



(19)  
Bundesrepublik Deutschland  
Deutsches Patent- und Markenamt

(10) **DE 698 35 314 T2** 2007.05.10

(12) **Übersetzung der europäischen Patentschrift**

(97) **EP 0 872 991 B1**

(51) Int Cl.<sup>8</sup>: **H04L 29/06** (2006.01)

(21) Deutsches Aktenzeichen: **698 35 314.5**

(96) Europäisches Aktenzeichen: **98 301 909.2**

(96) Europäischer Anmeldetag: **13.03.1998**

(97) Erstveröffentlichung durch das EPA: **21.10.1998**

(97) Veröffentlichungstag

der Patenterteilung beim EPA: **26.07.2006**

(47) Veröffentlichungstag im Patentblatt: **10.05.2007**

(30) Unionspriorität:

<b>9707551</b>	<b>15.04.1997</b>	<b>GB</b>
<b>54047 P</b>	<b>18.07.1997</b>	<b>US</b>

(84) Benannte Vertragsstaaten:

**DE, FR, GB**

(73) Patentinhaber:

**Hewlett-Packard Development Co., L.P., Houston, Tex., US**

(72) Erfinder:

**Arnold, Patrick Simon, Boise, Idaho 83706, US;**  
**Williams, Peter Michael, Portishead, North**  
**Somerset, GB; Willerup, Frederik, Boise, Idaho**  
**83703, US; Sowden, Anthony, Clifton, Bristol BS8**  
**4PG, GB**

(74) Vertreter:

**Schoppe, Zimmermann, Stöckeler & Zinkler, 82049 Pullach**

(54) Bezeichnung: **Verfahren und Vorrichtung zur formatgesteuerten Interaktion zwischen Geräten**

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]** Die vorliegende Erfindung bezieht sich auf Verfahren und Vorrichtungen für eine Vorrichtungswechselwirkung, insbesondere zum Weiterleiten von Informationen zwischen Vorrichtungen.

**[0002]** Die Forderung nach einem effizienten und erfolgreichen Austausch von Informationen zwischen Vorrichtungen ist überall vorhanden. Für eine Kommunikation zwischen elektronischen Geräten, wie z. B. einem Scanner und einem Drucker, einer Speichervorrichtung und einer Anzeigevorrichtung oder einem Camcorder und einem Fernsehsatz ist ein anwendungsspezifisches Protokoll erforderlich. In den ersten zwei Fällen ist es ferner normal, dass eine Rechenvorrichtung, wie z. B. ein Personalcomputer, eine Kommunikation vermittelt. Ein Personalcomputer benötigt vorrichtungsspezifische Treiber für jedes elektronische Gerät (normalerweise jene, die als „Peripheriegeräte“ für einen Personalcomputer betrachtet werden), um zu ermöglichen, dass diese Kommunikation auftritt.

**[0003]** In der physischen Welt erfordert eine Wechselwirkung mit physischen Vorrichtungen keine speziellen Protokolle. Ein kleiner Satz an generischen Anweisungen wird verwendet und die Wirkung dieser Wechselwirkungen wird durch die Vorrichtung bestimmt, mit der in Wechselwirkung getreten wird. Zum Beispiel ist die Wechselwirkung eines Benutzers mit einem Telefon und einem Photokopierer (üblicherweise) im Wesentlichen die gleiche: das Drücken einer Reihe von Knöpfen. Bei dem Telefon führt dies zum Einrichten einer Verbindung mit einem entfernten Telefon und einem offenen Kommunikationskanal, wohingegen bei dem Photokopierer das Ergebnis die Wiedergabe einer Anzahl von Bildern ist. In jedem Fall ist die grundlegende Wechselwirkung (Drücken von Knöpfen) dieselbe, aber das Ergebnis ist sehr unterschiedlich.

**[0004]** In spezifischen Kontexten finden sich Beispiele einer Wechselwirkung mit digital gehaltenen Informationen auf eine Weise, die vergleichbar mit einer Wechselwirkung mit physischen Vorrichtungen ist. Ein Beispiel ist das von elektronischen „Büchern“, die physischen Büchern ähneln – wie z. B. erörtert wird in „Page Flipping for User Interfaces“, IBM Technical Disclosure Bulletin, IBM Corp., New York, US, Band 36, Nr. 8, 1. August 1993, Seiten 237-239, XP000390208 ISSN: 0018-8869.

**[0005]** Die vorliegende Erfindung erreicht das Weiterleiten von Informationen zwischen Vorrichtungen mit der selben Leichtigkeit und Effektivität, wie sie in der physischen Welt erreicht wird. Genauer gesagt ermöglicht die vorliegende Erfindung, dass eine reiche und effektive Kommunikation mit generischen Anweisungen erreicht wird.

**[0006]** Dementsprechend schafft die Erfindung bei einem Aspekt ein Verfahren zum Freigeben des Sendens von Daten von einer Sende-Informationshandhabungsvorrichtung zu einer Empfangs-Informationshandhabungsvorrichtung in einem Datenformat, das durch die Empfangs-Informationshandhabungsvorrichtung ausgewählt wird, wobei eine Einrichtung für eine Kommunikation von Informationen zwischen den Informationshandhabungsvorrichtungen vorliegt, wobei das Verfahren aufweist, dass die Sende-Informationshandhabungsvorrichtung folgende Schritte ausführt: Bestimmen einer Mehrzahl von Datenformaten, in denen die Daten durch die Sende-Informationshandhabungsvorrichtung gesendet werden können, und Erzeugen einer Datenformathierarchie, bei der jedes der Mehrzahl von Datenformaten durch einen Weg durch die Datenformathierarchie dargestellt ist; und Senden der Datenformathierarchie zu der Empfangs-Informationshandhabungsvorrichtung, so dass die Empfangs-Informationshandhabungsvorrichtung die Daten in einem erforderlichen Datenformat durch Navigieren entlang eines Wegs durch die Datenformathierarchie erhält.

**[0007]** Bei einem weiteren Aspekt schafft die Erfindung ein Verfahren zum Freigeben des Empfangs von Daten von einer Sende-Informationshandhabungsvorrichtung zu einer Empfangs-Informationshandhabungsvorrichtung in einem Datenformat, das durch die Empfangs-Informationshandhabungsvorrichtung ausgewählt wird, wobei eine Einrichtung für eine Kommunikation von Informationen zwischen den Informationshandhabungsvorrichtungen vorliegt, wobei das Verfahren aufweist, dass die Empfangs-Informationshandhabungsvorrichtung folgende Schritte ausführt: Empfangen einer Datenformathierarchie mit jedem einer Mehrzahl von Datenformaten von der Sende-Informationshandhabungsvorrichtung, in denen die Daten durch die Sende-Informationshandhabungsvorrichtung gesendet werden können, dargestellt durch einen Weg durch die Datenformathierarchie; und Navigieren entlang eines Wegs durch die Datenformathierarchie, um aus der Mehrzahl von Datenformaten ein einzelnes Datenformat oder einen Teilsatz der Mehrzahl von Datenformaten auszuwählen.

**[0008]** Bei weiteren Aspekten schafft die Erfindung Informationshandhabungsvorrichtungen, die angepasst sind, um diese Verfahren zu implementieren.

**[0009]** Vorteilhafterweise bestimmt die empfangende Vorrichtung das Format, in dem die Daten dann empfangen werden, durch eine Antwort auf die Sendevorrichtung, die einen Weg durch die Datenformathierarchie aufweist, und alle Datenformate weisen einen oder mehrere einer Mehrzahl von Datenformattypen auf, und wobei für jeden Datenformattyp ein Datenformat existiert, das von allen Informationshandhabungsvorrichtungen empfangen werden kann, die diesen Datenformattyp unterstützen. Es ist bei der Implementierung vorteilhaft, wenn das Verfahren zum Weiterleiten von Informationen ferner Anforderungen nach Inhaltsdaten für einen ausgewählten Weg durch die Datenformathierarchie und Antworten auf solche Anforderungen aufweist.

**[0010]** Ein besonders vorteilhafter Kontext für eine Implementierung der Erfindung ist, wo die Datenformathierarchie in einer Beschreibung einer Oberfläche umfasst ist, wobei die Oberfläche eine Darstellung eines internen Zustands von einer der Informationshandhabungsvorrichtungen ist.

**[0011]** Ein bestimmtes, effektives weiteres Merkmal ist, dass jede Datenformathierarchie eine einzelne Codierung aufweist, wobei jede Codierung eine unterschiedliche grundlegende Form darstellt, in der Informationen präsentiert werden können. Solche Codierungen können Bild, Text, binäre Datei, Ebene oder Zuordnung sein. Vorzugsweise besitzt eine solche Ebene zwei Seiten, und Informationen können auf jeder Seite der Ebene aufbereitet werden. Sowohl Ebenen- als auch Zuordnungs-Codierungen können vorteilhafterweise Kindoberflächen besitzen (die weitere Codierungen anzeigen, die auf der Ebene aufbereitet werden sollen bzw. miteinander verarbeitet werden sollen).

**[0012]** Die Anwendung der Erfindung ermöglicht einen vorrichtungsunabhängigen Informationsaustausch. Ein vollständig reicher Informationsaustausch kann mit dem Kommunikationsmedium erreicht werden, das durch die Erfindung bereitgestellt wird. Das Medium kann bei Ausführungsbeispielen auf transportunabhängige, vorrichtungsunabhängige Weise ausgenutzt werden und die Fähigkeit zum Kommunizieren durch dieses Medium kann in jegliche Vorrichtung eingebaut sein, die einen potentiellen Bedarf für eine Verbindung mit anderen Vorrichtungen unabhängig von ihrer Funktion aufweist. Dies könnte sich von offensichtlichen Kommunikations- und Aufbereitungs-Geräten (wie z. B. Faxmaschinen, Druckern, Magnettafeln) hindurch zu Haushaltsgeräten, wie z. B. Thermostaten, Waschmaschinen und Toastern erstrecken. Bei bevorzugten Ausführungsbeispielen wird das selbe Medium für eine Kommunikation zwischen jedem Gerät verwendet und es wird garantiert, dass Vorrichtungen, die in der Lage sind, den selben Informationstyp zu handhaben, Informationen jeglicher Form weiterleiten können.

**[0013]** Eine geeignete Verwendung der Erfindung ermöglicht einen garantierten Austausch von Informationen mit existierenden und zukünftigen Vorrichtungen. Es ist auch möglich, dass Vorrichtungen kommunizieren, sogar wenn sie keine Kenntnis voneinander haben – kein Vorprogrammierungselement ist erforderlich, wie es für Host-Peripheriegeräte-Umgebungen der Fall ist. Weitere Vorteile sind, dass Systemkomponenten separat entwickelt werden können (z. B. kann ein Drucker entwickelt werden, ohne Berücksichtigung der Erzeugung von Treibern für einen Personalcomputer). Ferner sind bevorzugte Implementierungen der Erfindung derart, dass Produktmerkmale durch den Entwickler definiert sind, nicht das Protokoll: das Protokoll ist nur eine Röhre, durch die Informationen ausgetauscht werden, und hat keine Kenntnis von Vorrichtungen oder Implementierungen.

**[0014]** Spezifische Ausführungsbeispiele der Erfindung werden nachfolgend beispielhaft Bezug nehmend auf die beiliegenden Zeichnungen beschrieben, in denen:

**[0015]** [Fig. 1](#) die Erzeugung eines Oberflächenabdrucks bzw. einer Oberflächenimpression auf einer Vorrichtung aus dem Oberflächen Ausdruck einer anderen zeigt;

**[0016]** [Fig. 2](#) eine Darstellung eines zweiseitigen Dokuments in einer visuellen und einer Oberflächen-Darstellung zeigt;

**[0017]** [Fig. 3](#) die Komponenten der JetSend-Architektur und ihre logische Beziehung zueinander zeigt;

**[0018]** [Fig. 4](#) eine Kopie von Meldungen zeigt, die durch zwei Vorrichtungen ausgetauscht werden, wenn sie eine Sitzung einrichten;

**[0019]** [Fig. 5](#) eine schematische Darstellung einer Sitzung zwischen zwei JetSend-Vorrichtungen zeigt;

**[0020]** [Fig. 6](#) eine schematische Darstellung der Zustände in einer JetSend-Session-Protocol- bzw. -Sitzungsprotokoll-Sitzung zeigt;

