die Heim-URL. Hierin ist eine URL ein Beispiel für ein spezifisches Ausführungsbeispiel eines Betriebsmittellokalisierers.

[0175] Beispielsweise erhält das Client-Modul **702** beim Bekommen eines Heim-URL-Prozesses **801** eine Heim-URL, wie beispielsweise
<http://www.libris.com/airnet/home.cgi>

und speichert die Heim-URL im Arbeitsbereich. Der Teil der Heim-URL, <http://www.libris.com>, identifiziert einen bestimmten HTTP-Server, d.h. den Server **749**, im World Wide Web. Der Teil der URL/airnet/home.cgi spezifiziert ein bestimmtes gemeinsames Gateway-Schnittstellenprogramm innerhalb der CGI-Programme **761**. Die Verwendung einer URL, die auf einen Server im World Wide Web zeigt, ist nur illustrativ und soll die Erfindung nicht auf Anwendungen im World Wide Web beschränken. Im Allgemeinen erhält das Zellulartelefon **700** einen Identifizierer, d.h. einen Betriebsmittellokalisierer, eine Heimanwendung auf einem Heimserver, die durch den Server ausgeführt wird, wenn das Zellulartelefon anfangs auf dem Luftnetz-Netzwerk **750** aktiv wird, und spei-

chert den Betriebsmittellokalisierung im Arbeitsbereich.

[0176] Als nächstes wandelt das Client-Modul **702** beim Erzeugen eines HTTP-Anforderungsprozesses **802** die URL im Arbeitsbereich in eine HTTP-Anforderung um. Beispielsweise erzeugt für die obige URL ein Erzeugen eines HTTP-Anforderungsprozesses **802** ein Verfahrensfeld, wie beispielsweise GET/airnet/home.cgi HTTP/1.0

[0177] Das GET-Verfahren ist ein Teil von HTTP. Somit ist das Format für das GET-Verfahren Fachleuten auf dem Gebiet bekannt. Ebenso wird diese bestimmte Form des Verfahrens verwendet, weil eine spezifische Serververbindung durch das Zellulartelefon **700** eingerichtet ist, und somit eine Identifizierung des Servers unnötig ist. Nichts desto weniger weist, kurz gesagt, dieser Befehl den Server **749** an, eine Anwendung home.cgi auszuführen, und eine Ausführung der Anwendung home.cgi resultiert infolge davon in einer Erzeugung eines Heimpostens und einem darauf folgenden Senden des Heimpostens zum Zellulartelefon **700**. HTTP/1.0 spezifiziert die HTTP-Version, die durch das Client-Modul **702** im Zellulartelefon **700** verwendet wird.

[0178] Zusätzlich zu dem Verfahrensfeld könnte das Client-Modul **702** in dem Prozess **802** auch geeignete HTTP-Anforderungsfelder erzeugen, um Information zum Server **749** über die Fähigkeiten bzw. Kapazitäten des Client-Moduls **702** zu führen. Die Anforderungsfelder können Information enthalten, wie beispielsweise Listen der MIME-Inhaltstypen, die für den Client akzeptierbar sind; Listen von Datencodierungstypen, die für den Client akzeptierbar sind; Anwenderauthentifikations- und -verschlüsselungsschemainformation für den Server; die Länge in Bytes der Nachricht, die zum Server gesendet wird; und die Internet-Mailadresse des Anwenders, der auf den Server zugreift. Diese Liste von Information ist nur illustrativ und soll die Erfindung nicht auf die hierin beschriebenen bestimmten Anforderungsfelder beschränken. Irgendein Anforderungsfeld, das durch HTTP definiert ist, kann durch das Client-Modul **702** verwendet werden. Jedoch werden bei diesem Ausführungsbeispiel die Vorgaben verwendet, und somit werden keine HTTP-Anforderungsfelder erzeugt.

[0179] Typische HTTP-Verfahren, die in dem HTTP-Anforderungsprozess **802** erzeugt werden können, sind ein GET-Verfahren zum Anfordern von entweder einem TIL-Posten vom Server **749** oder einer Ausführung eines gemeinsamen Gateway-Schnittstellenprogramms auf dem Server **749**; und eine GET-Verfahrens-anforderung zu einem gemeinsamen Gateway-Schnittstellenprogramm mit Daten, wie z.B. eine Anfragekette, die an die URL angehängt ist. In jedem Fall wird eine URL zu dem Server **749** innerhalb der bestimmten Nachricht gesendet. Nachdem ein Erzeugen des HTTP-Anforderungsprozesses **802** beendet ist, transferiert der Client-Prozess zum Senden eines Anforderungsprozesses **804**.

[0180] Wenn jedoch das Übertragungssteuerungsprotokoll anstelle von UDP verwendet wird, würde das Client-Modul **702** beim Bilden des Serververbindungsprozesses **803** auf ein TCP-Modul zugreifen, das das UDP-Modul **714** ersetzen würde. Da bei diesem Ausführungsbeispiel UDP verwendet wird, ist ein Bilden des Verbindungsprozesses **803** in [Fig. 8A](#) durch eine gestrichelte Linie umgeben, um anzuzeigen, dass dieser Prozess unnötig ist, wenn UDP verwendet wird.

[0181] Beim Bilden des Serververbindungsprozesses **803** würde eine virtuelle Verbindung über das CDPD-Netzwerk **710** zwischen dem TCP-Schnittstellenmodul **714** und einem TCP-Schnittstellenmodul im HTTP-Server **749** hergestellt werden, so dass Daten zwischen dem Zellulartelefon **700** und dem Computer **743** unter Verwendung von TCP übertragen werden könnten, z.B. Puffer zum Unterstützen eines Datenaustauschs definiert werden. Das Bilden einer TCP-Verbindung ist wohlbekannt und wird somit nicht weiter beschrieben.

[0182] In [Fig. 8A](#) verbindet eine gestrichelte Linie ein Bilden eines Serververbindungsprozesses **803** mit einem Bilden eines Client-Verbindungsprozesses **860**, der auch gestrichelt ist, der durch den HTTP-Server **749** durchgeführt wird. Dies zeigt an, dass sowohl das Client-Modul **702** als auch der Server **749** zum Beenden des Prozesses **803** erforderlich sind.

[0183] Wenn die virtuelle TCP-Verbindung gebildet wird, transferiert das Client-Modul **702** eine Verarbeitung von einem Bilden des Serververbindungsprozesses **803** zu einem Senden eines Anforderungsprozesses **804**. Gleichermaßen transferiert der Server **749** zu einem Anfordern einer empfangenen Prüfung **861**, wobei der Server **749** wartet, bis eine Anforderung empfangen ist. Ein Bilden eines Client-Verbindungsprozesses **860** ist für UDP nicht nötig und somit initiiert der HTTP-Server **749** eine Verarbeitung in Bezug auf einen Prüfprozess **861** für eine empfangene Anforderung. Der Prozess **860** ist mit einem Kasten mit gestrichelter Linie umgeben, um anzuzeigen, dass der Prozess nur für TCP verwendet wird.

[0184] Beim Sendeanforderungsprozess **804** wird die HTTP-Anforderung von dem Arbeitsbereich im Telefon

700 zum HTTP-Server **749** gesendet. Wiederum verbindet eine gestrichelte Linie den Prozess **804** des Client-Moduls **702** mit einer Prüfung **861** für eine empfangene Anforderung, die durch den HTTP-Server **749** durchgeführt wird, um anzuzeigen, dass die Prüfung von Information vom Client-Modul **702** abhängt. Wenn die Übertragung bzw. das Senden der Anforderung beendet ist, transferiert das Client-Modul **702** zu einer Prüfung **806** für eine empfangene Antwort.

[0185] Auf einen Empfang und eine Speicherung der HTTP-Anforderung hin transferiert die Prüfung **861** für eine empfangene Anforderung zu einem Dienstanforderungsprozess **862**, in welchem der HTTP-Server **749** einen Dienst der empfangenen Anforderung initiiert. Beim Dienstanforderungsprozess **862** liest der HTTP-Server **749** dann, wenn die HTTP-Anforderung nur nach einem Transfer eines statischen Postens sucht, den angeforderten statischen Posten aus TIL-Posten **760** aus. Gegensätzlich dazu startet der Server **749** dann, wenn die Anforderung erfordert, dass der Server **749** Daten aus dem Internet erhält oder Daten an eine bestimmte Datei anhängt, die gemeinsame Gateway-Schnittstellenanwendung, die in der Anforderung adressiert ist, und führt die Daten in der HTTP-Anforderung zu dieser Anwendung zur weiteren Verarbeitung.

[0186] Wenn beispielsweise der Anwender des Zellulartelefons **700** ein Fax anforderte, wie in [Fig. 2F](#), identifiziert die HTTP-Anforderung eine gemeinsame Gateway-Schnittstellenanwendung in CGI-Programmen **761**, die als Eingangsdaten die Telefonnummer akzeptiert bzw. annimmt und die Information, die zu faxen ist, ergreift. Die CGI-Anwendung erzeugt eine e-Mail-Übertragung zu dem Fax-Gateway. Gleichermaßen startet der Server **749** für einen Aktienkurs in Reaktion auf die HTTP-Anforderung eine gemeinsame Gateway-Schnittstellenanwendung, die eine Aktienanforderung über das Internet **140** zu einem Aktienkurs-Serviceprovider unter Verwendung des Fernmelder-Bandsymbols aussendet, das als Eingangsdaten durch den Server **749** zu der gemeinsamen Gateway-Schnittstellenanwendung geführt bzw. übergeben ist. Wenn die Antwort auf die Aktienanfrage empfangen wird, bildet die gemeinsame Gateway-Schnittstellenanwendung einen PDIL-Posten, der die Daten in der Antwort auf die Aktienanfrage enthält.

[0187] Auf eine Beendigung eines Bedienens der Anforderung hin wandelt der HTTP-Server **749** den PIDL-Posten in einen TIL-Posten um und bringt den TIL-Posten unter Verwendung von UDP in einem Transferantwortprozess **863** zu dem Client-Modul **702** zurück, der durch eine gestrichelte Linie mit der Prüfung **806** für eine empfangene Antwort im Client-Modul **702** verbunden ist. Wenn der TIL-Posten transferiert ist, speichert das Client-Modul **702** den Posten im Speicher **716**.

[0188] Nachdem der TIL-Posten transferiert ist, schließt der HTTP-Server **749** den Prozess zum Antworten auf die Nachricht vom Zellulartelefon **700**. Alle Information, die durch das Client-Modul **702** zum Erzeugen einer Anwenderschnittstelle auf dem Anzeigebildschirm **705** und zum Antworten auf irgendeine Auswahl oder eine in der Anwenderschnittstelle präsentierte Dateneingabe nötig ist, ist in dem TIL-Posten enthalten. Folglich muss das Client-Modul **702** nur den TIL-Posten interpretieren und die Anwendereingabe interpretieren, um die nächste Nachricht zum HTTP-Server **749** zu senden. Der Zustand für den HTTP-Server ist in der nächsten Nachricht definiert. Folglich ist der HTTP-Server **749** zustandslos, weil der HTTP-Server **749** keine Zustandsinformation in Bezug auf eine Antwort auf eine Nachricht zurückhält, nachdem die Nachricht gesendet ist.

[0189] Jedoch könnte bei einem weiteren Ausführungsbeispiel (nicht gezeigt) ein Server eine Zustandsinformation in Bezug auf jede Interaktion mit einem Client-Modul zurückhalten. Wenn beispielsweise der Server eine Auswahlkarte zum Client-Modul sendete, würde der Server Zustandsinformation zurückhalten, die anzeigt, dass eine Auswahl vom Client-Modul anhängig war. Bei diesem Ausführungsbeispiel wird dann, wenn der Anwender eine Auswahl durchführt, z.B. eine Taste zwei drückt, um eine Auswahl zwei anzuzeigen, die Auswahl zu dem Server gesendet, welcher infolge davon auf die URL zugreift, die zu der Auswahl zwei gehört. Wenn diese URL eine weitere Anwendung adressiert, führt der Server diese Anwendung aus. Somit hält der Server bei diesem Ausführungsbeispiel Zustandsinformation in Bezug auf jede Interaktion mit einem Client-Modul zurück. Angesichts dieser Offenbarung können Fachleute auf dem Gebiet die Prinzipien der Erfindung unter Verwendung eines Servers implementieren, der Zustandsinformation zurückhält, wenn eine solche Client/Server-Kombination vorteilhaft ist.

[0190] Kehrt man zurück zu dem vorliegenden Ausführungsbeispiel, verlässt das Client-Modul **702** dann, wenn der TIL-Posten empfangen ist, den Prüfprozess **806** für eine empfangene Antwort und transferiert zu einem Verarbeiten einer ersten Karte **808**. Wenn jedoch TCP anstelle von UDP verwendet wird, würde das Client-Modul **702** auf ein Verlassen der Prüfung **806** hin die virtuelle TCP-Verbindung im Prozess **807** für eine beendete Übertragung schließen. Auf ein Schließen der virtuellen TCP-Verbindung hin würde eine Verarbeitung zu einem Verarbeiten der ersten Karte **808** transferieren. Wiederum ist der Prozess **807** für ein Beenden einer Übertragung innerhalb eines Kastens mit gestrichelter Linie umgeben, um anzuzeigen, dass der Prozess

807 nur mit TCP verwendet wird.

[0191] Beim Verarbeiten der ersten Karte **808** zerlegt das Client-Modul **702** den TIL-Posten und interpretiert die erste Karte. Eine Verarbeitung transferiert von einem Verarbeiten der ersten Karte **808** zu einem Erzeugen eines Anzeigeprozesses **809**.

[0192] Beim Erzeugen eines Anzeigeprozesses **809** führt das Client-Modul **702** die anzuzeigenden Daten in der ersten Karte zum Anzeigemodul **712**. Das Anzeigemodul **712** treibt in Reaktion auf die Daten den Text und die Bilder in den Daten auf dem Anzeigebildschirm **705** an. Ein Erzeugen des Anzeigeprozesses **809** transferiert eine Verarbeitung zu einer Prüfung **820** auf ein Drücken einer Taste durch einen Knoten **813**. In den [Fig. 8A](#) bis [Fig. 8D](#) ist jeder kreisförmige Knoten mit demselben alphanumerischen Zeichen und demselben Bezugszeichen derselbe Knoten. Die kreisförmigen Knoten werden zum Bilden von Verbindungen zwischen den verschiedenen Prozessen bei dem Verfahren der [Fig. 8A](#) bis [Fig. 8D](#) verwendet, ohne die Figuren mit einer Anzahl von Verbindungslinien zu überhäufen.

[0193] Das Client-Modul **702** wartet bei der Prüfung **820** auf ein Drücken einer Taste darauf, dass der Anwender eine Taste auf einem Tastenfeld **715** des Zellulartelefons **700** drückt. Bei diesem Ausführungsbeispiel ist angenommen, dass das Zellulartelefon **700** die Fähigkeit hat, zwei weiche Tasten, eine Taste zum Rollen nach oben, eine Taste zum Rollen nach unten, eine vorherige Taste, eine nächste Taste und Tasten Null bis 9 zu unterstützen, die in der standardmäßigen Telefontastenfeldkonfiguration konfiguriert sind. Angesichts der folgenden Offenbarung kann ein Fachmann auf dem Gebiet dann, wenn eine oder mehrere dieser Tasten nicht vorhanden sind, das Verfahren für die bestimmte Konfiguration des Zellulartelefonastufenfelds oder eines anderen Zweiwege-Datenkommunikationsvorrichtungstastenfelds abändern. Wenn beispielsweise das Zellulartelefon einen Heimschlüssel enthielt, würde die nachfolgend vollständiger beschriebene Verarbeitung für ein Drücken einer Taste eine Prüfung enthalten, die erfasste, wenn die Heimtaste gedrückt wurde, und infolge davon zu einem Bekommen eines Heim-URL-Prozesses **801** transferieren würde.

[0194] Kurz gesagt identifizieren die Prozesse in den [Fig. 8B](#) bis [Fig. 8C](#) die durch den Anwender gedrückte Taste, identifizieren sie die erforderliche Aktion und transferieren dann zu einem Prozess, der die erforderliche Aktion implementiert. Insbesondere dann, wenn eine Taste auf dem Tastenfeld gedrückt ist, speichert ein Tastenfeldmodul **711** einen Identifizierer für die Taste in einem Arbeitsspeicher **716** und benachrichtigt das Client-Modul **702** über das Drücken der Taste. Auf einen Empfang der Benachrichtigung vom Tastenfeldmodul **711** hin liest das Client-Modul **702** die Speicherstelle im Arbeitsspeicher **716**, um die gedrückte Taste zu bestimmen, und transferiert eine Verarbeitung von einer Prüfung **820** auf ein Drücken einer Taste zu einer Prüfung **821** auf eine Rolltaste.

[0195] Bei der Prüfung **821** auf eine Rolltaste bestimmt das Client-Modul **702**, ob der Anwender eine der Rolltasten drückte. Wenn eine Rolltaste gedrückt wurde, transferiert eine Verarbeitung zum Einstellen eines Anzeigeprozesses **822**, und sonst zu einer Prüfung **823** auf eine Anzeigekarte.

[0196] Beim Einstellen eines Anzeigeprozesses **822** bestimmt das Client-Modul **702**, welche der Tasten für ein Rollen nach oben oder ein Rollen nach unten gedrückt wurde. Das Client-Modul **702** sendet dann Information zum Anzeigemodul **712**, so dass die aktuelle Anzeige entweder um eine Zeile nach oben gerollt oder um eine Zeile nach unten gerollt wird. Wenn die Rolltaste die Anzeige über eine Grenze der aktuellen Karte hinausgehend bewegen würde, wird das Drücken der Rolltaste beim Einstellen des Anzeigeprozesses **822** ignoriert.

[0197] In Reaktion auf die Information vom Client-Modul **702** stellt das Anzeigemodul **712** die Bildschirmanzeige auf dem Anzeigebildschirm **705** ein. Das Client-Modul **702** transferiert eine Verarbeitung von einem Einstellen des Anzeigeprozesses **822** zu einer Prüfung **820** auf ein Drücken einer Taste durch den Knoten **813**.

[0198] Wenn keine Rolltaste gedrückt wurde, wird eine Verarbeitung durch die Prüfung **821** auf eine Rolltaste zu einer Prüfung **823** auf einer Anzeigekarte geführt. Das Client-Modul **702** nimmt eine Aktion vor, die von dem bestimmten Typ der Karte abhängt, die gegenwärtig auf dem Anzeigebildschirm **705** angezeigt wird. Wenn die aktuelle Karte eine Anzeigekarte ist, läuft das Client-Modul **702** durch die Prüfung **823** auf eine Anzeigekarte zu einer Prüfung **828** auf eine weiche Taste, und transferiert sonst zu einer Prüfung **824** auf eine Auswahlkarte.

[0199] Nimmt man für den Moment an, dass die aktuelle Karte weder eine Anzeigekarte noch eine Auswahlkarte ist, muss die aktuelle Karte eine Eingabekarte sein, weil bei diesem Ausführungsbeispiel nur drei Kartentypen definiert sind. Somit prüft das Client-Modul **702** nicht auf eine Eingabekarte. Vielmehr bestimmt eine Prü-

fung **826** auf eine Datentaste, ob eine gültige Datentaste gedrückt wurde. Bei diesem Ausführungsbeispiel sind die Datentasten null bis neun auf dem Tastenfeld und die #-Taste. Bei anderen Ausführungsbeispielen könnten andere Kombinationen von Tasten als Datentasten definiert sein. Wenn die gedrückte Taste eine der Datentasten war, transferiert die Prüfung **826** auf eine Datentaste zum Verarbeiten einer Dateneingabe **827** und transferiert sonst zu einer Prüfung **828** auf eine weiche Taste.

[0200] Beim Verarbeiten einer Dateneingabe **827** weiß das Client-Modul **702**, ob der Vorhersagetexteingabeprozess eingeschaltet ist, weil einer der Parameter auf der Eingabekarte spezifiziert, ob der Vorhersagetexteingabeprozess zu verwenden ist, wie es im Anhang I beschrieben ist, welcher hierin durch Bezugnahme in seiner Gesamtheit enthalten ist.

[0201] Wenn der Vorhersagetexteingabeprozess nicht eingeschaltet ist, gibt das Client-Modul **702** beim Verarbeiten einer Dateneingabe **827** den Wert der gedrückten Taste in einen Texteingabepuffer im Arbeitsspeicher **716** bei der geeigneten Stelle ein. Ebenso sendet das Client-Modul **702** Information zum Anzeigemodul **712**, so dass der Wert der gedrückten Taste bei der geeigneten Stelle auf dem Anzeigebildschirm **705** durch das Anzeigemodul **712** angezeigt wird.

[0202] Wenn der Vorhersagetexteingabeprozess eingeschaltet ist, verwendet das Client-Modul **702** den neuen Vorhersagetexteingabeprozess beim Verarbeiten einer Dateneingabe **827**, wie es nachfolgend in Bezug auf die [Fig. 9](#), [Fig. 10A](#) bis [Fig. 10T](#) und [Fig. 11](#) vollständiger beschrieben ist, um den Buchstaben zu bestimmen, der aus der Gruppe von Buchstaben auszuwählen ist, die zu der gedrückten Taste gehören. Nachdem der Vorhersagetexteingabeprozess den geeigneten Buchstaben bestimmt, wird ein Wert, der den Buchstaben darstellt, bei der geeigneten Stelle im Textpuffer im Arbeitsspeicher **716** gespeichert. Ebenso sendet das Client-Modul **702** Information zum Anzeigemodul **712**, so dass der Buchstabe bei der geeigneten Stelle auf dem Anzeigebildschirm **705** angezeigt wird. Auf eine Beendigung eines Verarbeitens einer Dateneingabe **827** hin transferiert das Client-Modul **702** eine Verarbeitung durch den Knoten **813** zu einer Prüfung **820** auf ein Drücken einer Taste.

[0203] Die vorherige Beschreibung nahm an, dass die aktuelle Karte eine Eingabekarte war, aber dann, wenn die aktuelle Karte ein Auswahlkarte ist, würde die Prüfung **824** auf eine Auswahlkarte zu einer Prüfung **826** auf eine Auswahlkarte transferiert. Beim Erzeugen eines Anzeige Prozesses **804** für die Auswahlkarte wird jede der Auswahlen gemäß Information auf der Auswahlkarte bezeichnet, und einige oder alle der Auswahlen werden auf dem Anzeigebildschirm **705** angezeigt. Somit bestimmt die Prüfung **826** auf eine Auswahlkarte, ob die gedrückte Taste einer der Auswahlen entspricht. Wenn die gedrückte Taste eine der Auswahlen ist, sendet das Client-Modul **702** bei einem Ausführungsbeispiel Information zum Anzeigemodul **712**, um die ausgewählte Auswahl anzuzeigen. Das Client-Modul **702** transferiert auch von der Prüfung **826** auf eine Auswahlkarte durch den Knoten **831** zu einem Prozess **850** für ein Speichern eines Identifizierers ([Fig. 8D](#)), der nachfolgend vollständiger beschrieben wird. Gegensätzlich dazu transferiert die Prüfung **826** auf eine Auswahlkarte dann, wenn die gedrückte Taste keine der Auswahlen ist, zu einer Prüfung **828** auf eine weiche Taste.

[0204] Weiche Tasten können sowohl für einen Posten als Gesamtes als auch pro Karte spezifiziert werden, d.h. eine physikalische Taste auf dem Tastenfeld wird als weiche Taste spezifiziert, wie es vollständiger im Anhang I beschrieben ist. Jede Spezifikation einer weichen Taste enthält einen Identifizierer, der die Aktion definiert, die dann vorzunehmen ist, wenn die weiche Taste gedrückt wird.

[0205] Wenn eine weiche Taste für einen Posten spezifiziert ist, bleibt die weiche Taste für den gesamten Posten effektiv. Jedoch dann, wenn eine weiche Taste für eine Karte spezifiziert wird, setzt sich die Spezifikation einer weichen Taste für eine Karte temporär über die entsprechende Spezifikation einer weichen Taste für einen Posten hinweg, d.h. die Spezifikation einer weichen Taste für einen Posten für diese physikalische Taste wie die Spezifikation einer weichen Taste für eine Karte, während die Karte sichtbar ist, d.h. auf dem Anzeigebildschirm **705** angezeigt wird. Dieses Hinwegsetzen wird unabhängig für die zwei weichen Tasten durchgeführt. Somit transferiert die Prüfung **828** auf eine weiche Taste eine Verarbeitung zu einer Prüfung **829** auf eine erste weiche Taste, wenn die gedrückte Taste eine der zwei möglichen physikalischen weichen Tasten ist. Gegensätzlich transferiert die Prüfung **828** auf eine weiche Taste eine Verarbeitung zu einer Prüfung **840** auf eine nächste Taste ([Fig. 8C](#)), wenn keine der zwei möglichen physikalischen weichen Tasten durch den Anwender gedrückt wird.

[0206] Bei der Prüfung **829** auf eine erste weiche Taste bestimmt das Client-Modul **702**, ob die gedrückte Taste der ersten weichen Taste entspricht. Wenn die gedrückte Taste die erste weiche Taste ist, führt bzw. übergibt die Prüfung **829** den aktiven Identifizieren für die erste weiche Taste zu einem Prozess **850** zum Speichern

eines Identifizierers über einen Knoten **831**. Gegensätzlich dazu transferiert eine Verarbeitung dann, wenn die gedrückte Taste nicht die erste weiche Taste ist, von der Prüfung **829** zu einer Prüfung **830** auf eine zweite weiche Taste.

[0207] Wenn die gedrückte Taste die zweite weiche Taste ist, führt die Prüfung **830** den aktiven Identifizierer für die zweite weiche Taste zu einem Prozess **850** zum Speichern eines Identifizierers über einen Knoten **831**. Gegensätzlich dazu transferiert eine Verarbeitung dann, wenn die gedrückte Taste nicht die zweite weiche Taste ist, z.B. eine physikalische Taste, die als weiche Taste definiert sein kann, gedrückt wurde, aber weder der aktuelle Posten noch die aktuelle Karte eine weiche Taste für diese physikalische Taste definiert, von der Prüfung **830** zu einer Prüfung **820** auf ein Drücken einer Taste über den Knoten **813**.

[0208] Wenn ein Drücken zu einer Prüfung **840** auf eine nächste Taste transferiert, bestimmt das Client-Modul **702**, ob die gedrückte Taste die nächste Taste war. Wenn die nächste Taste gedrückt wurde, transferiert eine Verarbeitung zu einer Prüfung **841** auf eine Anzeigekarte, und sonst zu einer Prüfung **846** auf eine vorherige Taste.

[0209] Wenn eine Anzeigekarte die aktuelle Karte ist, wird die nächste Taste zum Bewegen zu einer anderen Karte in einem Posten verwendet, oder alternativ dazu zu einem anderen Posten. Somit transferiert die Prüfung **841** auf eine Anzeigekarte eine Verarbeitung zu einer Prüfung **842** auf eine letzte Karte, wenn eine Anzeigekarte die aktuelle Karte ist, und sonst zu einer Prüfung **843** auf eine Eingabekarte.

[0210] Die Prüfung **842** auf eine letzte Karte bestimmt, ob die aktuelle Karte die letzte Karte im Posten ist. Wenn die aktuelle Karte nicht die letzte Karte im Posten ist, transferiert die Prüfung **842** auf eine letzte Karte eine Verarbeitung zu einem Prozess **845** zum Lesen der nächsten Karte, welcher infolge davon die nächste Karte im Posten liest und über einen Knoten **812** zu einem Prozess **809** zum Erzeugen einer Anzeige transferiert.

[0211] Wenn die aktuelle Anzeigekarte die letzte Karte im Posten ist, enthält der Posten einen Identifizierer, der die Stelle spezifiziert, zu welcher von der letzten Karte zu transferieren ist. Dieser Identifizierer kann beispielsweise eine URL zu einem anderen Posten, zu einem gemeinsamen Gateway-Schnittstellenprogramm oder eine Adresse für eine Karte innerhalb des aktuellen Postens sein. Somit transferiert die Prüfung **842** auf eine letzte Karte über einen Knoten **831** zu einem Prozess **850** zum Speichern eines Identifizierers, wenn die aktuelle Anzeigekarte die letzte Karte im Posten ist.

[0212] Wenn die aktuelle Karte keine Anzeigekarte ist, sondern eine Eingabekarte ist, transferiert die Prüfung **841** auf eine Anzeigekarte zu einer Prüfung **843** auf eine Eingabekarte. Bei diesem Ausführungsbeispiel ist die nächste Taste die vorbestimmte Taste, die zum Anzeigen verwendet wird, dass alle Daten für eine Eingabe auf einer Eingabekarte eingegeben worden sind. Somit transferiert die Prüfung **843** auf eine Eingabekarte dann, wenn die aktuelle Karte eine Eingabekarte ist, eine Verarbeitung zu einem Prozess **844** zum Speichern von Daten.

[0213] Der Prozess **844** zum Speichern von Daten speichert die eingegebenen Daten bei einer geeigneten Stelle im Speicher, die in der aktuellen Eingabekarte spezifiziert ist. Typischerweise werden die Daten als Argument mit einer URL kombiniert und gespeichert. Auf eine Beendigung hin transferiert der Prozess **844** zum Speichern von Daten über einen Knoten **810** zu einem Prozess **802** zum Erzeugen einer HTTP-Anforderung ([Fig. 8A](#)).

[0214] Wenn die nächste Taste gedrückt wird, ist dann, wenn die aktuelle Karte weder eine Anzeigekarte noch eine Eingabekarte ist, die aktuelle Karte eine Auswahlkarte. Jedoch erfordert, wie es oben angezeigt ist, bei diesem Ausführungsbeispiel das Client-Modul **702**, dass der Anwender eine Auswahl durchführt, und lässt nicht eine Verwendung der nächsten Taste zu. Folglich transferiert die Prüfung **843** auf eine Eingabekarte dann, wenn die aktuelle Karte nicht eine Eingabekarte ist, eine Verarbeitung über einen Knoten **813** zu einer Prüfung **820** auf ein Drücken einer Taste.

[0215] Die vorherige Diskussion nahm an, dass die nächste Taste gedrückt wurde und somit die Prüfung **840** auf eine nächste Taste eine Verarbeitung zu einer Prüfung **841** auf eine Anzeigekarte transferierte. Jedoch dann, wenn die nächste Taste nicht gedrückt wurde, transferiert die Prüfung **840** auf eine nächste Taste eine Verarbeitung zu einer Prüfung **846** auf eine vorherige Taste. Wenn die vorherige Taste gedrückt wurde, transferiert die Prüfung **846** zu einer Prüfung **847** auf eine erste Karte und bringt sonst eine Verarbeitung zurück zu einer Prüfung **820** auf ein Drücken einer Taste.

[0216] Die Prüfung **847** auf eine erste Karte bestimmt, ob die aktuelle Karte die erste Karte eines Postens ist. Wenn die aktuelle Karte nicht die erste Karte ist, transferiert eine Verarbeitung von der Prüfung **847** auf eine erste Karte zu einem Prozess **849** zum Lesen einer vorherigen Karte, welcher infolge davon die vorherige Karte liest und zu einem Prozess **809** zum Erzeugen einer Anzeige über einen Knoten **813** transferiert. Gegenätzlich dazu transferiert eine Verarbeitung dann, wenn die aktuelle Karte die erste Karte ist, zu einer Prüfung **848** auf einen Heimposten.

[0217] Wenn die aktuelle Karte die erste Karte im Heimposten ist, gibt es keine vorherige Karte, und somit transferiert die Prüfung auf einen Heimposten eine Verarbeitung zu einer Prüfung **820** auf ein Drücken einer Taste über einen Knoten **813**, und somit wird ein Drücken einer vorherigen Taste ignoriert. Wenn der aktuelle Posten nicht der Heimposten ist, liest die Prüfung **848** auf einen Heimposten den Identifizierer für den vorherigen Posten aus und transferiert über einen Knoten **831** zu einem Prozess **850** zum Speichern eines Identifizierers.

[0218] Der Prozess **850** zum Speichern eines Identifizierers wird über den Knoten **831** von mehreren unterschiedlichen Stellen aus erreicht. Die Operationen im Prozess **850** zum Speichern eines Identifizierers sind ungeachtet des bestimmten Prozesses dieselben, der zu dem Prozess **850** transferiert. In jedem Fall wird ein Identifizierer zu dem Prozess **850** zum Speichern eines Identifizierers übergeben bzw. geführt und sichert der Prozess **850** den Identifizierer im Arbeitsspeicher **716**. Der Identifizierer kann beispielsweise ein Zeiger zu einer anderen Stelle in der aktuellen Karte, eine Adresse einer anderen Karte im aktuellen Posten, eine URL zu einem Posten, der im Arbeitsspeicher **716** gespeichert ist, eine URL zu einem TIL-Posten in den TIL-Posten **760** auf dem Computer **743** oder vielleicht eine URL zu einem gemeinsamen Gateway-Schnittstellenprogramm in CGI-Programmen **761** auf dem Computer **743** sein. Somit prüft der Prozess **800** den gespeicherten Identifizierer, um die erforderliche Aktion zu bestimmen.