- [0021] [Fig. 7](#) eine schematische Darstellung des Startens einer JetSend-Sitzungsprotokoll-Sitzung zeigt;
- [0022] [Fig. 8](#) eine schematische Darstellung des Öffnens und Schließens eines Kanals bei dem JetSend-Sitzungsprotokoll zeigt;
- [0023] [Fig. 9](#) eine schematische Darstellung einer Teilmeldungsfragmentierung bei dem JetSend-Sitzungsprotokoll zeigt;
- [0024] [Fig. 10](#) einen Benutzer eines Zeitgebers beim Planen von Gateway-Rundsende-Meldungen unter Verwendung des JetSend-Sitzungsprotokolls zeigt;
- [0025] [Fig. 11](#) das allgemeine Format eines JetSend-Sitzungsprotokoll-Meldungsanfangsblocks zeigt;
- [0026] [Fig. 12a](#) bis [Fig. 12k](#) die Verwendung von Meldungen des JetSend-Wechselwirkungsprotokolls beim Austausch von Informationen zwischen Geräten zeigen;
- [0027] [Fig. 13](#) die Inhaltsbereitstellung auf einem Meldungskanal zeigt;
- [0028] [Fig. 14](#) die Inhaltsbereitstellung auf einem Stromkanal zeigt;
- [0029] [Fig. 15](#) eine graphische Darstellung einer Selbst-Oberfläche als ein Beispiel einer bekannten Oberfläche zeigt;
- [0030] [Fig. 16](#) eine Probedokumentübertragung gemäß der Auftragstaktik zeigt;
- [0031] [Fig. 17](#) eine graphische Darstellung einer bekannten Oberfläche zeigt;
- [0032] [Fig. 18](#) die Struktur einer generischen Selbst-Oberfläche zeigt;
- [0033] [Fig. 19a](#) bis [Fig. 19e](#) graphische Darstellungen einer Selbst-Oberfläche und ihrer Teil-Oberflächen zeigen;
- [0034] [Fig. 20](#) die Struktur einer generischen Statusoberfläche zeigt;
- [0035] [Fig. 21a](#) bis [Fig. 21e](#) graphische Darstellungen einer Statusoberfläche und ihrer Teil-Oberflächen zeigt;
- [0036] [Fig. 22a](#) eine graphische Darstellung einer Adresse-Innen-Oberfläche zeigt; und [Fig. 22b](#) die Struktur einer Adress-Karten-Oberfläche zeigt, die in Wechselwirkung mit einer Adresse-Innen-Oberfläche ist;
- [0037] [Fig. 23](#) eine Hierarchie von Attributen für die vImage-Codierung zeigt;
- [0038] [Fig. 24](#) eine Hierarchie von Attributen für die vText-Codierung zeigt;
- [0039] [Fig. 25](#) eine Hierarchie von Attributen für die vFile-Codierung zeigt;
- [0040] [Fig. 26](#) eine Hierarchie von Attributen für die vPlane-Codierung zeigt;
- [0041] [Fig. 27](#) schematisch die Anordnung von Elementen für eine vPlane-Codierung darstellt;
- [0042] [Fig. 28a](#) bis [Fig. 28d](#) die Herleitung von Werten für vBackCoord bei der vPlane-Codierung darstellen;
- [0043] [Fig. 29](#) ein Beispiel eines Dokuments zum Codieren mit vPlane darstellt, das vier Kind-Elemente einlagert;
- [0044] [Fig. 30](#) eine Attribut-Hierarchie für die vAssociation-Codierung zeigt;
- [0045] [Fig. 31](#) ein schematisches Diagramm für einen generischen e-Material-Block zeigt;
- [0046] [Fig. 32](#) eine Bilddarstellung eines spezifischen e-Material-Blocks zeigt;

**[0047]** [Fig. 33](#) eine schematische Darstellung einer Oberflächenbeschreibungscodierungshierarchie zeigt; und

**[0048]** [Fig. 34](#) eine Kommunikation zwischen Vorrichtungen darstellt, an die die vorliegende Erfindung anwendbar ist.

**[0049]** Die vorliegende Erfindung ist in der JetSend-Architektur der Hewlett-Packard Company verkörpert. Eine Einführung in diese Architektur wird im World Wide Web unter <http://www.jetsend.hp.com/> bereitgestellt, und eine volle Beschreibung wird in der HP JetSend Communications Technology Protocol Specification geliefert, die von der Hewlett-Packard Company erhältlich und hierin in ihrer Gesamtheit durch Bezugnahme aufgenommen ist.

**[0050]** Die Grundelemente der JetSend-Architektur werden nachfolgend erörtert. Die individuellen Elemente werden dann individuell detaillierter zu dem Ausmaß erörtert, das notwendig ist, um die Erfindung und vorteilhafte Merkmale derselben darzustellen. Eine Bezugnahme auf bevorzugte und erforderliche Merkmale bezieht sich nachfolgend auf bevorzugte und erforderliche Merkmale der JetSend-Architektur und nicht auf bevorzugte und erforderliche Merkmale der Erfindung selbst.

**[0051]** Grundlegend für die JetSend-Architektur sind vier Prinzipien, die definieren, wie das System arbeitet. Dies sind folgende:

Partner-zu-Partner-Wechselwirkung – Wenn ein Netzwerktransport es erlaubt, müssen Vorrichtungen in der Lage sein, andere Vorrichtungen direkt zu adressieren. Vorrichtungen sollten nicht das Vorhandensein eines PC oder einer anderen Zwischenvorrichtung benötigen, um einen Informationsaustausch zwischen zwei Nicht-PC-Vorrichtungen zu ermöglichen. Benutzer können somit eine Verbindung mit einer minimalen Konfiguration herstellen und ad-hoc-Informationsübertragungen ausführen.

**[0052]** Vorrichtungs- und Vorrichtungstyp-Unabhängigkeit – Keine vorrichtungsspezifischen Informationen über andere Vorrichtungen sollten in jegliche Vorrichtung vorprogrammiert sein. Eine Vorrichtung sollte sich nicht unterschiedlich für eine spezifische Vorrichtung oder einen Vorrichtungssatz verhalten müssen, auf die sie treffen kann. Bei der JetSend-Architektur kann eine Vorrichtung allgemeine Operationen ausführen, wie z. B. das Senden von Informationen und das Überwachen des Status, die an sich keinen vorrichtungsspezifischen Aspekt aufweisen. Dies ermöglicht jeglichen zwei für JetSend freigegebenen Vorrichtung, in Wechselwirkung zu treten, ohne Treiber zu verwenden, die spezifisch für eine Vorrichtung oder einen Vorrichtungstyp sind.

**[0053]** Verhandlung – Vorrichtungen sollen Datencodierungen bis zu dem höchsten gemeinsamen Nenner verhandeln, die Sender und Empfänger unterstützen. Es muss eine Standardcodierung für jeden Datentyp vorliegen, den alle Sender und Empfänger, die diesen Datentyp verwenden, unterstützen müssen. Die Fähigkeit zum Verhandeln stellt sicher, dass Vorrichtungen High-End- und sogar proprietäre Datentypen implementieren können, während die standardmäßige Codierung sicherstellt, dass immer ein bedeutungsvoller Datenaustausch stattfindet. Eine Verhandlung sollte flexibel und offen sein.

**[0054]** Konsistenz – Das selbe Protokoll wird verwendet, unabhängig davon, ob die Vorrichtungen Steuerung austauschen, Daten übertragen, Statusinformationen austauschen oder jegliche andere Form von Informationen weiterleiten.

**[0055]** [Fig. 34](#) stellt die Umgebung dar, in der JetSend-Vorrichtungen arbeiten. Ein Netzwerk **341** einer gewissen Art existiert zwischen Vorrichtungen, wie z. B. einem Drucker **342**, einem Personalcomputer **343** und einem Scanner **344**. Jede dieser Vorrichtungen muss einen Prozessor **346** einer bestimmten Art besitzen, natürlich zusammen mit einer Verbindungseinrichtung **345**, um eine Schnittstelle mit dem Netzwerk **341** zu ermöglichen. Es ist bei dieser Implementierung für jede der Vorrichtungen notwendig, ein Maß an Verarbeitungskapazität aufzuweisen, da eine solche Kapazität für Prozesse notwendig ist, die in JetSend integriert sind.

**[0056]** Die Basis für eine Kommunikation zwischen Vorrichtungen bei JetSend ist das Oberflächenwechselwirkungsmodell. Eine Oberfläche ist eine Darstellung eines gewissen Aspekts eines internen Zustands, den eine Vorrichtung besitzt. Die Darstellung ist universell und wird nicht durch die Funktionen bestimmt, die durch die Vorrichtung ausgeführt werden. Eine Oberfläche in diesem Kontext ist analog zu der Oberfläche, die durch ein physisches Objekt bereitgestellt wird (wie z. B. ein Telefon oder einen Grundbaustein). Die Art und Weise, auf die das Objekt arbeitet, wird dadurch bestimmt, wie die „funktionalen“ Teile des Objekts mit der Oberfläche verbunden sind, aber die Oberfläche selbst kann auf einfache, universelle Weise beschrieben werden, unab-

hängig von der Funktion, und liefert das Mittel, durch das das Objekt mit anderen physischen Objekten in der Welt verbindbar ist – das Wesen der Verbindung zwischen physischen Objekten ist hier unabhängig von jeglicher Vorrichtungsfunktion. Bei JetSend ist die Oberfläche die fundamentale Einheit eines Informationsaustauschs, und Bilder, Dokumente, Statusmeldungen, Vorrichtungsetiketten und ähnliches werden alle durch Verwendung von einer oder mehreren Oberflächen übertragen. Eine Oberfläche besteht aus einer Anzahl von Elementen: Beschreibung, Inhalt, Eigenschaft und Klasse – diese werden nachfolgend weiter erörtert.

**[0057]** Das Oberflächenwechselwirkungsmodell definiert Mechanismen zum Erzeugen, gemeinschaftlichen Verwenden, Modifizieren und Löschen von Oberflächen. Nur eine feste Anzahl von generischen Operationen kann auf einer Oberfläche ausgeführt werden, unter Verwendung der Meldungen, die durch das JetSend Interaction Protocol (JIP; JetSend-Wechselwirkungs-Protokoll) definiert sind, das nachfolgend hierin weiter erörtert wird.

**[0058]** Die Originalkopie einer Oberfläche wird hier eine Expression bzw. ein Ausdruck (expression) genannt. Es ist ein Ausdruck bei jeglicher Oberflächenwechselwirkung umfasst. Vorrichtungen, die Informationen zum gemeinschaftlichen Verwenden mit anderen Vorrichtungen aufweisen, erzeugen einen oder mehrere Ausdrücke zum Darstellen dieser Informationen.

**[0059]** Oberflächenausdrücke können auf andere Vorrichtungen abgedruckt werden. Dies erzeugt einen Abdruck bzw. eine Impression (impression) dieser Oberfläche – auch bekannt als gemeinschaftliches Verwenden der Oberfläche. Abdrucke sind Kopie der Ausdrücke anderer Vorrichtungen, und zwischen den Vorrichtungen werden Verbindungen beibehalten, um sicherzustellen, dass Abdrucke ihrem entsprechenden Ausdruck entsprechen. Üblicherweise verwenden zwei Vorrichtungen verschiedene Oberflächen zu einer gegebenen Zeit gemeinschaftlich; Oberflächen, die die Elemente eines Auftrags, Statusinformationen von jeder Vorrichtung, Sicherheitsinformationen usw. darstellen.

**[0060]** [Fig. 1](#) zeigt, wie ein Ausdruck **11** auf einer Vorrichtung (Vorrichtung A) mit einer anderen Vorrichtung (Vorrichtung B) gemeinschaftlich verwendet wird, wodurch ein neuer Abdruck **12** erzeugt wird. Der selbe Ausdruck kann mehrere Male auf mehrere Vorrichtungen abgedruckt werden. Dies erzeugt mehrere Abdrucke des selben Oberflächenausdrucks. Wenn sich der Ausdruck ändert oder gelöscht wird, werden alle Abdrucke benachrichtigt. Dies wird hier Änderungsausbreitung genannt.

**[0061]** Eine Änderungsausbreitung ermöglicht, dass interessierende dynamische Merkmale, wie z. B. Statusinformationen, für Vorrichtungen in der Architektur implementiert werden. Zum Beispiel erzeugt eine Vorrichtung mit Statusinformationen einen Oberflächenausdruck, der eine Statuszeichenfolge enthält, die auf eine Anzahl von angeschlossenen Vorrichtungen abgedruckt wird. Die Vorrichtung mit dem Status ändert die Oberfläche abhängig von ihrem Zustand. Alle JetSend-Vorrichtungen, die einen Abdruck der Statusoberfläche aufweisen, empfangen Aktualisierungsmeldungen mit dem aktuellen Status. Diese Funktionalität ist in die Architektur eingebaut und ihre Verwendung durch Vorrichtungsentwickler ist optional.

**[0062]** Von einem Implementierungsstandpunkt aus ist ein Ausdruck durch den Zustand definiert, der über eine gegebene Oberfläche beibehalten wird (ihren Namen, ihre Beschreibung, ihren Inhalt und den Satz aus Abdrucken, die an entfernten Vorrichtungen gemacht wurden). Es wird später gezeigt, dass von einem Protokollstandpunkt aus ein Ausdruck durch die Oberflächenhandhabung definiert ist, die verwendet wird, um JIP-Meldungen über dieselbe auszutauschen.

**[0063]** Eine Oberfläche weist eine Beschreibung der Oberfläche auf und den Inhalt, der in der Oberfläche enthalten ist. Die Unterscheidung ist grundlegend für JetSend und von Bedeutung bei Aspekten der vorliegenden Erfindung, da sie einen Mechanismus für eine Inhaltsverhandlung schafft.

**[0064]** Die Oberflächenbeschreibung richtet den vollen Bereich von möglichen Formen ein, in denen Informationen, die einer Oberfläche zugeordnet sind (wobei diese Informationen als Oberflächeninhaltsdaten bereitgestellt werden), mit einer anderen Vorrichtung gemeinschaftlich verwendet werden können. Die Beschreibung besteht aus einer Datenformathierarchie und kann als eine Baumstruktur mit einer Anzahl von Knoten dargestellt sein, die jeweils einen definierbaren Aspekt des Datenformats darstellen (bezeichnet als ein Attribut und hierin nachfolgend anderswo detaillierter erörtert). Ein spezifisches Datenformat für die Informationen, die der Oberfläche zugeordnet sind, ist ein Weg durch diese Baumstruktur von einem Wurzelknoten bis hinunter zu einem Anschlussknoten (einem Blattknoten des Baums). Solche spezifischen Wege werden durch einen Prozess einer Verhandlung erreicht. Der Verhandlungsprozess weist in diesem Kontext auf, dass der Oberflächeneigentümer die Oberflächenbeschreibung oder einen Teil der Oberflächenbeschreibung mit einer anderen

Vorrichtung gemeinschaftlich verwendet. Die andere Vorrichtung wählt dann die Optionen aus, die sie bevorzugt (im Allgemeinen werden solche Optionen so ausgewählt, dass die Empfangsvorrichtung ihre eigene Funktionalität verwenden kann, auf die beste Weise, die im Hinblick auf die gebotenen Auswahlmöglichkeiten verfügbar ist, aber dies ist im Wesentlichen ein Aspekt für den Entwickler jeglicher gegebenen Vorrichtung), und dieser Prozess wird fortgesetzt, bis ein spezifischer Weg und somit ein spezifisches Datenformat ausgewählt ist.