[0219] Insbesondere bestimmt das Client-Modul **702** im Identifizierer zu der Prüfung **851** auf einen aktuellen Posten, ob der Identifizierer zu einer Karte im aktuellen Posten ist. Wenn der Identifizierer zum aktuellen Posten zeigt, transferiert die Prüfung **851** eine Verarbeitung zu einem Prozess **852** zum Auslesen von Daten, und sonst zu einer URL zu einer Prüfung **853** auf einen lokalen Posten.

[0220] Im Prozess **852** zum Auslesen von Daten liest das Client-Modul **702** die Information aus, die bei der Stelle gespeichert ist, die durch den Identifizierer angezeigt ist, aus dem Arbeitsspeicher **716** und verarbeitet die Information. Der Prozess **852** zum Auslesen von Daten transferiert über einen Knoten **812** zu einem Prozess **809** zum Erzeugen einer Anzeige ([Fig. 8A](#)), der oben beschrieben wurde.

[0221] Die Prüfung **853** auf eine URL zu einem lokalen Posten bestimmt, ob der Identifizierer eine URL zu einem Posten ist, der im Arbeitsspeicher **716** gespeichert ist, z.B. im Cache gespeichert ist. Wenn der Posten lokal gespeichert ist, transferiert die Prüfung **853** zu einem Prozess **854** zum Auslesen eines lokalen Postens, welcher infolge davon den lokalen Posten in die Speicherstelle für den aktuellen Posten bewegt. Der Prozess **854** zum Auslesen eines lokalen Postens transferiert eine Verarbeitung über einen Knoten **811** zu einem Prozess **808** zum Verarbeiten einer ersten Karte ([Fig. 8A](#)), der oben beschrieben wurde.

[0222] Wenn der Identifizierer weder zu einer Stelle im aktuellen Posten ist, noch zu einem lokalen Posten, ist der Identifizierer eine URL zu einem Objekt auf dem Computer **743**. Somit bringt die Prüfung **853** in diesem Fall eine Verarbeitung zu einem Prozess **802** zum Erzeugen eines HTTP-Anforderung über einen Knoten **810** zurück.

[0223] Der Prozess **800** dauert so lange an, wie der Anwender damit fortfährt, einzugeben und die gelieferte Information zu verarbeiten. Bei diesem Ausführungsbeispiel wird der Prozess **800** beispielsweise entweder durch ein Abschalten des Zellulartelefons **700** durch den Anwender, ein Auswählen einer Auswahl- oder Eingabekarte, das Operationen des Client-Moduls **702** nicht fortführt, oder ein Inaktivbleiben für eine Zeit, die länger als eine Auszeitperiode ist, so dass sich das Client-Modul **702** selbst herunterfährt, beendet.

[0224] Um die Operationen im Prozess **800** weiter darzustellen, soll das folgende Beispiel betrachtet werden, das zu dem Client-Modul **702** als TIL-Posten in Reaktion auf eine durch den Prozess **802** erzeugte HTTP-Anforderung zurückgebracht ist. Der Lesbarkeit halber präsentiert eine Tabelle 2 den Posten in PIDL. Bei diesem Beispiel sind alle Auswahlen für Anwendungen auf demselben Server. Jedoch könnte bei einem anderen Ausführungsbeispiel jede URL irgendeine erwünschte Kombination von Servern adressieren.

TABELLE 2

BEISPIEL EINES PIDL-AUSWAHLPOSTENS

<PIDL>

<AUSWAHL>

<CE URL=http://www.libris.com/airnet/nnn>News

<CE URL=http://www.libris.com/airnet/nnn>Wheater

<CE URL=http://www.libris.com/airnet/nnn>Sports

</AUSWAHL>

</PIDL>

[0225] Im Prozess **808** zum Verarbeiten einer ersten Karte interpretiert das Client-Modul **702** die Information in der Tabelle 2 und transferiert zu einem Prozess **809** zum Erzeugen einer Anzeige. Im Prozess **809** zum Erzeugen einer Anzeige sendet das Client-Modul **702** Information zum Anzeigemodul **712**, so dass dem Anwender eine Liste von drei Auswahlen auf dem Anzeigebildschirm **705** präsentiert wird, d.h. eine Anwenderschnittstelle für die Auswahlkarte erzeugt wird:

1. Nachrichten
2. Wetter
3. Sport

[0226] Der Prozess **809** zum Erzeugen einer Anzeige ([Fig. 8A](#)) transferiert zu einer Prüfung **820** auf ein Drücken einer Taste ([Fig. 8B](#)). Wenn der Anwender die Taste Zwei auf dem Tastenfeld **715** drückt, transferiert die Prüfung **820** auf ein Drücken einer Taste über die Prüfung **821** zu einer Prüfung **823** auf eine Anzeigekarte.

[0227] Da die aktuelle Karte eine Auswahlkarte ist, transferiert die Prüfung **823** eine Verarbeitung zu einer Prüfung **824** auf eine Auswahlkarte, welche infolge davon zu einer Prüfung **826** auf eine Auswahlkarte transferiert. Da die Taste Zwei gedrückt wurde und diese Taste eine Auswahlkarte ist, transferiert die Prüfung **826** eine Verarbeitung zu einem Prozess **850** zum Speichern eines Identifizierers ([Fig. 8D](#)). Im Prozess **850** speichert das Client-Modul **702** die URL entsprechend zwei, d.h.
URL = http://www.libris.com/airnet/www

im Arbeitsspeicher **716**.

[0228] Da diese URL zu einem Objekt auf dem Computer **743** ist, transferiert eine Verarbeitung über die Prüfungen **851** und **853** zu einem Prozess **802** zum Erzeugen einer HTTP-Anforderung, welche infolge davon die Anforderung erzeugt. Wenn die HTTP-Anforderung zum Server **749** gesendet wird, wie es oben in Bezug auf den Prozess **804** beschrieben ist, liest der Server **749** in einem Prozess **862** zum Bedienen einer Anforderung einen Posten www von TIL-Posten **760** aus. Ein Beispiel für den Posten ist in einer Tabelle 3 angegeben. Wiederum der Lesbarkeit halber wird der Posten hierin in PIDL präsentiert.

TABELLE 3

BEISPIEL EINES ZWEITEN PIDL-AUSWAHLPOSTENS

```

<PIDL>
<AUSWAHL>
<CE URL=http://www.libris.com/airnet/www-1>World
<CE URL=http://www.libris.com/airnet
    /www-2>National
<CE URL=http://www.libris.com/airnet/www-3>State
<CE URL=http://www.libris.com/airnet/www-4>Local
</AUSWAHL>
</PIDL>

```

[0229] Der Posten in der Tabelle 3 wird zum Zellulartelefon **700** gesendet und im Speicher **716** gespeichert, wie es oben in Bezug auf den Prozess **806** beschrieben ist. Die Auswahlkarte wird in einem Prozess **808** verarbeitet und in einem Prozess **809** angezeigt. Als Ergebnis des Prozesses **809** wird dem Anwender eine Liste von Auswahlen präsentiert:

1. Welt
2. National
3. Staat
4. Lokal

[0230] Wenn der Anwender eine andere Auswahl vornimmt, wird dieselbe Sequenz von Prozessen, wie sie oben für die erste Auswahlkarte beschrieben ist, durch das Client-Modul **702** ausgeführt und wird eine weitere URL gespeichert, die zu einem Programm auf dem Server **749** zeigt, der die erwünschte Wetterinformation ausliest und einen Posten mit dieser Information erzeugt. Dieser Posten wird zum Zellulartelefon **700** transferiert und angezeigt.

[0231] Wie es oben beschrieben ist, erreicht der Client-Prozess **702** dann, wenn die aktuelle Karte eine Eingabekarte ist und eine Taste gedrückt wird, eine Prüfung **826** auf ein Drücken einer Datentaste ([Fig. 8B](#)). Wenn die gedrückte Taste eine gültige Datentaste ist, transferiert die Prüfung **826** zu einem Prozess **827** zum Verarbeiten einer Dateneingabe.

[0232] Bei einem Ausführungsbeispiel verwendet der Prozess **827** zum Verarbeiten einer Dateneingabe einen neuen Vorhersagetexteingabeprozess für eine Texteingabe. Es soll erneut die Aufmerksamkeit darauf gelenkt werden, dass die Tasten auf einem typischen Telefontastenfeld mit sowohl einer Zahl als auch zwei oder drei Buchstaben bezeichnet sind. Beispielsweise ist die Taste Zwei auch mit abc bezeichnet. Dies führt zu einer gewissen Vieldeutigkeit, wenn das Telefontastenfeld zum Eingeben von Text verwendet wird. Versucht der Anwender a, b oder c einzugeben, wenn die Taste Zwei gedrückt wird?

[0233] Bei einem Verfahren nach dem Stand der Technik waren zwei Tastenhübe erforderlich, um einen jeweiligen Buchstaben von Text einzugeben. Der erste Tastenhub identifizierte die erste Taste und der zweite Tastenhub identifizierte den spezifischen Buchstaben, der auf der ersten Taste erwünscht ist. Beispielsweise würde der Anwender zum Eingeben des Buchstabens s zuerst die Taste Sieben drücken, die mit dem Buchstaben p, r und s bezeichnet ist. Als nächstes würde der Anwender die Taste Drei drücken, um den Buchstaben s auszuwählen. Während dieses Verfahren für kurze Sequenzen gut arbeiten kann, die aus nur drei der vier Buchstaben bestehen, arbeitet das Verfahren für englischen Text nicht gut. Wenn beispielsweise der Anwender bereits th eingegeben hat und dann die Taste Drei drückt, die mit den Buchstaben d, e und f bezeichnet ist, ist der erwünschte nächste Buchstabe nahezu immer der Buchstabe e. Daher ist ein Veranlassen, dass der Anwender die Taste Zwei drückt, ein zusätzlicher und unnötiger Schritt.

[0234] Das Client-Modul **702** verwendet einen neuen Vorhersagetexteingabeprozess zum Reduzieren der Anzahl von Tastenhüben, die zum Eingeben von Text unter Verwendung eines Telefontastenfelds erforderlich sind, oder irgendeines ähnlichen Tastenfelds. Unter Verwendung dieses Prozesses genügt in den meisten Fällen ein einziger Tastenhub zum Eingeben eines einzelnen Buchstabens.

[0235] Während dieser Prozess in Bezug auf ein Telefontastenfeld beschrieben ist, sind die Prinzipien nicht auf nur ein Telefontastenfeld beschränkt. Im Allgemeinen kann der nachfolgend vollständiger beschriebene Prozess auf irgendein Tastenfeld erweitert werden, wo eine einzige Taste dazu verwendet wird, zwei oder mehr Buchstaben einzugeben. Weiterhin ist der Prozess nicht nur auf Buchstaben beschränkt, sondern ist vielmehr auf irgendein Tastenfeld anwendbar, wo eine einzige Taste dazu verwendet wird, zwei oder mehr Zeichen darzustellen. Angesichts der folgenden Offenbarung können Fachleute auf dem Gebiet die Prinzipien des Vorhersagetexteingabeprozesses in einer weiten Vielfalt von Anwendungen verwenden.

[0236] Das System für eine Vorhersagetexteingabe enthält ein Vorhersagetexteingabemodul **901**, das bei diesem Ausführungsbeispiel im Client-Modul **702** enthalten ist, ein Tastaturmodul **711** und eine Buchstabenhäufigkeitstabelle **902**, die in den Speicher **716** geladen wird, wenn das Client-Modul **702** aktiviert wird. Das Vorhersagetexteingabemodul **901** wird in einem Prozess **827** zum Verarbeiten einer Dateneingabe verwendet, wenn es durch die aktuelle Eingabekarte spezifiziert ist. Das Vorhersagetexteingabemodul **901** führt Programmbaupuffermanagementprozesse durch, die einem Fachmann auf dem Gebiet bekannt sind und somit nicht weiter beschrieben werden, um ein Wegführen von dem Prozess zu vermeiden.

[0237] Das Vorhersagetexteingabemodul **901** speichert eine Buchstabeneingabe für jeden Buchstaben, der in einen Textpuffer **903** eingegeben ist, im Speicher **716**. Bei diesem Ausführungsbeispiel sind die Buchstaben Q und Z der Taste Eins zugeordnet, und die Taste Null wird zum Eingeben eines Raums, einer Periode und eines Kommas, verwendet, d.h. die Taste Null sorgt für die Interpunktion. Jedoch sind diese Zuordnungen nur illustrativ und sollen die Erfindung nicht auf dieses bestimmte Ausführungsbeispiel beschränken.

[0238] Der erste eingegebene Buchstabe wird am linken Ende des Puffers platziert und jeder zusätzliche Buchstabe wird im äußersten linken nicht verwendeten Raum im Puffer **903** platziert. Somit ist der letzte Buchstabe, der in den Textpuffer **903** eingegeben wird, das Zeichen am weitesten rechts. Die Buchstabenhäufigkeitstabelle **902**, die manchmal Tabelle für Vorhersagebuchstabeneingaben genannt wird, ist eine Nachschantabelle, in welcher jede Eingabe in der Nachschantabelle durch drei Indizes adressiert ist. Die ersten zwei Indizes stellen die zwei zuletzt eingegebenen Buchstaben im Textpuffer **903** dar und der dritte Index stellt die Taste dar, die gedrückt wurde. Jede Vorhersagebuchstabeneingabe, die in der Buchstabenhäufigkeitstabelle **902** gespeichert ist, definiert, welcher der Buchstaben, die zu der gedrückten Taste gehören, die zu verwenden ist, die vorherigen zwei Buchstaben ergaben. Beispielsweise bringt deshalb, weil the eine allgemein auftretende Kette ist, die Eingabe in der Tabelle **902**, die durch (t, h, 3) adressiert ist, e zurück, oder genauer wird die Vorhersagebuchstabeneingabe 2 zurückgebracht, um anzuzeigen, dass der zweite Buchstabe der Gruppe von Buchstaben d, e und f, die zu der Taste Drei gehören, der vorhergesagte Buchstabe ist. Natürlich könnte die Buchstabenhäufigkeitstabelle **902** abgeändert werden, um mehr als einen einzigen Buchstaben zurückzubringen.

[0239] Bei diesem Ausführungsbeispiel wurde die Buchstabenhäufigkeitstabelle **902** unter Verwendung einer Sammlung von e-Mails empirisch erzeugt. Der Anhang II ist eine Computerprogrammauflistung, die zum Erzeugen einer Buchstabenhäufigkeitstabelle **902** verwendet wurde, die in den [Fig. 10A](#) bis [Fig. 10T](#) dargestellt ist. Kurz gesagt implementiert das Computerprogramm einen Prozess, der sequenziell durch die Daten schreitet, die geliefert sind, und (i) für jeden möglichen einzelnen Buchstaben den wahrscheinlichsten Buchstaben bestimmt, der für jede Taste auf dem Tastenfeld folgt; und (ii) für jede mögliche Kombination von zwei Buchstaben den wahrscheinlichsten Buchstaben bestimmt, der folgt, und zwar für jede Taste auf dem Tastenfeld. Bei diesem Ausführungsbeispiel ist der wahrscheinlichste Buchstabe der Buchstabe mit der größten Häufigkeit nach dem einzelnen Buchstaben. Gleichmaßen ist der wahrscheinlichste Buchstabe der Buchstabe mit der größten Häufigkeit nach der Kombination von zwei Buchstaben. Wenn es einen Zusammenhang bezüglich der Häufigkeit gibt, wird der erste Buchstabe ausgewählt, der zu einer Taste gehört. Natürlich könnten andere Maßnahmen einer Wahrscheinlichkeit dazu verwendet werden, die Eingaben in die Tabelle **902** zu erzeugen.

[0240] Somit ist in den [Fig. 10A](#) bis [Fig. 10T](#) die erste der zehn Spalten, d.h. die Spalte am weitesten links, die Sequenz aus zwei Buchstaben und ist die erste Zeile, d.h. die oberste Zeile, die Tasten auf dem Tastenfeld, die zum Eingeben von Text verwendet werden. Eine Kombination einer Eingabe in der ersten Spalte und einer Taste in der obersten Zeile wird zum Auswählen der vorhergesagten Texteingabe verwendet. Somit erscheint unter Verwendung des Beispiels von th diese Sequenz der Taste Zwei in der ersten Spalte der [Fig. 10Q](#). Wenn die Taste Drei gedrückt wird, wird der Buchstabe in der Zeile mit th als die erste Eingabe und in der Spalte mit Drei als die erste Eingabe, d.h. e, ausgelesen. Alternativ dazu wird dann, wenn die Taste Vier gedrückt wird, der Buchstabe i aus der Tabelle ausgelesen.

[0241] Bei diesem Ausführungsbeispiel ist die Tabelle **902** ein Puffer von Zahlen aus zwei Bits. Jede Zwei-

bit-Zahl hat einen Wert im Bereich von Null bis Drei und die Zweibit-Zahl stellt einen vorhergesagten Buchstaben für die gedrückte Taste dar. Somit stellt für eine Taste Zwei, die mit den Buchstaben A, B und C bezeichnet ist, eine Null A dar, eine Eins B dar und eine Zwei C dar. Im Allgemeinen wird die Anzahl von verwendeten Bits durch die Taste bestimmt, die die maximale Anzahl von Zeichen darstellt. Bei diesem Ausführungsbeispiel ist die maximale Anzahl von Zeichen, die durch eine Taste dargestellt werden, Drei. Die Anzahl von Speicherbits, die erforderlich sind, ist eine ganze Zahl S, wobei S die kleinste Zahl ist, so dass 2^{**S} größer als maximale Anzahl von Zeichen oder gleich dieser ist, die durch eine Taste dargestellt werden.

[0242] Bei diesem Ausführungsbeispiel werden drei Indizes i0, i1 und i2 dazu verwendet, um einen Tabellenindex zu erzeugen, der infolge davon dazu verwendet wird, auf eine bestimmte Vorhersagebuchstabeneingabe in der Tabelle **902** aus zwei Bit-Zahlen zuzugreifen. Jeder Buchstabe wird als Zahl dargestellt, d.h. eine Buchstabeneingabe, wobei der Buchstabe A Null ist, wobei der Buchstabe B Eins ist, wobei der Buchstabe C Zwei ist, usw., wobei der Buchstabe Z Fünfundzwanzig ist. Ein Raumelement bzw. Blank-Element ist einem Raumelementwert von Sechszwanzig zugeordnet. Somit gibt es bei diesem Ausführungsbeispiel siebenundzwanzig mögliche Zeichen.

[0243] Auf die anfängliche Eingabe zum Prozess **1100** hin ([Fig. 11](#)), wurden Buchstabenindizes i0, i1 und i2 bei der anfänglichen Verarbeitung der Eingabekarte auf Sechszwanzig eingestellt, um anzuzeigen, dass der Textpuffer leer ist. Ebenso werden, wie es nachfolgend vollständiger erklärt wird, wenn ein jeweiliger Buchstabe von Text eingegeben wird, Buchstabenindizes i0 und i1 aktualisiert und im Speicher **716** gespeichert.

[0244] Jedoch ist bei einem weiteren Ausführungsbeispiel ein Prozess zum Initialisieren von Indizes die erste Operation im Vorhersagetexteingabeprozess **1100**. Bei diesem Ausführungsbeispiel werden für den ersten eingegebenen Buchstaben die Buchstabenindizes i0 und i1 auf Sechszwanzig eingestellt; für den zweiten eingegebenen Buchstaben wird ein Buchstabenindex i0 auf Sechszwanzig eingestellt und wird der Buchstabenindex i1 auf den Wert des Buchstabens im Textpuffer **903** eingestellt; und für alle Buchstaben, die nach den ersten beiden eingegeben werden, wird der Wert, der zu dem nächsten bis zu dem letzten Buchstaben im Textpuffer **903** gehört, dem Buchstabenindex i0 zugeordnet, und wird der Wert, der zu dem letzten Buchstaben im Textpuffer **903** gehört, dem Buchstabenindex i1 zugeordnet.

[0245] Eine Prüfung **1101** auf eine Interpunktions Taste bestimmt, ob die Taste Null gedrückt wurde, d.h. die Taste auswählte, eine Interpunkt darzustellen.

[0246] Wenn die Taste Null gedrückt wurde, transferiert eine Verarbeitung von der Prüfung **1101** zu einem Prozess **1102** zum Verarbeiten einer Interpunktions Eingabe. Der Prozess **1102** zum Verarbeiten einer Interpunktions Eingabe stellt den Index i2 auf Sechszwanzig ein und sendet den Raumelementwert zu einem Prozess **1108** zum Anzeigen eines Buchstabens. Der Prozess **1108** zum Anzeigen eines Buchstabens transferiert den Raumelementwert zum Anzeigemodul **712**, das infolge davon einen Raum bzw. ein Blank in der Texteingabe auf dem Anzeigebildschirm **705** antreibt. Dies beendet die Operation des Prozesses zum Verarbeiten einer Dateneingabe für ein Drücken der Taste Null, und somit kehrt eine Verarbeitung zu der Prüfung **820** auf ein Drücken einer Taste zurück.

[0247] Wenn die Taste Null nicht gedrückt wurde, transferiert eine Verarbeitung über die Prüfung **1101** auf eine Interpunktions Taste im Dateneingabeprozess **1100** zu einer Prüfung **1103** auf eine Taste von Eins bis Neun, d.h. zu einer Prüfung auf eine Dateneingabetaste. Wenn die gedrückte Taste irgendeine der Tasten Eins bis Neun war, transferiert die Prüfung **1103** zu einem Prozess **1104** zum Einstellen des Buchstabenindex und sonst zu einem Prozess **1109** zum Drehen der letzten Eingabe.

[0248] Im Prozess **1104** zum Einstellen des Buchstabenindex wird Eins von dem numerischen Wert der gedrückten Taste subtrahiert und wird der resultierende Wert dem Index i2 zugeordnet. Der Prozess **1104** zum Einstellen des Index transferiert zu einem Prozess **1105** zum Erzeugen eines Tabellenindex.

[0249] Der Prozess **1105** zum Erzeugen eines Tabellenindex kombiniert die Indizes i0, i1 und i2, um einen Tabellenindex zu erzeugen. Bei diesem Ausführungsbeispiel ist der Tabellenindex TABLE_INDEX definiert als:

$$\text{TABLE_INDEX} = (((i0 * 27) + i1) * 9) + i2$$

[0250] Auf eine Beendigung des Prozesses **1105** zum Erzeugen eines Tabellenindex hin liest ein Prozess **1106** zum Erzeugen einer Texteingabe den Zweibit-Wert in der Tabelle bei der Stelle, auf die durch den Tabellenindex TABLE_INDEX gezeigt wird, und wandelt den Zweibit-Wert in einen Buchstaben um, der durch den

Zweibit-Wert dargestellt wird.

[0251] Der Prozess **1106** zum Erzeugen einer Texteingabe transferiert zu einem Prozess **1107** zum Aktualisieren eines Index, welcher infolge davon den Wert des Buchstabenindex i1 als Buchstabenindex i0 speichert; den Wert des ausgelesenen Buchstabens im Buchstabenindex i1 speichert und den vorhergesagten Buchstaben im Textpuffer **903** speichert. Während dieser Schritt annimmt, dass die Buchstabenindizes i0 und i1 gespeichert werden und auf diese in einem Prozess **827** jedes Mal zugegriffen wird, und zwar abwechselnd, können die letzten zwei Buchstaben im Textpuffer **903** ausgelesen und jeweils den Indizes i0 und i1 zugeordnet werden, wie es oben beschrieben ist.

[0252] Der Prozess zum Aktualisieren eines Index **1107** transferiert zu einem Prozess **1108** zum Anzeigen eines Buchstabens. Der Prozess **1108** zum Anzeigen eines Buchstabens sendet Information zum Anzeigemodul **712**, das infolge davon den vorhergesagten Buchstaben auf dem Anzeigebildschirm **705** erzeugt.

[0253] Wenn die gedrückte Taste keine der Tasten Eins bis Neun ist, d.h. wenn sie keine Dateneingabetaste ist, transferiert eine Verarbeitung von der Prüfung **1103** zu einem Prozess **1109** zum Drehen der letzten Eingabe. Es soll daran erinnert werden, dass die Prüfung **826** auf eine Datentaste bestimmte, ob die gedrückte Taste eine der Tasten Null bis Neun oder die #-Taste war. Somit ist deshalb, weil die Prüfungen **1101** und **1103** bestimmten, dass die Tasten Null bis Neun nicht gedrückt wurden, die einzige übrig gebliebene gedrückte Taste die #-Taste, d.h. die Taste zum Drehen einer Eingabe, was anzeigt, dass der Anwender einen Buchstaben wünscht, der ein anderer als derjenige ist, der zuletzt in den Textpuffer **903** eingegeben wurde. Im Prozess **1109** zum Drehen der letzten Eingabe wird das letzte Zeichen, d.h. das Zeichen ganz rechts, im Textpuffer **903** durch das nächste Zeichen in der Gruppe von Zeichen ersetzt, die der zuletzt gedrückten Taste zugeordnet sind, bevor die #-Taste gedrückt wurde. Wiederum ist die Verwendung der #-Taste lediglich illustrativ und soll die Erfindung nicht auf die Verwendung von dieser bestimmten Taste zum Drehen einer Eingabe beschränken.

[0254] Wenn beispielsweise das letzte Zeichen im Textpuffer **903** ein t war und die #-Taste gedrückt wird, ändert der Prozess **1109** das t in u. Wenn die #-Taste wieder gedrückt wird, wird das u zu einem v geändert. Alternativ dazu ändert dann, wenn das letzte Zeichen im Textpuffer **903** ein u war und die #-Taste gedrückt wird, der Prozess **1109** das u zu einem v. Wenn das letzte Zeichen im Textpuffer **903** ein v war und die #-Taste gedrückt wird, ändert der Prozess **1109** das v zu einem t. Wenn der Index i1 gespeichert ist, wenn das letzte Zeichen im Textpuffer **903** gedreht wird, wird der Index i1 aktualisiert.

[0255] Eine Texteingabe in das Zellulartelefon **700** in anderen Sprachen oder Zusammenhängen kann durch Verwenden von anderen Buchstabenhäufigkeitstabellen unterstützt werden. Beispielsweise kann die Vorhersagetabelle für Klempner auf einem Text über Klempnerprozeduren basieren. Für Franzosen kann die Vorhersagetabelle auf französischem Text basieren. Ebenso könnten mehrere Buchstabenhäufigkeitstabellen im Zellulartelefon **700** gespeichert werden, oder selektiv zum Zellulartelefon **700** übertragen werden, und eine bestimmte Buchstabenhäufigkeitstabelle würde auf einer Eingabekarte ausgewählt werden.

[0256] Zusätzlich kann ein Eintrag in der Tabelle mehr als ein einzelner Buchstabe sein und somit sogar mehr Tastenhübe sichern. Wenn beispielsweise der Textpuffer sche enthält, dann könnte ein Tippen von 3 eher dule als nur d zurückbringen. Weiterhin kann dieses neue Verfahren einer Texteingabe mit einer anderen Vorrichtung als einem Zellulartelefon verwendet werden. Das Verfahren ist auf irgendeine Vorrichtung anwendbar, die mehrere Zeichen hat, die zu einer einzigen Taste auf einem Tastenfeld zugeordnet sind.

[0257] Beim obigen Ausführungsbeispiel wurde das englische Alphabet und ein Blank-Element als die Zeichengruppe verwendet. Somit ist die Zahl 27, die beim Definieren des Tabellenindex verwendet wird, genau die Anzahl N von Zeichen in der Gruppe. Gleichmaßen ist die Zahl 9, die beim Definieren des Tabellenindex verwendet wird, genau die Anzahl M von Tasten im Tastenfeld, die zwei oder mehr unterschiedliche Zeichen darstellen. Somit ist ein Vorhersagetexteintragsverfahren nicht auf Text beschränkt und ist direkt auf irgendein Tastenfeld anwendbar, bei welchem jede Taste eine Vielzahl von unterschiedlichen Zeichen darstellt.

[0258] Bei dem Ausführungsbeispiel der [Fig. 7](#), 8 und [Fig. 9](#) kommunizieren das Client-Modul **702** und das Servermodul **749** über das CDPD-Netzwerk **710**. Jedoch ist diese Architektur nur für die Prinzipien der Erfindung illustrativ und soll die Erfindung nicht auf die beschriebene bestimmte Architektur begrenzen. Das Client-Modul **702** und das Servermodul **729** können eine weite Vielfalt von Zweiwege-Datenkommunikationsverbindungen zum Austauschen von Betriebsmittellokalisierern, wie z.B. URLs, und TIL-Posten auszutauschen. Beispielsweise könnte die Kommunikationsverbindung eine geschaltete Sprachschaltung sein, wobei das Client-Modul und das Servermodul unter Verwendung von Modems kommunizieren. Alternativ dazu könnte die

Kommunikationsverbindung irgendein anderes paketgeschaltetes Netzwerk sein, solange es eine Art für das Client-Modul **702** gibt, Anforderungen zu dem Servermodul **749** zu bekommen, und für das Servermodul **749**, Daten zum Client-Modul **702** zurückzusenden. Weiterhin könnte ein Server für spezielle Zwecke anstelle des HTTP-Servers **749** verwendet werden. Beispielsweise können die Prinzipien dieser Erfindung über verschiedene Datentransportmechanismen verwendet werden, einschließlich von schaltungsgeschalteten Daten und paketgeschalteten Daten. Diese Datentransportmechanismen sind für die meisten der Zellulernetzwerkstandards, einschließlich GSM, TDMA und CDMA, definiert und implementiert.

[0259] Bei der Konfiguration des Luftnetz-Netzwerks **750** ([Fig. 7](#)) kommunizierte das Client-Modul **702** direkt mit dem Servercomputer **743**. Bei einem anderen Ausführungsbeispiel kommuniziert, wie es in [Fig. 5](#) dargestellt ist, die Zweiwege-Datenkommunikationsvorrichtung zuerst mit einer Luftnetz-Netzwerkumsetzungseinheit **500**, die infolge davon mit dem geeigneten Server kommuniziert. Bei diesem Ausführungsbeispiel ist die Operation von Zweiwege-Datenkommunikationsvorrichtungen **100** und **101** gleich derjenigen, die oben für das Zellulartelefon **700** beschrieben ist, außer dass das Verfahrensfeld in der im Prozess **802** erzeugten Anforderung eine andere Form hat. Beispielsweise ist unter Verwendung derselben Information wie zuvor das Verfahrensfeld bei diesem Ausführungsbeispiel folgendes:

```
GET http://www.libris.com/airnet/home.cgi?&cost=1 ANTP/1.0
```

[0260] Das Verfahrensfeld enthält die vollständige Adresse des Servers, die erwarteten Kosten des Dienstes und die Version des Protokolls, das zum Kommunizieren mit der Luftnetz-Netzwerkumsetzungseinheit **500** verwendet wird. Die Zweiwege-Datenkommunikationsvorrichtung sendet die HTTP-Anforderung einschließlich der vollständigen URL zu der Luftnetz-Netzwerkumsetzungseinheit **500**.

[0261] [Fig. 12](#) ist ein detaillierteres Blockdiagramm, das die Strukturen bei einem Ausführungsbeispiel der Luftnetz-Netzwerkumsetzungseinheit **500** darstellt. Bei diesem Ausführungsbeispiel ist die Luftnetz-Netzwerkumsetzungseinheit **500** ein Computer, der unter dem UNIX-Betriebssystem läuft, mit einer Schnittstelle zum CDPD-Netzwerk **710**. Solche Computer sind Fachleuten auf dem Gebiet wohlbekannt. Somit werden hierin nur die Strukturen und Prozesse, die zu einem solchen hinzugefügt werden müssen, beschrieben.

[0262] Die Luftnetz-Netzwerkumsetzungseinheit **500** unterstützt Internetprotokoll-(IP)-Verbindungen über das CDPD-Netzwerk **710** und mit jedem Computernetzwerk, mit welchem die Umsetzungseinheit **500** interagieren kann. Bei diesem Ausführungsbeispiel ist jedes der Module in der Netzwerkumsetzungseinheit **500** ein Prozess, der durch den Prozessor im Computer ausgeführt wird. Ein Steuermodul **1201** ist eine höhere Macht, die auf Übertragungen über eine IP-Verbindung vom CDPD-Netzwerk **710** hört. Wenn das Steuermodul **1201** eine Übertragung annimmt, bringt das Steuermodul **1201** einen ANT-Anforderungsprozessor **1204** hervor, der bei diesem Ausführungsbeispiel ein Prozess ist, wie es oben angezeigt ist. Während in [Fig. 12](#) nur ein ANT-Anforderungsprozessor **1204** gezeigt ist, gibt es einen ANT-Anforderungsprozessor, der für jede Übertragung hervorgebracht ist, die das Steuermodul **1201** annimmt, und der ANT-Anforderungsprozessor bleibt aktiv, bis die Kommunikation beendet ist.

[0263] [Fig. 13](#) ist ein Verfahrensablaufdiagramm, das den Betrieb eines ANT-Anforderungsprozessors **1204** darstellt. Dieses Verfahrens- bzw. Prozessablaufdiagramm betrachtet Übertragungen, die sowohl TCP/IP als auch UDP/IP verwenden. Jedoch sind die Prozesse, die nur für TCP/IP spezifisch sind, in Kästen mit gestrichelter Linie umgeben. Daraufhin, dass er für TCP/IP hervorgebracht ist, bildet der ANT-Anforderungsprozessor **1204** bei dem Prozess **1300** zum Bilden einer Verbindung eine TCP-Verbindung unter Verwendung eines TCP-Moduls im Server mit dem Client-Modul über das CDPD-Netzwerk **710**. Nachdem die Verbindung gebildet ist, transferiert eine Verarbeitung vom Prozess **1300** zu einer Prüfung **1301** auf eine empfangene Anforderung.

[0264] Wenn UDP verwendet wird, initiiert ein ANT-Anforderungsprozessor **1204** daraufhin, dass er hervorgebracht ist, eine Verarbeitung in der Prüfung **1301** auf eine empfangene Anforderung. Bei der Prüfung **1301** bestimmt der ANT-Anforderungsprozessor **1204**, ob die Anforderung vom Zellulartelefon **700** ([Fig. 12](#)) empfangen worden ist und im Speicher **1210** gespeichert worden ist. Der Speicher **1210** stellt sowohl einen RAM als auch einen nichtflüchtigen Speicher bei diesem Ausführungsbeispiel dar. Wenn die Anforderung empfangen und gespeichert worden ist, transferiert eine Verarbeitung von der Prüfung **1301** zu einem Prozess **1302** zum Auslesen von Daten.

[0265] Im Prozess **1302** zum Auslesen von Daten liest der ANT-Anforderungsprozessor **1204** Information in Bezug auf die Quelle der URL aus, d.h. das Client-Modul **702** des Zellulartelefons **700** von einer Kunden-Datenbank **1213**, und den in der URL spezifizierten Zielort, d.h. den bestimmten Server, aus der Server-Daten-

bank **1212**. Beide Datenbanken **1212** und **1213** sind im Speicher **1210** gespeichert. Eine Kundenaufzeichnung in der Datenbank **1213** enthält beispielsweise eine Trägeradresse, z.B. eine IP-Nummer, eine Luftnetz-Netzwerkumsetzungseinheits-Kontierungsnummer, eine Buchungsinformation und Serveranteile. Eine Serveraufzeichnung in der Datenbank **1212** enthält eine Server-IP-Adresse, einen Namen, eine Kategorie und eine Dienstklasse. Eine Dienstklasse bezieht sich auf den Preis des Dienstes, wie z.B. Basisdienste, Premium-Dienste oder Dienste, für die pro Ansehen bezahlt wird. Andere Preisschemen können in anderen Implementierungen unterstützt werden. Wenn die Information für den Server und den Dienst, die in der URL spezifiziert ist, und für den Kunden ausgelesen wird, transferiert eine Verarbeitung zu einer Prüfung **1303** auf eine gültige Anforderung.

[0266] Bei der Prüfung **1303** auf eine gültige Anforderung bestimmt der ANT-Anforderungsprozessor **1204** beispielsweise, ob das Client-Modul **702**, d.h. der Kunde, dazu autorisiert ist, auf die Luftnetz-Netzwerkumsetzungseinheit **500** zuzugreifen; ob das Client-Modul **702** dazu autorisiert ist, auf den in der URL spezifizierten Server zuzugreifen; ob der spezifizierte Server durch die Umsetzungseinheit **500** verfügbar ist; und ob der spezifizierte Server den angeforderten Dienst unterstützt. Somit erklärt die Prüfung **1303** auf eine gültige Anforderung den Client, den Server und das Client/Server-Paar für gültig. Ebenso könnten deshalb, weil geschätzte Kosten in der Anforderung enthalten sind, der Status und Kreditgrenzen auf dem Konto des Kunden geprüft werden, um zu bestimmen, ob die geschätzten Kosten akzeptierbar sind.

[0267] Wenn die gesamten Prüfungen wahr sind, transferiert eine Verarbeitung zu einem Prozess **1306** zum Erzeugen einer HTTP-Anforderung. Gegensätzlich dazu führt die Prüfung **1303** auf eine gültige Anforderung dann, wenn irgendeine der Prüfungen nicht wahr ist, Information in Bezug auf den Fehler zu einem Prozess **1304** zum Zurückbringen des Fehlers.

[0268] Der Prozess **1304** zum Zurückbringen eines Fehlers startet ein im Speicher **1210** gespeichertes CGI-Programm basierend auf der empfangenen Information und führt geeignete Information zu dem CGI-Programm. Das CGI-Programm bildet einen geeigneten PIDL-Posten, der den Fehler beschreibt, und wandelt den PIDL-Posten in einen TIL-Posten um, wie es oben beschrieben ist. Wenn der TIL-Posten, der den Fehler beschreibt, vollständig ist, transferiert der Prozess **1304** zum Zurückbringen eines Fehlers eine Verarbeitung zu einem Prozess **1315** zum Protokollieren einer Transaktion, der nachfolgend vollständig beschrieben wird.

[0269] Wenn alle Prüfungen in der Prüfung **1303** auf eine gültige Anforderung wahr sind, wandelt der Prozess **1306** zum Erzeugen einer HTTP-Anforderung die Anforderung im Speicher **1211** in eine Anforderung um, die für den spezifizierten Server spezifisch ist, welche bei diesem Ausführungsbeispiel eine HTTP-Anforderung ist. Beispielsweise erzeugt der Prozess **1306** zum Erzeugen einer HTTP-Anforderung für die obige Anforderung ein Verfahrensfeld, wie beispielsweise:

```
GET/airnet/home.cgi?&client=xyz&cost=1 HTTP/1.0
```

[0270] Bei diesem Ausführungsbeispiel enthält das Verfahrensfeld dieselbe Information wie bei dem oben beschriebenen Ausführungsbeispiel und zusätzlich enthält das Verfahrensfeld eine Client-Identifikation und die geschätzten Kosten.

[0271] Nachdem der Prozess **1306** zum Erzeugen einer HTTP-Anforderung beendet ist, greift der ANT-Anforderungsprozessor **1204** auf das TCP-Modul **1203** im Prozess **1307** zum Bilden einer Serververbindung für TCP/IP zu und transferiert zu einer Prüfung **1308** zum Sichern einer Übertragung für UDP/IP. Im Prozess **1307** zum Bilden einer Verbindung wird eine Verbindung zwischen dem in der Client-Anforderung bestimmten Server und dem TCP-Schnittstellenmodul (nicht gezeigt) hergestellt, so dass Daten zwischen der Luftnetz-Netzwerkumsetzungseinheit **500** und dem Server übertragen werden können. Wenn die TCP-Verbindung zu dem Server gebildet ist, transferiert der ANT-Anforderungsprozessor **1204** eine Verarbeitung von dem Prozess **1307** zum Bilden einer Serververbindung zu einer Prüfung **1308** auf eine sichere Übertragung.

[0272] In der Prüfung **1308** auf eine sichere Übertragung bestimmt der ANT-Anforderungsprozessor **1204**, ob die HTTP-Anforderung von dem Client einen Server anforderte, der ein Protokoll verwendet, das eine Verschlüsselung unterstützt. Wenn ein solcher Server angefordert wurde, transferiert eine Verarbeitung zu einem Prozess **1306** zum Verhandeln und sonst zu einem Prozess **1310** zum Übertragen einer Anforderung.

[0273] Im Prozess **1309** zum Verhandeln verhandelt der ANT-Anforderungsprozessor **1204** über eine Verschlüsselungstechnik mit dem Server. Auf eine Beendigung der Verhandlung hin transferiert die Verarbeitung vom Prozess **1309** zu einem Verschlüsselungsprozess **1311**. Im Verschlüsselungsprozess **1311** wird die HTTP-Anforderung unter Verwendung der verhandelten Verschlüsselungstechnik verschlüsselt und dann

transferiert die Verarbeitung zu einem Prozess **1310** zum Senden einer Anforderung.

[0274] Im Prozess **1310** zum Übertragen einer Anforderung wird die HTTP-Anforderung vom Speicher **1210** zum HTTP-Server gesendet. Wenn die Übertragung beendet ist, geht der ANT-Anforderungsprozessor **1204** zu einer Prüfung **1312** auf ein empfangenes Ergebnis.

[0275] Wie es oben beschrieben ist, bedient der HTTP-Server auf einen Empfang der Anforderung hin die Anforderung. Auf eine Beendigung eines Bedienens der Anforderung hin bringt der HTTP-Server entweder einen PIDL-Posten oder einen TIL-Posten zur Luftnetz-Netzwerkumsetzungseinheit **500** zurück. Der Posten wird im Speicher **1210** gespeichert. Wenn der Server den PIDL-Posten nicht in einen TIL-Posten umwandelt, wird die Umsetzung durch die Luftnetz-Netzwerkumsetzungseinheit **500** durchgeführt.

[0276] Wenn der Posten empfangen und gespeichert ist, geht der ANT-Anforderungsprozessor **1204** von der Prüfung **1312** zu einem Prozess **1313** für eine beendete Übertragung für TCP/IP und zu einer Prüfung **1314** für eine sichere Übertragung für UDP/IP über. Der ANT-Anforderungsprozessor **1204** schließt die TCP-Schaltung mit dem Server im Prozess **1313** für eine beendete Übertragung. Auf ein Schließen der Server-TCP-Verbindung hin transferiert die Verarbeitung zu einer Prüfung **1314** auf eine sichere Übertragung.

[0277] Wenn der Server eine Verschlüsselung verwendete, wird der im Speicher **1210** gespeicherte Posten verschlüsselt. Somit transferiert die Prüfung **1314** auf eine sichere Übertragung eine Verarbeitung zu einem Entschlüsselungsprozess **1316**, wenn eine Verschlüsselung verwendet wurde, und sonst zu einem Prozess **1314** zum Protokollieren einer Transaktion.

[0278] Im Entschlüsselungsprozess **1316** wird der verschlüsselte Posten decodiert und im Speicher **1210** gespeichert. Ebenso wird nach der Decodierung dann, wenn der Posten in einen TIL-Posten umgewandelt werden muss, die Umsetzung durchgeführt. Der Entschlüsselungsprozess **1316** transferiert zu einem Prozess **1315** zum Protokollieren einer Transaktion.

[0279] Im Prozess **1315** zum Protokollieren einer Transaktion schreibt der ANT-Anforderungsprozessor **1204** eine Beschreibung der Transaktion zu einem Prozess **1211** zum Protokollieren einer Transaktion im Speicher **1210**. Bei diesem Ausführungsbeispiel enthält jede Transaktionsaufzeichnung eine Kundenidentifikation, eine Serveridentifikation, eine Zeit, die für die Transaktion erforderlich ist, Kosten für die Transaktion und einen Beendigungscode. Bei einem Ausführungsbeispiel ist jedes Zellularteleson zu Sicherheitszwecken nur einem Kunden und nur einem Konto zugeordnet.

[0280] Nachdem die Transaktion protokolliert ist, transferiert eine Verarbeitung zu einem Prozess **1317** zum Senden bzw. Übertragen eines Ergebnisses. Im Prozess **1317** zum Übertragen eines Ergebnisses bringt der ANT-Anforderungsprozessor **1204** den Posten zum Client **702** zurück. nachdem der Posten übertragen ist, wird ANT-Anforderungsprozessor **1204** beendet.

[0281] Bei einem Ausführungsbeispiel bringt die Umsetzungseinheit dann, wenn eine Luftnetz-Netzwerkumsetzungseinheit vollständig geladen ist und eine weitere Übertragung hereinkommt, die Adresse einer anderen Luftnetz-Netzwerkumsetzungseinheit zurück und verweigert die Übertragung. Das Zellularteleson überträgt die Nachricht zu der anderen Luftnetz-Netzwerkumsetzungseinheit. Bei einem weiteren Ausführungsbeispiel werden alle ankommenden Übertragungen zu einem Router geführt.

[0282] Eine Vielzahl von Luftnetz-Netzwerkumsetzungseinheiten ist mit dem Router verbunden. Der Router überwacht den Zustand jeder Umsetzungseinheit. Jede ankommende Übertragung wird zu der am wenigsten ausgelasteten Umsetzungseinheit geführt, die infolge davon auf die Übertragung antwortet und die nötigen Operationen zum Fortführen von Kommunikationen mit dem Client-Modul durchführt.

[0283] In der obigen Beschreibung des Client-Moduls **702** interagierte das Modul **702** mit Komponenten innerhalb des Zellulartelesons, um die verschiedenen Operationen durchzuführen, die durch den Anwender spezifiziert sind. Zum Isolieren des Client-Moduls **702** vor den Dringlichkeiten von verschiedenen Zellulartelesonen zu dem möglichen Ausmaß wird eine allgemeine Architektur für das Client-Modul **702** nachfolgend vollständiger beschrieben. Diese allgemeine Architektur ist entwickelt, um spezifische Managermodule zu haben, die mit den oben beschriebenen Modulen innerhalb des Zellulartelesons interagieren, und um Standardinformation zu den übrigen Managermodulen innerhalb des Client-Moduls **702** zu liefern. Die Managermodule mit dem Client-Modul **702** bilden einen Interpretierer, der TIL-Posten interpretiert, um eine Anwenderschnittstelle zu erzeugen; eine Dateneingabe durch den Anwender interpretiert; und die TIL-Posten interpretiert, so dass die Da-

teneingabe durch den Anwender mit einem geeigneten Betriebsmittellokalisierer kombiniert wird und entweder eine Nachricht zu einem geeigneten Server gesendet wird oder ein weiterer lokaler TIL-Posten durch das Client-Modul **702** interpretiert wird. Während dieses Ausführungsbeispiel für eine Zellulartelefon ist, sind die Managermodule allgemein und sind somit auf irgendein Client-Modul in einer Zweiwege-Datenkommunikationsvorrichtung anwendbar.

[0284] Dieser Ansatz beschränkt die Modifikationen, die an dem Client-Modul **702** durchgeführt werden müssen, um die Prinzipien dieser Erfindung in einer weiten Vielfalt von Zweiwege-Datenkommunikationsvorrichtungen über eine weite Vielfalt von Zweiwege-Datenkommunikationsnetzwerken zu implementieren. Ebenso unterstützte das Client-Modul **702** bei dem obigen Ausführungsbeispiel Kommunikationen und Interaktionen über das Zellulartelefonnetzwerk. Jedoch kann das Client-Modul **702** auch lokale Dienste am Zellulartelefon **700** unterstützen. Typische lokale Dienste enthalten lokale Nachrichten, ein Adressbuch und vorkonfigurierte e-Mail-Antworten oder irgendeine Kombination von solchen Diensten.

[0285] Bei diesem Ausführungsbeispiel enthält das Client-Modul **702** eine Vielzahl von Managermodulen, die folgende enthalten: ein Navigations-Managermodul **1401**, ein Netzwerk-Managermodul **1402**, ein TIL-Managermodul **1403**, ein Archi-Managermodul **1404**, ein lokales Managermodul **1405**, ein Ereignis-Managermodul **1406**, ein Zeitgeber-Managermodul **1407**, ein Anwenderschnittstellen-Managermodul **1408**, ein Speicher-Managermodul **1409** und ein vorrichtungsabhängiges Modul **1410**.

[0286] Das Navigations-Managermodul **1401** handhabt eine Karten- und Postennavigation und managt auch irgendwelche Caches. Das Navigations-Managermodul **1401** besitzt und managt eine Vorgeschichtenliste sowie auch eine gestoßene Kartenliste. Zusätzlich fungiert das Navigations-Managermodul **1401** als die Hauptleitung des Client-Moduls **702**; führt die gesamte Ereignisverteilung durch; und unterstützt lokale Dienste.

[0287] Für lokale Dienste, wie eine lokale Nachrichtenspeicherung, gibt es zwei grundsätzliche Ansätze, die verwendet werden können. Als erstes werden lokale Dienste auf eine CGI-artige Weise implementiert. Jeder lokale Dienst hat eine Eintragsstelle, welche mit einer Argumentenliste aufgerufen wird. Ein TIL-Posten wird über den Ereignismanager zurückgebracht. Von dieser Stelle an wird der TIL-Posten auf die standardmäßige Weise verarbeitet. Dieser Ansatz beschränkt lokale Dienste auf dieselben Beschränkungen wie entfernte Dienste. Ein weniger restriktiver Ansatz besteht im Zulassen des lokalen Dienstes zu Feldereignissen anstelle der standardmäßigen Ereignisschleife. Der lokale Dienst würde TIL-Karten im Flug bilden und sie zum Anwenderschnittstellen-Managermodul **1406** zu führen. Es ist zu beachten, dass der lokale Dienst mit der standardmäßigen Ereignisschleife in Bezug auf die Vorgeschichte, der gestoßenen Kartenliste und irgendeinem anderen Zustand, der normalerweise durch die Ereignisschleife gemanagt wird, kooperieren müsste. Eine Tabelle 4 ist eine Auflistung von Prozessen für die Architektur für das Navigations-Managermodul **1401**.

TABELLE 4

ARCHITEKTUR FÜR EIN NAVIGATIONS-MANAGERMODUL 1401

```
ProcessEvents (void);
PushLocation (void * location, Boolean forStack);
void * PopLocation (Boolean forStack);
```

```

void * CurrentLocation ();
struct LOCAL_SEERVICE {
    char name [50];
    FUNC HandleEvent (Event * pevent);
    FUNC StartLocalService (void);
    FUNC StopLocalService (void);
};
static LOCAL_SERVICE localServices [] = { ...};
STATUS HandleEvent (Event * prevent);
STATUS StartLocalService ();
STATUS StopLocalService ();

```

[0288] Die Routine ProcessEvents ist die Haupteintrittsstelle für eine Ereignisverarbeitung im Client-Modul **702**. Typische Ereignisse enthalten Tastendrücke auf dem Tastenfeld, eine Auswahlselektion für eine Auswahlkarte, eine Texteingabe für eine Eingabekarte, Netzwerkereignisse und Vorgeschichtenereignisse. Die Routine Process Events kann zu jeder Zeit aufgerufen werden, um ein Ereignis oder Ereignisse zu verarbeiten. Die Routine ProcessEvents springt nicht zurück, bis alle Ereignisse in einer Warteschlange, die durch das Ereignis-Managermodul **1406** erzeugt ist, verarbeitet sind. Wenn ein lokaler Dienst abläuft, werden Ereignisse zu dem lokalen Dienst verteilt, bevor sie durch die Routine ProcessEvents verarbeitet werden.

[0289] Die übrigen Routinen in der Tabelle 4 werden intern vom Navigations-Managermodul **1401** und durch lokale Dienste aufgerufen. Die Routine PushLocation legt auf eine Stelle auf der Vorgeschichtenliste ab und gibt eine Anforderung für diese Stelle aus. Das Flag forStack zeigt einen Auflegen eines Stapels von lokalen Karten an.

[0290] Die Routine *PopLocation hebt eine Stelle auf dem Vorgeschichtenstapel ab und gibt eine Anforderung für die oberste Stelle des Vorgeschichtenstapels aus. In der Routine *PopLocation zeigt das Flag forStack an, dass alle Karten seit dem letzten Drauflegen auf den Stapel abgehoben werden sollten.

[0291] Die Routine *CurrentLocation bringt die aktuelle Stelle zurück, wobei die aktuelle URL angezeigt wird.

[0292] Wie es in der Tabelle 4 gezeigt ist, stellt jeder lokale Dienst eine Anzahl von Funktionen zur Verfügung. Wenn ein lokaler Dienst läuft, wird eine Funktion HandleEvent, der Ereignishandhaber des lokalen Dienstes, vor irgendeiner Verarbeitung durch das Navigations-Managermodul **1401** aufgerufen. Wenn das Ereignis durch den lokalen Dienst gehandhabt wird, wird das Ereignis nicht irgendwie weiterverarbeitet.

[0293] Die Funktion StartLocalService ist die Funktion zum Starten von lokalen Diensten. Die Funktion StartLocalService wird aufgerufen, bevor irgendwelche Ereignisse zu der lokalen Funktion verteilt werden. Gleichmaßen ist die Funktion StopLocal Service die Stoppfunktion für den bestimmten lokalen Dienst. Die Funktion Stop LocalService wird aufgerufen, wenn keine weiteren Ereignisse zum lokalen Dienst verteilt werden.

[0294] Das Netzwerk-Managermodul **1402** isoliert den Rest des Client-Moduls **702** von dem spezifischen Netzwerkbildungsprotokoll, das über das Zellulartelefonnetzwerk verwendet wird. Das Netzwerk-Managermodul **1402** liefert Anforderungen zu dem Server, der in der URL spezifiziert ist, über die Zellulartelefonnetzwerk-Schnittstelle; segmentiert Antworten von dem Server für eine niedrigere Latenzzeit; liefert Antworten von lokalen Diensten zum Navigations-Managermodul **1401** über das Ereignis-Managermodul **1406**; handhabt einen Anforderung/Antwort-Zyklus (z.B. eine Strategie eines Löschens, eines erneuten Versuchs) mit dem Server über das Zellulartelefonnetzwerk; kann asynchrone Nachrichten vom Server empfangen; führt ein Speichermanagement von TIL-Posten durch; führt ein Speichern von TIL-Posten in Caches durch; handhabt alle Verhandlungen in Bezug auf Protokolle und eine Serverskalierung mit dem Server; handhabt irgendeine Verschlüsselung für Information, die zwischen dem Zellulartelefon **700** und dem Server ausgetauscht ist.

[0295] Bei einigen Zellulartelefonen ist die maximale Nachrichtengröße festgelegt. Jedoch wird für UDP- und TCP-Nachrichten eine direktere Schnittstelle verwendet, die diese Beschränkung einer Nachrichtenführung umgeht. Es ist wichtig, ein Kopieren von Netzwerkdaten von einem Speicherpuffer zu einem Speicherpuffer zu

vermeiden, da ein solches Kopieren die "Hochwassermarke" des Speichers erhöht sowie die Leistungsfähigkeit erniedrigt. Da unterschiedliche Zellulartelefone unterschiedliche Schnittstellen zum Liefern von Netzwerkdaten haben, managt das Netzwerk-Managermodul **1402** die Netzwerkdaten. Auf diese Weise werden Netzwerkdaten nur von dem Netzwerkpuffer für eine langzeitige Speicher kopiert.

[0296] Wenn eine Nachricht oder Antwort ankommt, verwendet das Netzwerk-Managermodul **1402** das Ereignis-Managermodul **1406** zum Berichten über diese Tatsache. Jedoch erfolgt ein Zugriff zu den Daten durch andere Managermodule im Client-Modul **702** durch ein Protokoll, das eine Speicherung von Daten in einer Vielfalt von Arten bei unterschiedlichen Telefonen zulässt. Irgendein transparentes, kurzzeitiges Speichern von TIL-Daten in Caches wird durch das Netzwerk-Managermodul **1402** gehandhabt. Eine Tabelle 5 ist eine Architektur für das Netzwer-Managermodul **1402**.

TABELLE 5

SPEZIFIKATION FÜR EIN NETZWWERK-MANAGERMODUL 1402

```
typedef short TID;
void NM_Init (void);
void NM_Terminate (void);
TID NM_SendRequest (void *requestData, int length,
Boolean ignoreCache);
NM_CancelRequest (TID TRANSACTIONId);
NM_DataType (TID TRANSACTIONId);
NM_GetData (TID TRANSACTIONId, void *data, int
*length, Boolean *complete);
void *NM_HoldData (TID TRANSACTIONId);
NM_ReleaseData (TID TRANSACTIONId);
TID NM_StartData (int data Type, char *requestData,
int length);
STATUS NM_EndData (TID TRANSACTIONId);
STATUS NM_SetDataLentgh (TID TRANSACTIONId, int
length);
STATUS NM-GrowDataLength (TID TRANSACTIONId, Int
grow);
int NM_GetDataLength (TID TRANSACTIONId);
void *NM_GetDataPointer (TID TRANSACTIONId);
STATUS NM_DeliverData (TID TRANSACTIONId);
```

[0297] Das Netzwek-Managermodul **1402** identifiziert jede Netzwerkdatentransaktion durch einen 16-Bit-Transaktionsidentifikationscode TID. Das Network-Managermodul **1402** inkrementiert den Transaktionsidentifikationscode TID für jede neue Transaktion um Eins. Der Transaktionsidentifikationscode TID läuft nach 0xffff über.

[0298] Die Routine NM_Init initialisiert das Netzwerk-Managermodul **1402** und wird somit vor irgendwelchen anderen Aufrufen im Netzwerk-Managermodul **1402** aufgerufen. Die Routine NM_Terminate schließt eine Verarbeitung des Netzwerk-Managermoduls **1402** und wird somit nach allen anderen Aufrufen im Netzwerk-Managermodul **1402** aufgerufen.

[0299] Das Netzwerk-Managermodul **1402** verwendet eine Routine TID NM_SendRequest als den Standardprozess zum Senden einer Anforderung zum Server. Ein Zeiger *requestData in dem Aufruf zu der Routine TID

MN_SendRequest ist durch das Serverprotokoll definiert. Gleichmaßen wird der Zustand, z.B. der Boolesche Wert, einer variablen ignoreCache dazu verwendet, anzuzeigen, ob irgendwelche im Cache gespeicherten Duplikate bzw. Kopien ignoriert werden sollten. Nach einem Senden der Anforderung bringt diese Routine einen Server-Transaktionsidentifikationscode TRANSACTIONId zurück. Ein lokaler Dienst kann auch eine Anforderung zum Server senden.

[0300] Wenn der Anwender das Client-Modul **702** anweist, eine Anforderung zu löschen, ruft das Netzwerk-Managermodul **1402** eine Routine NM_CancelRequest mit einem Zellulartelefon-Transaktionsidentifikationscode TID und einem Server-Transaktionsidentifikationscode TRANSACTIONId auf. Die Routine NM_CancelRequest gibt einen Befehl zu dem Server aus, um die spezifizierte Anforderung zu löschen.

[0301] Wenn Daten vom Netzwerk empfangen werden, können die Daten entweder eine Antwort auf eine durch die Routine TID MN_SendRequest gesendete Anforderung sein, oder durch einen lokalen Dienst. Somit erzeugt das Netzwerk-Managermodul **1402** in Antwort auf ein Empfangen von Daten vom Server ein Ereignis, das einen Server-Transaktionsidentifikationscode TRANSACTIONId und den Typ von Daten DATAType enthält. Für Antworten auf Anforderungen, die durch die Routine TID MN_SendRequest gesendet sind, ist der Server-Transaktionsidentifikationscode TRANSACTIONId derselbe wie derjenige, der durch den passenden Aufruf zu der Routine TID MN_SendRequest zurückgebracht ist, und Datentyp DATAType zeigt an, dass die Daten eine Antwort sind. Für bei einem lokalen Dienst entstandene Nachrichten ist eine Servertransaktions-ID neu und hängt der Datentyp DATAType davon ab, ob die Daten eine e-Mail, eine draufgelegte TIL oder ein anderer Typ sind.

[0302] Nachdem das Netzwerkereignis durch das Ereignis-Managermodul **1406** empfangen ist, und das Navigations-Managermodul **1401** eine Steuerung des Ereignisses zum Netzwerk-Managermodul **1402** verteilt, verwendet das Netzwerk-Managermodul **1402** den Server-Transaktionsidentifikationscode TRANSACTIONId und die übrigen Routinen in der Tabelle 5 zum Verarbeiten der Daten.

[0303] Die Routine NM_DataType wird dazu verwendet, den bestimmten Datentyp dataTYPE, z.B. Antwort, MIME, Drauflegen durch den Server, etc., zurückzubringen. Die Routine NM_GetData stellt einen Zeiger zu den Daten ein, die durch den Server-Transaktionsidentifikationscode TRANSACTIONId identifiziert sind, liest die Länge der Daten aus und bestimmt, ob alle Daten empfangen worden sind. Die durch diese Routine zur Verfügung gestellte Schnittstelle lässt zu, dass der erste Teil eines Datenstroms, z.B. die erste Karte eines TIL-Postens, durch das Client-Modul **702** verarbeitet wird, bevor der Rest des Postens empfangen wird.

[0304] Die Routine NM_HoldData wird vor einem Aufrufen der Routine NM_GetData aufgerufen, um die Daten zu halten und somit sicherzustellen, dass die Daten während einer Verarbeitung durch das Client-Modul **702** gültig bleiben. Wenn die Daten nicht gehalten werden, könnten die Daten gelöscht oder mit den internen Puffern des Netzwerk-Managermoduls **1402** bewegt werden. Wenn die Daten gehalten werden, wird die Routine NM_ReleaseData aufgerufen, nachdem Netzwerkdaten verarbeitet worden sind, um die Daten freizugeben.

[0305] Die Routinen TID NM_StartData, NM_EndData, NM_SetDataLength, NM_GrowDataLength, NM_GetDataLength, NM_GetDataPointer und NM_DeliverData werden intern durch das Netzwerk-Managermodul **1402** und durch lokale Dienste zum Liefern von Daten verwendet. Durch Zulassen, dass lokale Dienste diese Routinen verwenden, können dieselben Puffer zum Speichern von sowohl Netzwerkdaten als auch lokal erzeugten Daten verwendet werden, um dadurch das Ausmaß an Speicher zu reduzieren, das zum Unterstützen des Client-Moduls **702** erforderlich ist.

[0306] Die Routine TID NM_StartData erzeugt eine neue Datentransaktion und triggert ein Datenlieferereignis. Die Routine NM_EndData wird aufgerufen, wenn alle Daten für den gegebenen Server-Transaktionsidentifikationscode TRANSACTIONId übertragen worden sind. Die Routine NM_SetDataLength stellt das Datensegment auf eine gegebene Länge ein und kann veranlassen, dass sich diese Stelle der Daten ändert. Die Routine NM_GrowDataLength lässt das Datensegment um eine gegebene Länge anwachsen und kann auch veranlassen, dass sich die Stelle der Daten ändert. Die Routine NM_GetDataLength bringt die Länge des Datensegments zurück. Die Routine NM_GetDataPointer bringt einen Zeiger zu den Daten zurück. Diese Routine wird vorzugsweise vor einem Schreiben in den Datenpuffer aufgerufen. Ebenso wird diese Routine vorzugsweise aufgerufen, wann immer die Stelle von Daten sich geändert haben kann. Die Routine NM_DeliverData kann aufgerufen werden, wenn wenigstens eine Karte gespeichert worden ist, um eine Latenzzeit zu reduzieren, während die anderen Karten gerade erzeugt werden.

[0307] Das TIL-Managermodul **1403** isoliert den Rest des Client-Moduls **702** von Änderungen zu der TIL-Spezifikation. Die durch das TIL-Managermodul **1403** zur Verfügung gestellte Schnittstelle hat die folgenden Eigenschaften: sie entfernt die Notwendigkeit zum Zerlegen durch den Rest des Client-Moduls **702**; sie verwendet Cursor zum Vermeiden eines Erzeugens von Datenstrukturen im Flug; sie benötigt keinen gesamten Posten zum Arbeiten; und sie handhabt eine TIL-Versionenbildung.

[0308] Jeder TIL-Posten enthält eine größere bzw. höhere und eine kleinere bzw. niedrigere Versionsnummer. Die niedrigere Versionsnummer wird inkrementiert, wenn sich TIL auf eine Weise ändert, die existierende TIL-Managermodule nicht zerstört. Die höhere Versionsnummer wird für nicht kompatible Versionen von TIL inkrementiert.

[0309] Jeder TIL-Posten hat dieselbe Hierarchie. Ein Ausführungsbeispiel dieser Hierarchie wird in der Tabelle 6 präsentiert. In der Tabelle 6 wird eine Absatzbildung zum Darstellen der Beziehungen der verschiedenen Hierarchieebenen verwendet.