**[0065]** Der Oberflächeninhalt sind die Informationen, die der Oberfläche zugeordnet sind, und werden in einem Format bereitgestellt, das durch Verhandlung bestimmt wird, wie oben angezeigt ist. Die Inhaltsdaten können in allen Formaten vorgesehen sein, die durch die Datenformathierarchie angezeigt sind, die in der Oberflächenbeschreibung verkörpert ist, aber eine gegebene Oberflächenwechselwirkung wird allgemein nur in einem der verfügbaren Formate geliefert. Es besteht die Möglichkeit zum Liefern von Informationen in mehr als einem Format für einen einzelnen Abdruck: ein Fall, in dem sich dies als nützlich herausstellen kann, ist bei der Wiedergewinnung von Informationen aus einer Speichervorrichtung, wo z. B. ein Bild anfänglich als eine Skizze mit niedriger Auflösung geliefert werden kann und später, nach Bestätigung, dass das Bild erwünscht ist, mit voller Größe und hoher Auflösung ohne Änderung des Abdrucks geliefert wird. Eine Forderung nach weiterem Inhalt wäre jedoch in einer solchen Situation normal. Inhaltsdaten müssen vor der Fertigstellung der Verhandlung nicht existieren: für einige oder alle der Auswahlmöglichkeiten in der Datenformathierarchie können die Daten erst erzeugt werden, nachdem die Formatauswahl bekannt ist. Alternativ können für einige Auswahlmöglichkeiten die Inhaltsdaten direkt zusammen mit der Beschreibung geliefert werden, die einen bestimmten Punkt in der Datenformathierarchie als einen Teil des Verhandlungsprozesses identifiziert (so dass kein separater Schritt zur Bereitstellung von Inhalt für die Formatauswahl erforderlich ist). Dies wird als das eingereihte Bereitstellen von Inhaltsdaten bezeichnet.

**[0066]** Wie oben angezeigt ist, werden Beschreibungen zusammen mit dem Originaloberflächenabdruck gesendet, aber in dem allgemeinen Fall wird Inhalt separat angefordert. Zum Beispiel könnte eine Oberfläche, die ein Rasterbild darstellt, eine Oberflächenbeschreibung definieren, mit Auswahlmöglichkeiten über Farbräume, Auflösungen und Komprimierungen für das Bild. Der Empfänger des Abdrucks fordert die tatsächlichen Pixeldaten (den Inhalt) an, der ein bevorzugtes Codieren für den Empfänger aus den verfügbaren Auswahlmöglichkeiten spezifiziert.

**[0067]** Eine Oberflächenbeschreibung kann ferner Bezugnahmen auf andere Oberflächen enthalten. Diese Oberflächen sind als Teiloberflächen (oder Kindoberflächen) bekannt, und sie müssen separat von dem Eigentümer der Elternoberfläche angefordert werden. Jede Teiloberfläche enthält ihre eigene Beschreibung und Inhalt, unabhängig von seinen Eltern. Ein Dokument kann z. B. als mehrere Oberflächen codiert sein, die jeweils eine separate Seite darstellen. Das Dokument selbst ist als eine Oberfläche dargestellt, die Bezugnahmen auf diese Seitenoberflächen enthält.

**[0068]** Ein Dokument hat in diesem Kontext eine umfassendere Bedeutung als eine Sammlung an Seiten, die gebunden oder anderweitig in einer einzelnen Einheit gehalten sind. Hier ist ein Dokument jegliche Sammlung aus Informationen, die in einer solchen Form gehalten werden, dass sie durch eine Vorrichtung zum Verarbeiten durch eine andere Vorrichtung geliefert werden können. In dem spezifischen Kontext von JetSend weist ein Dokument eine oder mehrere Oberflächen auf, wobei jede Oberfläche mit allen anderen Oberflächen entweder direkt durch eine einzelne Eltern-Kind-Beziehung oder indirekt durch eine Kette aus Eltern-Kind- oder Kind-Eltern-Beziehungen verbunden ist.

**[0069]** Eine Oberflächenbezugnahme, oder ein Oberflächenname, ist einfach eine ASCII-Zeichenfolge, wie z. B. ein URL. Alle Oberflächen werden durch ihren Namen identifiziert und können zu einer bestimmten Klasse gehören. Alle Oberflächen besitzen eine Klasse. Klasse ist eine Eigenschaft, die den Zweck anzeigt, für den die Oberfläche verwendet wird; bei der vorliegenden Implementierung muss jede Oberfläche eine und nur eine Klasse aufweisen. Klassen von Oberflächen umfassen die Selbst-Oberfläche, Innen-Oberfläche, Status-Oberfläche und Adress-Oberfläche. Es existieren spezifische Regeln oder können erdacht werden, zur Verwendung von Oberflächen einer bestimmten Klasse. Klassen werden insbesondere im Kontext von spezifischen Taktiken verwendet. Zum Beispiel verwendet die Auftragstaktik eine bestimmte Oberflächenklassik, die Innen-Oberfläche, als Mittel zum Liefern von Aufträgen zu einer Auftragsverarbeitungsvorrichtung. Ein Drucker hat eine Innen-Oberfläche, und Oberflächen, die auf die Ein-Oberfläche abgedruckt sind, werden als neue Aufträge behandelt und gedruckt. Ein PC, eine Kamera oder eine andere Sendevorrichtung kann Oberflächen auf diese Innen-Oberfläche abdrucken. Wo eine bestimmte Klasse einer Oberfläche eine grundlegende Rolle bei der Ausführung einer gegebenen Taktik spielt, werden Oberflächen dieser Klasse als „bekannte Oberflächen“ für diese Taktik bezeichnet.

**[0070]** Eine Taktik, oder Wechselwirkungstaktik, ist ein Satz aus Konventionen bzw. Übereinkünften zur Verwendung von generischen Oberflächenwechselwirkungen zum Erreichen einer bestimmten Aufgabe. Der Vorteil des Verwendens einer Taktik für eine bestimmte Aufgabe ist, dass es ermöglicht, dass alle JetSend-Vorrichtungen angepasst sind, die bestimmte Aufgabe auszuführen oder eine andere Vorrichtung erfordern, um die bestimmte Aufgabe auszuführen, um erfolgreich und auf vorhersehbare Weise mit anderen Vorrichtungen in Wechselwirkung zu treten, um diese bestimmte Aufgabe zu erreichen. Dies schließt die Möglichkeit nicht aus, dass z. B. ein Vorrichtungspaar, das zum Ausführen einer solchen Aufgabe auf eine unterschiedliche Weise angepasst ist, die anderen JetSend-Vorrichtungen nicht zugänglich ist, eine andere Oberflächenklasse verwendet, um dies auszuführen. Es ist jedoch wünschenswert, dass solche Vorrichtungen weiterhin die relevante JetSend-Taktik unterstützen, so dass sie diese Aufgabe auch mit anderen JetSend-Vorrichtungen erreichen können.

**[0071]** Alle Oberflächenwechselwirkungen zwischen zwei Vorrichtungen werden mit den folgenden Meldungen ausgeführt:

SurfaceRequestMsg – Anforderung einer Oberfläche eines bestimmten Namens (und Klasse)  
 SurfaceMsg – Abdrucken einer Oberfläche (Senden ihrer Beschreibung)  
 ContentRequestMsg und ContentReplyMsg – Übertragen von Oberflächeninhalt  
 DescriptionRequestMsg und DescriptionReplyMsg – Übertragung zusätzlicher Beschreibung  
 SurfaceChangeMsg – Benachrichtigen und Anfordern von Änderungen einer Oberfläche  
 SurfaceReleaseMsg – Beseitigen der Verbindung zwischen einem Abdruck und seinem Ausdruck

**[0072]** Das Format dieser Meldungen und ihre Implementierung wird zu einem späteren Zeitpunkt beschrieben.

**[0073]** E-Material, ein Ausdruck, der als abgekürzte Version von „elektronischem Material“ geprägt ist, ist die Form, die von Informationen angenommen wird, durch die Oberflächen ausgedruckt und gemeinschaftlich verwendet werden. Es weist Beschreibung und Inhalt auf. Die Beschreibung zeigt eine Hierarchie aus Auswahlmöglichkeiten von Format an, in dem die zugeordneten Inhaltsdaten bereitgestellt werden können. Der Inhalt sind die Daten, die der Oberfläche selbst zugeordnet sind. Die Beschreibung zeigt die Möglichkeiten an, auf die Daten präsentiert werden können, wobei der Inhalt die Informationen selbst sind, präsentiert in einer Form, die als geeignet für die Vorrichtung ausgewählt ist, die sie sendet, und die Vorrichtung, die sie empfängt. Die Existenz dieser Auswahl ist der Schlüssel für das Konzept von E-Material, und der Mechanismus, durch den die Auswahl getroffen wird, wird nachfolgend weiter erörtert.

**[0074]** Eine Oberflächenwechselwirkung und E-Material sind das Wesen der JetSend-Architektur, und Aspekte der vorliegenden Erfindung sind von besonderer Bedeutung für die implementierte Kombination aus Oberflächenwechselwirkung und E-Material. Die zwei Konzepte sind getrennt aber eng verwandt. Eine Oberflächenwechselwirkung ist das Protokoll zum Austausch von Informationen, die als E-Material codiert sind. Die Kombination aus Oberflächen und E-Material ermöglicht es JetSend-Vorrichtungen, Datentypen zu verhandeln und Informationen auf vorrichtungsunabhängige Weise auszutauschen.

**[0075]** E-Material ist kein Dateiformat. E-Material definiert das Format der Daten, die Vorrichtungen austauschen, aber es definiert nicht, wie Vorrichtungen die Daten verarbeiten oder speichern. Vorrichtungen, die E-Material verbrauchen, tun dies auf eine Weise, die für diese Vorrichtung spezifisch ist. Zum Beispiel verarbeitet eine Empfangsvorrichtung, wie z. B. ein Drucker, E-Material unterschiedlich zu einer Empfangsvorrichtung, wie z. B. einem PC. Der PC kann das E-Material in eine Art von Dateiformat setzen, während der Drucker es in PCL oder PostScript™ umwandeln kann und schließlich in Punkte auf einer Seite. Auf ähnliche Weise werden Vorrichtungen, die E-Material erzeugen, dies auf eine Weise tun, die für diese Vorrichtung spezifisch ist.

**[0076]** Die grundlegendste Unterteilung von E-Material-Typen ist in Codierungen. Eine Codierung ist eine E-Material-Darstellung einer grundlegenden Form, in der Informationen in der Welt präsentiert werden können. Die Vorstellung von einer „grundlegenden Form“ kann durch die unterschiedlichen Codierungen dargestellt werden, die bislang definiert sind.

vText bezieht sich auf eine Präsentation von Informationen in der Form von geschriebenem Text. Eine Zeichenfolge aus ASCII oder Unicode-Text, die eine rechteckige Ebene füllt (keine Schriftart- oder Beabstandungs-Informationen). Die einfache Textcodierung wird hauptsächlich für Statusinformationen verwendet, kann aber auch zum Markieren von Seiten verwendet werden. Der Symbolsatz ist verhandelbar.

vlmage bezieht sich auf die Präsentation von Informationen in der Form von graphischen Bildern. Die Codierung bezieht sich auf ein Rasterbild, für das die Möglichkeit zum Verhandeln von Qualitäten besteht, wie z. B.

Farbraum, Pixeltiefe, Auflösung, Pixelcodierung und Komprimierung.

vPlane bezieht sich auf die Präsentation von Informationen auf einem planaren Objekt, von dem die allgemeinste Form ein Blatt Papier ist – wie im echten Leben bezieht sich vPlane auf planare Objekte mit zwei Seiten. Genauer gesagt liefert diese Codierung eine zweidimensionale, rechteckige Ebene mit einer Hintergrundfarbe und einer geordneten „Schicht“ aus Kindelementen auf seiner Vorder- und Rückseite. Eine Seite ist eine Ebene, die andere Codierungen enthält, wie z. B. Bilder oder Text.

vFile bezieht sich auf die Präsentation von Informationen in der Form einer binären Datei. Binäre Dateien umfassen solche Dinge wie Dokumente in anwendungsspezifischen Formaten, Ausführerelemente, etc. Ein Dateiname, Mime-Typ und ein Bildsymbol bzw. Icon kann den Daten zugeordnet sein. Der Mime-Typ kann verhandelt werden.

vAssociation bezieht sich auf die Darstellung von Informationen in der Form einer Anordnung von Informationen, die in einem oder mehreren Formaten präsentiert sind (die jeweils durch ihre eigene Codierung dargestellt werden). Im Wesentlichen ist eine Zuordnung eine geordnete Sequenz aus Kindelementen. Ein herkömmliches Dokument könnte als eine Zuordnung von Seiten dargestellt sein.

**[0077]** Weitere Codierungen gemäß der vorliegenden Erfindung sind ohne weiteres denkbar und sind geeignet für Vorrichtungstypen und Funktionen, die nicht in der existierenden JetSend-Beschreibung adressiert werden: z. B. können Stimm- oder Klang-Codierungen erdacht sein, um Informationen in der Form einer menschlichen Stimme oder einem Klang im Allgemeinen darzustellen. Es sollte auch erwähnt werden, dass der Ausdruck „Codierung“ hier nicht nur verwendet wird, um grundlegende Formattypen zu bezeichnen, sondern auch eine spezifische Datenformathierarchie (z. B. eine, die am Ende eines bestimmten Verhandlungsprozesses erreicht wird). Solche Verwendungen des Ausdrucks „Codierung“ sind jedoch ohne weiteres durch den Kontext unterscheidbar.