TABELLE 6

TIL-POSTEN-HIERARCHIE

Posten

Optionen

weiche Tasten

Optionen

Karte

Optionen

weiche Tasten

Optionen

formatierter Text

formatierte Linien bzw. Zeilen

Einträge

Optionen

formatierte Linie bzw. Zeile

[0310] Die in der Tabelle 7 für das TIL-Managermodul **1403** präsentierte Schnittstelle ist unter der Annahme entwickelt, dass TIL- eine direkte Tokenisierung von PIDL ist, wie es im Anhang I beschrieben ist. Jedoch hat die Schnittstelle keinerlei Abhängigkeiten von dieser Tokenisierung und kann andere PIDL-Codiertechiken unterstützen. Unter der Voraussetzung der obigen Annahme sind die nachfolgend beschriebenen opaken Zeiger aktuelle Zeiger in den TIL-Posten selbst. Ein rudimentäres Objekttypisierungsschema, das darauf basiert, wo im Posten der opake Zeiger hinzeigt, kann zum Implementieren der nachfolgend beschriebenen allgemeinen bzw. generischen Funktionen verwendet werden. Wenn diese Objekttypisierung aufgrund von Details einer TIL-Codierung nicht möglich ist, können die allgemeinen Funktionen durch spezifische Funktionen ersetzt werden.

TABELLE 7

ARCHITEKTUR FÜR EIN TIL-MANAGERMODUL 1403

typedef char *opaque

typedef opaque Deck;

typedef opaque Card;

typedef opaque Text;


```
typedef opaque Entry;
typedef opaque Option;
typedef opaque SoftKey;
typedef opaque Object;
```

```
/* Allgemeine Funktionen */
```

```
FirstOption (Object obj, Optionn *o);
    /* obj is a card, softkey, entry or deck */
GetSoftkey (Object obj, Option *o);
    /* obj is a card or deck */
GetText (Object obj, Option *o);
    /* obj is a card or entry */
```

```
/* Postenfunktionen */
```

```
SetDeck (Deck d, int length)
    (* tells module which deck to use *)
DeckGetCard (Card *c, int num);
```

```
-oder-
```

```
DeckGetCard (Deck d, Card *c, int num);
```

```
/* Kartenfunktionen */
```

```
int CardType (Card c);
CardFirstEntry (Card c, Entry *e);
CardLookupSoftkey (Card c, int num, Softkey *s);
CardIsLast (Card c);
```

```
/* Options-Cursorfunktionen */
```

```
OptionNext (Option *o);
char *OptionKey (Option *o);
char *OptionValue (Option *o)
```

```
/* Eingabe-Cursorfunktionen */
```

```
/* Text- (und Bild-)Cursorfunktionen */
```

```
TextNextToken (Text *t, int *type, int *subtype,
               int *length, char *data);
```

[0311] Das Archiv-Managermodul **1404** speichert langlebige Information und liest sie aus. Diese Information enthält folgendes: Daten, die auf die Serverstelle bezogen sind und/oder die zum Unterstützen einer Serverskalierung erforderlich sind; Daten, die auf eine Verschlüsselung bezogen sind; ein Speichern von TIL in Caches (für den Anwender transparent); eine TIL-Speicherung (durch einen Anwender spezifiziert); und eine Nachrichtenspeicherung und -auslesung (siehe lokales Managermodul). Das Archiv-Managermodul **1404** sollte eine Vielfalt von nichtflüchtigen Speicherschemen unterstützen, die durch die Zweige-Datenkommunikationsvorrichtungen zur Verfügung gestellt werden.

[0312] Das lokale Managermodul **1405** ist eine Schnittstelle zu lokalen Vorrichtungsbetriebsmitteln, wie beispielsweise lokalen Nachrichten, Adressbucheinträgen und vorkonfigurierten e-Mail-Antworten. Das lokale Managermodul **1405** sollte auch eine abstrakte Schnittstelle zum Navigations-Managermodul **1401** zur Verwendung durch das Archiv-Managermodul **1404** definieren.

[0313] Die Tabelle 8 ist eine Architektur für eine Schnittstelle innerhalb des lokalen Managermoduls **1405** zum Zugreifen auf ein beim Zellulartelefon **700** gespeichertes Adressbuch. Der Name einer Routine in der Tabelle 8 ist beschreibend für die durch die Routine durchgeführten Operationen.

TABELLE 8

ARCHITEKTUR FÜR EINEN ADRESSENBUCHZUGRIFF

```
int NumAddresses ();
char *AddressName(int num);
char *AddressGetEMail(int num);
    // returns e-mail address
char *AddressGetPhone(int num);
    // returns phone number
char *AddressGetFax(int num);
    // returns fax number
SetAddress(int num, char *name, char *email,
           char *phone, char *fax);
DeleteAddress(int num);
InsertAddress(int before);
```

[0314] Die Tabelle 9 ist eine Architektur für eine Schnittstelle innerhalb des lokalen Managermoduls **1405** zum Zugreifen auf vorbestimmte Antworten, die beim Zellulartelefon **700** gespeichert sind. Der Name einer Routine in der Tabelle 9 ist beschreibend für die durch die Routine durchgeführten Operationen.

TABELLE 9

ARCHITEKTUR FÜR EINEN ZUGRIFF AUF EINE VORBESTIMMTE ANTWORT

```
int NumReplies();
char * GetReply(int num);
DeleteReply(int num);
SetReply(int num, char *text);
InsertReply (int before);
```

[0315] Die Tabelle 10 ist eine Architektur für eine Schnittstelle innerhalb des lokalen Managermoduls **1405** zum Zugreifen auf Nachrichten, die beim Zellulartelefon **700** lokal gespeichert sind. Der Name einer Routine in der Tabelle 10 ist beschreibend für die durch die Routine durchgeführten Operationen.

ARCHITEKTUR FÜR EINEN ZUGRIFF AUF LOKAL GESPEICHERTE NACHRICHTEN

```

int NumMessages();
void *FirstMessage();
void *NextMessage();
int MessageType (void *msg);
    // e. g. e-mai, TIL, etc.
void *MessageContent (void *msg);
void *SaveMessage (int type, void *content, int
                    contentLength);
DeleteMessage (void *msg);

```

[0316] Das Ereignis-Managermodul **1406** handhabt die Verteilung von Ereignissen. Bei diesem Ausführungsbeispiel enthalten Ereignisse Ereignisse niedriger Ebene, wie Tastendrucke, und eine Navigation höherer Ebene und Anwenderschnittstellenereignisse. Es gibt typischerweise nur eine geringe Anzahl von Ereignissen zu irgendeinem Zeitpunkt. Die Hauptereignisschleife in dem von der Zweiwege-Datenkommunikationsvorrichtung abhängigen Modul hält ein Aufrufen von EM_GetNextEvent(), bis keine Ereignisse in der Warteschlange gelassen sind. Es ist zu beachten, dass eine Verarbeitung von einem Ereignis veranlassen kann, dass ein anderes Ereignis auf die Warteschlange draufgelegt wird. Die Hauptereignisschleife wird nicht erneut gestartet, bis ein weiteres Ereignis auf die Warteschlange draufgelegt wird, und zwar aufgrund eines Tastendrucks durch einen Anwender oder ein Netzwerkereignis.

[0317] Bei diesem Ausführungsbeispiel enthalten die Ereignistypen folgendes:

- 1) Tastenfeldereignisse, d.h. Eindrücken einer Taste;
- 2) Auswahlereignisse, die sich auf eine aktuelle Auswahlkarte beziehen, wobei z.B. der Anwender eine Auswahl Drei auswählt;
- 3) Texteintragsereignisse in Bezug auf eine aktuelle Eingabekarte, wobei z.B. der Anwender "Hallo" über Tasten eingibt;
- 4) Netzwerkereignisse, z.B. Antwort angekommen, Anforderung angekommen, Transaktion beendet, Netzwerkstatus; und
- 5) Vorgeschichtenereignisse, z.B. Abheben, Abheben zu einem Markierer.

[0318] Die Tabelle 11 ist eine Architektur für das Ereignis-Managermodul **1406**. Wie bei den anderen Tabellen hierin ist der Name einer Routine in der Tabelle 11 für die durch die Routine durchgeführten Operationen beschreibend und wird zusätzlich eine kurze Beschreibung im Kommentierungsfeld angegeben.

TABELLE 11

ARCHITEKTUR FÜR EIN EREIGNIS-MANAGERMODUL 1406

```

struct Event {
    int type;
    void *data;
    /* z.B. Tastencode, Auswahlnummer, Eingabetext,
       Statuscode, andere Daten */
}

```

```

EM_QueueEvent(int type, void * data);
    /* Fügt ein Ereignis an ein Ende einer Warteschlange
       hinzu*/
EM_GetNextEvent(Event * event);
    /*Hebt nächstes Ereignis ab*/
EM_PeekNextEvent (Event event);
    /*Spitzen bei nächstem Ereignis*/

```

[0319] Das Zeitgeber-Managermodul **1407** lässt zu, dass Zeitgeberereignisse Auszeiten, eine Animation und andere Zeitbereichsmerkmale unterstützen. Auszeiten werden über das Ereignis-Managermodul **1406** geliefert.

[0320] Die Tabelle 12 ist eine Architektur für das Zeitgeber-Managermodul **1407**. Wie in den anderen Tabellen hierin ist der Name einer Routine in der Tabelle 12 für die durch die Routine durchgeführten Operationen beschreibend.

TABELLE 12

ARCHITEKTUR FÜR EIN ZEITGEBER-MANAGERMODUL 1407

```

TimerInit();
int TimerSet(int milliseconds, int code, void
             *clientData);
    /*Bringt eine Zeitgeberidentifikation timerId zurück,
       um für Löschungen verwendet zu werden*/
TimerCancel(int timerId);
TimerCancelAll(?)

```

[0321] Das Anwenderschnittstellen-Managermodul **1408** handhabt Interaktionen mit dem Tastenfeld und der Anzeige. Jeder der drei Typen von Anwenderschnittstellen, die in der obigen Tabelle 1 definiert sind, erfordert eine andere Version eines Anwenderschnittstellen-Managermoduls **1408**. Für die meisten Zellulartelefone wird nur eine Karte gleichzeitig verwendet. Jedoch können einige Zellulartelefone mehrere Karten auf einmal anzeigen, und würden somit eine andere Version eines Anwenderschnittstellen-Managermoduls **1408** gegenüber der Version erfordern, die eine Anzeige von nur einer Karte gleichzeitig handhabte.

[0322] Bei diesem Ausführungsbeispiel stellt das Anwenderschnittstellen-Managermodul eine Anwenderschnittstelle für die drei Typen von Karten, nämlich Anzeige, Auswahl und Eingabe, zur Verfügung; stellt Haken für kundenangepasste Anwenderschnittstellen für die Adressenlisten- und e-Email-Antworteingabe zur Verfügung; sorgt sich nur um die Anwenderschnittstellenaspekte von Karten und stellt keine Navigations-, Argumenten- oder Optionsverarbeitung zur Verfügung; handhabt das gesamte Text- und Graphiklayout, einschließlich eines Wortumbruchs; handhabt ein Rollen eines Textes; arbeitet von PIDL-Datenstrukturen; Erzeugt Tastaturereignisse von welchem einige durch weiche Tasten erzeugt werden können; und erzeugt Ereignisse hoher Ebene, wie z.B. eine nächste Karte, eine Auswahl eingabe 3, eine Texteingabe "IBM".

[0323] Die Tabelle 13 ist eine Architektur zum Verarbeiten von Karten durch das Anwenderschnittstellen-Managermodul **1408**. Wie in den anderen Tabellen hierin ist der Name einer Routine in der Tabelle 13 für die durch die Routine durchgeführten Operationen beschreibend.

TABELLE 13

ARCHITEKTUR FÜR EINE KARTENVERARBEITUNG DURCH EIN UI- BZW. ANWENDERSCHNITTSTELLEN-MANAGERMODUL 1408

```
void UI_StartCard(Card c);
    /* aufgerufen, um eine Anzeige und eine Verarbeitung einer
       gegebenen Karte zu beginnen*/
void UI_EndCard(Card c);
    /*aufgerufen, wenn eine Karte nicht mehr anzuzeigen ist*/
Boolean UI_HandleEvent(Event *pevent);
    /*bringt wahr zurück, wenn das Ereignis gehandhabt wird,
       falsch, wenn nicht*/
```

[0324] Die Tabelle 14 ist eine Architektur für die Anwenderschnittstellenimplementierung durch das Anwenderschnittstellen-Managermodul **1408**. Wie in den anderen Tabellen hierin ist der Name einer Routine in der Tabelle 14 für die durch die Routine durchgeführten Operationen beschreibend.

TABELLE 14

ARCHITEKTUR FÜR EINE ANWENDERSCHNITTSTELLENIMPLEMENTIERUNG DURCH EIN ANWENDERSCHNITTSTELLEN-MANAGERMODUL 1408

```
UI_LayoutCard(Card c, Boolean draw, Proc callback)
    /* beruht auf globalen Daten; muss fähig sein zum:
       Zeichnen, wenn es geht; und die spezielle Funktion
       der aktuellen Linie notieren (z.B. nichts, Auswahl,
       weiche Taste)*/
int numLine, firstVisible, lastVisible, currentLine;
char currentEntry[80];
int currentChoice;
void *currentSoftkey;
Card currentCard; and
    ... other info as needed for in-line scrolling
```

[0325] Die Rückfunktoutine wird über die spezielle Funktion jeder Linie benachrichtigt, wenn die Linie ausgelegt bzw. aufgezeichnet wird. Somit kann die Routine UI_LayoutCard dazu verwendet werden, zu einer bestimmten Auswahl zu rollen. Wenn die aktuelle Linie zu weit ist, um alles auf einmal anzuzeigen, wird ein horizontales Rollen verwendet, um die vollständige Linie anzuzeigen, und zwar mit einer Anzeigebreite gleichzeitig.

[0326] Das Speicher-Managermodul **1409** ist optional und wird in Zweiwege-Datenkommunikationsvorrichtungen verwendet, die keine dynamische Speicherzuteilung unterstützen. In diesen Vorrichtungen müssen alle Speicherzuteilungen und – freigaben durch das Speicher-Managermodul **1409** gehen. Ebenso läuft durch Zuteilen eines Speichers im Voraus über das Speicher-Managermodul **1409** das Client-Modul **702** nicht aus dem Speicher heraus, und zwar aufgrund von irgendeinem anderen Prozess auf der Vorrichtung unter Verwendung des Speichers.

[0327] Bei einem Ausführungsbeispiel wurde die oben beschriebene Luftnetz-Netzwerkumsetzungseinheit mit einem Internetserver verwendet, um mit dem Client-Modul zu kommunizieren. Der Internetserver war ein UNIX-Computer, der den Mosaik-HTTP-Server laufen lässt. Der ausführbare Code wurde durch Kompilieren

des Quellencodes auf einem Computer erzeugt, der das Betriebssystem Solaris 2.4 von Sun Microsystems unter Verwendung eines Compilers SunPro C und C# von Sun Microsystems und der SDK-Aufbauvorrichtung von Sun Microsystems laufen lässt. Alle diese Produkte sind von Sun Microsystems aus Mountain View, California verfügbar.

[0328] Verschiedene Ausführungsbeispiele eines neuen interaktiven Zweiwege-Datenkommunikationssystems, einer Zweiwege-Datenkommunikationsvorrichtung und einer Luftnetz-Netzwerkarchitektur sind hierin beschrieben worden. Diese Ausführungsbeispiele sind nur für die Prinzipien der Erfindung illustrativ.

ANHANG I

EINE BESCHREIBUNG VON PIDL UND TIL

nicht veröffentlichtes © 1995 Libris, Inc.

[0329] Die Hauptstruktur von PIDL wird durch eine abstrakte Syntax beschrieben. Dieser Anhang beschreibt die Elemente der Sprache und ihrer Semantik. In der Syntaxbeschreibung jedes Elements wird ein Element in einem verstärkten BNF definiert.

<code>a :: = b</code>	das Element a wird als b definiert
<code>a :: = b :: = c</code>	das Element a wird als b oder c definiert
<code>b c</code>	das Element b, dem das Element c folgt, der dazwischen liegende Raum dient nur der Klarheit
<code>a b c</code>	Element a oder Element b oder Element c
<code>{a}</code>	das Element a ist optional
<code>{a}*</code>	das Element a kann Null oder mehrere Male in einer Zeile bzw. Reihe erscheinen
<code>{a}+</code>	das Element a kann ein oder mehrere Male in einer Reihe erscheinen
<code><u>abc</u></code>	die Zeichen abc literal
<code>ol(a)</code>	eine Optionsliste mit Null oder mehreren Optionen des Elements a, siehe Optionen unten

[0330] Im Allgemeinen kann das Element Blank-Raum optional zwischen irgend zwei anderen Elementen erscheinen. Um das Diagramm klar zu halten, ist es weggelassen worden, außer dort, wo es erforderlich ist. Wenn ein Blank-Raum illegal ist oder speziell behandelt wird, ist es angemerkt.

Die PIDL-ELEMENTE

Posten	<code>::= deck-header {softkey}~{card}+ deck-footer</code>
Posten-Anfangsblock	<code>::= <PIDL ol(deck-options) ></code>
Posten-Optionen	<code>::= o-args o-cost o-ttl</code>
Posten-Fußende	<code>::= </PIDL></code>

Ein Posten besteht aus einer oder mehreren Karten. Es muss wenigstens eine Karte geben. Ein Posten kann auch eine Anzahl von weichen Tasten haben, die definiert sind, welche für den gesamten Posten in Kraft bleiben. Siehe weiche Tasten unten in Bezug auf die Syntax- und die vollständige Beschreibung.

o-cost
o-ttl

::= cost= value
::= ttl= integer

Zusätzliche Argumente, um zu der nächsten Posten-anforderung geführt zu werden, sind in o-args angegeben. Siehe Argumente unten für die Syntax und die vollständige Beschreibung.

Die Kosten zum Auslesen dieser Seite (ausschließlich von Telefonsystemgebühren) sind in o-Kosten dargestellt. Wenn keine o-Kosten gegeben sind, sind die Postenkosten in dem standardmäßigen Dienstvertrag des Anwenders enthalten.

Posten können durch das Zellulartelefon für eine Zeitperiode in einen Cache gespeichert werden. Die o-ttl-Eingabe zeigt die Anzahl von Sekunden an, für welche der Posten ab einer Empfangszeit in einen Cache gespeichert werden kann. Wenn keine o-ttl-Eingabe angegeben ist, kann der Posten nur für kurze Zeitperioden in einen Cache gespeichert werden, wie beispielsweise zum Implementieren einer Stützfunktion gleich derjenigen der meisten Web-Browser. Wenn der Wert von o-ttl Null ist, muss der Posten nicht in einen Cache gespeichert werden.

KARTENELEMENTE

Karte

::=Anzeigekarte | Auswahlkarte | Eingabekarte

Eine Karte ist eine von drei Typen einer Karte bei diesem Ausführungsbeispiel. Diese werden in den nachfolgenden Abschnitten beschrieben.

Kartenoptionen

o-Name

o-nächste

o-vorherige

::= o-Name | o-nächste | o-vorherige

::= Name= Identifizierer

::= nächste= Zielort

::= vorherige= Zielort

Alle Karten können diese Optionen haben. Die optionale o-Name-Option gibt einen Namen zu der Karte. Wenn eine Karte einen Namen hat, kann auf die Karte durch einen Zielort Bezug genommen werden.

o-nächste und o-vorherige geben Zielorte für die Tasten NEXT und PREV an. Wenn sie weggelassen sind, sind die Vorgaben die nächste und die vorherige sequenzielle in dem Posten. Wenn o-vorherige von der ersten Karte weggelassen ist, springt die Taste PREV zu dem Posten zurück, der zuletzt besucht wurde. Wenn o-nächste von der letzten Karte weggelassen ist, springt die Taste NEXT zu der ersten Karte des aktuellen Postens zurück. Jedoch ist dieses Vorgabeverhalten nur eine Ausfallsicherung: die letzte Karte in einem Posten sollte immer entweder eine Option o-nächste haben oder eine Auswahlkarte sein, wo jede Auswahl eingabe einen neuen Zielort anzeigt.

ANZEIGEKARTE

Anzeige-Karte

Anzeige-Anfangsblock

Anzeige-Optionen

Anzeige-Inhalt

Anzeige-Fußende

::= Anzeige-Anfangsblock, Anzeige-Inhalt, Anzeige-Fußende

::= <ANZEIGE Optionsliste>

::= Kartenoptionen

::= { weiche Taste } * formatierter Text

::= </ANZEIGE>

Anzeigekarten geben Information für den Anwender zu lesen. Siehe formatierter Text unten für eine vollständige Beschreibung des Formats der Information, die angezeigt werden kann. Weiche Tasten können nur für diese Karte beschrieben werden, siehe weiche Tasten unten.

AUSWAHLKARTE

Auswahlkarte	::= Auswahl-Anfangsblock, Anzeige-Inhalt { Eingaben } Auswahl-Fußende
Auswahl-Anfangsblock	::= <AUSWAHL ol{Auswahl-Optionen} >
Auswahl-Optionen	::= Karten-Optionen o-Verfahren o-Taste o-Vorgabe
o-Verfahren	::= Verfahren= Verfahrenstyp
Verfahrenstyp	::= Anzahl Liste Alpha Gruppe
Eingaben	::= { Auswahleingabe }+
	::= { Gruppeneingabe { Auswahleingabe }+ }+
Auswahl-Fußende	::= </AUSWAHL>

Auswahlen lassen einen Anwender Eins aus einer Liste herausholen. Der anfängliche Anzeigeinhalt wird dem Anwender gezeigt, wonach die Auswahlen folgen. Jede Auswahl kann eine Linie von formatiertem Text haben (der durch das Telefon ungebrochen oder gerollt werden kann, wenn er zu lang ist).

Wie die Auswahlen angezeigt und ausgewählt werden, basiert auf der o-Verfahrensoption. Es ist zu beachten, dass diese Option nur ein Hinweis ist und durch das Telefon unbeachtet bleiben kann. Das Nummernverfahren ist die Vorgabe und zeigt an, dass die Auswahlen von Eins an sequenziell nummeriert sind und durch Drücken der geeigneten Ziffer auf dem Tastenfeld ausgewählt werden. Wenn es mehr als neun Optionen gibt, kann das Telefon irgendein anderes Verfahren einer Auswahl auswählen.

Das Listenverfahren zeigt an, dass die Liste nicht nummeriert werden sollte und dass der Anwender durch die Liste rollen und irgendeine bestimmte Eingabetaste treffen sollte, um eine Eingabe auszuwählen. Das Alphaverfahren ist wie die Liste, nur ist es eine Anzeige dafür, dass der Text der Eingaben dazu verwendet werden sollte, bei einer Auswahl zu helfen, wenn es überhaupt möglich ist. In diesem Fall ist angenommen, dass die Eingaben alphabetisch sortiert sind. Das Gruppenverfahren wird nachfolgend detaillierter beschrieben.

Die o-Tastenooption zeigt dann, wenn sie vorhanden ist, an, dass die Taste eines Arguments zu der Argumentenliste hinzuzufügen ist. Siehe Argumente unten für mehr Information. Der Wert des Arguments kommt von der Auswahl eingabe; siehe unten. Die o-Vorgabenoption zeigt den Vorgabenwert an, wenn der Anwender nur ENTER trifft. Sie o-Vorgabe unter Eingabekarte, unten, für mehr Information.

Auswahleingabe

::= <CE ol(Eingabeoptionen) >

Eingabeoptionen

formatierte Linie

::= Aktionsoptionen | o-Wert

Jede Auswahl hat einen Text, der dem Anwender angezeigt wird. Wenn die Aktionsoptionen gegeben sind, wird die angezeigte Option durchgeführt, wenn die Auswahl durchgeführt ist. Wenn die o-Wertoption vorhanden ist, führt sie den Wert zu dem Argument zu, der mit der o-Tastenooption in dem Auswahl-Anfangsblock identifiziert ist. Wenn kein o-Wert gegeben ist, wird der Text der Eingabe verwendet (ohne irgendeine Formatierung), und zwar als der Argumentenwert.

Gruppeneingabe

::= < GE ol(Gruppenoptionen) >

Gruppenoptionen

::= Etikett= Wert

Wenn das Gruppenverfahren verwendet wird, werden die Auswahlen in einer Anzahl von Gruppen aufgeteilt. Jede Gruppe wird durch eine Gruppeneingabe angeführt, die über die Etikettenoption der Gruppe einen kurzen Namen gibt. Das Telefon kann dann dem Anwender eine hierarchische Schnittstelle zum Auswählen unter einer großen Anzahl von Auswahlen zu teilen. Der Text des Etiketts sollte auf acht Zeichen beschränkt werden und kann durch das Telefon verketet werden.

EINGABEKARTE

Eingabekarte	::= Eingabe-Anfangsblock, Anzeige-Inhalt, Eingabe-Fußende
Eingabe-Anfangsblock	::= <EINGABE ol(Eingabeoptionen) >
Eingabeoptionen	::= Kartenoptionen o-Format o-Taste o-Vorgabe
Eingabe-Fußende	::= </EINGABE> Eingaben lassen den Anwender einen Wert eingeben. Der Anzeigeinhalt wird dem Anwender gezeigt, wonach eine Eingabelinie folgt. Die Eingabe des Anwenders wird durch das Format gesteuert. Die o-Tastenoption zeigt das Argument an, das durch diese Eingabe eingestellt wird. Der Wert des Arguments sind die Eingaben des Anwenders.
o-Format	::= Format= Wert { ; Formathinweis }
Formathinweis	::= Wert Diese Option spezifiziert das Format für Anwender-eingabeeinträge. Die Kette besteht aus Formatsteuerzeichen und statischen Text, der im Eingabebereich angezeigt wird. Die meisten der Formatsteuerzeichen steuern, welche Daten erwartungsgemäß durch den Anwender über Tasten eingegeben werden. Sie werden als Blanks angezeigt, bis der Anwender sie eintippt. Die Formatcodes sind: A: Eingabe von irgendeinem alphabetischen Zeichen 9: Eingabe von irgendeinem numerischen Zeichen X: Eingabe von irgendeinem alphabetischen oder numerischen Zeichen *f: lässt eine Eingabe von irgendeiner Anzahl von Zeichen zu; das nächste Zeichen f ist eines von A, 9 oder X und spezifiziert, welche Art von Zeichen eingegeben werden kann. \c: zeigt das nächste Zeichen c im Eingabefeld an; lässt eine Anzeige der formatierenden Zeichen im Eingabefeld zu. Formathinweise zeigen an, welche Art von Wert erwartet wird. Wenn ein Formathinweis nicht verstanden wird, wird er ignoriert. Gegenwärtig definierte Formathinweise sind: Text: Es wird erwartet, dass Text Text ist, ein Verwenden spezieller Eingabetechniken; allgemein folgt *A oder *X Postantwort: wie Text, aber erwarteter Text dient für eine e-Mail-Nachricht oder eine Seite; kann einen Eingabealgorithmus beeinflussen Adressenliste: Eingabe ist eine Liste von e-Mail-Adressen
o-Vorgabe:	::= Vorgabe= Wert Die o-Vorgabeoption führt einen Wert zu, der dann verwendet wird, wenn der Anwender einfach NEXT trifft bzw. drückt. Wenn kein Vorgabewert gegeben ist, dann muss der Anwender einen Wert zuführen.

FORMATIERTER TEXT

formatierter Text	::= { { Flussbild } { Linienformat } Textlinie }*
formatierte Linie	::= Textlinie
Textlinie	::= {Text Bild Textformat Ausrichtungsformat }* Formatierter Text ist das, was dem Anwender in den meisten Karten gezeigt wird. Formatierte Linien werden für Auswahleingaben verwendet.
Textformat	::= <I> <BL> ::= <I> <BL> Die Formatcodes steuern Bold, Italic und Blinken. Die Schrägstrich-Versionen löschen die Formatierung. Ungleich HTML müssen diese nicht streng vernetzt sein und werden über eine Anwendung oder über ein Löschen toleriert. Elemente in formatiertem Text und formatierter Linie beginnen in einem Normalmode (nicht Fett, Italic oder Blinken).
Ausrichtungsformat	::= <Zentriert> <Fett> <TAB> Die Ausrichtungscodes spezifizieren, wie Teile einer Linie auszulegen sind. Der Text, der dem Ausrichtungscodes folgt, ist entweder zentriert oder auf rechts auf derselben Linie wie der andere Text eingestellt. Der Text oder das Bild, der oder das dem Code folgt, wird derart angesehen, dass es der gesamte Text bis zu dem nächsten Ausrichtungscodes oder einem Zeilenumbruch ist. Alle Linien beginnen implizit links ausgerichtet. Es ist zu beachten, dass diese keinen impliziten Zeilenumbruch enthalten, so dass man sowohl links als auch rechts eingestellten Text auf einer einzigen Zeile haben kann. Wenn es zu viel Text und nicht genügend Platz auf der Linie gibt, dann wird in einem Umbruchmode der nicht passende Text zu der nächsten Linie bzw. Zeile bewegt und auf dieselbe Weise ausgerichtet. Wenn man in einem

Zeilenmode ist, kann die Zeile bis zu einem Zusammenlaufen mit zwei Räumen zwischen den links, zentriert und rechts eingestellten Segmenten enden.

Der Tabulatorcode wird dazu verwendet, ausgerichtete Spalten zu erzeugen. Eher als den Tabulator auf spezifische Zeichenpositionen einzustellen, trennt der Tabulatorcode den Text für jede Spalte. Die Breite der Spalte wird durch die maximale Breite des Textes (oder der Bilder) in jeder Zeile bestimmt. Das Ausmaß der Spalten ist von der ersten Zeile mit Tabulatorcodes über die letzte fortgesetzte Zeile mit Tabulatorcodes. Einige Zeilen können weniger Tabulatorcodes als andere haben, in welchem Fall angenommen ist, dass sie keinen Text für die Zusatzspalten haben.

Zeilenformat

::= <UMBRUCH> | <ZEILE>

::=
.

Mehrere Zeilen eines Textes werden durch den
-Code getrennt. Wenn eine Zeile zu lang ist, um auf den Bildschirm zu passen, und wenn man in einem Umbruchmode ist, wird die Zeile bezüglich eines Words auf mehrere Zeilen umgebrochen bzw. getrennt. Wenn man in einem Zeilenmode ist, wird die Zeile als eine Zeile gelassen und wird in horizontaler Richtung gerollt. Elemente für formatierten Text und für formatierte Zeile starten in einem Umbruchmode und können mit entweder dem <UMBRUCH>-Code oder dem <ZEILE>-Code geändert werden. Diese Codes sind ein implizierter Zeilenumbruch.

BILDER

Bild

::= <BILD ol(Bildoptionen)>

::= <INNERHALB EINER ZEILE ol(Bildoptionen)>

Flussbild	<p>Daten innerhalb einer Zeile </INNERHALB EINER ZEILE> ::= <BILD ol(Flussbildoptionen)> ::= <INNERHALB EINER ZEILE ol(Flussbildpotio- nen)> Daten innerhalb einer Zeile </INNERHALB EINER ZEILE> ::= o-Quelle ::= Bildoptionen o-Fluss</p>
Bildoptionen Flussbildoptionen Daten innerhalb einer Zeile: Zeile:	<p>::= ASCII85-Codierung von Bilddaten</p> <p>Bilder werden als große Worte behandelt und werden durch eine Vorgabe einfach als Teil des Textes angezeigt. Flussbilder haben eine Flussoption, die veranlasst, dass sie anders behandelt werden. Die Bilddaten werden in einem getrennten Datenstrom gespeichert, wie es durch die Quellenoption identifiziert ist.</p> <p>Bilder innerhalb einer Zeile werden identisch behandelt, nur die Daten sind ein Teil des aktuellen Datenstroms. ASCII85 ist eine standardmäßige zum Codieren von binären Daten in druckbarem ASCII, wobei je alle vier Bytes von Daten in fünf Zeichen codiert werden. Es ist zu beachten, dass TIL nur Bilder innerhalb einer Zeile verwendet und eine andere Codierung verwendet.</p>
o-Quelle	<p>::= src= Stelle Diese Option spezifiziert die Stelle der Quelle für Bilder.</p>
o-Fluss	<p>::= Fluss= { links rechts } Diese Option steuert die Ausrichtung von Flussbildern. Die Option spezifiziert, dass das Bild links ausgerichtet oder rechts ausgerichtet mit dem Bildschirm ist. Nachfolgende Zeilen eines Textes fließen in den übrigen rechten oder linken Raum.</p>

WEICHE TASTEN BZW. SOFTKEYS

Softkey
Softkey-Optionen

::= <SOFTKEY ol(Softkey-Optionen) >
::= o-Etikett | o-Taste | Aktionsoptionen

Softkeys führen Definitionen für zwei Tasten zu, die als SOFT-L und SOFT-R bekannt sind. Sie zeigen sich nicht im normalen Text und Graphikbereich, die dem Anwender gezeigt werden, sondern auf einer separaten Linie für Softkey-Etiketten. (Beachte: bei einigen Implementierungen, bei welchem ein Grundbesitz bzw. Raum für den Bildschirm spärlich ist, kann diese Etikettenlinie dahin gelangen, für eine Anzeige von normalem Text und Graphik verwendet zu werden, wenn es keine Softkeys gibt, die auf der aktuellen Karte definiert sind).

Wenn die weiche Taste bzw. der Softkey gedrückt wird, findet die angezeigte Aktion statt.

o-Taste
Seite
o-Etikett

::= Taste= Seite
::= links | rechts
::= Etikett= Wert

Die Tastenoption spezifiziert, auf welche physikalische Taste der Softkey angewendet wird. Die Etikettenoption ist der Text, der auf einem Bildschirm für diese Taste angezeigt wird. Das Telefon kann das Etikett verketteten. Es wird vorgeschlagen, dass Etiketten weniger als acht Zeichen sind.

Softkeys können sowohl für den Posten als Gesamtes als auch pro Karte spezifiziert werden. Wenn sie

für den Posten spezifiziert werden (hinter dem Posten-Anfangsblock, aber vor der ersten Karte), bleiben sie für den gesamten Posten in Wirkung. Wenn sie für eine Karte spezifiziert sind (am Anfang des formatierten Textes für die Karte), gehen sie temporär über irgendwelche Posten-Softkeys hinaus, während die Karte sichtbar ist. Es ist zu beachten, dass das darüber hinausgehen unabhängig für die zwei Tasten durchgeführt wird (eine Karte kann über irgendeinen Softkey hinausgehen, über nicht über den anderen). Zum Hinausgehen über einen Posten-Softkey mit keinem Softkey (in der Wirkung zum Entfernen eines Softkeys für die Dauer einer Karte) erfolgt eine Verwendung eines Softkeys ohne Etikett und ohne Aktion:

SYNTAX: OPTIONEN

Viele der Syntaxelemente von PIDL haben Optionslisten, die zu ihnen gehören. Optionen verfeinern die Operation der Elemente, von welcher sie ein Teil sind. Solange es nicht anders angegeben ist, verschachteln sich Optionen nicht, selbst wenn dieselbe Option in zwei verschachtelten Elementen gegeben ist. Optionen, die für ein Element nicht definiert sind, werden ignoriert, selbst wenn sie für ein umgebendes Element gültig sind.

ol(gültige Option)

Optionsliste

Option

Taste

Wert

::= { Blank-Raum gültige Option }*

{Blank-Räume }

::= ol(Option)

::= Taste = Wert

::= Identifizierer

::= Klartext bzw. unverschlüsselter Text

::= "{ Text }"

Eine Optionsliste enthält Null oder mehr Optionen. Jede Option ist durch einen Blank-Raum (erforderlich!) getrennt und optional folgt ihr ein Blank-Raum. In den Syntaxdiagrammen sind Optionslisten gezeigt als: ol(gültige Option), wobei gültige Option durch ein Element ersetzt wird, das die möglichen Optionen in diesem Zusammenhang definiert. ol ist eine allgemeine Syntaxbeschreibung von Optionslisten.

Jede Option ist eine Taste und ein Wert. Sie können in irgendeiner Reihenfolge innerhalb der Liste von Optionen angegeben sein. Die Taste ist immer ein alphanumerischer Name, der gegenüber einem Fall unempfindlich ist. Der Wert kann dann, wenn er aus nur alphanumerischen Zeichen zusammengesetzt ist, direkt nach dem Gleichheitszeichen erscheinen. Sonst muss der Wert in Anführungszeichen gesetzt werden. In Anführungszeichen wird ein Blank-Raum literal behandelt und wird als Teil des Werts angesehen.

In den Syntaxdiagrammen sind die möglichen Werte für verschiedene Optionen ohne Anführungszeichen spezifiziert. Jedoch sind Anführungszeichen um einen Optionswert immer akzeptierbar.

Ungleich nahe zu allen anderen Syntaxelementen ist ein Blank-Raum zwischen der Taste und dem Gleichheitszeichen oder zwischen dem Gleichheitszeichen und dem Wert nicht zugelassen.

Viele Optionen haben eine beschränktere Gruppe von möglichen Werten als es durch die obige Syntax dargestellt ist. Siehe die individuellen Optionen für Details.

ZIELOORTE

Zielort	::= Stelle { ; Animation } ::= Kartenstelle { ; Animation } ::= Stapeloperation { ; Animation }
Stelle	::= vollständige Stelle Teilstelle relative Stelle
vollständige Stelle	::= Dienst-Id / Postenpfad ::= Dienst-Host / Postenpfad
Teilstelle	::= Postenpfad
relative Stelle	{ ../ }* Postenpfad
Kartenstelle	::= # Identifizierer
Postenpfad	::= Klartext { / Klartext }*
Dienst-Id	::= % Klartext
Dienst-Host	::= Klartext
	Zielorte werden in einigen Optionen zum Anzeigen des nächsten oder vorherigen Postens, oder Karte, zum Zeigen verwendet. Ein Posten ist entweder mit einer vollständigen Stelle (Dienst-Id und Postenpfad), nur einem Postenpfad (in welchem Fall der Dienst derselbe wie der aktuelle Dienst des Postens ist) oder einem relativen Postenpfad spezifiziert. Im letzteren Fall ist die letzte Komponente des aktuellen Posten- pfads entfernt (und eine zusätzliche Komponente für jeden ../ im relativen Postenpfad) und der Postenpfad angehängt.
	Eine bestimmte Karte kann ein Zielort sein und ist durch ein Kartenstellenelement spezifiziert.
Stapeloperation	::= + Kartenstelle ::= – Zusätzlich zu der normalen Vorgeschichtenliste darü- ber, wo ein Anwender gewesen ist, die durch ein

Telefon gehalten wird, hält das Telefon auch einen kurzen Stapel von Stellen bzw. Lokalisierungen. Ein Verwenden einer Pluszeichenform veranlasst, dass die aktuelle Lokalisierung (Posten und Karte und Lokalisierung in der Vorgeschichtenliste und verwendete Animation) auf den Stapel gelegt wird, bevor zur neuen Karte gegangen wird. Ein Verwenden der Minuszeichenform veranlasst einen Rücksprung zu der Stelle bzw. Lokalisierung am obersten Ende des Stapels und das die Vorgeschichtenliste zur gesicherten Stelle gekürzt wird. Wenn keine Animation gegeben ist, wird die inverse Animation verwendet. Vom Stapel wird abgehoben.

Animation

::= slideN | slideS
 ::= slideW | slideE
 ::= slideSW | slideNE
 ::= slideSE | slideNW
 ::= flipV
 ::= flipH
 ::= fade bzw. schwinden
 ::= nichts

Das optionale Animationsargument zeigt an, welche Form von Bildschirmanimation, wenn sie verfügbar ist, zu verwenden ist, wenn zum Zielort gegangen wird. An die Animation wird sich mit dem Zielort in der Vorgeschichte und dem Zielortstapel erinnert. Wenn sich der Anwender zu einem Zielort über eine "geh"- oder "nächste"-Operation bewegt, dann wird die Animation durchgeführt. Wenn sich der Anwender zu einem Zielort über eine "vorherige" oder "Abnahme"-Operation bewegt, wird die zu der aktuellen Stelle gehörende umgekehrte Animation durchgeführt.

AKTIONEN

o-gehen
 o-Aufruf

Auswahleingaben und Softkeys können Aktionen spezifizieren, die durchzuführen sind, wenn der Anwender die Auswahl oder den Softkey auswählt. Aktionsoptionen ::= o-args | o-Aufruf | o-Seite
 ::= gehen= Zielort
 ::= Aufruf= Wert
 Die Geh-Operation zeigt an, dass der Zielort bewegt werden sollte.

ARGUMENTENVERARBEITUNG

Jedes Mal dann, wenn ein Posten angefordert wird, können Argumente zusammen mit der Anforderung geführt bzw. übergeben werden. Diese Argumente können durch das Dienstende verwendet werden, um einen Posten zu berechnen, der für den Anwender spezifisch ist, eher als nur einen im Voraus geschriebenen zurückzubringen.

Argumente werden gebildet, wenn der Anwender den Posten durchquert. Jedes Argument ist ein Tasten/Wert-Paar. Während Argumente oberflächlich wie Optionen ausschauen, sind diese zwei Einheiten bzw. Größen völlig unterschiedlich: Optionen sind ein Teil von PIDL und beeinflussen die Operation des Telefons. Argumente sind Informationen, die durch das Telefon gesammelt und zum Dienst zurückgebracht werden. Weder PIDL noch das Telefon versteht die Argumente über ihre Basis-Syntaxstruktur hinausgehend.

Argumente kommen von drei Stellen: Auswahlkarten, Eingabekarten und der Argumentenoption. Jedes von diesen spezifiziert ein Tasten-Wert-Paar, das zu einem Puffer von zu sendenden Argumenten hinzu

gefügt ist. In dem Fall der Argumentenoption können mehrere Argumente spezifiziert werden. Wenn ein Argument Tasten/Wert-Paar zu dem Argumentenpuffer hinzugefügt wird, wird dann, wenn die Taste bereits im Puffer vorhanden ist, ihr Wert ersetzt.

o-Argumente
Argumentenliste

Argument-Taste-Wert
Argument-Taste
Argument-Wert

::= Argumente= Argumentenliste
::= Argument-Taste-Wert { { & | ; }
Taste-Wert }*
::= Argument-Taste = Argument-Wert
::= Identifizierer
::= Klartext

Es zu beachten: Das gesamte o-Argumentenelement ist aktuell der Wert einer Option. Wenn es mehr als einen Argument-Taste-Wert hat, wird es in Anführungszeichen zu setzen sein. Da das Und-Zeichen (&) und der Strichpunkt (;) als Tasten/Wert-Paar Trennungen verwendet werden, können diese Zeichen kein Teil von Argumentenwerten sein.

o-Taste
o-Wert

::= Taste= Argumententaste
::= Wert= Argumentenwert
Diese Optionen werden bei Auswahl- und Eingabekarten verwendet, um die Taste und den Wert für die Argumente zu spezifizieren, für welche diese Karten eingesetzt sind.

BASISELEMENTE

alpha	::= irgendein alphabetisches Zeichen
numerisch	::= irgendeine Ziffer
alphanumerisch	::= alpha numerisch
hex	::= numerisch irgendein Buchstabe von A bis F, je nach Fall
Blank-Raum	::= {Raum Tabulator neue Zeile }+
Raum	::= das Raumzeichen
Tabulator	::= das Tabulatorzeichen
neue Zeile	::= das Wagenrückführzeichen ::= das Zeilenzufuhrzeichen ::= die Sequenzwagenrückführung, Zeilenzufuhr
Wort	::= { alphanumerisch }+
Identifizierer	::= alpha { alphanumerisch }*
ganze Zahl	::= { + - } { numerisch }+
Text	::= irgendein 7-Bit-ASCII-Zeichen außer <, >, " oder & ::= > < " & ; ::= irgendeine Einheit, die ISO-Latin-1 genannt wird ::= &# hex hex; In einem Text werden Durchläufe eines Blank-Raums als einzelne Räume behandelt und können als Stelle für einen Wortumbruch bzw. eine Worttrennung verwendet werden.
Klartext	
Sicher	::= { alpha numerisch sicher }Ü ::= \$ - _ @ . & ! *,

TIL-CODIERUNG

Außer wo es angemerkt ist, ist TIL identisch zu PIDL bezüglich der Struktur. Zum Umsetzen von PIDL in TIL sind konzeptmäßig mehrere Schritte nötig (diese können in einem Durchlauf durch eine Umsetzungseinheit durchgeführt werden):

1. Escape-Zeichen mit hoher Bitgruppe.
2. Komprimieren oder entfernen von allen Blank-Räumen, wo es möglich ist.
3. Tokenisierungs-Kommentierungselemente mit einem einzelnen Byte mit der hohen Bitgruppe.
4. Bilder innerhalb einer Zeile.

Grundsätzlich ist TIL nur PIDL mit bestimmten gemeinsamen Zeichensequenzen, die durch einzelne Bytes mit der Gruppe hoher Bits ersetzt sind. Die obigen ersten zwei Schritte unterstützen dies. Zusätzlich werden Bilder dadurch weiter kompakt gemacht, dass sie in einem dichten Format innerhalb einer Zeile enthalten sind.

Die Tokenisierung folgt der in der nachfolgenden Tabelle angegebenen Codierung. Es ist zu beachten, dass zu Zwecken einer Elementtrennung die Token, die Optionstastenidentifizierer (mit dem gleichen Vorzeichen) darstellen, derart betrachtet werden können, dass sie alle vorangehenden Blank-Räume enthalten. Gleichmaßen können die Token, die Optionswerte darstellen, derart angesehen werden, dass sie den gesamten folgenden Blank-Raum enthalten.

<PIDL>	90	Argumente=	C0	Alpha	E0
</PIDL>	91	Taste=	C1	Zentriert	E1
<ANZEIGE>	92	Aufruf=	C2	Fade bzw. Schwund	E2
</ANZEIGE>	93	Kosten=	C3	flipH	E3
<AUSWAHL>	94	Vorgabe=	C4	flipV	E4
</AUSWAHL>	95	Fluss=	C5	Gruppe	E5
<EINGABE>	96	Format=	C6	Innerhalb einer Zeile	E6
</EINGABE>	97	Geh=	C7	Links	E7
<CE	A0	Taste=	C8	Liste	E8
<GE	A1	Etikett=	C9	Nichts	E9
<BILD	A2	Verfahren=	CA	Zahl	EA
<INNERHALB EINER ZEILE	A3	Name=	CB	Rechts	EB
<SOFTKEY	A4	Nächste=	CC	slideE	EC
	B0	Seite=	CD	slideN	ED
	B1	Vorherige=	CE	slideNE	EE
<I>	B2	src=	CF	slideNW	EF
</I>	B3	ttl=	D0	slideS	F0
<BL>	B4	Wert=	D1	slideSE	D1
</BL>	B5			slideSW	F2
<ZENTRIERT>	B6			slideW	F3
<RECHTS>	B7				
<UMBRUCH>	B8				
<ZEILE>	B9				
 	BA				

ANHANG II

COMPUTERPROGRAMM ZUM ERZEUGEN EINER BUCHSTABENHÄUFIGKEITSTABELLE ZUR VERWENDUNG IN DEM VORHERSAGEDATENEINGABEPROZESS

Nicht veröffentlichtes © 1995 Libris, Inc.

```
/* Dieses Programm öffnet eine durch den Anwender ausgewählte Textdatei,
   erzeugt die Häufigkeitstabelle für diese Datei und schreibt dann die Häufig-
   keitstabelle zu einer anderen Datei, die auch durch den Anwender ausgewählt
   ist.
```

```
*/
```

```
#include <stdio.h>
#include <string.h>
#include <console.h>
#include <assert.h>
```

```
typedef unsigned char byte;
typedef byte triplet[3];
typedef byte tristorage[27][27][27];
```

```
IncrementTrigram(triplet t, tristorage trigrams)
{
    byte * pb;
    assert(t[0] < 27);
    assert(t[1] < 27);
    assert(t[2] < 27);
    pb = &trigrams[t[0]][t[1]][t[2]];
    if (*pb < 255) *pb = *pb + 1;
    return *pb;
}
```

```
StoreTrigramValue(triplet t, tristorage trigrams, byte
value)
{
```

```

    assert(t[0] < 27);
    assert(t[1] < 27);
    assert(t[2] < 27);
    trigrams[t[0]][t[1]][t[2]] = value;
}

```

```

byte FetchTrigramvalue(triplet t, tristorage trigrams)
{
    assert(t[0] < 27);
    assert(t[1] < 27);
    assert(t[2] < 27);
    return trigrams[t[0]][t[1]][t[2]];
}

```

```

byte DumpTrigram(triplet t, tristorage trigrams)
{
    byte value;
    assert(t[0] < 27);
    assert(t[1] < 27);
    assert(t[2] < 27);
    value = FetchTrigramValue(t, trigrams);
    if (value != 0)
    {
        printf("%c%c%c = ", t[0] + 'a', t[1] + 'a', t[2] +
'a');
        if (value == 255) printf("***");
        else printf("%3d", value);
    }
    return value;
}

```

```

int IdFromChar(short c)
{
    c = tolower(c);
    if (c < 'a' || c > 'z') return 26;
    return c - 'a';
}

```



```

}
```

```

AddChar(tristorage trigrams, triplet t, byte b)
```

```

{
```

```
    byte value;
```

```
    unsigned short r;
```

```
    assert(b <= 26);
```

```
    if (b == 26) { t[0] = t[1] = t[2] = 26; return; }
```

```
    t[0] = t[1];
```

```
    t[1] = t[2];
```

```
    t[2] = b;
```

```
    value = FetchTrigramValue(t, trigrams);
```

```
    if (value == 255) return;
```

```

#endif
```

```
    if (value > 64) {
```

```
        r = Random();
```

```
        if (value > 192 && r & 0xE000) return;
```

```
        else if (value > 128 && r & 0xC000) return;
```

```
        else if (value > 64 && r & 0x8000) return;
```

```
    }
```

```

#endif
```

```
    StoreTrigramValue(t, trigrams, value + 1);
```

```

}
```

```

DumpTrigrams(tristorage trigrams)
```

```

{
```

```
    int i, j, k;
```

```
    int x;
```

```
    triplet t;
```

```
    x = 0;
```

```
    for (i = 0; i < 26; ++i)
```

```

for (j = 0; j < 26; ++j)
    for (k = 0; k < 26; ++k)
    {
        byte value;
        t[0] = i;
        t[1] = j;
        t[2] = k;

        value = DumpTrigram(t, trigrams);

        if (value == 0) continue;
        if (++x == 6) {
            printf("\n"); x = 0;
        }
        else
            printf(" ");
    }
}

OSErr BuildTrigram(short refNum, tristorage trigrams)
{
    OSErr err;
    triplet t;
    t[0] = t[1] = t[2] = 26;

    while (true)
    {
        long count = 80;
        char buf[80];
        int i;

        err = FSRead(refNum, &count, buf);
        if (count == 0) return err;
        for (i = 0; i < count; ++i) {
            AddChar(trigrams, t, IdFromChar(buf[i]));
        }
    }
}

```

```

    }
    if (err) return err;
}
return 0;
}

```

```

Handle OpenTrigrams(void)

```

```

{
    OSErr err;
    OSType type;
    StandardFileReply reply;
    short refNum;
    short id;
    Handle h;
    Str63 name;
    tristorage *trigrams;

    type = 'TEXT';
    StandardGetFile(nil, 1, &type, &reply);
    if (!reply.sfGood) return nil;
    err = FSpOpenDF(&reply.sfFile, fsCurPerm, &refNum);
    if (err) return nil;

    memcpy(name, reply.sfFile.name, sizeof(name));

    h = NewHandle(sizeof(tristorage));

    HLock(h);
    trigrams = (tristorage *) (*h);
    memset(*trigrams, 0, sizeof(tristorage));

    BuildTrigram(refNum, *trigrams);
    FSClose(refNum);
    DumpTrigrams(*trigrams);
    HUnlock(h);
}

```

```

type = 'rsrc';
StandardGetFile(nil, 1, &type, &reply);
if (!reply.sfGood) return;
refNum = FSpOpenResFile(&reply.sfFile, fsCurPerm);
if (refNum == -1) return;
UseResFile(refNum);

id = UniqueID('smrt');
//id = 128;
AddResource(h, 'smrt', id, name);
UpdateResFile(refNum);
FSClose(refNum);
return h;
}

main()
{
    OSErr    err;
    Handle    h;

    cshow(stdout);
    TEInit();
    InitDialogs(OL);
    InitCursor();
    h = OpenTrigrams();

```

Patentansprüche

1. Verfahren, das in einem Netzwerkknoten (**500, 743**) implementiert ist, der mit einem drahtlosen Telekommunikationsnetzwerk (**110, 111**) und mit einem drahtgebundenen Computernetzwerk (**120, 130, 140**) gekoppelt ist, zum Ermöglichen, dass jeweilige zu einer drahtlosen Kommunikation fähige mobile Vorrichtungen (**100, 101**) in Taschengröße für Zweirichtungsdaten eine interaktive Zweirichtungskommunikation mit einer Anwendung haben, wobei das Verfahren folgendes aufweist:

über das drahtlose Telekommunikationsnetzwerk (**110, 111**) Empfangen einer Anfrage von der mobilen Vorrichtung, eine Anwendung auf einem Server (**121, 131, 141**) auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) auszuführen;

Senden einer Anfrage über das drahtgebundene Computernetzwerk zu dem Server, um auf die Anwendung zuzugreifen;

gekennzeichnet durch

Empfangen einer Antwort über das drahtgebundene Computernetzwerk gemäß der Ausführung der Anwendung, auf die zugegriffen ist, wobei die Antwort aus durch die mobile Vorrichtung zu interpretierender Information besteht und wobei daraus eine interaktive Anwenderschnittstelle erzeugt wird und wobei daraus eine weitere An-

frage zu dem oder zu einem anderen Server gemäß der von dieser erzeugten Anwenderschnittstelle durchgeführten Verwendung erzeugt wird; und
Senden der Antwort zu der mobilen Vorrichtung über das drahtlose Telekommunikationsnetzwerk.

2. Verfahren nach Anspruch 1, wobei eine Gruppe von einer oder mehreren zugehörigen interaktiven Anwenderschnittstellen aus der Information erzeugt wird.

3. Verfahren nach Anspruch 2, wobei die oder jede interaktive Anwenderschnittstelle eine oder mehrere Bildschirmanzeigen (**300, 304, 306, 308**) an der mobilen Kommunikationsvorrichtung (**100, 101**) aufweist.

4. Verfahren nach Anspruch 2 oder 3, das weiterhin ein Betreiben des Netzwerkknotens (**500, 743**) aufweist, um die Gruppe in Antwort auf die Anfrage dynamisch zu erzeugen.

5. Verfahren nach einem der Ansprüche 2 bis 4, wobei eine oder mehrere Anwenderschnittstellen der Gruppe eine oder mehrere Aufgaben spezifizieren, um auf der mobilen Kommunikationsvorrichtung (**100, 101**) durchgeführt zu werden.

6. Verfahren nach einem der vorangehenden Ansprüche, das weiterhin ein Umwandeln der Antwort aus einer ersten Sprache, die auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) verwendet wird, in eine zweite Sprache, die auf dem drahtlosen Telekommunikationsnetzwerk (**110, 111**) verwendet wird, aufweist.

7. Verfahren nach Anspruch 6, wobei die Antwort ein Markup-Language-Dokument aufweist.

8. Verfahren nach Anspruch 6 oder 7, wobei die zweite Sprache eine destillierte Form der ersten Sprache ist und wobei ein Senden der übersetzten Antwort zu der mobilen Kommunikationsvorrichtung (**100, 101**) ein Senden der Antwort zu der mobilen Kommunikationsvorrichtung (**100, 101**) in der zweiten Sprache über das drahtlose Telekommunikationsnetzwerk (**110, 111**) aufweist, so dass die zu der mobilen Kommunikationsvorrichtung (**100, 101**) gesendete Antwort eine komprimierte Form der durch den Netzwerkknoten (**500, 743**) von einem Server an dem drahtgebundenen Computernetzwerk (**120, 130, 140**) empfangenen Antwort ist.

9. Verfahren nach einem der vorangehenden Ansprüche, wobei der Netzwerkknoten (**500, 743**) einen Gateway-Server aufweist, um das drahtlose Telekommunikationsnetzwerk (**110, 111**) mit dem drahtgebundenen Datennetzwerk zu koppeln.

10. Verfahren nach einem der Ansprüche 1 bis 8, wobei der Netzwerkknoten (**500, 743**) einen Proxy-Server aufweist, um Anfragen von der mobilen Kommunikationsvorrichtung (**100, 101**) zu entfernten Servern auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) zu vertreten bzw. zu bevollmächtigen.

11. Verfahren nach Anspruch 1, das weiterhin folgendes aufweist:
Betreiben des Netzwerkknotens (**500, 743**), um mit der mobilen Kommunikationsvorrichtung (**100, 101**) über das drahtlose Telekommunikationsnetzwerk (**110, 111, 112**) unter Verwendung eines ersten Protokolls zu kommunizieren; und
Betreiben des Netzwerkknotens (**500, 743**), um über das drahtgebundene Computernetzwerk (**120, 130, 140**) unter Verwendung eines zweiten Protokolls zu kommunizieren, das unterschiedlich von dem ersten Protokoll ist.

12. Verfahren nach einem der vorangehenden Ansprüche, wobei der Netzwerkknoten (**500, 743**) einen HTTP-Server aufweist.

13. Verfahren nach Anspruch 12, wobei der Netzwerkknoten (**500, 743**) ein UDP-Modul zusätzlich zu dem HTTP-Server aufweist und wobei der HTTP-Server das UDP-Modul verwendet, um Daten mit dem drahtlosen Telekommunikationsnetzwerk (**110, 111**) zu kommunizieren.

14. Verfahren nach einem der vorangehenden Ansprüche, das weiterhin ein Betreiben des Netzwerkknotens (**500, 743**) aufweist, um einen Zugriff auf Betriebsmittel auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) durch die mobile Kommunikationsvorrichtung (**100, 101**) zu steuern.

15. Verfahren nach Anspruch 14, wobei ein Betreiben des Netzwerkknotens (**500, 743**) zum Steuern eines Zugriffs auf Betriebsmittel auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) ein Betreiben des

Netzwerkknotens (**500, 743**) zum Autorisieren von Anfragen durch die mobile Vorrichtung (**100, 101**) aufweist.

16. Verfahren nach einem der vorangehenden Ansprüche, wobei die mobile Kommunikationsvorrichtung ein Client-Modul (**702**) enthält, das nur eine leichtgewichtige Speicher- und Verarbeitungsenergie zum Arbeiten erfordert.

17. Verfahren nach einem der vorangehenden Ansprüche, das weiterhin ein Steuern eines Zugriffs durch die mobilen Vorrichtung auf bezahlungsbasierende Dienste auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**), einschließlich eines Sammelns von Transaktions- und Berechnungsinformation, die zu einem Bereitstellen von Betriebsmitteln auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) zu den mobilen Vorrichtungen gehört, aufweist.

18. Netzwerkknoten (**500, 743**), der folgendes aufweist:
 eine erste Kommunikationsschnittstelle (**748, 1202**) zum Kommunizieren mit zu einer drahtlosen Kommunikation fähigen mobilen Vorrichtungen (**100, 101**) in Taschengröße für Zweiwegedaten über ein drahtloses Telekommunikationsnetzwerk (**110, 111**);
 eine zweite Kommunikationsschnittstelle (**1203**), die zu einer Kommunikation mit einem Server (**121, 131, 141**) über ein drahtgebundenes Computernetzwerk (**120, 130, 140**) fähig ist; und
 einen Prozessor (**1204**), der mit der ersten Kommunikationsschnittstelle und der zweiten Kommunikationsschnittstelle gekoppelt ist, um einen Prozess auszuführen, wobei der Prozess folgendes enthält:
 über das drahtlose Telekommunikationsnetzwerk Empfangen einer Anfrage von der mobilen Vorrichtung, um eine Anwendung auszuführen, auf einem Server auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**);
 Senden der Anfrage über das drahtgebundene Computernetzwerk zu dem Server, um auf die Anwendung zuzugreifen; gekennzeichnet durch
 Empfangen einer Antwort über das drahtgebundene Computernetzwerk gemäß der Ausführung der Anwendung, auf die zugegriffen ist, wobei die Antwort aus durch die mobile Vorrichtung zu interpretierender Information besteht und wobei daraus eine interaktive Anwenderschnittstelle erzeugt wird und wobei daraus eine weitere Anfrage zu dem oder einem weiteren Server gemäß der von der erzeugten Anwenderschnittstelle gemachten bzw. durchgeführten Anwendung erzeugt wird; und
 Senden der Antwort zu der mobilen Vorrichtung über das drahtlose Telekommunikationsnetzwerk.

19. Netzwerkknoten nach Anspruch 18, wobei eine Gruppe von einer oder mehreren zugehörigen interaktiven Anwenderschnittstellen aus der Information erzeugt wird.

20. Netzwerkknoten nach Anspruch 19, wobei die oder jede interaktive Anwenderschnittstelle eine oder mehrere Bildschirmanzeigen (**300, 304, 306, 308**) an der mobilen Kommunikationsvorrichtung (**100, 101**) aufweist.

21. Netzwerkknoten nach Anspruch 19 oder 20, der weiterhin ein dynamisches Erzeugen der Gruppe in Antwort auf die Anfrage aufweist.

22. Netzwerkknoten nach einem der Ansprüche 19 bis 21, wobei eine oder mehrere Anwenderschnittstellen der Gruppe eine oder mehrere Aufgaben spezifizieren, um auf der mobilen Kommunikationsvorrichtung (**100, 101**) durchgeführt zu werden.

23. Netzwerkknoten nach einem der Ansprüche 18 bis 22, der weiterhin ein Umwandeln der Antwort von einer ersten Sprache, die auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) verwendet wird, in eine zweite Sprache, die auf dem drahtlosen Telekommunikationsnetzwerk (**110, 111**) verwendet wird, aufweist.

24. Netzwerkknoten nach Anspruch 23, wobei die Antwort ein Markup-Language-Dokument aufweist.

25. Netzwerkknoten nach Anspruch 23 oder 24, wobei die zweite Sprache eine destillierte Form der ersten Sprache ist und wobei ein Senden der übersetzten Antwort zu der mobilen Kommunikationsvorrichtung (**100, 101**) ein Senden der Antwort zu der mobilen Kommunikationsvorrichtung (**100, 101**) in der zweiten Sprache über das drahtlose Telekommunikationsnetzwerk (**110, 111**) aufweist, so dass die zu der mobilen Kommunikationsvorrichtung (**100, 101**) gesendete Antwort eine komprimierte Form der durch den Netzwerkknoten (**500, 743**) von einem Server auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) empfangenen Antwort ist.

26. Netzwerkknoten nach einem der Ansprüche 18 bis 25, wobei der Netzwerkknoten (**500, 743**) einen Gateway-Server aufweist, um das drahtlose Telekommunikationsnetzwerk (**110, 111**) mit dem drahtgebundenen Datennetzwerk zu koppeln.

27. Netzwerkknoten nach einem der Ansprüche 18 bis 25, wobei der Netzwerkknoten (**500, 743**) einen Proxy-Server aufweist, um Anfragen von der mobilen Kommunikationsvorrichtung (**100, 101**) zu entfernten Servern auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) zu bevollmächtigen bzw. zu vertreten.

28. Netzwerkknoten nach Anspruch 18, der weiterhin folgendes aufweist:
Betreiben des Netzwerkknotens (**500, 743**) zum Kommunizieren mit der mobilen Kommunikationsvorrichtung (**100, 101**) über das drahtlose Telekommunikationsnetzwerk (**110, 111, 112**) unter Verwendung eines ersten Protokolls; und
Betreiben des Netzwerkknotens (**500, 743**) zum Kommunizieren über das drahtgebundene Computernetzwerk (**120, 130, 140**) unter Verwendung eines zweiten Protokolls, das unterschiedlich vom ersten Protokoll ist.

29. Netzwerkknoten nach einem der Ansprüche 18 bis 28, wobei der Netzwerkknoten (**500, 743**) einen HTTP-Server aufweist.

30. Netzwerkknoten nach Anspruch 29, wobei der Netzwerkknoten (**500, 743**) ein UDP-Modul zusätzlich zu dem HTTP-Server aufweist, und wobei der HTTP-Server das UDP-Modul zum Kommunizieren von Daten mit dem drahtlosen Telekommunikationsnetzwerk (**110, 111**) verwendet.

31. Netzwerkknoten nach einem der Ansprüche 18 bis 30, der weiterhin ein Steuern eines Zugriffs auf Betriebsmittel auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) durch die mobile Kommunikationsvorrichtung (**100, 101**) aufweist.

32. Netzwerkknoten nach Anspruch 31, wobei ein Betreiben des Netzwerkknotens (**500, 743**) zum Steuern eines Zugriffs auf Betriebsmittel auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) ein Betreiben des Netzwerkknotens (**500, 743**) zum Autorisieren von Anfragen durch die mobile Vorrichtung (**100, 101**) aufweist.

33. Netzwerkknoten nach einem der Ansprüche 18 bis 32, wobei die mobile Kommunikationsvorrichtung ein Client-Modul (**702**) aufweist, das nur eine leichtgewichtige Speicher- und Verarbeitungsenergie zum Arbeiten erfordert.

34. Netzwerkknoten nach einem der Ansprüche 18 bis 33, der weiterhin ein Steuern eines Zugriffs durch die mobilen Vorrichtungen auf bezahlungsbasierende Dienste auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**), einschließlich eines Sammelns von Transaktions- und Buchungsinformation, die zu einem Bereitstellen von Betriebsmitteln auf dem drahtgebundenen Computernetzwerk (**120, 130, 140**) zu den mobilen Vorrichtungen gehört, aufweist.