**[0078]** Für jede Codierung (vImage, vText etc.) ist eine einzelne Standardcodierung definiert. Dies stellt sicher, dass alle Vorrichtungen, die in der Lage zu einem Austausch von Daten eines bestimmten Typs sind (z. B. alle Vorrichtungen, die Bilder in einer gewissen Form darstellen können – Personalcomputer, Drucker, Scanner, aber kein Mikrophon), in der Lage sind, Daten in einer bestimmten Form auszutauschen. In dem Fall eines Bildes ist die Basiscodierung bei der vorliegenden Implementierung vImage.vGray.1.(300, 300).vRLE: alle Vorrichtungen, die in der Lage sind, Bildinformationen zu verwenden, können Informationen in diesem Format austauschen.

**[0079]** Es ist allgemein ein besseres Format für den Austausch von Daten verfügbar als die Standardcodierung – die natürlich zur Verwendung durch die Vorrichtungen mit dem am wenigsten reichen Funktionssatz verfügbar sein muss. Es ist trotzdem wünschenswert, Standardformate zu haben, die allgemein für einen Austausch durch Vorrichtungen höherer Funktionalität verfügbar sind. Solche Codierungen werden Basiscodierungen genannt: Basiscodierungen werden nicht durch alle Vorrichtungen unterstützt, aber es würde erwartet werden, dass alle Vorrichtungen mit einer ausreichend reichen Funktionalität, um dies zu tun, den Austausch von Daten gemäß einer Basiscodierung unterstützen würden.

**[0080]** Ein Attribut ist ein Merkmal eines Stücks aus E-Material, das in der E-Material-Beschreibung definiert ist (oder für das Stück aus E-Material definiert ist nach der Fertigstellung des Verhandlungsprozesses). Im Wesentlichen stellt jeder Knoten in der Datenformathierarchie, der in der E-Material-Beschreibung vorgesehen ist, ein Attribut dar. Aus dieser Äquivalenz zwischen Knoten und Attributen wird das Konzept einer „Attributebene“ hergeleitet: im Wesentlichen wird eine Attributebene gemeinschaftlich durch alle Knoten verwendet, die gleich weit entfernt von dem Wurzelknoten der Datenformathierarchie sind. Attribute weisen eine Qualität auf, die für den E-Material-Inhalt vorgesehen sein kann, und eine Auswahl oder einen Wert für diese Qualität: dies kann von der Größe reichen, in der ein Bild vorgesehen ist, oder der Sprache, in der Text vorgesehen ist, zu etwas weniger greifbaren Qualitäten (wie z. B. der Reihenfolge, in der Elemente auf einer Ebene aufbereitet sind), und die Auswahl oder der Wert kann selbst ein weiteres Attribut darstellen (eine Qualität, die eine Auswahl oder einen Wert erfordert). Die Qualität selbst ist durch einen Attributnamen dargestellt (manchmal abgekürzt auf „Attribut“ in dieser Beschreibung), wohingegen die Auswahl oder der Wert durch einen „Attributwert“ dargestellt ist (manchmal abgekürzt auf Wert in dieser Beschreibung): folglich können die Knoten der Datenformathierarchie derart betrachtet werden, dass sie eine spezifische Qualität liefern und entweder eine spezifische Auswahl oder einen Bereich von Auswahlmöglichkeiten für diese Qualität. Attributnamen sind Schlüsselwörter, für die verfügbaren Werte oder Auswahlmöglichkeiten vordefiniert sind, und so müssen die Optionen, die in einer gegebenen Oberflächenbeschreibung für ein gegebenes Schlüsselwort angeboten sind, eine gewisse Auswahl aus diesen vordefinierten Auswahlmöglichkeiten sein. Für unterschiedliche Codierungen sind bestimmte Attribute erforderlich, andere sind optional und andere werden überhaupt nicht verwendet.

**[0081]** E-Material drückt somit die Codierungsauswahlmöglichkeiten einer Oberfläche mit einer Anzahl von Attributwertpaaren aus, die in einer Hierarchie angeordnet sind. Alle Oberflächenbeschreibungen enthalten das Attribut vEncoding (alle Attribut- und Wert-Namen haben als Präfix den Kleinbuchstaben v). Wie oben angezeigt ist, bestimmt dies die Gesamtcodierung der Oberfläche – z. B. wenn es Rasterbild ist, ist vEncoding vImage. Es kann auch eine Liste aus Auswahlmöglichkeiten sein, wo der Eigentümer dieser Oberfläche ihre Inhalte in mehr als einer Codierung bereitstellen kann.

**[0082]** Jede Codierung weist eine Anzahl von standardmäßigen Attribut-Wert-Paaren auf, die derselben zugeordnet sind. Zum Beispiel definiert vImage Attribute, wie z. B. Auflösung, Pixelgröße, Farbraum und Komprimierung. Einige dieser Attribute können auch Auswahlmöglichkeiten enthalten. Tatsächlich kann der Eigentümer der Oberfläche relativ komplexe Auswahlmöglichkeiten ausdrücken durch Hinzufügen von Optionen zu vielen der Attribute in der Oberflächenbeschreibung.

**[0083]** Die Attribute sind in einer Hierarchie angeordnet, die unterschiedliche Auswahlmöglichkeiten für ein unterschiedliches Attribut erlaubt, abhängig von vorangehenden Wahlen. Wenn z. B. eine Bildcodierung für eine Oberfläche angeboten wird, könnte sie eine Auswahl aus JPEG- oder JPEG-LS-Komprimierung enthalten, wenn SRGB als der Farbraum ausgewählt ist, aber RLE oder Group-3-Faxkomprimierung, wenn der Farbraum Monochrom ist.

**[0084]** Oberflächenbeschreibungen, die Codierungsauswahlmöglichkeiten auflisten, sind in einer Tabellenform geschrieben, genannt eine Beschreibungstabelle. Beispiele von Beschreibungstabellen werden später in dieser Spezifizierung gezeigt. Sobald eine Vorrichtung eine Oberflächenbeschreibung empfängt, wird sie vor die Entscheidung gestellt, welche Codierung sie für diese Oberfläche bevorzugt. Sobald sie getroffen ist, wird die Auswahl zu dem Eigentümer der Oberfläche gesendet, mit einer Anforderung nach den tatsächlichen Inhaltsdaten.

**[0085]** Die Beschreibungstabelle listet die Attribute in den drei Spalten auf: dem Pegel, dem Attribut und seinem Wert. Der Pegel spezifiziert die Werte anderer Attribute, auf die ein gegebenes Attribut zutrifft. Zum Beispiel kann das Attribut „vResolution“ mit dem Wert „(150, 150)“ nur gelten, wenn die Auswahlen „vImage.vS-RGB.24“ für vorangehende Attribute getroffen wurden, die in der selben Beschreibung aufgelistet sind.

**[0086]** Alle Codierungen können im Hinblick auf das Attribut vLanguage verhandelt werden. Jegliche Oberfläche kann in mehreren Sprachen zu Lokalisierungszwecken angeboten werden. Für jede Sprache können unterschiedliche Auswahlmöglichkeiten über die Codierung gegeben sein. Es ist sogar möglich z. B. Text für eine Sprache und ein Bild für eine andere anzubieten.

**[0087]** Die JetSend-Spezifikation, die hier beschrieben ist, konzentriert sich auf das Modellieren von statischen, zweidimensionalen, visuellen Informationen. Sie spezifiziert daher Codierungen, die nützlich zum Darstellen von Informationen sind, die auf physischem Papier enthalten sein können. Das E-Material ist nicht auf diesen Informationstyp beschränkt, und neue Codierungen können ohne weiteres innerhalb des Kontexts der vorliegenden Erfindung erdacht werden, die im Prinzip auf jegliche Form von Informationen anwendbar sind, die zwischen Vorrichtungen ausgetauscht werden können. Es ist ebenfalls möglich, proprietäre Codierungen zu erfinden, so lange bestimmte Regeln befolgt werden. Dies ermöglicht eine effektive Kommunikation zwischen Paaren oder Gruppen aus Vorrichtungen.

**[0088]** [Fig. 2](#) zeigt ein zweiseitiges Dokument. Eine visuelle Darstellung **21** ist links gezeigt und ihre Oberflächendarstellung rechts. In diesem Fall ist sie unter Verwendung von sieben Oberflächen dargestellt, jede mit ihrer eigenen Codierung. Die Rechtecke sind Oberflächen, die Bezugnahmen auf andere Oberflächen enthalten. Die Oberfläche **22** oberer Ebene ist eine Zuordnung zu einer Bezugnahme auf jede der zwei Seiten des Dokuments. Jede Seite ist als eine Ebenencodierung **23**, **24** dargestellt. Jede Ebene enthält zwei Bezugnahmen auf Oberflächen, die die Informationen auf den Seiten darstellen. Die erste Ebene **23** zeigt auf eine Bildoberfläche **25** und eine Textoberfläche **26**. Die zweite Ebene **24** zeigt auf zwei Bildoberflächen **27**, **28**.

**[0089]** Jede Bildoberfläche wird separat verhandelt und könnte unterschiedliche Codierungsangebote enthalten. Dies ist nur eine Möglichkeit, dass dieses Dokument unter Verwendung von Oberflächen und E-Material dargestellt werden kann. In diesem Fall wird es in die Elemente unterteilt, die auf der Seite sichtbar sind – d. h., die drei Bilder und das Textsegment. Eine andere Möglichkeit ist das Codieren der Informationen als zwei Ebenen, jede mit einer großen Bildregion, die die gesamte Seite abdeckt. Es ist eine Entwurfsauswahl des Eigentümers eines Dokuments, wie seine Informationen codiert sein sollen.

**[0090]** Wie oben angezeigt ist, ist in diese Architektur das Prinzip der Bereitstellung einer Standardcodierung integriert. Für jede Codierung besteht ein Standardattributsatz, den alle JetSend-Vorrichtungen unterstützen müssen. Die Standardcodierung stellt sicher, dass jegliche Kombination aus Vorrichtungen, die dieselbe Klasse aus Informationen austauschen, in der Lage ist, zu kommunizieren. Sie stellt ferner eine Vorwärtskompatibilität sicher: neue Vorrichtungen können klügere Codierungen implementieren, so lang sie die Standardmäßige unterstützen.

**[0091]** Alle Vorrichtungen müssen die Zuordnungscodierung unterstützen, die einfach ein Behälter für Oberflächen mit anderen Codierungen ist. Ferner müssen alle Vorrichtungen, die mit statischen, zweidimensionalen Informationen umgehen, die Ebenencodierung unterstützen.

**[0092]** Die Standardcodierung für Bilderzeugungsvorrichtungen ist 300 × 300 dpi, Monochrom, RLE-komprimiertes Raster. Das bedeutet, dass alle JetSend-Bildvorrichtungen (Drucker, Scanner, Kopierer, Kameras, Magnettafeln etc.) in der Lage sein müssen, diese Codierung zu senden und/oder zu empfangen. Text- und Datei-Codierungen sind optional für solche Vorrichtungen. Da Vorrichtungen, die andere Datenklassen unterstützen, implementiert sind, werden Standardcodierungen für diese Datentypen ebenfalls eingeführt. Dies würde solche Datentypen wie Audio und Video umfassen. Der Standard ist als niedrigster gemeinsamer Nenner für eine gegebene Codierung gedacht.

**[0093]** Wie oben weiter angezeigt ist, ist zusätzlich zu Standardcodierungen, die Bereitstellung von Basiscodierungen ein optionales aber vorteilhaftes Merkmal. Dieses sind empfohlene Codierungen höherer Ebene, die bessere Ergebnisse für Vorrichtungen mit höheren Fähigkeiten sicherstellen. Zum Beispiel besteht für Farbbilderzeugungsvorrichtungen eine Basiscodierung für Farbbilder, die diese Vorrichtungen zusätzlich zu den Standardcodierungen unterstützen sollten. Einer Vorrichtung steht es frei, zusätzliche Codierungen zu bieten, so lange die Standardcodierungen ebenfalls angeboten werden. Die Vorrichtung kann einen besseren Verhaltens- oder Qualitäts-Pegel dadurch bieten, dass auch die Basiscodierungen implementiert sind, die am besten für diesen bestimmten Vorrichtungstyp geeignet sind.

**[0094]** Die JetSend-Protokolle sind entworfen, um unabhängig von Vorrichtungstyp, Plattform und Transport zu sein. Ein kurzer Überblick über jede der Funktionskomponenten der JetSend-Geräte. Gerät ist ein Ausdruck, der verbreitet in dieser Spezifikation verwendet wird und dem eine spezifischere Bedeutung gegeben ist als Vorrichtung. Ein Gerät ist eine Vorrichtung mit einer zweckgebundenen Funktion, das in der Lage zu einer im Wesentlichen unabhängigen Handhabung der Daten ist, die zu demselben geliefert werden, um diese zweckgebundene Funktion zu erreichen. Es hat somit eine breitere Anwendung als ein reines Peripheriegerät z. B. für einen Personalcomputer. Ein Gerät ist im Allgemeinen nicht neu programmierbar und benötigt im Allgemeinen nur eine minimale Menge an Konfiguration zum Arbeiten. Vorrichtungen, wie z. B. Scanner und Drucker, funktionieren wünschenswerterweise als Geräte, wenn sie angepasst sind, um die vorliegende Implementierung der vorliegenden Erfindung zu unterstützen.