Es folgen 36 Blatt Zeichnungen

Anhängende Zeichnungen

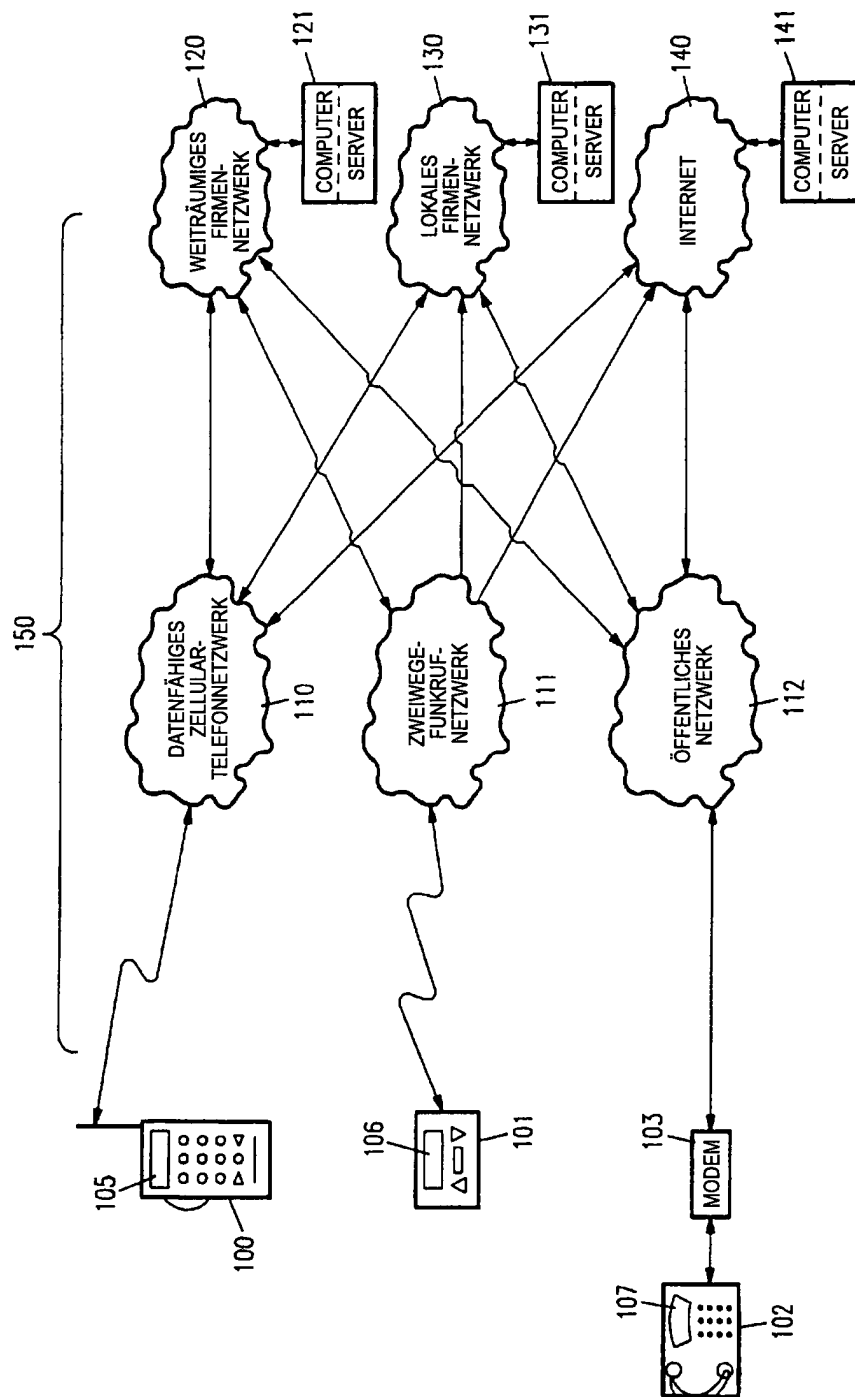


FIG. 1

FIG. 2A

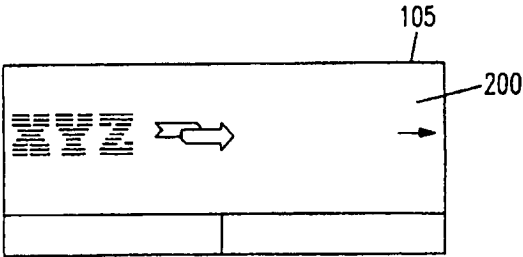


FIG. 2B

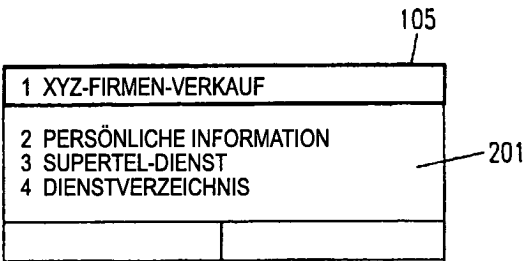


FIG. 2C

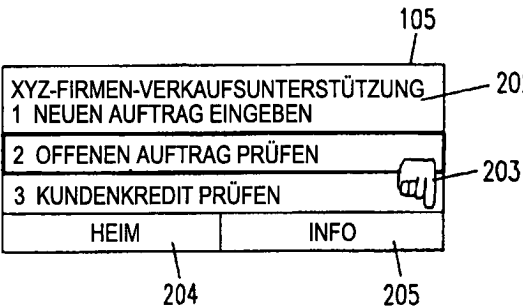


FIG. 2D

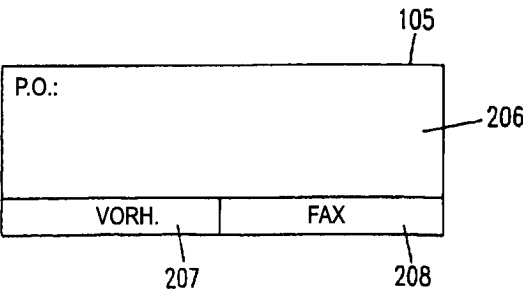


FIG. 2E

105

P.O.: 11-11-11	
VORH.	FAX

206

207

208

FIG. 2F

105

P.O.: 11-11-11 KUNDE: ABC Designs DATUM: 3. MÄRZ 1994 VERSAND: 7. MÄRZ 1994	
VORH.	FAX

209

203

207

208

FIG. 2G

105

FAXDETAILS ZU WELCHER NUMMER:	
VORH.	

FIG. 2H

105

FAXDETAILS ZU WELCHER NUMMER:	
(415) 341-4473	
VORH.	

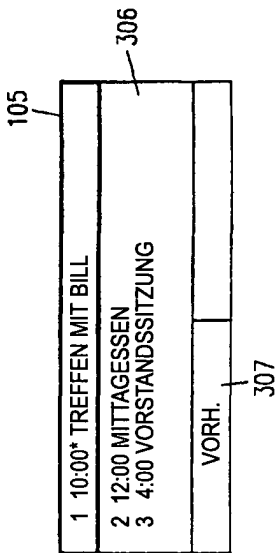


FIG. 3D

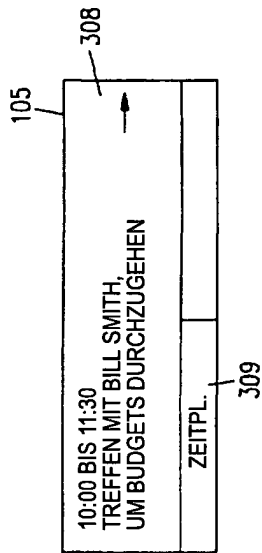


FIG. 3E

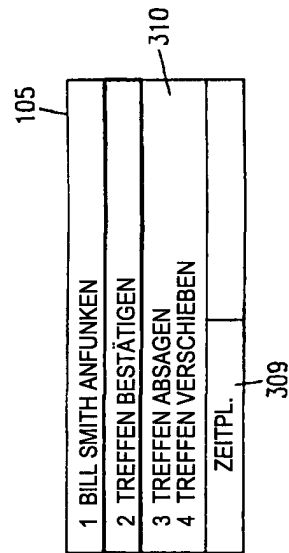


FIG. 3F

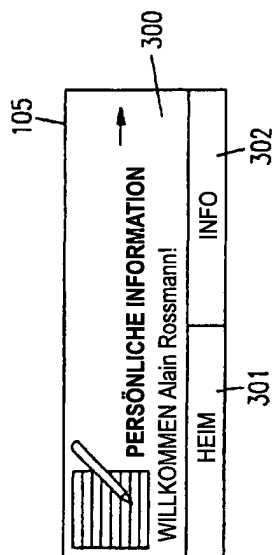


FIG. 3A

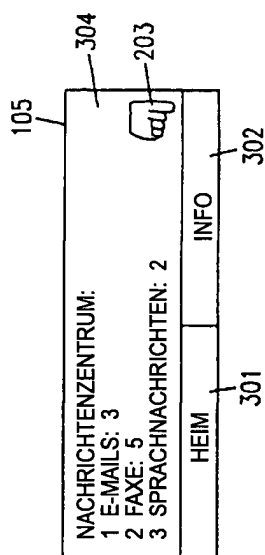


FIG. 3B

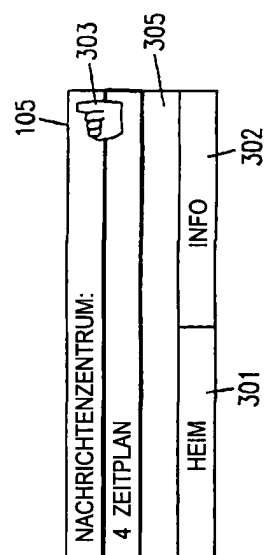
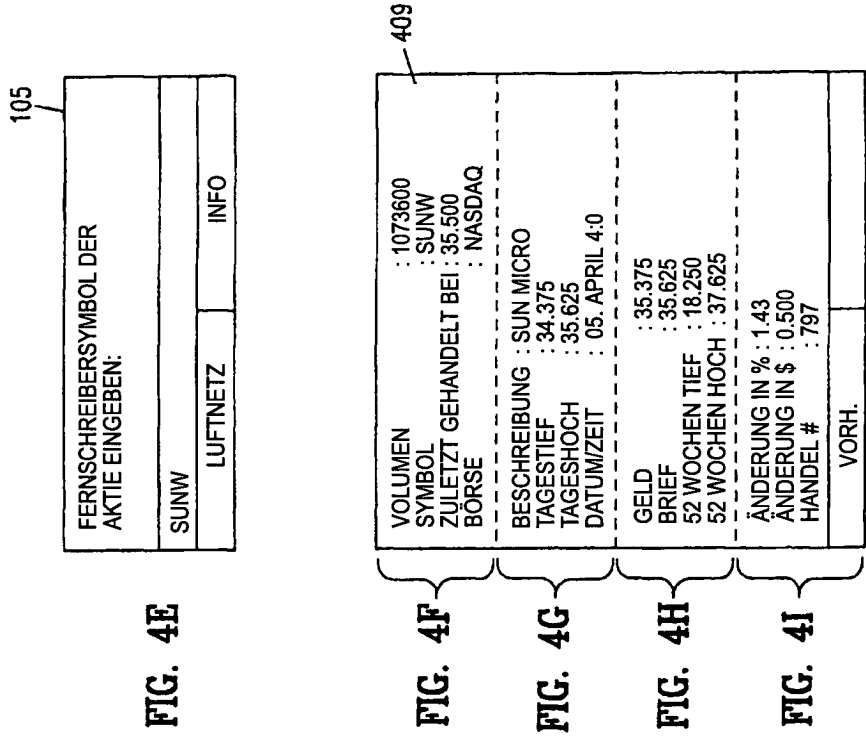
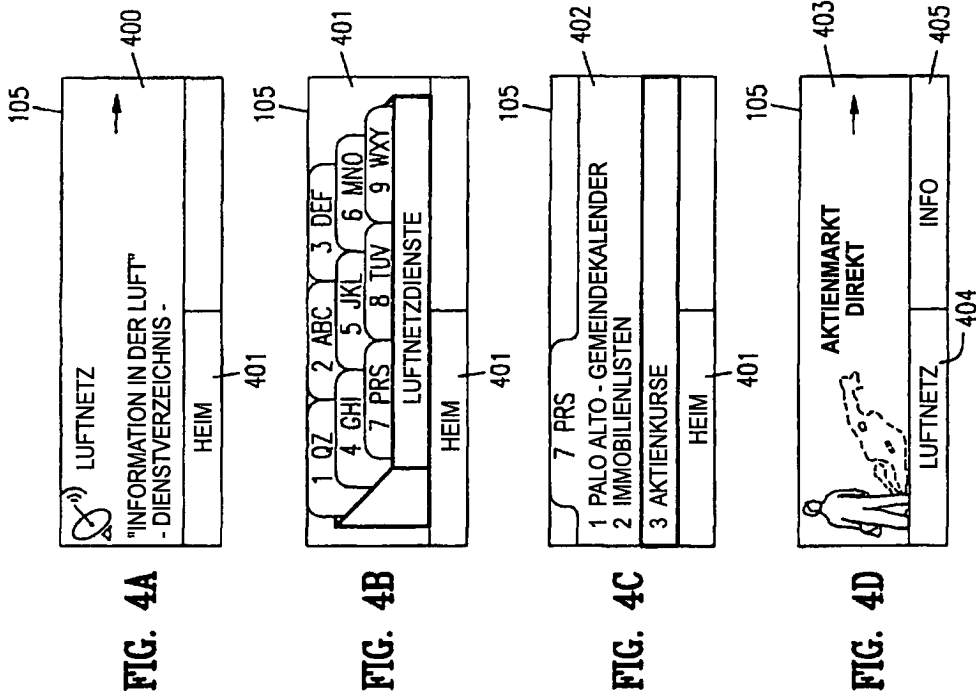