**[0095]** Es gibt drei primäre Funktionalitätsbereiche, die ein JetSend-Gerät bilden; die Transporte in der Vorrichtung, die JetSend-Protokolle selbst und den vorrichtungsspezifischen Code. [Fig. 3](#) identifiziert die Komponenten der JetSend-Architektur und ihre logischen Beziehungen zueinander. Dies wird gefolgt von einem Überblick über jede der Komponenten. Details für jede Komponente werden zu einem späteren Zeitpunkt gegeben. Es sollte darauf hingewiesen werden, dass [Fig. 3](#) kein Implementierungsdiagramm ist. Sie zeigt die Beziehung zwischen dem Protokoll, nicht zwischen den Softwarekomponenten. Tatsächliche Implementierungen können ähnliche Komponenten aufweisen oder kombinieren die Implementierung von mehreren Protokollen zu einem einzelnen Modul.

**[0096]** Die JetSend-Architektur ist anwendbar, unabhängig vom Transport. JetSend-Vorrichtungen können einander direkt über jeglichen bidirektionalen Transport [36](#) unter Verwendung einer eindeutigen Adressierung adressieren. Es ist notwendig, dass der Transport zuverlässig ist: daher muss für einen unzuverlässigen Transport, wie z. B. UDP, eine weitere Protokollschicht hinzugefügt werden, um Übertragungen über den Transport zuverlässig zu machen (ein weiteres Protokoll, hier genannt zuverlässiges Meldungstransportprotokoll (RMTP; Reliable Message Transport Protocol) [37](#) wird zu diesem Zweck verwendet). Mögliche Transporte umfassen TCP/IP, SPX/IPX, IrDA, IEEE1284, IEEE1394 und andere. Eine Vorrichtung kann einen oder mehrere Transporte implementieren, wodurch ihr ermöglicht wird, mit anderen Vorrichtungen unter Verwendung derselben Transporte zu kommunizieren.

**[0097]** Eine Kommunikation zwischen JetSend-Geräten tritt auf unter Verwendung einer Anzahl von geschichteten Protokollen, wie aus [Fig. 3](#) ersichtlich ist. Diese Schichten sind ähnlich zu den meisten Vernet-

zungssystemen, bei denen jede Protokollschicht mit der entsprechenden Schicht in der anderen Vorrichtung kommuniziert. Die Schichten, die das JetSend-Protokoll aufweisen, sind die Wechselwirkungstaktiken (Interaction Policies) **35**, das Wechselwirkungsprotokoll (Interaction Protocol) **33**, das Sitzungsprotokoll (Session Protocol) **34** und das RMTP-Protokoll **37**.

**[0098]** Die Wechselwirkungstaktiken definieren verschiedene typische Sequenzen von Interaktionen, denen Vorrichtungen folgen. Die Wechselwirkungstaktiken werden als allgemeine Baublöcke verwendet, um spezifischere Informationsaustauschvorgänge zwischen Vorrichtungen zu erreichen. Die nachfolgenden Wechselwirkungstaktiken wurden definiert und werden nachfolgend weiter erörtert:

Job Policy (Auftragstaktik) –	Wie Dokumente zwischen Sendern und Empfängern gesendet werden.
Self Policy (Selbsttaktik) –	Wie globale Informationen über eine Vorrichtung ausgetauscht werden, wie z. B. Etikett, Icon und Passwörter.
Status Policy (Statustaktik) –	Wie ein Status über eine Vorrichtung und über Aufträge gegeben wird.
Address Policy (Adresstaktik) –	Wie Vorrichtungen mit neuen Zieladressen programmiert werden.

**[0099]** Vorrichtungen müssen keine der Taktiken implementieren, aber die meisten Vorrichtungen implementieren die Auftragstaktik. Weitere Taktiken können innerhalb der JetSend-Architektur entwickelt werden, um unterschiedliche Bereiche zu adressieren, wo es wünschenswert ist, ein generisches Verhalten bei der Kommunikation einzurichten.

**[0100]** Das Wechselwirkungsprotokoll verwendet das Sitzungsprotokoll, um Informationen über Oberflächen auszutauschen. Das JetSend-Wechselwirkungsprotokoll enthält Meldungen zum Anfordern und Übertragen von Oberflächenbeschreibungen, Übertragen von Inhaltsdaten für eine Oberfläche und Aktualisieren von Oberflächen. In Verbindung mit E-Material liefert dies den Mechanismus zum Verhandeln von Datentypen zwischen Vorrichtungen.

**[0101]** Das Sitzungsprotokoll definiert Meldungen zum Einrichten von Sitzungen und Kanälen zwischen zwei Vorrichtungen. Eine Sitzung verwaltet die Einrichtung und den Abschluss eines Datentransports zwischen den zwei JetSend-Entitäten. Sie kann Logikkanäle innerhalb dieses Kontexts für Kommunikationen erzeugen. All diese Wechselwirkungen finden über egal welchem Transport statt, der durch die darunter liegende Transportschicht geliefert wird. JSP wird auch in Gateways zwischen JetSend- und Nicht-JetSend-Vorrichtungen verwendet.

**[0102]** Wenn ein Kanal einen unzuverlässigen Transport verwendet, wie z. B. UDP, liefert RMTP den Zuverlässigkeitsdienst für diesen Kanal. RMTP ist ein zuverlässiges, sequenziertes Lieferprotokoll. RMTP ist nicht transportspezifisch und eine einzelne Instanz desselben kann Verbindungen durch alle Transportstapel gleichzeitig beibehalten.

**[0103]** Der Vorrichtungscodex (Device Code) ist der Ausdruck, der für die Logik verwendet wird, die benötigt wird, um die JetSend-Protokolle in eine tatsächliche Vorrichtung zu binden. Der Vorrichtungscodex **31** in [Fig. 3](#) kann in jeglicher Vorrichtung existieren, die in der Lage ist, Informationen entweder zu erzeugen oder zu verbrauchen. Typische Erzeugungsvorrichtungen sind Scanner und PC-Anwendungen. Typische Verbraucher sind Drucker und PC-Betrachter. Eine Vorrichtung muss in der Lage sein, zwischen den vorrichtungsspezifischen Informations-/Daten-Formaten und dem E-Material, das durch die JetSend-Protokolle verwendet wird, umzuwandeln.

**[0104]** Das Verhandeln, das durch jedes JIP ausgeführt wird, ist spezifisch für eine bestimmte Datenklasse. Jede Vorrichtung weist ihre eigenen Präferenzen für Datencodierungen auf, und verhandelt somit für unterschiedliche Attribute und trifft unterschiedliche Auswahlen. Alle Vorrichtungen verwenden die JetSend-Protokolle zur Verhandlung, ein Prozess, bei dem der Sender ein Angebot macht, der Empfänger seine bevorzugte Codierung auswählt und eine Anforderung macht, und der Sender die Anforderung erfüllt. Der Vorrichtungscodex ist die sehr vorrichtungsspezifische Implementierung der Komponente, die zwischen den JetSend-Protokollen und den Aktionen der Vorrichtungen in Wechselwirkung tritt. Die Funktionalität, die der Vorrichtung zum Implementieren überlassen wird, umfasst:

Benutzerschnittstellen-Implementierung und -Steuerung

E-Material-Verhandlung und -Verbrauch (im Sinn des Bereitstellens von E-Material-Codierungen in Reihenfolge der Präferenz und bei Handhaben von eingehendem E-Material)

E-Material-Erzeugung

Fehler-Benachrichtigung und -Erholung

**[0105]** [Fig. 4](#) zeigt eine Kopie der Meldungen, die zwei Vorrichtungen austauschen, wenn sie eine Sitzung einrichten, ein Bild von einer zu der anderen senden und die Sitzung beenden. Die Meldungen sind nicht vollständig, in dem Sinn, dass nur ein paar der wichtigen Meldungsfelder in den Anfangsblöcken spezifiziert wurden (vollständigere Beispiele werden später in der Beschreibung gegeben). Jede Meldung hat einen JSP-Anfangsblock: jede Meldung muss in der gesendeten Reihenfolge empfangen werden, somit die Anforderung nach einem zuverlässigen Datenkanal. Jeder JSP-Anfangsblock enthält den JSP-Meldungstyp (SYN, ASYN, CSYN, CHN etc. – diese Meldungstypen werden nachfolgend weiter erörtert). Die CHN-Meldung enthält den JIP-Anfangsblock und potentiell einige E-Material-Daten. Der JIP-Anfangsblock enthält einen Meldungstyp, der SurfaceMsg, ContentRequestMsg und ContentReplyMsg enthält. Die SurfaceMsg-Meldung enthält das E-Material-Codierungsangebot für eine Oberfläche.

**[0106]** Das JetSend-Sitzungsprotokoll kann mehrere Transportschichten unterstützen, wie z. B. TCP/IP, IPX/SPX, IrDA, Rückschleifung etc. Es liegt an dem Entwickler jeglicher gegebenen Vorrichtung, zu bestimmen, welche Transporte verfügbar gemacht werden. Die Transportimplementierung muss mit dem Transportcode in einer JetSend-Vorrichtung kompatibel sein. Es gibt zwei spezifische Codebereiche, die implementiert sein müssen für eine Kompatibilität mit dem JetSend-Transportcode. Diese sind die Sitzungsschicht des Protokolls und die Bereitstellung eines zuverlässigen Meldungsübermittlungssystems.

**[0107]** Die Transportarchitektur weist das JetSend-Sitzungsprotokoll in Verbindung mit den Transporten auf, die sich die Vorrichtung entscheidet, zu unterstützen. Vorrichtungen, die JetSend unter Verwendung von TCP/IP implementieren, müssen das JetSend-Sitzungsprotokoll und RMTP für unzuverlässige Datagrammdienste implementieren. Die Netzwerkimplementierung von JetSend wird nachfolgend kurz erörtert.

**[0108]** Das JetSend-Sitzungsprotokoll (JSP; JetSend-Session Protocol) ist eine Sitzungsschicht, die ermöglicht, dass Vorrichtungen mehrere gleichzeitige Sitzungen mit anderen JetSend-Vorrichtungen einrichten. Für jede offene Sitzung werden mehrere Datenkanäle für eine Datenübertragung und zum Senden von JIP-Meldungen geöffnet. [Fig. 5](#) stellt eine Sitzung mit mehreren Kanälen zwischen zwei JetSend-Vorrichtungen **51**, **52** dar. Zwei Meldungskanäle **53** und zwei Stromkanäle **54** sind vorgesehen.

**[0109]** Eine JetSend-Sitzung ist eine Übereinkunft zwischen zwei JetSend-Vorrichtungen, die es einer Vorrichtung erlaubt, Kanäle zu der anderen zu öffnen. Damit zwei JetSend-Vorrichtungen kommunizieren können, muss eine Sitzung zwischen denselben eingerichtet werden. Das Einrichten einer Sitzung umfasst das Öffnen einer MESSAGE-(Meldungs-)Verbindung zwischen den Vorrichtungen und das Verwenden des JSP-Sitzungseinrichtungsprotokolls über die Verbindung, um zu verhandeln, dass eine Sitzung einberufen wird. Die Architektur von JSP ermöglicht es jeder JetSend-Vorrichtung, sowohl aktiv als auch passiv Sitzungen einzurichten. Damit eine Einrichtung einer Sitzung auftritt, muss eine Seite beginnen, passiv nach einer Sitzung zu hören, und die andere Seite muss aktiv versuchen, die Sitzung mit der passiven Seite einzurichten. JSP verweigert eine Sitzungseinrichtungsanforderung, wenn eine Sitzung bereits zwischen den zwei Vorrichtungen existiert, außer eine der Vorrichtungen ist ein JetSend-Gateway (ein JetSend-Gateway ist nicht dasselbe wie ein IP-Gateway: ein JetSend-Gateway ist eine Vorrichtung, die zwischen Vorrichtungen übersetzt, die die JetSend-Protokolle „aussprechen“, und jenen, die dies nicht tun (z. B. eine entfernte Faxmaschine). Wenn das JIP, das das Ziel der Anforderung ist, nicht nach Sitzungseinrichtungsanforderungen auf dem Netzwerk hört, durch das die Anforderung angekommen ist, lehnt das JSP die Anforderung ab. Diese Situation kann existieren, wenn ein Benutzer die Vorrichtung gesperrt hat.

**[0110]** Die Architektur des JSP ermöglicht, dass jegliche JetSend-Vorrichtung gleichzeitige Sitzungen mit mehreren anderen JetSend-Vorrichtungen einrichtet. Jegliche JetSend-Vorrichtung kann – gleichzeitig – sowohl Sitzungen haben, die sie passiv erzeugt hat, als auch Sitzungen, die sie aktiv erzeugt hat.

**[0111]** Jede Seite einer Sitzung kann den Abschluss der Sitzung initiieren. Die anfordernde Seite muss nicht warten, sollte aber, wenn sie kann, bis sie keine gepufferten Daten mehr hat, bevor sie den Abschluss anfordert. Wenn ein Abschluss angefordert wird, stoppen beide Seiten das Senden von Daten durch alle offenen Kanäle und schließen dieselben sofort.

**[0112]** Das JIP kann zwei Typen von Kanälen durch eine JSP-Sitzung öffnen, STREAM-(Strom-) und MES-

SAGE-(Meldungs-)Kanäle. Beide Kanaltypen bieten einen zuverlässig geordneten Lieferdienst. STREAM-Kanäle benötigen mehr Mehraufwand, um beibehalten zu werden als MESSAGE-Kanäle, da sie unter Verwendung von verbindungsorientierten Transportprotokollen implementiert werden, wie z. B. TCP. MESSAGE-Kanäle werden unter Verwendung von RMTP implementiert. STREAM-Kanäle bieten ein besseres Verhalten als MESSAGE-Kanäle für die Lieferung großer Datenmengen. JIP muss MESSAGE-Kanäle unterstützen: das JIP kann jedoch einen beliebigen Kanaltyp für alles wählen, das es ausführen möchte. Einige Implementierungen können nur eine sehr begrenzte Anzahl von STREAM-Kanälen unterstützen oder können STREAM-Kanäle gar nicht unterstützen.

**[0113]** Eine JetSend-Sitzung allein überträgt oder empfängt keine JIP-Daten. Das JIP muss anfordern, dass das JSP Kanäle anstatt seiner öffnet zum Kommunizieren mit seinem JIP-Partner durch die Sitzung. Ein Paar von JIPs kann eine offene Sitzung zwischen denselben haben, ohne jegliche Kanäle während der Sitzung offen zu haben. Das JIP an einem Ende der Sitzung kann das Öffnen und Schließen von Kanälen initiieren. Beide JIPs können Kanäle durch die Sitzung nach Wunsch öffnen und schließen. Es gibt keine architekturellen Grenzen für die Anzahl oder die Mischung aus STREAM- und MESSAGE-Kanälen innerhalb einer Sitzung. Wenn eine bestimmte JetSend-Implementierung nicht in der Lage ist, eine Anforderung zum Öffnen eines neuen Kanals unterzubringen, antwortet sie auf die Anforderung und zeigt an, dass sie keine Kanäle dieses Typs mehr öffnen kann.

**[0114]** RMTP ist ein zuverlässiges, sequenziertes Lieferprotokoll. Es wird implementiert durch Hinzufügen eines Minimal-Zuverlässigkeits-Mechanismus und einer Sequenznummerierung zu einem Datagrammtransportprotokoll, wie z. B. UDP oder IPX. RMTP verspricht nicht den selben Pegel an Verhalten oder Zuverlässigkeit, den STREAM-Protokolle wie TCP und SPX versprechen. Abhängig von den Netzwerken, die eine bestimmte Implementierung von JetSend unterstützt, können RMTP-Implementierungen über einen oder mehrere der nachfolgenden unzuverlässigen Transporte laufen: UDP, IPX etc. Seine Funktion ist das Liefern eines Voll-Duplex-Kommunikationskanals zwischen den zwei Ports jeder Transportverbindung, und es versucht, zuverlässig alle Benutzermeldungen zu liefern und berichtet dem Benutzer einen Ausfall, wenn es eine Meldung nicht liefern kann. RMTP erweitert den Datagrammdienst von UDP (und/oder anderer Datagrammprotokolle), um eine zuverlässige Lieferung einzuschließen. Es versucht, alle beschädigten und doppelten Meldungen zu erfassen und zu verwerfen und liefert eine sequenzierte Lieferung von Benutzermeldungen.

**[0115]** RMTP ist ein Transportprotokoll. Es erweitert die Ende-zu-Ende-Dienste, die ein Datagramm-Transportprotokoll, wie UDP oder IPX liefert, um Zuverlässigkeit und geordnete Lieferung einzuschließen. RMTP akzeptiert Daten von JSP in Meldungseinheiten. RMTP packt jede übermittelte JSP-Meldung für einen Transport als ein einzelnes RMTP-Segment und leitet das Segment durch den Datagrammtransport für eine Lieferung weiter. Um eine JSP-Meldung in ein RMTP-Segment zu packen, hängt das RMTP seinen eigenen Anfangsblock vor der JSP-Meldung an. Jedes RMTP-Segment ist als ein RMTP-Anfangsblock und ein oder mehrere Datenoktette gepackt. RMTP versucht nicht, eine große JSP-Meldung in kleinere Segmente zu fragmentieren und die Meldung an dem Empfangsende neu anzuordnen. Dies unterscheidet sich von einem Byte-Strom-Protokoll, wie z. B. TCP, das die Übertragung einer unbestimmten Länge eines Datenstroms zwischen Ports unterstützt, und Daten puffert, bis sie durch den Empfänger angefordert werden.

**[0116]** RMTP muss einen unzuverlässigen Transport zuverlässig machen. RMTP ist nicht wesentlich für die Architektur von JetSend, und die Erfindung kann ohne Zurückgreifen auf RMTP ausgeführt werden – RMTP ist jedoch vorteilhaft beim Erweitern des Bereichs von verfügbaren Transporten.

**[0117]** Wie oben angezeigt ist, ist ein JetSend-Gateway eine Vorrichtung, die mit JetSend-Vorrichtungen an einem Ende und Vorrichtungen eines anderen Typs an dem Anderen kommunizieren kann. Gateways übersetzen zwischen dem JetSend-Protokoll und Nicht-JetSend-Protokollen. Dies umfasst das Umwandeln von E-Mail-Datentypen in Datentypen, die für ein gegebenes Gateway geeignet sind. Ein Gateway hat seine eigene JetSend-Adresse, die eine JetSend-Sendevorrichtung verwendet, um eine Sitzung mit demselben einzurichten. Der Sender muss ferner die Endzielortadresse zu dem Gateway weiterleiten. Dies ist bekannt als die Alias-Adresse und ist spezifisch für einen gegebenen Typ von Gateway. Zum Beispiel erfordert ein Fax-Gateway, dass eine Telefonnummer als die Alias-Adresse weitergeleitet wird. Die Adresse von Gateways kann in Sendevorrichtungen vorkonfiguriert sein. Gateways können ferner ihr Vorhandensein zu Vorrichtungen auf ihrem lokalen Teilnetz rundsenden.

**[0118]** Spezifische Aspekte der JetSend-Architektur werden nun nachfolgend detaillierter in dem Kontext der Übersicht beschrieben, die oben ausgeführt ist. Die Erörterung führt von den unteren Protokollschichten aufwärts. Zuerst wird kurz RMTP erörtert, dann das JetSend-Sitzungsprotokoll, JetSend-Wechselwirkungsproto-

koll, JetSend-Wechselwirkungstaktiken und E-Material.

### Zuverlässiges Meldungstransportprotokoll

**[0119]** Das zuverlässige Meldungstransportprotokoll (RMTP; Reliable Message Transport Protocol) ist nicht grundlegend für JetSend, aber der Bedarf nach einem zuverlässigen Transport ist es. RMTP ist daher vorteilhaft als eine leichte Protokollschicht, um einen existierenden unzuverlässigen Transport zuverlässig zu machen.

**[0120]** RMTP ist ein verbindungsorientiertes Protokoll, bei dem jede Verbindung als ein Vollduplexkommunikationskanal zwischen zwei Prozessen wirkt. Segmente von einem Sender werden zu einem Port an dem Zielort-Host geleitet. RMTP kann unter Verwendung jeglichen Transportprotokolls implementiert sein. Es ist für eine Implementierung unter Verwendung der Dienste vorgesehen, die durch einen Datagrammtransport geliefert werden, wie z. B. UDP oder IPX. Es fügt Wert zu Datagrammtransporten hinzu durch Implementieren von Zuverlässigkeit und geordneter Lieferung.

**[0121]** RMTP verwendet den allgemeinen Internet-3-Wege-Handshake, um eine Verbindung einzurichten. RMTP verwendet dieses Verbindungs-Handshake-Verfahren, um maximale Segmentgrößen zu verhandeln, Sequenzzahlen zu synchronisieren und Verbindungsidentifizierer auszutauschen. Es enthält ferner eine Prozedur für eine Verhandlung von RMTP-Versionen zwischen Hosts.

**[0122]** Auf der RMTP-Ebene werden ausgehende Segmente, wenn sie erzeugt werden, als Eingabe zu der Datagrammschicht in Warteschlange gesetzt. Jedes Segment wird durch das Sende-RMTP gehalten, bis es durch den fremden Host bestätigt ist.

**[0123]** Eingehende Segmente werden als Eingabe zu dem Benutzerprozess in Warteschlange gestellt. Segmente werden bestätigt, wenn sie als Eingabe zu dem Benutzerprozess in Warteschlange gestellt wurden, oder in bestimmten Fällen, wenn sie durch die Empfangsseite akzeptiert wurden.

**[0124]** Das Empfangsende jeder Verbindung spezifiziert die maximale Segmentgröße, die sie akzeptiert. Jeder Versuch durch den Sender, ein größeres Segment zu übertragen, ist ein Fehler. Wenn RMTP bestimmt, dass ein Puffer, der mit einer Sende-Anforderung übermittelt wurde, die maximale Segmentgröße überschreitet, die an der Verbindung erlaubt ist, sendet RMTP einen Fehler zu dem Benutzer zurück. Zusätzlich dazu bricht RMTP eine Verbindung ab, wenn ein eingehendes Segment mehr Daten als die maximal akzeptable Segmentgröße enthält. Es wird kein Versuch unternommen, sich von diesem Fehlerzustand zu erholen oder diesen anderweitig zu überwinden.

**[0125]** Eine zuverlässige Meldungsübermittlung wird durch eine Anzahl von Mechanismen wie folgt implementiert:

Segment-Sequenznummern – jedes Segment, das Daten transportiert, hat eine Sequenznummer, die es eindeutig unter allen anderen Segmenten in der selben Verbindung identifiziert. Die Anfangssequenznummer wird ausgewählt, wenn die Verbindung geöffnet wird. Jedes Mal, wenn ein Segment übermittelt wird, das Daten enthält, wird die Sequenznummer inkrementiert.

**[0126]** Prüfsummen – Prüfsummen sind in Paketen umfasst, so dass Fehler, die während der Übertragung angetroffen werden, erfasst werden können. Für Internetprotokolle, wie z. B. UDP, ist dies besonders wichtig, da Pakete möglicherweise drahtlose Netzwerke überqueren müssen, wie z. B. das Paketfunknetzwerk (Packet Radio Network) und das Atlantik-Satellitennetzwerk (Atlantic Satellite Network), wo Pakete verfälscht werden können.

**[0127]** Positive Bestätigung des Segments – RMTP nimmt an, dass es nur einen unzuverlässigen Datagrammdienst zum Liefern von Segmenten hat. Um die Lieferung von Segmenten in dieser Umgebung zu garantieren, verwendet RMTP eine positive Bestätigung und Neuübertragung von Segmenten. Jedes Segment, das eine eindeutige Sequenznummer enthält, wird bestätigt, wenn es korrekt empfangen und durch den Zielthost akzeptiert wird. Segmente, die nur eine Bestätigung enthalten, werden nicht bestätigt. Beschädigte Segmente werden verworfen und werden nicht bestätigt. Segmente werden neu übertragen, wenn keine rechtzeitige Bestätigung des Segments durch die Zielhosts erfolgt.

**[0128]** Neuübertragungszeitbegrenzung – Segmente können bei der Übertragung aus zwei Gründen verloren gehen: sie können verloren gehen oder aufgrund der Wirkungen der verlustbehafteten Übertragungsmedien

beschädigt werden; oder sie können durch das Empfangs-RMTP verworfen werden. Die Positiv-Bestätigungstaktik erfordert, dass der Empfänger ein Segment nur bestätigt, wenn das Segment korrekt empfangen und akzeptiert wurde. Um fehlende Segmente zu erfassen, muss das Sende-RMTP einen Neuübertragungszeitgeber für jedes übertragene Segment verwenden. Der Zeitgeber wird auf einen Wert eingestellt, der annähernd die Übertragungszeit eines Segments in dem Netzwerk ist. Wenn eine Bestätigung für ein Segment empfangen wird, wird der Zeitgeber für dieses Segment abgebrochen. Wenn der Zeitgeber abläuft, bevor eine Bestätigung für ein Segment empfangen wird, wird dieses Segment neu übertragen und der Zeitgeber wird neu gestartet.

#### JetSend-Sitzungsprotokoll

**[0129]** Das JetSend-Sitzungsprotokoll (JSP; JetSend Session Protocol) ist ein sitzungsorientiertes Protokoll, innerhalb dem zwei Prozesse mehrere Vollduplexkommunikationskanäle zwischen denselben öffnen können. Meldungen von einem Sender werden zu einem JSP-Port an dem Zielorthost gerichtet. JSP kann implementiert sein, um mehrere gleichzeitige Sitzungen zu unterstützen, obwohl dies keine Anforderung ist.

**[0130]** Die JSP-Sitzungsverwaltung ist über einen spezifischen Transport implementiert. Der Transport ist verantwortlich für das Liefern der Funktionalität, die schließlich die JSP-Meldungen sendet und empfängt. Der Transport muss einen Mechanismus einer zuverlässigen, geordneten Lieferung für einen ordnungsgemäßen Meldungs austausch unterstützen. Über ein IP-Netzwerk z. B. würde das JSP das Reliable Message Transport Protocol (RMTP) verwenden, um diese benötigten MESSAGE-Kanäle zu liefern.

**[0131]** Zusätzlich zu den erforderlichen MESSAGE-Kanälen kann die JSP-Implementierung auch STREAM-Kanäle liefern. Diese können unter Verwendung eines zuverlässigen, geordneten Liefertransportprotokolls implementiert sein, wie z. B. TCP, TIP oder SPX. Es wird jedoch darauf hingewiesen, dass die Unterstützung von STREAM-Kanälen optional ist und kein erforderlicher Teil des Protokolls ist.

**[0132]** Jede JSP-Sitzung schreitet während ihrer Lebensdauer durch eine Reihe von Zuständen fort. Die Zustände sind in [Fig. 6](#) gezeigt und werden nachfolgend individuell beschrieben. In [Fig. 6](#) stellen die Ellipsen die Zustände der JSP-Endzustandsmaschine dar und die Bögen stellen die Änderungen bei dem Zustand dar. Jeder Bogen ist mit dem Ereignis kommentiert, das die Zustandsänderung und die resultierende Ausgabe verursacht.