FIG. 3C



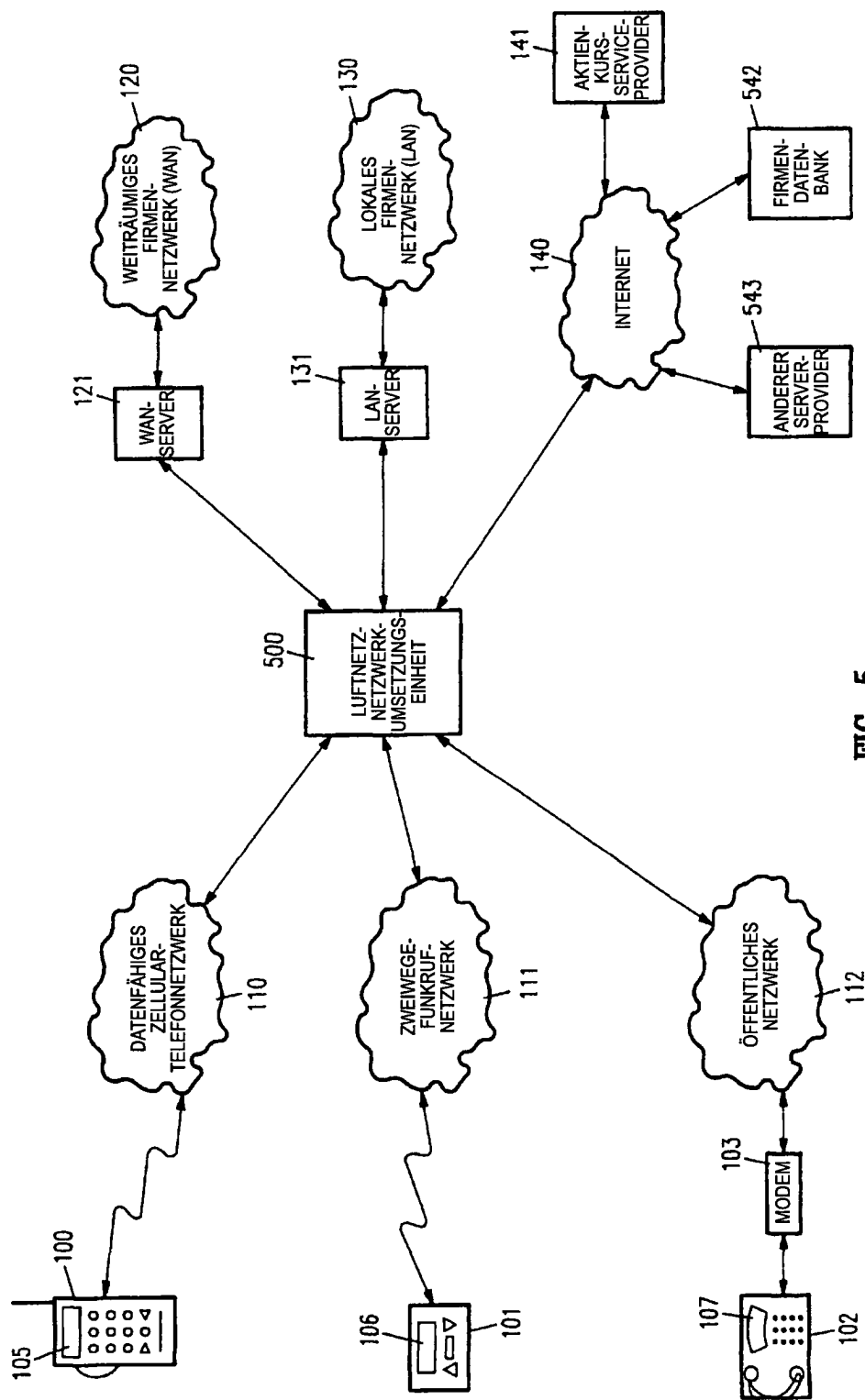


FIG. 5

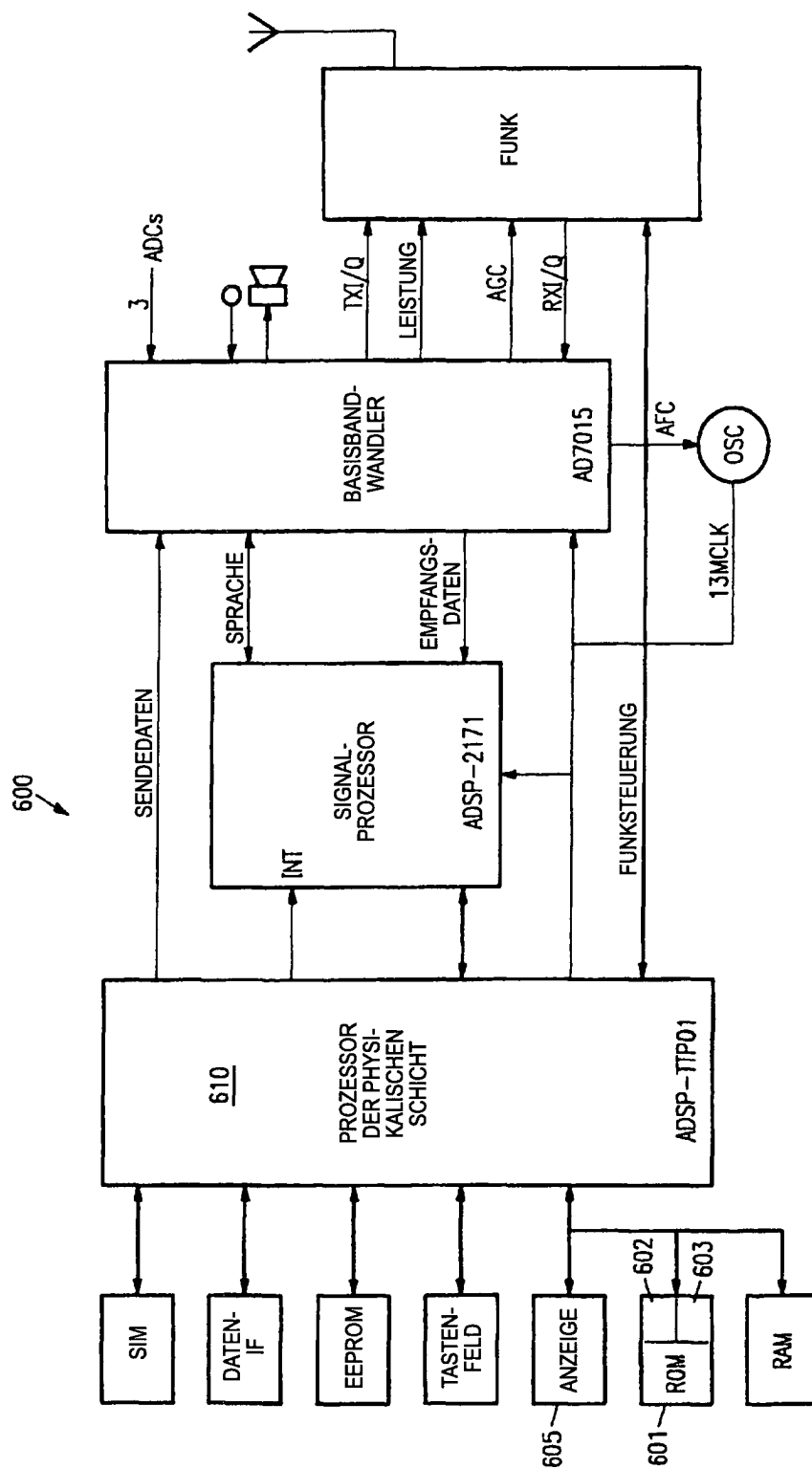


FIG. 6

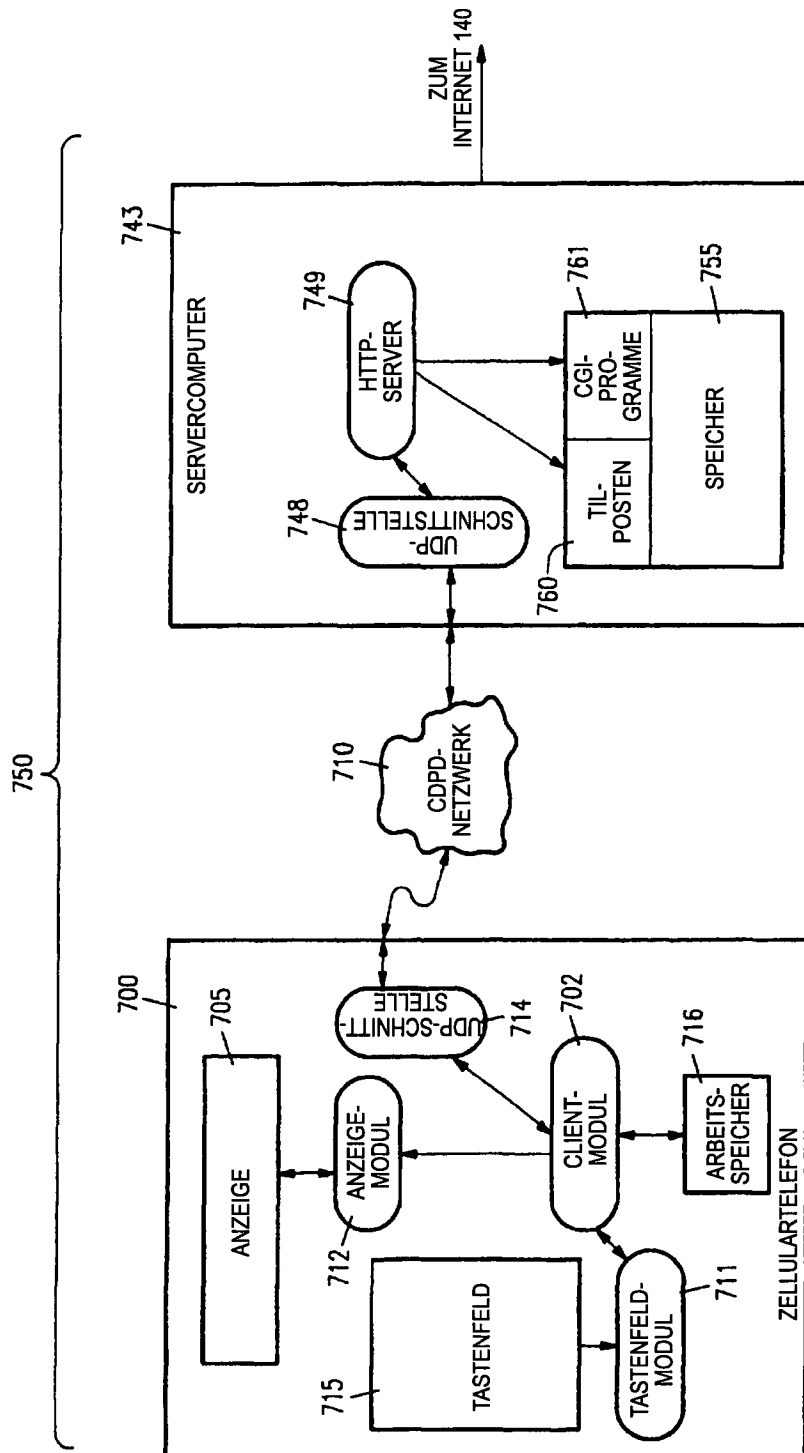


FIG. 7

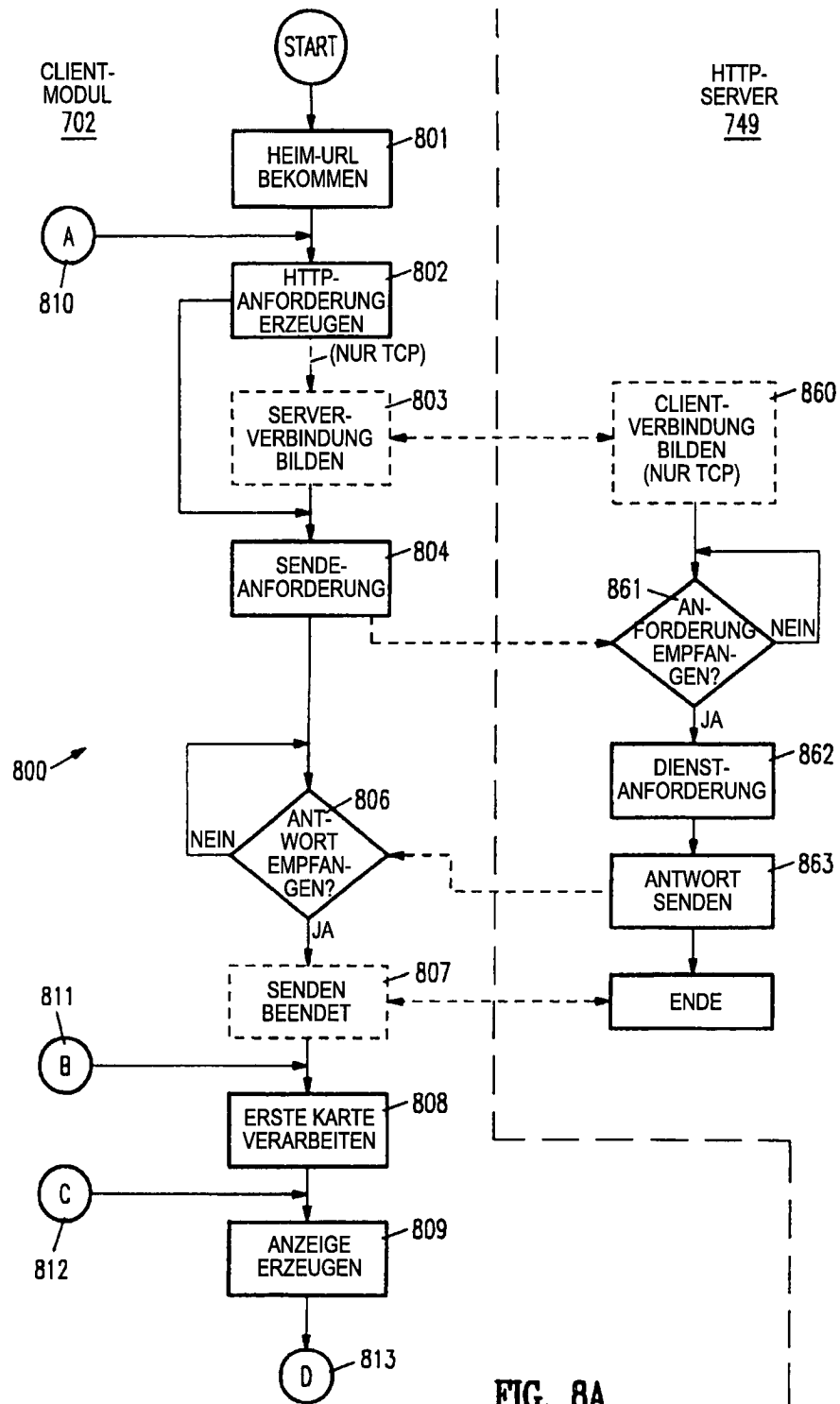


FIG. 8A

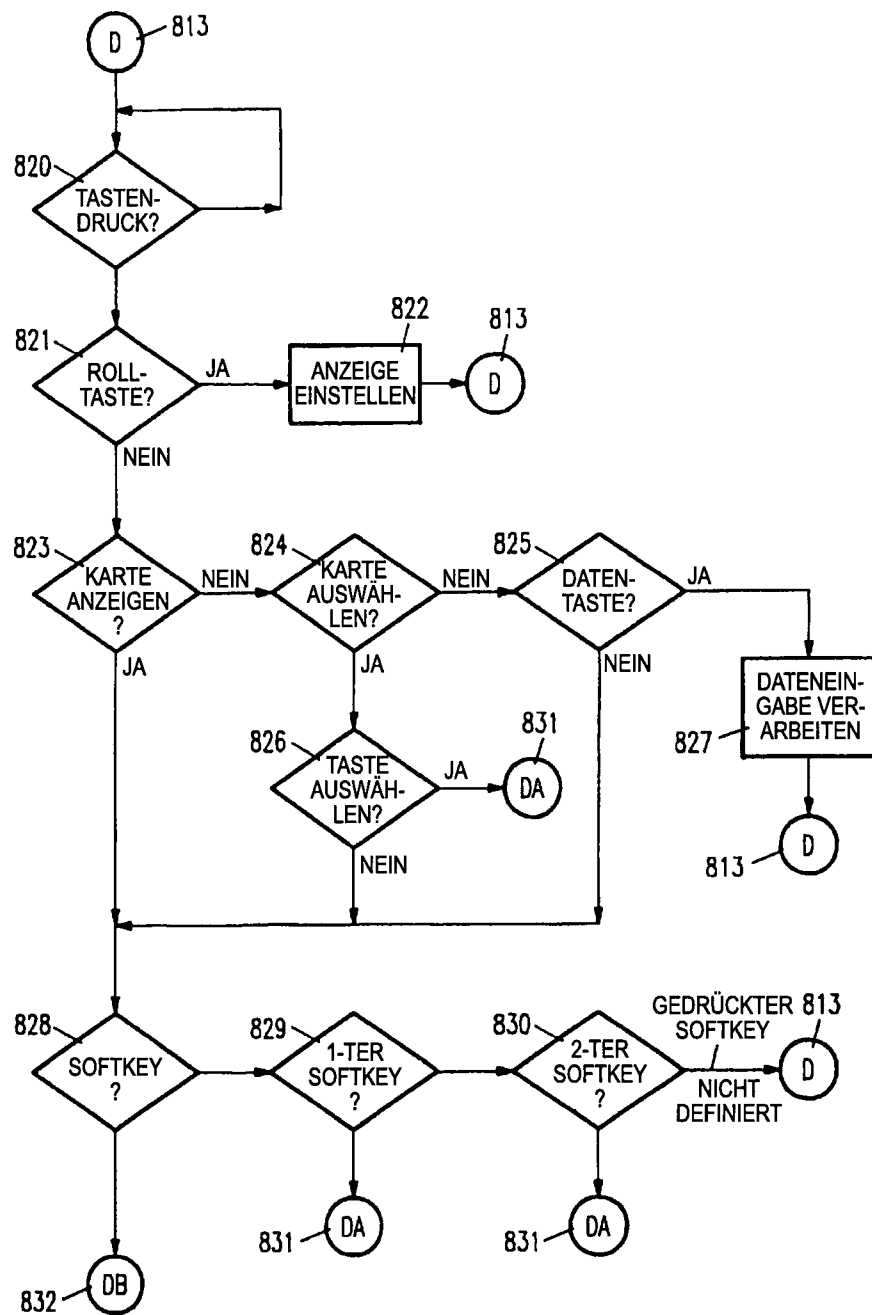


FIG. 8B

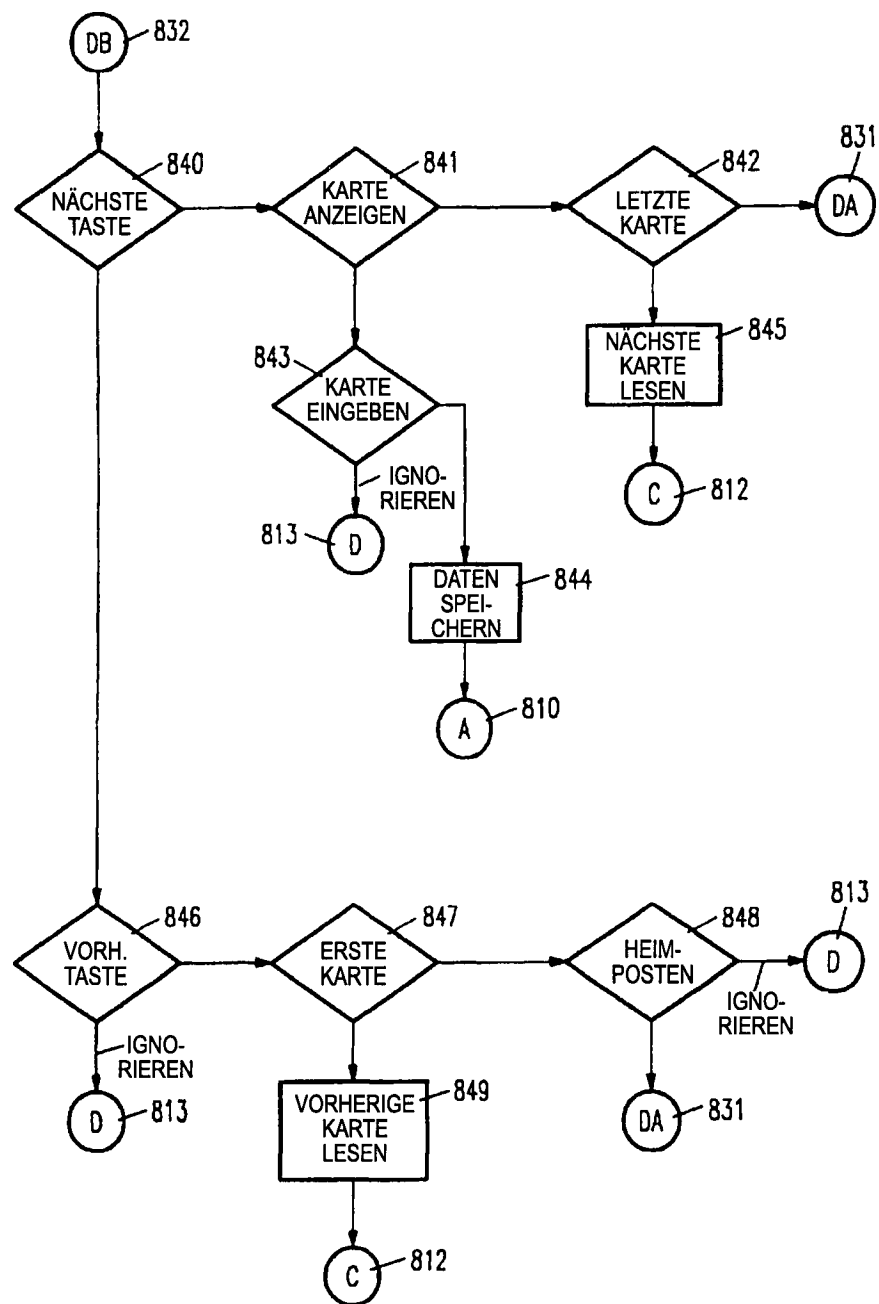


FIG. 8C

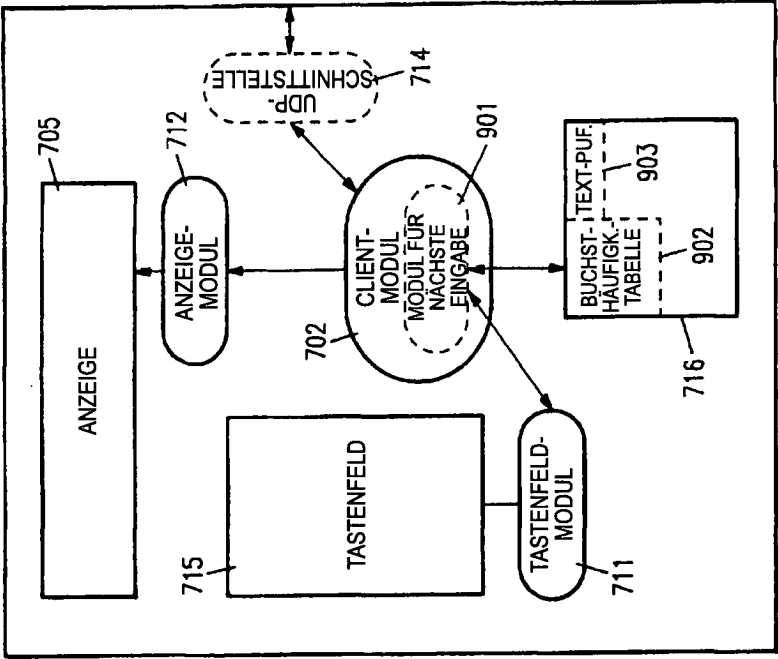


FIG. 9

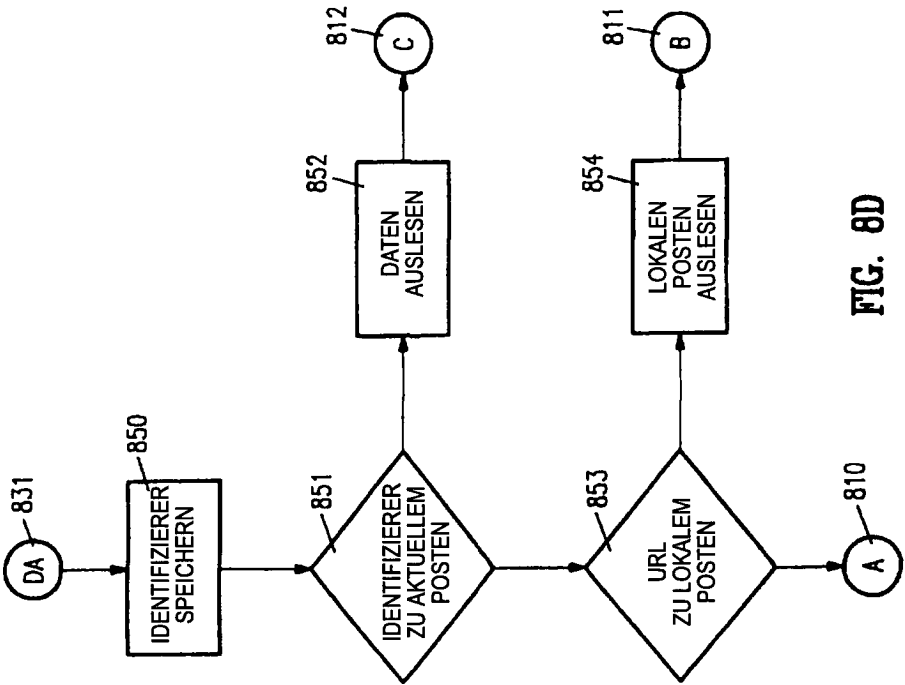


FIG. 8D

"_	_"	1	2	3	4	5	6	7	8	9
"a	a"	q	a	d	g	j	m	r	t	w
"a	b"	q	a	e	i	l	o	r	u	w
"a	c"	q	c	e	h	k	m	r	t	w
"a	d"	q	a	d	i	j	m	s	v	y
"a	e"	q	a	d	g	j	m	p	t	w
"a	f"	q	a	f	g	j	m	r	t	w
"a	g"	q	a	e	i	j	o	r	t	w
"a	h"	q	a	e	g	j	m	p	u	w
"a	i"	q	c	d	g	l	n	r	t	w
"a	j"	q	a	d	g	j	o	p	t	w
"a	k"	q	a	e	i	j	m	s	t	w
"a	l"	q	a	e	i	l	o	s	t	w
"a	m"	q	a	e	i	l	o	p	u	w
"a	n"	q	c	d	g	k	n	s	t	y
"a	o"	q	a	d	g	j	m	p	t	w
"a	p"	q	a	e	h	l	o	p	t	w
"a	q"	q	a	d	g	j	m	s	t	w
"a	r"	q	c	e	g	k	o	r	t	y
"a	s"	q	c	e	i	k	o	s	t	y
"a	t"	q	a	e	i	l	o	s	t	w
"a	u"	q	c	d	g	l	m	s	t	w
"a	v"	q	a	e	i	j	o	p	t	w
"a	w"	q	a	d	g	k	m	p	t	w
"a	x"	q	a	e	i	l	m	s	t	w
"a	y"	q	b	e	i	j	m	s	t	w
"a	z"	q	a	d	i	j	m	p	t	w
"a	"	q	a	d	g	j	m	p	t	w
"b	a"	q	c	d	g	k	n	s	t	y
"b	b"	q	a	d	i	j	m	s	t	y
"b	c"	q	a	d	g	j	o	r	t	w
"b	d"	q	a	d	g	j	m	s	t	w
"b	e"	q	c	e	g	l	n	s	t	x
"b	f"	q	a	d	g	j	m	p	t	w
"b	g"	q	a	d	g	j	m	p	t	w
"b	h"	q	a	d	g	j	m	p	t	w
"b	i"	q	c	d	g	l	n	s	t	w
"b	j"	q	a	e	g	j	m	p	t	w

FIG. 10A

"-	-"	1	2	3	4	5	6	7	8	9
"b	k"	q	a	d	g	j	m	p	t	w
"b	l"	q	a	e	i	j	o	p	t	y
"b	m"	q	a	d	g	j	m	r	t	w
"b	n"	q	a	d	g	j	m	p	t	w
"b	o"	q	a	d	i	j	o	s	u	x
"b	p"	q	a	d	g	j	m	s	t	w
"b	q"	q	a	d	g	j	m	p	t	w
"b	r"	q	a	e	i	j	o	p	u	w
"b	s"	q	c	d	i	j	o	s	t	w
"b	t"	q	a	d	g	j	m	p	t	w
"b	u"	q	a	d	i	l	m	s	t	y
"b	v"	q	a	d	i	j	m	p	t	w
"b	w"	q	a	e	g	j	m	p	t	w
"b	x"	q	a	d	g	j	m	p	t	w
"b	y"	q	a	d	g	j	m	p	t	w
"b	z"	q	a	d	g	j	m	p	t	w
"b	"	q	a	d	g	j	m	p	t	w
"c	a"	q	b	d	g	l	n	r	t	w
"c	b"	q	a	e	g	j	m	p	t	w
"c	c"	q	a	e	i	j	o	s	t	w
"c	d"	q	a	d	g	j	m	p	t	w
"c	e"	q	a	d	i	l	n	s	t	w
"c	f"	q	a	d	i	j	m	p	t	w
"c	g"	q	a	d	i	j	m	p	t	w
"c	h"	q	a	e	i	j	o	r	t	w
"c	i"	q	a	f	i	l	n	p	t	w
"c	j"	q	a	d	g	j	m	p	t	w
"c	k"	q	b	e	i	l	m	s	t	y
"c	l"	q	a	e	i	j	o	r	u	w
"c	m"	q	c	d	g	j	m	p	t	w
"c	n"	q	a	d	g	j	m	p	t	w
"c	o"	q	a	d	g	l	m	r	u	w
"c	p"	q	a	d	g	j	m	p	t	w
"c	q"	q	a	d	g	j	m	p	t	w
"c	r"	q	a	e	i	j	o	p	u	y
"c	s"	q	a	e	i	l	o	s	t	w
"c	t"	q	a	e	i	l	o	r	u	w

FIG. 10B

".	."	1	2	3	4	5	6	7	8	9
"c	u"	q	b	e	i	l	m	s	t	w
"c	v"	q	a	d	g	j	m	p	t	w
"c	w"	q	a	d	g	j	m	p	t	w
"c	x"	q	a	d	g	j	m	p	t	w
"c	y"	q	c	d	g	j	m	p	t	w
"c	z"	q	a	d	g	j	n	p	t	w
"c	"	q	a	d	g	j	m	p	t	w
"d	a"	q	b	d	g	j	n	r	t	y
"d	b"	q	a	d	g	j	m	p	t	w
"d	c"	q	a	d	g	j	m	p	t	w
"d	d"	q	a	e	i	l	m	r	t	w
"d	e"	q	a	d	g	l	m	r	v	x
"d	f"	q	a	d	i	j	m	p	t	w
"d	g"	q	a	e	g	j	m	p	t	w
"d	h"	q	a	e	g	j	m	p	t	w
"d	i"	z	a	f	g	l	n	s	t	w
"d	j"	q	a	d	g	j	m	p	t	w
"d	k"	q	a	d	g	j	m	p	t	w
"d	l"	q	a	e	i	j	m	p	t	y
"d	m"	q	a	d	i	j	m	s	t	w
"d	n"	q	a	e	g	j	m	p	t	w
"d	o"	q	c	e	i	j	n	p	t	w
"d	p"	q	a	d	g	j	m	p	t	w
"d	q"	q	a	d	g	j	m	p	u	w
"d	r"	q	a	e	i	j	o	p	t	w
"d	s"	q	a	e	i	j	m	p	t	w
"d	t"	q	a	d	g	j	m	r	t	w
"d	u"	q	c	e	g	l	m	r	t	w
"d	v"	q	a	d	i	j	m	p	t	w
"d	w"	q	a	d	i	j	o	p	t	w
"d	x"	q	a	d	g	j	m	p	t	w
"d	y"	q	a	d	g	j	m	p	t	w
"d	z"	q	a	d	g	j	m	p	t	w
"d	"	q	a	d	g	j	m	p	t	w
"e	a"	q	c	d	g	l	n	s	t	w
"e	b"	q	c	f	g	j	m	r	t	y
"e	c"	q	a	e	h	k	o	s	t	w

FIG. 10C

"-	-"	1	2	3	4	5	6	7	8	9
"e	d"	q	b	e	i	l	n	s	u	w
"e	e"	z	a	d	i	k	n	r	t	w
"e	f"	q	a	e	i	l	o	r	t	w
"e	g"	q	a	d	i	j	o	r	u	y
"e	h"	q	a	e	i	j	m	p	t	w
"e	i"	q	a	d	g	j	n	r	v	w
"e	j"	q	a	e	g	j	m	p	t	w
"e	k"	q	a	e	i	j	m	s	t	w
"e	l"	q	a	e	i	l	o	p	v	y
"e	m"	q	a	e	i	j	o	s	u	w
"e	n"	q	c	d	g	l	o	s	t	w
"e	o"	q	a	f	g	j	n	p	u	w
"e	p"	q	a	e	h	l	o	r	t	w
"e	q"	q	a	d	g	j	m	p	u	w
"e	r"	q	a	e	i	l	n	s	v	y
"e	s"	q	c	e	i	k	n	s	t	w
"e	t"	z	c	e	i	j	m	s	t	w
"e	u"	q	a	d	g	j	m	r	t	w
"e	v"	q	a	e	i	j	o	p	t	w
"e	w"	q	a	e	h	j	o	s	t	w
"e	x"	q	a	e	i	j	m	p	t	w
"e	y"	q	a	d	g	j	o	s	t	w
"e	z"	q	a	d	g	j	m	p	t	w
"e	"	q	a	d	g	j	m	p	t	w
"f	a"	q	c	d	i	k	m	s	v	x
"f	b"	q	a	d	g	j	m	p	t	w
"f	c"	q	c	d	g	j	m	p	t	w
"f	d"	q	a	d	g	j	m	p	t	w
"f	e"	q	a	e	g	l	m	r	t	w
"f	f"	q	a	e	i	j	o	s	t	w
"f	g"	q	a	d	g	j	m	p	t	w
"f	h"	q	a	d	g	j	m	p	t	w
"f	i"	q	c	e	g	l	n	r	t	x
"f	j"	q	a	d	g	j	m	p	t	w
"f	k"	q	a	d	g	j	m	p	t	w
"f	l"	q	a	e	i	j	o	p	t	y
"f	m"	q	a	d	g	j	m	p	t	w

FIG. 10D

"	"	1	2	3	4	5	6	7	8	9
"f	n"	q	a	d	g	j	m	p	t	w
"f	o"	q	c	d	g	l	n	r	u	w
"f	p"	q	a	d	g	j	m	p	t	w
"f	q"	q	a	d	g	j	m	p	t	w
"f	r"	q	a	e	i	j	o	p	t	y
"f	s"	q	a	d	g	j	m	p	t	w
"f	t"	q	a	e	g	k	m	p	t	w
"f	u"	q	a	d	g	l	n	r	t	w
"f	v"	q	a	d	g	j	m	p	t	w
"f	w"	q	a	d	g	j	m	p	t	w
"f	x"	q	a	d	g	j	m	p	t	w
"f	y"	q	a	d	i	j	m	p	t	w
"f	z"	q	a	d	g	j	m	p	t	w
"f	"	q	a	d	g	j	m	p	t	w
"g	a"	z	a	d	i	l	n	r	t	w
"g	b"	q	a	d	g	j	m	p	t	w
"g	c"	q	a	d	g	j	m	s	t	w
"g	d"	q	a	d	g	j	o	p	t	w
"g	e"	q	a	d	g	l	n	s	t	w
"g	f"	q	a	d	g	j	m	p	u	w
"g	g"	q	a	e	g	l	m	p	t	w
"g	h"	q	a	e	g	j	o	p	t	w
"g	i"	q	c	f	g	j	n	s	v	w
"g	j"	q	a	d	g	j	m	p	t	w
"g	k"	q	a	d	g	j	m	p	t	w
"g	l"	q	a	e	i	j	o	p	t	y
"g	m"	q	a	d	g	l	m	s	t	w
"g	n"	q	a	e	i	j	o	p	t	w
"g	o"	q	a	e	i	j	o	p	t	w
"g	p"	q	a	d	g	j	m	p	t	w
"g	q"	q	a	d	g	j	m	p	t	w
"g	r"	q	a	e	g	j	o	p	t	w
"g	s"	q	a	d	g	j	m	p	t	w
"g	t"	q	a	e	h	j	o	p	t	w
"g	u"	q	a	e	i	l	m	r	t	y
"g	v"	q	a	d	g	j	m	p	t	w
"g	w"	q	a	d	g	j	m	p	t	w

FIG. 10E

"	-	1	2	3	4	5	6	7	8	9
"g	x"	q	a	d	g	j	m	p	t	w
"g	y"	q	a	d	g	j	m	p	t	w
"g	z"	q	a	d	g	j	m	p	t	w
"g	"	q	a	d	g	j	m	p	t	w
"h	a"	q	b	d	i	l	n	s	t	w
"h	b"	q	a	d	g	j	m	p	t	w
"h	c"	q	a	d	g	j	o	p	t	w
"h	d"	q	a	d	g	j	m	r	t	w
"h	e"	q	c	d	i	l	n	r	t	y
"h	f"	q	a	d	g	j	m	p	t	w
"h	g"	q	a	d	g	j	m	p	t	w
"h	h"	q	a	d	g	j	m	p	t	w
"h	i"	q	c	e	g	l	n	s	t	w
"h	j"	q	a	d	g	j	m	p	t	w
"h	k"	q	a	d	g	j	m	p	t	w
"h	l"	q	a	d	g	j	m	p	t	y
"h	m"	q	a	e	g	j	m	p	t	w
"h	n"	q	a	d	i	j	o	p	t	w
"h	o"	q	b	d	i	l	n	r	u	w
"h	p"	q	a	d	g	j	m	p	u	w
"h	q"	q	a	d	g	j	m	p	t	w
"h	r"	q	a	e	i	j	o	p	t	w
"h	s"	q	a	d	g	j	m	p	t	w
"h	t"	q	a	d	g	l	m	s	t	w
"h	u"	q	a	d	g	l	m	r	t	w
"h	v"	q	a	d	g	j	m	p	t	w
"h	w"	q	a	d	g	j	m	p	t	w
"h	x"	q	a	d	g	j	m	p	t	w
"h	y"	q	a	d	g	j	m	p	t	w
"h	z"	q	a	d	g	j	m	p	t	w
"h	"	q	a	d	g	j	m	p	t	w
"i	a"	q	b	d	g	l	n	r	t	w
"i	b"	q	a	e	i	l	m	r	u	w
"i	c"	q	a	e	h	k	o	r	t	y
"i	d"	q	a	e	i	l	n	p	u	w
"i	e"	q	c	d	g	l	n	r	v	w
"i	f"	q	a	f	i	j	o	p	t	y

FIG. 10F

"-	."	1	2	3	4	5	6	7	8	9
"i	g"	q	a	e	h	j	n	r	u	w
"i	h"	q	a	d	g	j	m	p	t	w
"i	i"	q	a	d	g	j	m	p	t	w
"i	j"	q	a	d	g	j	o	p	t	w
"i	k"	q	a	e	g	k	m	p	t	w
"i	l"	q	a	e	i	l	o	s	t	y
"i	m"	q	a	e	i	j	m	p	u	w
"i	n"	q	c	e	g	k	n	s	t	w
"i	o"	q	a	d	g	l	n	r	u	w
"i	p"	q	a	d	i	l	m	s	t	w
"i	q"	q	a	d	g	j	m	p	u	w
"i	r"	q	c	e	i	l	n	s	t	w
"i	s"	q	c	e	h	k	n	s	t	w
"i	t"	q	a	e	h	l	o	s	t	y
"i	u"	q	a	d	g	j	m	p	t	w
"i	v"	q	a	e	i	j	m	p	t	w
"i	w"	q	a	d	g	j	m	p	t	w
"i	x"	q	a	e	g	j	m	p	t	w
"i	y"	q	a	d	g	j	m	p	t	w
"i	z"	q	a	e	i	j	o	p	t	w
"i	"	q	a	d	g	j	m	p	t	w
"j	a"	q	c	d	g	j	n	p	t	w
"j	b"	q	a	d	g	j	m	p	t	w
"j	c"	q	a	d	g	j	m	p	t	w
"j	d"	q	a	d	g	j	m	p	t	w
"j	e"	q	c	d	g	j	m	r	t	w
"j	f"	q	a	d	g	j	m	p	t	w
"j	g"	q	a	d	g	j	m	p	t	w
"j	h"	q	a	d	g	j	m	p	t	w
"j	i"	q	a	d	g	j	m	p	t	w
"j	j"	q	a	d	g	j	m	p	t	w
"j	k"	q	a	d	g	j	m	p	t	w
"j	l"	q	a	d	g	j	m	p	t	w
"j	m"	q	a	d	g	j	m	p	t	w
"j	n"	q	a	d	g	j	m	p	t	w
"j	o"	q	b	e	i	j	m	r	t	w
"j	p"	q	a	d	g	j	m	p	t	w

FIG. 10G

"	-"	1	2	3	4	5	6	7	8	9
"j	q"	q	a	d	g	j	m	p	t	w
"j	r"	q	a	d	g	j	m	p	t	w
"j	s"	q	a	d	g	j	m	p	t	w
"j	t"	q	a	d	g	j	m	p	t	w
"j	u"	q	a	d	g	j	m	s	t	w
"j	v"	q	a	d	g	j	m	p	t	w
"j	w"	q	a	d	g	j	m	p	t	w
"j	x"	q	a	d	g	j	m	p	t	w
"j	y"	q	a	d	g	j	m	p	t	w
"j	z"	q	a	d	g	j	m	p	t	w
"j	"	q	a	d	g	j	m	p	t	w
"k	a"	q	a	d	g	j	n	s	t	y
"k	b"	q	a	d	g	j	o	p	t	w
"k	c"	q	a	d	g	j	m	p	t	w
"k	d"	q	a	d	g	j	o	p	t	w
"k	e"	q	a	d	g	j	n	s	t	y
"k	f"	q	a	d	g	j	m	p	t	w
"k	g"	q	a	d	g	j	m	r	t	w
"k	h"	q	a	d	g	j	m	p	t	w
"k	i"	q	a	d	g	j	n	p	t	w
"k	j"	q	a	d	g	j	m	p	t	w
"k	k"	q	a	d	g	j	o	p	t	w
"k	l"	q	a	d	i	j	m	p	t	y
"k	m"	q	a	d	g	j	m	p	t	w
"k	n"	q	a	d	g	j	o	p	t	w
"k	o"	q	a	d	g	j	n	p	t	w
"k	p"	q	a	d	g	j	m	p	t	w
"k	q"	q	a	d	g	j	m	p	t	w
"k	r"	q	a	d	g	j	m	p	t	w
"k	s"	q	a	d	g	j	m	p	t	w
"k	t"	q	a	d	g	j	m	p	t	w
"k	u"	q	a	d	g	j	m	p	t	w
"k	v"	q	a	d	g	j	m	p	t	w
"k	w"	q	a	d	g	j	m	p	t	w
"k	x"	q	a	d	g	j	m	p	t	w
"k	y"	q	a	d	g	j	m	p	t	w
"k	z"	q	a	d	g	j	m	p	t	w

FIG. 10H

"	"	1	2	3	4	5	6	7	8	9
"k	"	q	a	d	g	j	m	p	t	w
"l	a"	q	b	d	i	j	n	r	t	y
"l	b"	q	a	d	g	j	o	p	t	w
"l	c"	q	a	d	g	j	o	p	t	w
"l	d"	q	a	e	i	j	n	s	t	w
"l	e"	q	a	d	g	l	m	s	t	x
"l	f"	q	a	d	g	j	m	p	t	w
"l	g"	q	a	d	g	j	m	p	t	w
"l	h"	q	a	d	g	j	m	p	t	w
"l	i"	z	c	e	g	k	n	s	t	w
"l	j"	q	a	d	g	j	m	p	t	w
"l	k"	q	a	e	i	j	m	s	t	w
"l	l"	q	c	e	i	j	o	s	u	y
"l	m"	q	a	d	g	j	o	s	t	w
"l	n"	q	a	e	g	j	m	p	t	w
"l	o"	q	c	d	g	j	o	s	t	w
"l	p"	q	a	d	h	j	m	s	t	w
"l	q"	q	a	d	g	j	m	p	t	w
"l	r"	q	a	e	g	j	m	p	t	w
"l	s"	q	a	e	i	j	o	p	t	w
"l	t"	q	a	e	i	j	o	s	t	w
"l	u"	q	c	d	g	l	m	s	t	w
"l	v"	q	a	e	i	j	m	p	t	w
"l	w"	q	a	e	g	j	m	p	t	w
"l	x"	q	a	d	g	j	m	p	t	w
"l	y"	z	a	d	i	j	n	s	t	w
"l	z"	q	a	d	g	j	m	p	t	w
"l	"	q	a	d	g	j	m	p	t	w
"m	a"	q	c	d	i	k	n	r	t	y
"m	b"	q	a	e	i	l	m	p	t	y
"m	c"	q	c	d	i	j	o	p	t	w
"m	d"	q	a	d	g	j	m	p	t	w
"m	e"	q	a	e	g	l	n	s	t	w
"m	f"	q	a	d	g	j	o	p	t	w
"m	g"	q	a	d	g	j	m	p	t	w
"m	h"	z	a	d	g	j	m	p	t	w
"m	i"	z	c	d	g	l	n	s	t	x

FIG. 10I

"_	_"	1	2	3	4	5	6	7	8	9
"m	j"	q	a	d	g	j	m	p	t	w
"m	k"	q	a	d	g	j	m	p	t	w
"m	l"	q	a	e	g	j	m	p	t	y
"m	m"	q	a	e	i	j	o	p	u	w
"m	n"	q	a	d	g	j	m	p	t	w
"m	o"	q	b	d	i	j	n	r	v	w
"m	p"	q	a	e	i	l	o	r	t	w
"m	q"	q	a	d	g	j	m	p	t	w
"m	r"	q	a	d	g	j	m	p	t	w
"m	s"	q	a	e	g	j	m	p	t	w
"m	t"	q	a	d	g	j	m	p	t	w
"m	u"	q	c	d	g	l	n	s	t	w
"m	v"	q	a	d	g	j	m	p	t	w
"m	w"	q	a	d	h	j	m	p	t	w
"m	x"	q	a	d	g	j	m	p	t	w
"m	y"	q	c	d	g	j	m	s	t	w
"m	z"	q	a	d	g	j	m	p	t	w
"m	"	q	a	d	g	j	m	p	t	w
"n	a"	q	b	d	g	l	m	r	t	w
"n	b"	q	a	d	g	j	m	p	t	w
"n	c"	q	a	e	i	l	o	r	t	y
"n	d"	q	a	e	i	l	o	s	t	y
"n	e"	q	c	e	i	l	n	s	t	w
"n	f"	q	a	e	i	j	o	s	u	w
"n	g"	q	a	e	i	l	m	s	u	w
"n	h"	q	a	d	g	j	m	p	t	w
"n	i"	z	c	e	g	l	n	s	t	x
"n	j"	q	a	d	g	j	m	p	u	w
"n	k"	q	a	e	i	j	n	s	t	w
"n	l"	q	a	e	i	j	o	p	t	y
"n	m"	q	a	e	g	j	m	p	t	w
"n	n"	q	a	e	i	j	o	p	u	y
"n	o"	q	b	d	g	l	n	r	t	w
"n	p"	q	a	d	g	j	o	p	u	w
"n	q"	q	a	d	g	j	m	p	u	w
"n	r"	q	a	d	g	j	m	p	t	w
"n	s"	q	a	e	i	l	m	p	t	w

FIG. 10J

1	2	3	4	5	6	7	8	9		
"n	t"	q	a	e	i	l	o	s	u	y
"n	u"	q	a	f	g	j	m	s	t	x
"n	v"	q	a	e	i	j	o	p	t	w
"n	w"	q	a	e	g	j	m	p	t	w
"n	x"	q	a	d	g	j	m	p	t	w
"n	y"	q	b	d	h	j	m	p	t	w
"n	z"	q	a	d	g	j	m	p	t	w
"n	"	q	a	d	g	j	m	p	t	w
"o	a"	q	b	d	g	l	m	s	t	w
"o	b"	q	a	e	i	l	o	s	t	w
"o	c"	q	a	e	h	k	o	s	u	w
"o	d"	q	a	e	i	l	o	s	u	y
"o	e"	q	a	d	g	j	m	s	t	w
"o	f"	q	a	f	g	j	m	p	t	w
"o	g"	q	a	e	i	j	o	r	t	y
"o	h"	q	a	d	g	j	n	p	t	w
"o	i"	q	c	d	g	l	n	p	t	w
"o	j"	q	a	e	g	j	m	p	t	w
"o	k"	q	a	e	i	j	m	s	t	w
"o	l"	q	b	d	i	l	o	s	u	w
"o	m"	q	a	e	i	l	m	p	t	w
"o	n"	q	a	e	g	l	n	s	t	y
"o	o"	q	a	d	g	k	n	s	t	w
"o	p"	q	a	e	h	l	m	p	t	y
"o	q"	q	a	d	g	j	m	p	t	w
"o	r"	q	a	e	g	k	m	r	t	w
"o	s"	q	c	e	i	j	o	s	t	y
"o	t"	q	a	e	h	j	o	s	t	y
"o	u"	q	c	d	g	l	n	r	t	y
"o	v"	q	a	e	i	j	m	p	t	w
"o	w"	q	a	e	i	j	n	s	t	w
"o	x"	q	a	d	i	j	m	p	t	w
"o	y"	q	a	e	g	j	m	s	t	w
"o	z"	q	a	d	g	j	m	p	t	w
"o	"	q	a	d	g	j	m	p	t	w
"p	a"	q	c	d	g	l	n	r	t	y
"p	b"	q	a	e	g	j	m	p	t	w

FIG. 10K

"	."	1	2	3	4	5	6	7	8	9
"p	c"	q	a	d	g	j	m	s	t	w
"p	d"	q	a	d	g	j	m	p	t	w
"p	e"	q	c	d	g	j	n	r	t	w
"p	f"	q	a	d	g	j	m	r	t	w
"p	g"	q	a	d	g	j	m	p	t	w
"p	h"	q	a	e	i	j	o	s	t	y
"p	i"	q	c	d	g	j	n	r	t	x
"p	j"	q	a	d	g	j	m	p	t	w
"p	k"	q	a	d	g	j	m	p	t	w
"p	l"	q	a	e	i	j	o	p	u	y
"p	m"	q	a	e	g	j	m	p	t	w
"p	n"	q	a	d	g	j	m	p	t	w
"p	o"	q	c	d	i	j	n	r	t	w
"p	p"	q	a	e	i	j	o	p	t	y
"p	q"	q	a	d	g	j	m	p	t	w
"p	r"	q	a	e	i	j	o	p	t	y
"p	s"	q	a	d	g	j	m	p	t	w
"p	t"	q	c	e	i	j	o	s	t	w
"p	u"	q	b	d	g	j	m	r	t	w
"p	v"	q	a	d	g	j	m	p	t	w
"p	w"	q	a	d	g	j	m	p	t	w
"p	x"	q	a	d	g	j	m	p	t	w
"p	y"	q	a	d	g	j	m	r	t	w
"p	z"	q	a	d	g	j	m	p	t	w
"p	"	q	a	d	g	j	m	p	t	w
"q	a"	q	a	d	g	j	m	p	t	w
"q	b"	q	a	d	g	j	m	p	t	w
"q	c"	q	a	d	g	j	m	p	t	w
"q	d"	q	a	d	g	j	m	p	t	w
"q	e"	q	a	d	g	j	m	p	t	w
"q	f"	q	a	d	g	j	m	p	t	w
"q	g"	q	a	d	g	j	m	p	t	w
"q	h"	q	a	d	g	j	m	p	t	w
"q	i"	q	a	d	g	j	m	p	t	w
"q	j"	q	a	d	g	j	m	p	t	w
"q	k"	q	a	d	g	j	m	p	t	w
"q	l"	q	a	d	g	j	m	p	t	w