#### CLOSED (GESCHLOSSEN)

**[0133]** Der Zustand CLOSED **61** existiert, wenn keine Sitzung existiert (CLOSED = GESCHLOSSEN).

#### LISTEN (HÖREN)

**[0134]** In den Zustand LISTEN **62** wird eingetreten, nachdem eine passive offene Anforderung **67a** verarbeitet ist. Das JSP wartet auf eine aktive Anforderung, um eine Transportverbindung von einem entfernten Host einzurichten.

#### CONNECTING (VERBINDEN)

**[0135]** In den Zustand CONNECTING **63** wird eingetreten, nachdem eine aktive offene Anforderung **67b** verarbeitet ist. Das JSP versucht, eine Transportverbindung zu einem entfernten Host aktiv zu öffnen.

#### SYN-WAIT (SYN-WARTEN)

**[0136]** In den Zustand SYN-WAIT **64** wird eingetreten nach dem passiven Öffnen einer Transportverbindung. Das JSP wartet dann in dem Zustand SYN-WAIT, dass eine SYN-Meldung von dem entfernten Host ankommt.

#### SYN-SENT (SYN-GESENDET)

**[0137]** In den Zustand SYN-SENT **65** wird eingetreten nach dem aktiven Öffnen einer Transportverbindung. Eine SYN-Meldung **68b** wird zu dem entfernten Host gesendet. Das JSP wartet dann in dem Zustand SYN-SENT **65** auf die Bestätigung (ASYN-Meldung **69b**) seiner Anforderung Öffnen.

## OPEN (ÖFFNEN)

**[0138]** Der Zustand OPEN **66** kann entweder aus dem Zustand SYN-WAIT **64** oder dem Zustand SYN-Sent **65** erreicht werden. Er wird aus dem Zustand SYN\_WAIT **64** erreicht, wenn eine SYN-Meldung von dem entfernten Host empfangen **69a** wird. Er wird aus dem Zustand SYN\_SENT **65** erreicht, wenn eine ASYN für den SYN von dem entfernten Host empfangen wird **69b**. In dem Zustand OPEN **66** können Kanäle geöffnet werden und Daten können zwischen den zwei Parteien der Sitzung gesendet werden.

**[0139]** Sitzungen werden geöffnet durch Ausgeben der Anforderung OPEN, die entweder aktiv oder passiv sein kann. Eine passive Anforderung OPEN **67a** setzt das JSP in den Zustand LISTEN **62**, während dem es passiv nach einer Anforderung hört, um eine Sitzung von einem entfernten Host zu öffnen. Die aktive Anforderung OPEN **67b** versucht, eine Sitzung mit einem spezifizierten JSP-Port an einem entfernten Host einzurichten. Im Wesentlichen ist ein JSP passiv, wenn es nach einer Verbindung hört und ist aktiv, wenn es die Öffnung einer Verbindung mit einem anderen JSP initiiert.

**[0140]** In spezifischen Fällen kann die Implementierung des JSP derart sein, um nur passive Verbindungen, nur aktive Verbindungen oder beide Typen von Verbindungen zu unterstützen. Diese Implementierungsentscheidung beeinflusst jedoch die verfügbare Funktionalität der oberen Schichten des JetSend-Protokolls.

**[0141]** JSP erfordert, dass die Seite, die die Transportverbindung aktiv geöffnet hat, auch die Sitzung aktiv öffnet. Die Sitzung-Öffnen-Verhandlung existiert, um Sitzungsidentifizierer auszutauschen und zu verhindern, dass Sitzungen erzeugt werden, wenn eine Nicht-JetSend-Netzwerkentität fälschlicherweise eine Transportverbindung mit einem JetSend-Gerät öffnet. [Fig. 7](#) zeigt ein schematisches Beispiel des Startens einer Sitzung.

**[0142]** Zuerst wird eine passive Sitzungsöffnung beschrieben.

**[0143]** Das JetSend-Wechselwirkungsprotokoll, das sich selbst mit einer zugeordneten JSP-Portnummer identifiziert (auch bezeichnet als JetSend-Vorrichtungsnummer, JetSend-Teiladresse und JIP-Handle), führt eine Anforderung zu dem JSP aus, um nach einer Sitzung zu hören (eine passive Öffnen-Anforderung). Das JSP antwortet dadurch, dass es seinen eigenen Hören-Ruf zu dem zugrunde liegenden MESSAGE-Transportdienst aussendet. Das JSP, das sich an die JSP-Portnummer erinnert, die zu diesem Hören-Ruf gehört, wartet auf den zugrunde liegenden Transport, um benachrichtigende Passive-Verbindung-Antworten zu geben. Jede Passive-Verbindung-Antwort (Passive Connect) sollte eine eindeutige Verbindungs-Handhabung haben (connHandle), die derselben zugeordnet ist. Ferner sollte das JSP für jede resultierende Passive-Verbindung einen neuen Sitzungssteuerblock (SCB; session control block) erzeugen der die JSP-Portnummer und connHandle darin speichert.

**[0144]** Mit der eingerichteten Verbindung und dem aufgebauten SCB wartet das JSP dann darauf, dass die aktive Seite ihm eine SYN-Meldung sendet. Wenn die erste ankommende Meldung keine SYN-Meldung ist oder keine SYN-Meldung innerhalb einer spezifizierten Zeitperiode ankommt (vorzugsweise zwei Minuten), dann schließt das JSP die Verbindung und zerstört den SCB. Wenn die erste Meldung eine SYN-Meldung ist und sie innerhalb der Warteperiode ankommt, sendet das JSP ein ASYN oder ein NSYN über die Verbindung zu dem entfernten Host. Das JSP sendet ein ASYN, um die Sitzung zu akzeptieren, wenn die SYN-Meldung die selbe JSP-Portnummer spezifiziert hat wie die, die in dem SCB gespeichert ist. Es sendet ein NSYN, um die Sitzung anderweitig abzulehnen.

**[0145]** Wenn das JSP die Sitzung abgelehnt hat (NSYN), wartet es, bis die NSYN erfolgreich zu dem entfernten Host gesendet wurde und schließt dann die Verbindung und zerstört den SCB. Wenn das JSP mehrere JIPs unterstützt und mehr als ein JIP anfordert, dass das JSP nach einer Sitzung auf dem selben Netzwerk hört, sollte das JSP nur einmal ein LISTEN auf diesem Netzwerk aufrufen. Es behält eine Liste der JSP-Portnummern bei, nach denen es auf diesem Netzwerk hört.

**[0146]** Wenn eine Transportverbindung aus dem HÖREN (LISTEN) erzeugt wird und das SYN empfangen wird, lehnt das JSP die Sitzungsanforderung ab, wenn die JSP-Portnummer bei dem SYN nicht mit jeglichen derer in der Liste übereinstimmt.

**[0147]** Wenn das JSP die Sitzung akzeptiert hat (ASYN), lässt es die Verbindung offen, wartet 4 Mal die Umlaufzeit (RoundTrip Time) und sendet dann eine Sitzungshandhabung zu dem JIP zurück. (Die Umlaufzeit ist die Zeitperiode, die beginnt, wenn das SYN gesendet wird und endet, wenn das entsprechende ASYN emp-

fangen wurde). Diese Wartezeit soll sicherstellen, dass jegliche Neuverhandlung der JSP-Protokollversion gehandhabt wird, bevor das JIP über die neue Sitzungshandhabung benachrichtigt wird. Ein Mechanismus wird innerhalb des JSP bereitgestellt, um eine Verhandlung der Version während des Handshaking-Verfahrens zu ermöglichen, um sicherzustellen, dass eine Kommunikation unter den selben Versionen von JSP auftritt.

**[0148]** Nun wird eine aktive Sitzungsöffnung beschrieben.

**[0149]** Das JIP, das sich selbst mit einer JSP-Portnummer und sein Ziel mit einer Adresszeichenfolge identifiziert, führt eine Anforderung zu dem JSP aus, um eine Sitzung zu öffnen (eine aktive Öffnen-Anforderung).

**[0150]** Das JSP antwortet dadurch, dass es seinen eigenen aktiven Ruf (Active Call) zu dem zugrunde liegenden MESSAGE-Dienst ausführt. Das JSP, das sich an die JSP-Portnummer erinnert, die den aktiven Ruf begleitet, wartet darauf, dass der aktive Ruf eine Aktive-Verbindung-Antwort (Active Connect) aus dem zugrunde liegenden Transport erzeugt. Jeder aktiven Verbindung sollte eine eindeutige Verbindungshandhabung zugeordnet sein. Für die resultierende aktive Verbindung erzeugt das JSP einen neuen Sitzungssteuerblock (SCB) und speichert die JSP-Portnummer und die entsprechende Verbindungshandhabung in demselben.

**[0151]** Wenn die Verbindung eingerichtet und der SCB aufgebaut ist, sendet das JSP dann eine SYN-Meldung zu der passiven Seite. Die SYN-Meldung enthält die passive JSP-Portnummer, die das JSP von der JMN-Formadresse des Ziels hergeleitet hat. Das JSP misst den Zeitbetrag, den der zugrunde liegende MESSAGE-Transport benötigt, um zu bestätigen, dass er erfolgreich die SYN-Meldung gesendet hat (Umlaufzeit). Es wartet dann, dass die passive Seite demselben eine Meldung ASYN oder NSYN sendet. Wenn die erste ankommende Meldung keine ASYN- oder NSYN-Meldung ist, schließt das JSP die Verbindung und zerstört den SCB. Wenn die ASYN oder NSYN nicht innerhalb der größeren Spanne von einer Minute oder 4 Mal der Umlaufzeit ankommt, schließt das JSP die Verbindung und zerstört den SCB.

**[0152]** Das Warten für 4 Mal die Umlaufzeit stellt sicher, dass die antwortende Seite ausreichend Zeit hat, über eine langsame Verknüpfung zu antworten. Das Warten für eine Minute stellt sicher, dass die antwortende Seite ausreichend Zeit hat, zu antworten, wenn es eine relativ langsame Maschine ist, die über eine schnelle Verknüpfung kommuniziert.

**[0153]** Wenn der entfernte Host die Sitzung akzeptiert hat (ASYN), lässt das JSP die Verbindung offen und bringt eine Sitzungshandhabung zu dem JIP zurück. Wenn der entfernte Host die Sitzung abgelehnt hat (NSYN), schließt das JSP die Verbindung und zerstört den SCB.

**[0154]** Für jegliche eine Sitzung behält das JSP nicht mehr als eine Transportmeldungsverbindung bei, aber sie kann mehr als eine Transportstromverbindung (TCP, SPX etc.) mit dem entfernten Host beibehalten. Das JSP multiplext alle zusätzlichen MESSAGE-Kanäle über die einzelne Meldungsverbindung. Alle zusätzlichen STREAM-Kanäle entsprechen direkt zusätzlichen Transportstromverbindungen.

**[0155]** Nachdem die Sitzung eingerichtet ist, existiert bereits eine MESSAGE-Verbindung. Sie wird während des Prozesses des Einrichtens der Sitzung erzeugt. Wenn das JIP eine Anforderung zu dem JSP erstellt, einen MESSAGE-Kanal zu öffnen, verhandelt das JSP den Kanal über diese existierende MESSAGE-Verbindung ins Leben.

**[0156]** Wenn das JIP eine Anforderung zum passiven Hören nach STREAM-Kanälen trifft, muss das JSP zuerst ein LISTEN (HÖREN) an seinem lokalen, bekannten Stromtransportport abschicken. Nachfolgend, wenn ein entferntes JIP anfordert, einen STREAM-Kanal zu öffnen, erzeugt das JSP den Kanal über eine neu erzeugte Transportverbindung.

**[0157]** Wenn das JIP eine Anforderung ausführt, aktiv einen STREAM-Kanal zu öffnen, öffnet das JSP eine neue Stromtransportverbindung zu dem bekannten Stromtransportport an dem entfernten Host. Wenn der entfernte Host nach eine Verbindung an diesem Port hört, wird die Verbindung unter Verwendung einer neuen Transportverbindung erzeugt. Wenn der Kanal geschlossen ist, schließt das JSP die Transportstromverbindung.

**[0158]** Die physische MESSAGE-Verbindung ist nie während der Lebensdauer der Sitzung geschlossen.

**[0159]** Wenn das JSP eine Sitzung schließt, muss es alle Transportverbindungen schließen, die es innerhalb der Sitzung offen hat und alle Transport-Hörvorgänge abbrechen, die aktiv sind. Das JSP schließt eine Sitzung

in drei Situationen: es empfängt eine Schließen-Anforderung von einer Schicht oben; es empfängt eine RST-Meldung von dem entfernten Host; oder seine Transport-MESSAGE-Verbindung schlägt fehl.

**[0160]** Wenn dem JSP befohlen wurde, die Sitzung zu schließen, sendet es eine RST-Meldung über seine MESSAGE-Verbindung, schließt alle seine Transportverbindungen, bricht alle seine Hörvorgänge ab und zerstört den SCB. Diese einfache Sitzungs-Schließ-Einrichtung erfordert, dass die höheren Schichten in dem Jet-Send-Protokoll bestimmt haben, dass alle Daten zuverlässig geliefert wurden, bevor ein Schließen der Sitzung angefordert wurde.

**[0161]** Wenn das JSP eine RST-Meldung über seine MESSAGE-Verbindung empfangen hat, schließt es alle seine Transportverbindungen, bricht alle seine Hörvorgänge ab und zerstört den SCB.

**[0162]** Wenn die MESSAGE-Verbindung des JSP fehlschlägt, schließt es alle seine Transportverbindungen, bricht alle seine Hörvorgänge ab und zerstört den SCB. Es wird darauf hingewiesen, dass das JSP die Sitzung nicht schließt, wenn seine STREAM-Transportverbindung fehlschlägt. Der Grund dafür ist, dass eine MESSAGE-Verbindung erforderlich ist, während eine STREAM-Verbindung nur eine optionale Möglichkeit zum Senden/Empfangen von Daten ist.