FIG. 10L

"-	-"	1	2	3	4	5	6	7	8	9
"q	m"	q	a	d	g	j	m	p	t	w
"q	n"	q	a	d	g	j	m	p	t	w
"q	o"	q	a	d	g	j	m	p	t	w
"q	p"	q	a	d	g	j	m	p	t	w
"q	q"	q	a	d	g	j	m	p	t	w
"q	r"	q	a	d	g	j	m	p	t	w
"q	s"	q	a	d	g	j	m	p	t	w
"q	t"	q	a	d	g	j	m	p	t	w
"q	u"	q	a	e	i	j	o	p	t	w
"q	v"	q	a	d	g	j	m	p	t	w
"q	w"	q	a	d	g	j	m	p	t	w
"q	x"	q	a	d	g	j	m	p	t	w
"q	y"	q	a	d	g	j	m	p	t	w
"q	z"	q	a	d	g	j	m	p	t	w
"q	"	q	a	d	g	j	m	p	t	w
"r	a"	q	c	d	g	l	n	p	t	w
"r	b"	q	a	d	g	j	o	p	t	w
"r	c"	q	a	e	h	l	m	p	u	w
"r	d"	q	a	e	i	j	n	s	v	w
"r	e"	q	a	e	g	l	n	s	v	w
"r	f"	q	a	e	g	j	o	p	t	w
"r	g"	q	a	e	i	j	m	s	u	w
"r	h"	q	a	e	g	j	o	p	t	w
"r	i"	z	c	e	g	l	n	p	t	w
"r	j"	q	a	d	g	j	m	p	t	w
"r	k"	q	a	e	i	l	m	s	t	y
"r	l"	q	a	d	i	j	o	p	t	y
"r	m"	q	a	e	i	j	m	s	t	w
"r	n"	q	a	e	i	j	o	s	t	w
"r	o"	q	b	d	g	l	m	p	u	w
"r	p"	q	a	d	g	j	o	r	t	w
"r	q"	q	a	d	g	j	m	p	t	w
"r	r"	q	a	e	i	j	o	p	t	y
"r	s"	q	a	e	i	j	o	p	t	w
"r	t"	z	a	e	h	j	n	s	u	w
"r	u"	q	c	e	g	l	n	s	t	w
"r	v"	q	a	e	i	j	m	p	t	w

FIG. 10M

"	"	1	2	3	4	5	6	7	8	9
"r	w"	q	a	d	i	j	o	p	t	w
"r	x"	q	a	d	g	j	m	p	t	w
"r	y"	q	b	d	i	j	m	p	t	w
"r	z"	q	a	d	g	j	m	p	t	w
"r	"	q	a	d	g	j	m	p	t	w
"s	a"	q	b	d	g	l	m	p	v	y
"s	b"	q	a	d	g	j	m	p	t	w
"s	c"	q	a	d	h	l	o	r	u	w
"s	d"	q	a	d	g	j	n	p	t	w
"s	e"	q	c	e	g	l	n	r	t	y
"s	f"	q	a	e	g	j	m	p	u	w
"s	g"	q	a	d	g	j	m	r	t	w
"s	h"	q	a	e	i	l	o	r	t	w
"s	i"	z	c	d	g	l	n	s	t	x
"s	j"	q	a	d	g	j	m	p	t	w
"s	k"	q	a	e	i	j	m	p	t	w
"s	l"	q	a	e	i	j	o	p	t	y
"s	m"	q	a	e	i	j	o	s	t	w
"s	n"	q	a	e	g	j	m	p	t	w
"s	o"	q	c	f	g	l	m	r	u	w
"s	p"	q	a	e	g	l	o	r	t	w
"s	q"	q	a	d	g	l	m	p	u	w
"s	r"	q	c	d	g	j	m	p	t	w
"s	s"	q	a	e	i	j	o	p	u	w
"s	t"	q	a	e	i	l	o	r	u	y
"s	u"	q	b	e	g	l	m	p	t	w
"s	v"	q	a	d	g	j	m	p	t	w
"s	w"	q	a	e	i	j	o	p	t	w
"s	x"	q	a	d	g	j	m	p	t	w
"s	y"	q	a	d	g	j	m	s	t	w
"s	z"	q	a	d	g	j	m	p	t	w
"s	"	q	a	d	g	j	m	p	t	w
"t	a"	q	c	f	i	l	n	r	t	y
"t	b"	q	a	d	g	j	m	p	t	w
"t	c"	z	a	d	h	j	o	p	t	w
"t	d"	q	a	d	g	j	m	p	t	w
"t	e"	q	c	d	g	l	m	r	u	x

FIG. 10N

"	"	1	2	3	4	5	6	7	8	9
"t	f"	q	a	d	i	j	o	p	t	w
"t	g"	q	a	e	g	j	o	p	t	w
"t	h"	q	a	e	i	l	o	r	u	w
"t	i"	z	c	e	g	l	o	s	v	w
"t	j"	q	a	d	g	j	m	p	t	w
"t	k"	q	a	e	g	j	m	p	t	w
"t	l"	q	a	e	i	j	m	p	t	y
"t	m"	q	a	e	g	l	o	p	t	w
"t	n"	q	a	e	g	j	m	p	t	w
"t	o"	q	c	d	g	l	m	r	u	w
"t	p"	q	c	d	g	j	m	p	u	w
"t	q"	q	a	d	g	j	m	p	t	w
"t	r"	q	a	e	i	j	o	p	u	y
"t	s"	q	c	e	i	j	o	p	u	w
"t	t"	q	a	e	i	l	o	p	t	y
"t	u"	q	a	f	i	j	n	r	t	w
"t	v"	q	a	d	g	j	m	p	t	w
"t	w"	q	a	e	i	j	o	p	t	w
"t	x"	q	a	d	g	j	m	p	t	w
"t	y"	q	a	d	g	l	m	p	t	w
"t	z"	q	a	d	g	j	m	p	t	w
"t	"	q	a	d	g	j	m	p	t	w
"u	a"	q	a	d	g	l	n	r	t	w
"u	b"	q	a	e	i	j	m	s	t	w
"u	c"	q	c	e	h	k	m	p	t	w
"u	d"	q	a	e	i	j	o	p	t	y
"u	e"	q	a	d	g	j	n	s	t	w
"u	f"	q	a	f	g	j	m	p	t	w
"u	g"	q	a	e	h	j	m	s	t	w
"u	h"	q	a	d	g	j	m	p	t	w
"u	i"	q	c	d	g	l	n	r	t	w
"u	j"	q	a	d	g	j	m	p	t	w
"u	k"	q	a	d	g	j	m	p	t	w
"u	l"	q	a	d	g	l	m	p	t	y
"u	m"	q	b	e	i	j	m	p	t	w
"u	n"	q	c	d	i	l	n	s	t	w
"u	o"	q	a	d	g	j	m	p	t	w

FIG. 100

"	"	1	2	3	4	5	6	7	8	9
"u	p"	q	a	d	i	l	o	p	t	y
"u	q"	q	a	d	g	j	m	p	t	w
"u	r"	q	c	e	i	l	n	s	t	y
"u	s"	q	a	e	i	l	m	s	t	y
"u	t"	q	a	e	i	l	o	s	t	w
"u	u"	q	c	d	g	j	m	p	t	w
"u	v"	q	a	d	g	j	m	p	t	w
"u	w"	q	a	d	g	j	m	p	t	w
"u	x"	q	a	d	g	j	m	p	t	w
"u	y"	q	a	e	g	j	m	s	t	w
"u	z"	q	a	d	g	j	m	p	t	w
"u	"	q	a	d	g	j	m	p	t	w
"v	a"	q	c	d	i	l	n	r	t	w
"v	b"	q	a	d	g	j	m	p	t	w
"v	c"	q	a	d	g	j	m	s	t	w
"v	d"	q	a	d	g	j	m	p	t	w
"v	e"	q	a	d	g	l	n	r	t	w
"v	f"	q	a	d	g	j	m	p	t	w
"v	g"	q	a	d	g	j	m	p	t	w
"v	h"	q	a	d	g	j	m	p	t	w
"v	i"	q	c	e	g	j	n	s	t	w
"v	j"	q	a	d	g	j	m	p	t	w
"v	k"	q	a	d	g	j	m	p	t	w
"v	l"	q	a	d	g	j	m	s	t	w
"v	m"	q	a	d	g	j	m	p	t	w
"v	n"	q	a	d	g	j	m	p	t	w
"v	o"	q	a	d	i	l	o	r	t	w
"v	p"	q	a	d	g	j	m	p	t	w
"v	q"	q	a	d	g	j	m	p	t	w
"v	r"	q	a	d	g	j	m	p	t	w
"v	s"	q	a	d	g	j	m	p	t	w
"v	t"	q	a	d	g	j	m	p	t	w
"v	u"	q	a	d	g	j	m	p	t	w
"v	v"	q	a	d	g	j	m	p	t	w
"v	w"	q	a	d	g	j	m	p	t	w
"v	x"	q	a	d	g	j	m	p	t	w
"v	y"	q	a	d	g	j	m	p	t	w

FIG. 10P

"_	"	1	2	3	4	5	6	7	8	9
"v	z"	q	a	d	g	j	m	p	t	w
"v	"	q	a	d	g	j	m	p	t	w
"w	a"	q	b	d	i	l	n	r	v	y
"w	b"	q	a	d	g	j	m	p	t	w
"w	c"	q	a	e	g	j	m	p	t	w
"w	d"	q	a	f	g	j	m	p	t	w
"w	e"	q	b	e	i	l	n	r	v	w
"w	f"	q	a	d	g	j	m	p	t	w
"w	g"	q	a	d	g	j	m	p	t	w
"w	h"	q	a	e	i	j	o	p	t	y
"w	i"	q	c	d	g	l	n	s	t	w
"w	j"	q	a	d	g	j	m	p	t	w
"w	k"	q	a	d	g	j	m	p	t	w
"w	l"	q	a	d	g	j	m	p	t	w
"w	m"	q	a	d	g	j	m	p	t	w
"w	n"	q	a	d	g	l	m	s	t	w
"w	o"	q	a	d	g	j	n	r	u	w
"w	p"	q	a	d	h	j	m	p	t	w
"w	q"	q	a	d	g	j	m	p	t	w
"w	r"	q	a	d	i	j	o	p	t	w
"w	s"	q	a	e	g	j	m	p	t	w
"w	t"	q	a	d	g	j	m	p	t	w
"w	u"	q	a	d	g	j	m	p	t	w
"w	v"	q	a	d	g	j	m	p	t	w
"w	w"	q	a	d	g	j	m	p	t	w
"w	x"	q	a	d	g	j	m	p	t	w
"w	y"	q	a	d	g	j	m	p	t	w
"w	z"	q	a	d	g	j	m	p	t	w
"w	"	q	a	d	g	j	m	p	t	w
"x	a"	q	c	d	g	j	m	p	t	w
"x	b"	q	a	d	g	j	m	p	t	w
"x	c"	q	a	e	h	j	m	p	t	w
"x	d"	q	a	d	g	j	m	p	t	w
"x	e"	q	c	d	g	l	m	s	t	x
"x	f"	q	a	d	g	j	m	p	t	w
"x	g"	q	a	d	g	j	m	p	t	w
"x	h"	q	a	d	g	j	m	p	t	w

FIG. 10Q

1	2	3	4	5	6	7	8	9		
"x	i"	q	c	d	g	j	n	s	t	w
"x	j"	q	a	d	g	j	m	p	t	w
"x	k"	q	a	d	g	j	m	p	t	w
"x	l"	q	a	d	i	j	m	p	t	w
"x	m"	q	a	d	g	j	m	p	t	w
"x	n"	q	a	d	g	j	m	p	t	w
"x	o"	q	a	d	g	j	m	p	t	w
"x	p"	q	a	e	i	l	o	p	t	w
"x	q"	q	a	d	g	j	m	p	t	w
"x	r"	q	a	d	g	j	m	p	t	w
"x	s"	q	a	e	i	j	m	p	t	w
"x	t"	q	a	e	g	j	m	r	u	w
"x	u"	q	a	d	g	j	m	p	t	w
"x	v"	q	a	d	g	j	m	p	t	w
"x	w"	q	a	d	g	j	m	p	t	w
"x	x"	q	a	d	g	j	m	p	t	w
"x	y"	z	a	d	g	j	m	p	t	w
"x	z"	q	a	d	g	j	m	p	t	w
"x	"	q	a	d	g	j	m	p	t	w
"y	a"	q	a	d	g	j	m	p	t	w
"y	b"	q	a	e	i	j	o	p	t	w
"y	c"	q	a	d	g	l	o	p	t	w
"y	d"	q	a	d	g	j	m	p	t	w
"y	e"	q	a	e	g	j	m	s	t	w
"y	f"	q	a	d	g	j	m	p	t	w
"y	g"	q	a	d	g	j	m	p	t	w
"y	h"	q	a	d	g	j	o	p	t	w
"y	i"	q	c	d	g	j	n	p	t	w
"y	j"	q	a	d	g	j	m	p	t	w
"y	k"	q	a	d	g	j	m	p	t	w
"y	l"	q	a	e	g	j	m	p	t	w
"y	m"	q	a	e	g	j	o	p	t	w
"y	n"	q	c	d	g	j	m	p	t	x
"y	o"	q	a	d	g	j	n	r	u	w
"y	p"	q	a	e	i	j	m	p	t	w
"y	q"	q	a	d	g	j	m	p	t	w
"y	r"	q	a	d	i	j	m	p	t	w

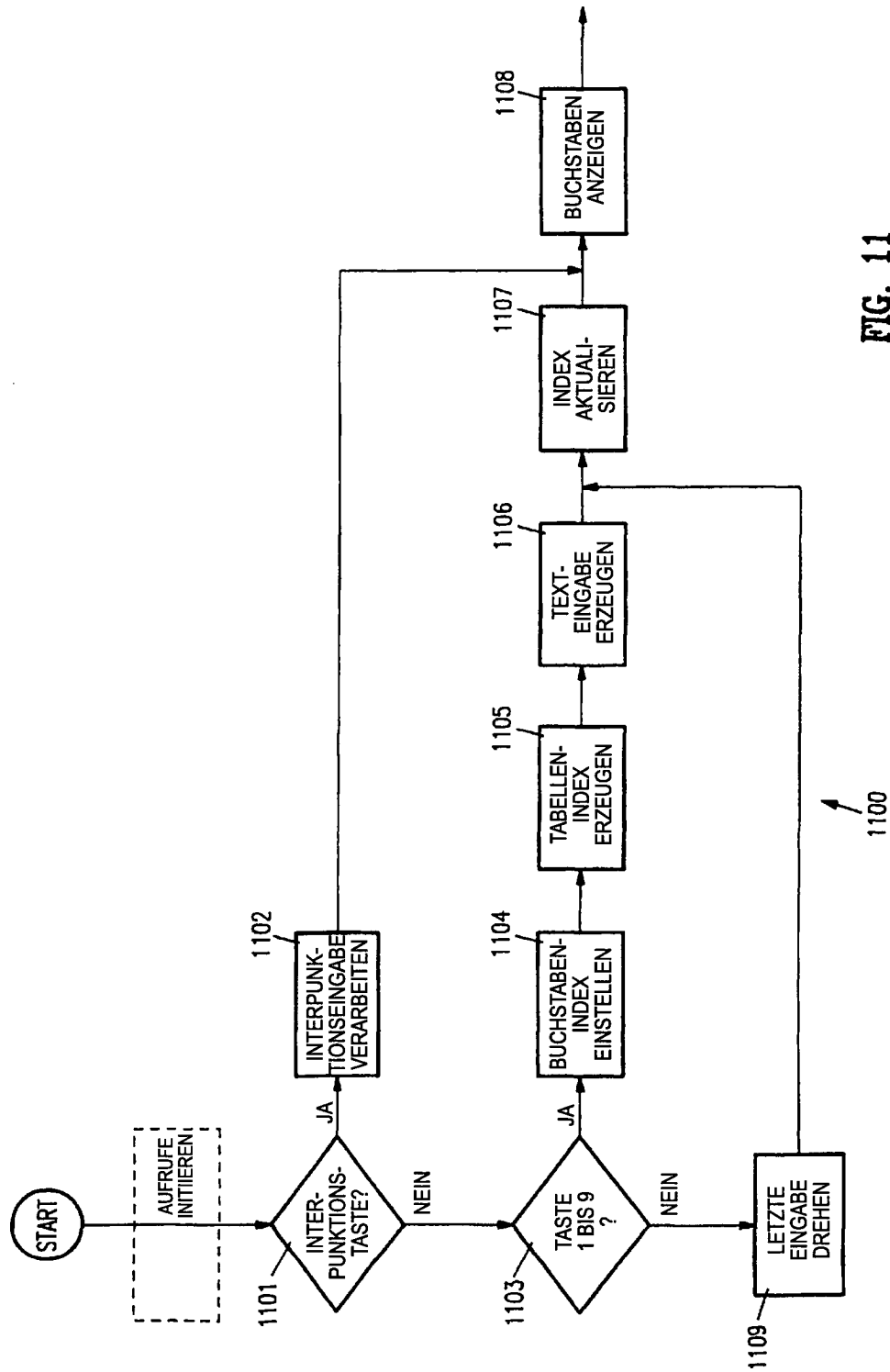
FIG. 10R

"-	-"	1	2	3	4	5	6	7	8	9
"y	s"	q	c	e	i	j	m	p	t	w
"y	t"	q	a	e	h	j	m	p	t	w
"y	u"	q	a	d	g	k	m	r	t	w
"y	v"	q	a	d	g	j	m	p	t	w
"y	w"	q	a	d	h	j	m	p	t	w
"y	x"	q	a	d	g	j	m	p	t	w
"y	y"	q	a	d	g	j	m	p	t	w
"y	z"	q	a	e	g	j	m	p	t	w
"y	"	q	a	d	g	j	m	p	t	w
"z	a"	q	a	d	g	j	m	p	t	w
"z	b"	q	a	d	g	j	m	p	t	w
"z	c"	q	a	d	g	j	m	p	t	w
"z	d"	q	a	d	g	j	m	p	t	w
"z	e"	q	a	d	g	j	n	r	t	w
"z	f"	q	a	d	g	j	m	p	t	w
"z	g"	q	a	d	g	j	m	p	t	w
"z	h"	q	a	d	g	j	m	p	t	w
"z	i"	q	a	d	g	j	n	p	t	w
"z	j"	q	a	d	g	j	m	p	t	w
"z	k"	q	a	d	g	j	m	p	t	w
"z	l"	q	a	d	g	j	m	p	t	w
"z	m"	q	a	d	g	j	m	p	t	w
"z	n"	q	a	e	g	j	m	p	t	w
"z	o"	q	a	d	g	j	n	p	t	w
"z	p"	q	a	d	g	j	m	p	t	w
"z	q"	q	a	d	g	j	m	p	t	w
"z	r"	q	a	d	g	j	m	p	t	w
"z	s"	q	a	d	g	j	m	p	t	w
"z	t"	q	a	d	g	j	m	p	t	w
"z	u"	q	a	d	g	j	m	p	t	w
"z	v"	q	a	d	g	j	m	p	t	w
"z	w"	q	a	d	g	j	m	p	t	w
"z	x"	q	a	d	g	j	m	p	t	w
"z	y"	q	a	d	g	j	m	p	t	w
"z	z"	q	a	d	g	j	m	p	t	w
"z	"	q	a	d	g	j	m	p	t	w
"	a"	q	c	d	i	l	n	r	t	w

FIG. 10S

"	"	1	2	3	4	5	6	7	8	9
"	b"	q	a	e	i	l	o	r	u	y
"	c"	q	a	e	h	l	o	r	u	y
"	d"	q	a	e	i	j	o	r	u	w
"	e"	q	a	d	i	l	n	s	v	x
"	f"	q	a	e	i	l	o	r	u	w
"	g"	q	a	e	i	l	o	r	u	w
"	h"	q	a	e	i	j	o	p	t	w
"	i"	q	c	f	g	j	n	s	t	w
"	j"	q	a	e	g	j	o	p	u	w
"	k"	q	b	e	i	j	n	p	t	w
"	l"	q	a	e	i	l	o	p	u	y
"	m"	q	a	e	i	k	o	s	u	y
"	n"	q	a	e	i	j	o	p	u	w
"	o"	z	b	f	g	k	n	r	u	w
"	p"	q	a	e	h	l	o	r	u	w
"	q"	z	a	d	g	j	m	p	u	w
"	r"	q	a	e	i	j	o	s	u	w
"	s"	q	c	e	h	l	o	p	t	y
"	t"	q	a	e	h	j	o	r	u	w
"	u"	q	a	d	i	k	n	s	t	w
"	v"	q	a	e	i	l	o	p	t	w
"	w"	q	a	e	i	j	o	r	t	w
"	x"	q	a	d	g	j	m	p	t	y
"	y"	q	a	e	i	j	o	p	u	w
"	z"	q	a	e	i	j	o	p	t	y
"	"	q	a	f	i	l	o	s	t	w

FIG. 10T



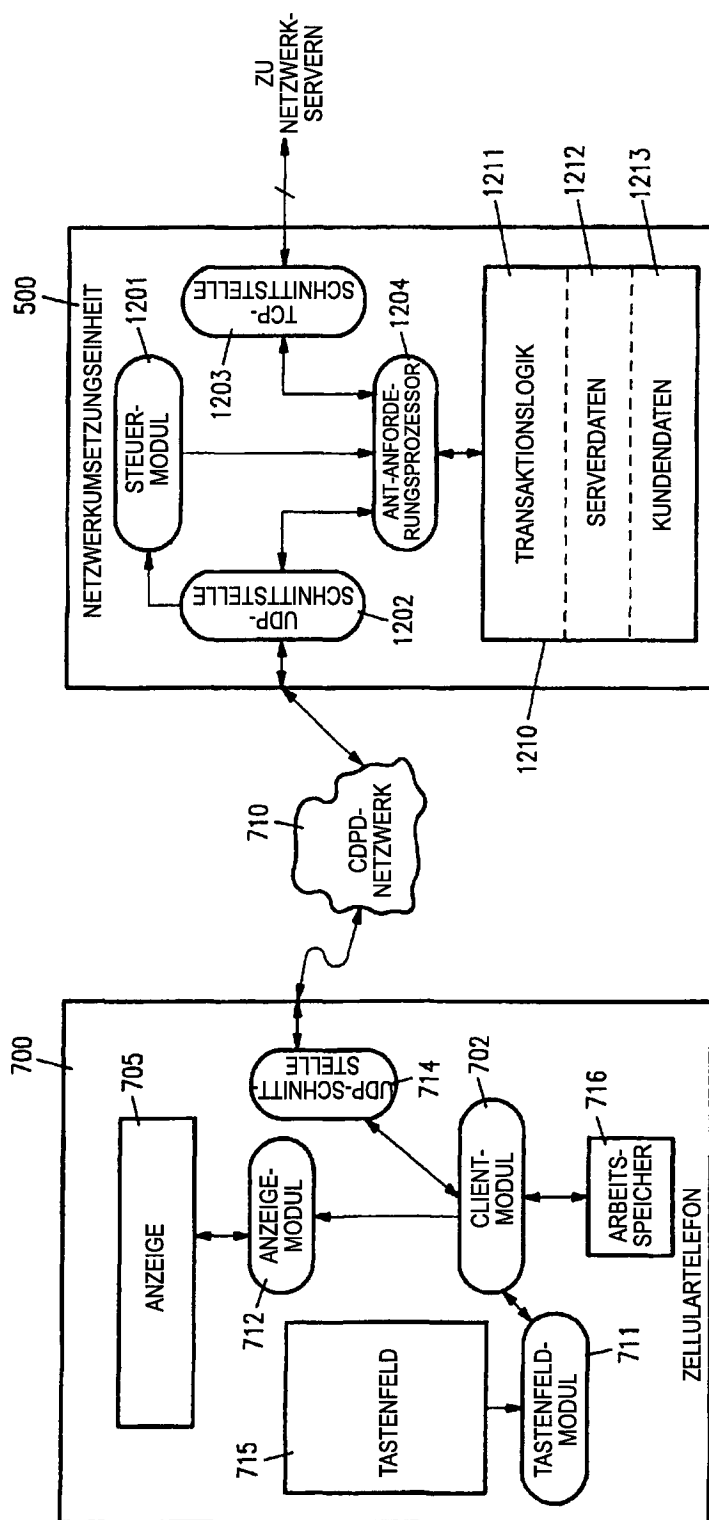
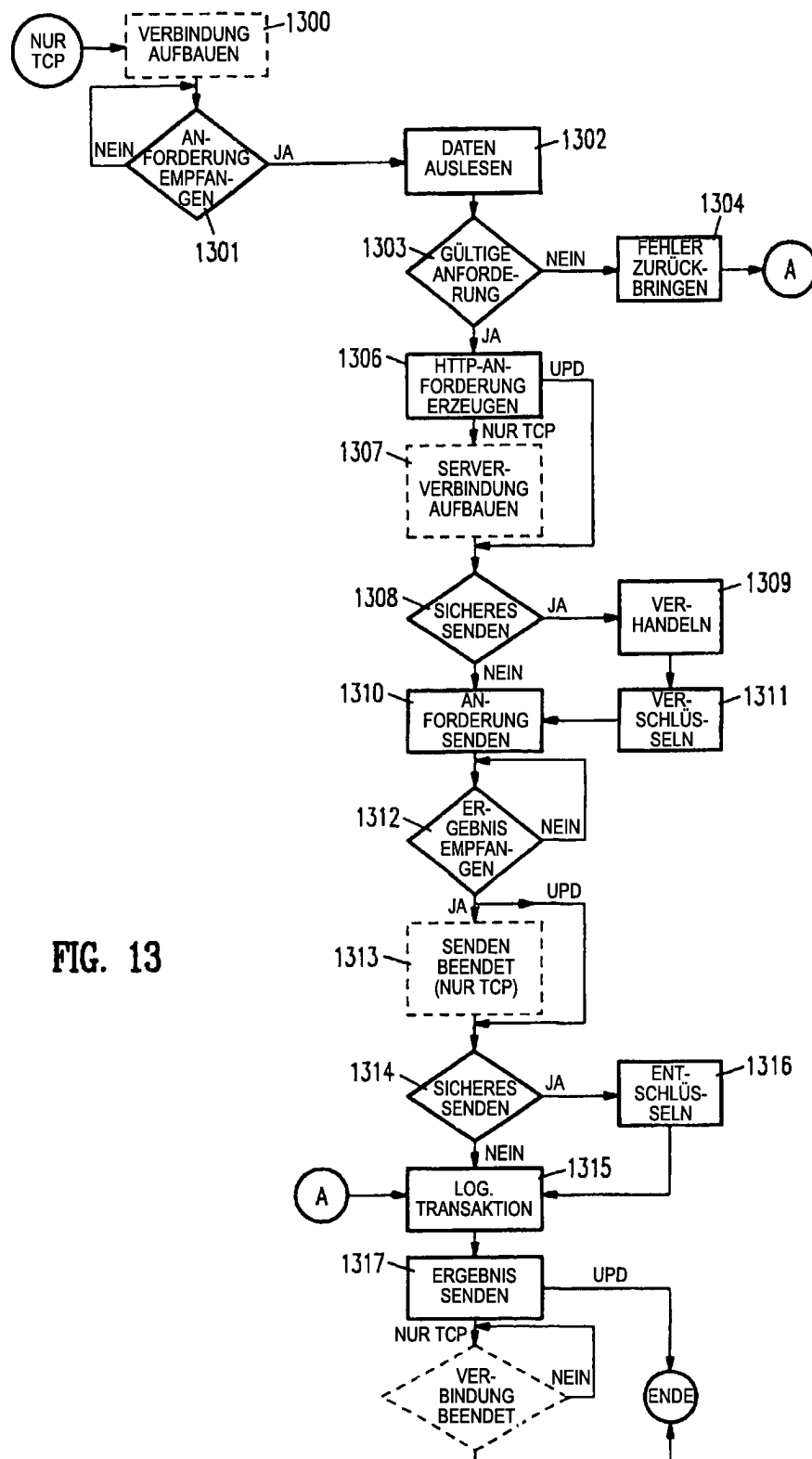


FIG. 12



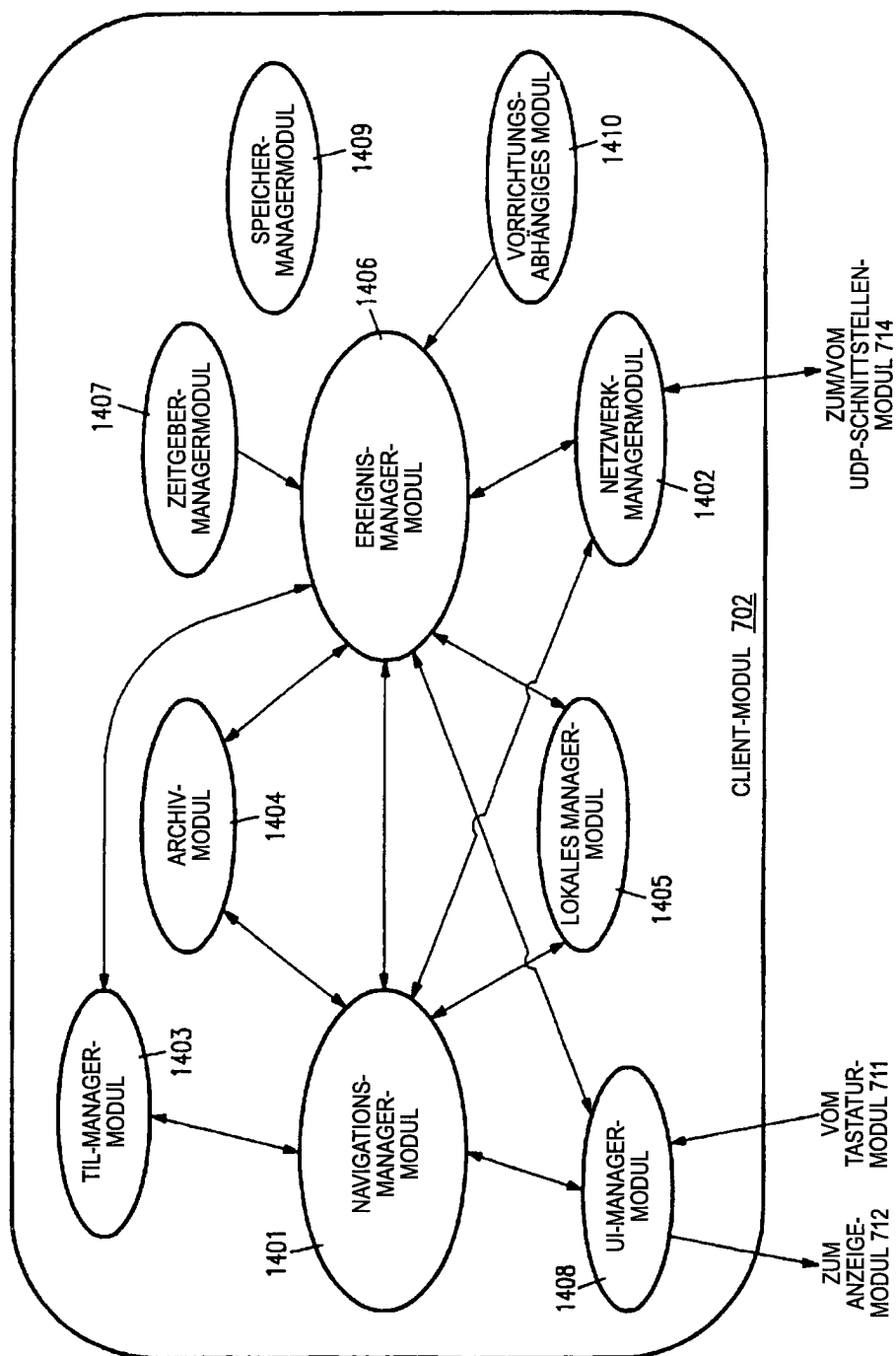


FIG. 14