**[0163]** Wenn eine Seite einer Sitzung zusammenbricht, kann die Sitzung gelassen werden, wobei die andere Seite noch aktiv ist. Diese Situation wird eine halboffene Sitzung genannt.

**[0164]** Um sicherzustellen, dass halboffene Sitzungen/Verbindungen nicht offen bleiben und Ressourcen verbrauchen, sendet das JSP Lebenserhaltungsmeldungen während langer Inaktivitätsperioden in der Sitzung. Wenn keine Aktivität auf einem Kanal innerhalb der Sitzung für mehr als zehn Minuten stattfand, sendet das JSP eine Lebenserhaltungsmeldung (NULL-Meldung) zu der anderen Seite über die MESSAGE-Verbindung. Wenn die MESSAGE-Verbindung fehlschlägt (das JSP empfängt eine Benachrichtigung Disconnect (Abtrennen) von dem zugrunde liegenden Transport), wird die Sitzung abgebrochen. Um zu verhindern, dass sich Lebenserhaltungsmeldungen in dem MESSAGE-Transport anhäufen, sendet das JSP keine weitere Lebenserhaltungsmeldung, bis es eine Benachrichtigung von dem zugrunde liegenden Transport empfängt, dass die letzte gesendet wurde.

**[0165]** Das JSP liefert die Abstraktion von Kanälen, die durch eine Sitzung laufen. Wenn das Sitzungsprotokoll in dem Zustand OPEN ist, kann das JSP Kanäle verfügbarer Typen öffnen. Jeder JSP-Kanal wirkt als ein Vollduplexkommunikationskanal zwischen zwei JetSend-Vorrichtungen. Meldungen von einem Sender werden zu einem Kanalendpunkt an dem Zielorthost gerichtet.

**[0166]** Kanäle werden geöffnet durch Ausgeben der Anforderung ÖFFNEN, die entweder aktiv oder passiv sein kann. Eine passive Anforderung ÖFFNEN verursacht, dass das JSP passiv nach einer Anforderung zum Öffnen eines Kanals von einem entfernten Host hört. Die aktive Anforderung ÖFFNEN sendet eine Anforderung zum Öffnen eines Kanals zu einem entfernten Host.

**[0167]** Bevor das JSP versuchen kann, aktiv einen Kanal zu öffnen, muss es wissen, welche Kanäle, falls überhaupt, zum Öffnen verfügbar sind. Die Meldung CEXTLSN wird durch das JSP zu diesem Zweck verwendet. Sobald eine Sitzung durch das oben erwähnte Handshaking bzw. den Quittungsaustausch eingerichtet wurde, sollten die JSPs, die an der Sitzung beteiligt sind, jeweils eine CEXTLSN-Meldung aussenden, die die Kanaltypen enthält, die es zum Öffnen verfügbar hat. Ein JSP versucht vielleicht nicht, aktiv einen Kanal mit einem anderen JSP zu öffnen, bis diese Meldung empfangen wurde.

**[0168]** Sobald die CEXTLSN empfangen wurde, kann das JSP versuchen, einen Kanal zu öffnen. Die Kanalöffnungsanforderung wird in einer SYN-Meldung getragen, wobei das CHN-Bit ebenfalls gesetzt ist (CSYN-Meldung). Sie umfasst die Kanalnummer von der Sendeseite. Die Antwort auf die CSYN ist eine SYN-Meldung, bei der das CHN- und ACK-Bit ebenfalls gesetzt sind (CASYN-Meldung). Sie enthält die Kanalnummer von der Antwortseite. Die passive Seite einer Kanalöffnung empfängt eine CSYN-Meldung von der aktiven Seite. Die passive Seite zeichnet die Kanalnummer der aktiven Seite in einem Kanalsteuerblock (CCB; channel control block) auf. Dann sendet sie eine CASYN zurück zu der aktiven Seite. Sie umfasst ihre eigene Kanalnummer (eine Handhabung zu dem CCB) in der CASYN. Von diesem Punkt an enthalten alle Meldungen, die die passive Seite zu der aktiven Seite sendet, die Kanalnummer der aktiven Seite in dem Zielortkanalfeld (Destination Channel Field).

**[0169]** Die aktive Seite einer Kanalöffnung erzeugt einen CCB und sendet eine CSYN-Meldung zu der passi-

ven Seite. Sie umfasst ihre eigene Kanalnummer (eine Handhabung zu dem CCB) in der CSYN. Sie empfängt dann eine CASYN zurück von der passiven Seite. Die aktive Seite zeichnet die Kanalnummer der passiven Seite in ihren CCB auf. Von diesem Punkt an enthalten alle Meldungen, die die aktive Seite zu der passiven Seite sendet, die Kanalnummer der passiven Seite in dem Zielortkanalfeld.

**[0170]** Die passive Seite kann auf eine CSYN mit einer SYN-Meldung antworten, bei der das CHN- und NAK-Bit ebenfalls gesetzt sind (CNSYN-Meldung). Dies ist eine Ablehnung der Kanalöffnung. Kein Kanal wird erzeugt.

**[0171]** Das Schließen eines Kanals kann initiiert werden durch eine Anforderung Schließen (Close) von dem JIP oder durch den Empfang einer CRST-Meldung von dem anderen Ende des Kanals. In dem Fall der Anforderung Schließen sendet das JSP eine CRST-Meldung zu der anderen Seite des Kanals und schließt dann sein Ende des Kanals. Das JSP verwirft Meldungen, die zu einem geschlossenen Kanal gerichtet sind. Diese einfache Kanalschließenrichtung erfordert, dass Benutzer bestimmen, dass alle Daten geliefert wurden, bevor ein Schließen des Kanals angefordert wird.

**[0172]** Ein visuelles Beispiel des Öffnens und Schließens eines Kanals ist in [Fig. 8](#) gezeigt.

**[0173]** Daten fließen durch einen JSP-Kanal in der Form von CHN-Meldungen. Alle CHN-Meldungen haben das Letzte-Fragment-Bit gesetzt (LF; Last Fragment), außer Nicht-Anschluss-Fragmente. Auf der JSP-Ebene werden ausgehende Meldungen, wenn sie erzeugt werden, als Eingabe zu der Transportschicht in Warteschlange gesetzt. Jede Meldung wird zuverlässig durch den zugrunde liegenden Transport geliefert. Wenn sie nicht geliefert werden kann, erzeugt der Transport ein Abtrennungseignis an dieser Verbindung und folglich schließt das JSP alle Kanäle, die über diese Verbindung laufen. Zusätzlich zum Schließen aller offenen Kanäle schließt das JSP auch die Sitzung.

**[0174]** Damit Informationen zwischen zwei Vorrichtungen geleitet werden, müssen bei dieser Implementierung alle Daten auf einer 4-Byte-Grenze gesendet und empfangen werden. Diese Datenblöcke werden im Rest dieses Dokuments als Task Block Control Buffers (TBCB; Aufgabenblocksteuerpuffer) identifiziert. Eine NULL-Auffüllung muss in den Fällen hinzugefügt werden, in denen der TBCB nicht durch Vier teilbar ist. Wenn das JSP Daten von einem TBCB zu einem entfernten Partner sendet, sollte es die Auffüllung umfassen.

**[0175]** Wenn das JSP eine Verbindung öffnet, ist eines der Informationselemente, die es von dem zugrunde liegenden Transport empfangen muss, die Maximalmeldungsgröße für diese Verbindung. Diese `max_transport_buff` ist die Größe der größten Meldung, die der Transport zum Senden von dem JSP auf dieser Verbindung akzeptiert. Wenn das JSP von dem JIP eine Anforderung empfängt zum Senden eines Datenpuffers, der größer ist als die Maximalmeldungsgröße, fragmentiert das JSP die Daten in mehrere JSP-Meldungen. Es hängt seinen Anfangsblock hinten an jede der mehreren Meldungen an, stellt das LF-Bit auf 0 an allen abgesehen von der Letzten, die sie auf 1 setzt, und sendet dieselben dann zu dem zugrunde liegenden Transport zur Übertragung. Die Empfangsseite ordnet die Meldungen in einen Datenblock zur Lieferung zu dem JIP neu an.

**[0176]** Als Teil des JSP-Protokolls tauschen CSYN- und CASYN-Pakete Informationen über die Maximalpuffergröße aus, die jedes JIP verwalten kann (`max_buf_size`; Maximalpuffergröße). Das JSP muss dann die entfernte `max_buf_size` (`remote_max_buf_size`) berücksichtigen vor der Fragmentierung des Puffers, insbesondere da die zwei Größen unterschiedlich sein können. Das JSP sollte nie einen Puffer zu der entfernten Vorrichtung senden, der größer ist als die `remote_max_buf_size`. In einem Fall, in dem die `max_buf_size` an dem lokalen JIP größer ist als die `remote_max_buf_size`, empfängt das JSP einen Puffer der `max_buf_size` (`this_buf_size`; diese Puffergröße), zum Senden zu einer entfernten Vorrichtung. Das JSP muss Puffer (die möglicherweise auch fragmentiert werden müssen) bis zu der `remote_max_buf_size` zu der entfernten Vorrichtung für den `this_buf_size`-Puffer senden. Das entfernte JIP empfängt die korrekte Datenmenge (`this_buf_size`), nicht in einem einzelnen Puffer, sondern in Teilmeldungen, die `this_buf_size` ergeben. Ferner, da 4K die Minimalaufzeichnungsgröße ist, die durch das JIP interpretiert werden kann, muss das JSP sicherstellen, dass die Teilmeldungen auch ein Mehrfaches dieser Aufzeichnungsgröße sind: ansonsten könnte das entfernte JIP nicht in der Lage sein, die Daten zu interpretieren, die in einer Teilmeldung empfangen werden. Ein Beispiel einer solchen Meldungsfragmentierung ist in [Fig. 9](#) gezeigt.

**[0177]** Das JSP fügt nur einen JSP-Anfangsblock zu einer Meldung hinzu, wenn sie über einen MES-SAGE-Kanal geht. Datenblöcken, die in einen Stromkanal durch das JIP geschrieben werden, wird nie der JSP-Anfangsblock hinzugefügt. Es ist eine Eigenschaft von STREAM-Kanälen, dass Datenblöcke nicht not-

wendigerweise zu dem anderen Ende der Verbindung geliefert werden, genau so wie sie am Sendeende geschrieben wurden. So kann z. B. das, was als zwei Blöcke von 64 Kilobyte geschrieben wurde, als ein einzelner Block von 128 Kilobyte geliefert werden. Die Reihenfolge der Bytes in dem Strom wird jedoch bewahrt.

**[0178]** JetSend-Vorrichtungen treten üblicherweise Partner-zu-Partner über einen einzelnen Netzwerktyp in Wechselwirkung (z. B. nur IP oder IPX). Dies bedeutet, dass JetSend-Vorrichtungen von einem Netzwerktyp nur mit JetSendVorrichtungen des selben Netzwerktyps kommunizieren können, die physisch mit dem selben Netzwerk verbunden sind. Das Problem tritt auf, wenn eine JetSend-Vorrichtung eine Verbindung mit einer Vorrichtung herstellen muss, die auf einem Netzwerk ist, das durch die erste Vorrichtung nicht unterstützt wird oder ein Protokoll betreibt, das durch die erste Vorrichtung nicht unterstützt wird. Es gibt zwei Lösungsansätze zum Lösen dieses Problems.

**[0179]** Der erste Ansatz ist das Laden mehrerer Kommunikationsstapel auf eine einzelne JetSend-Vorrichtung. Dies ermöglicht, dass die Vorrichtung mit jeglichem Typ einer JetSend-Vorrichtung in Wechselwirkung tritt, vorausgesetzt, sie hat einen geeigneten Kommunikationsstapel. Dieser Ansatz funktioniert gut für die Situation, in der der zugrunde liegende Transport unterschiedlich ist, aber die Vorrichtungen das volle JetSend-Protokoll verstehen (z. B. in dem Fall von IP und IPX).

**[0180]** Eine zweite Lösung für das Problem von mehreren Netzwerken ist das Verwenden eines Gateways. Ein Gateway ist eine Vorrichtung, die unsichtbar in dem Netzwerk sitzt und von einem Netzwerktyp zu einem anderen übersetzt. Der Gateway-Lösungsansatz ist geeignet für eine Situation, in der die Zielortvorrichtungen nicht JetSend-gemäß sind, und jede Quellvorrichtung dann den Zielort modellieren müsste, um mit demselben zu kommunizieren. Dies ist der Fall für Fax und E-Mail. Anstatt die Vorrichtung mit der Fähigkeit zu überlasten, mit Ursprungsnetzwerken zu sprechen, wird diese Funktionalität in den Gateway gesetzt, der üblicherweise auf einer programmierbaren Vorrichtung untergebracht sein kann.

**[0181]** Wenn das JIP eine Sitzung mit einer entfernten Vorrichtung einrichten möchte, die einen JetSend-Gateway zum Zugreifen benötigt, versteckt das JSP die Existenz des Gateways vor dem JIP. Das JSP erzeugt die Sitzung statt dessen mit dem Gateway und leitet sie zu der Adresse der entfernten Vorrichtung in der SYN-Meldung weiter. Von diesem Punkt an, wenn der Gateway die JIP-Daten durch einen der Kanäle in der Sitzung empfängt, gibt er die Daten zu der entfernten Vorrichtung über die Gateway-Verknüpfung weiter.

**[0182]** Wenn das JIP eine Sitzung zu einer zweiten entfernten Vorrichtung erzeugen möchte, auf die durch den selben Gateway zugegriffen wird, öffnet das JSP einfach eine zweite Sitzung mit dem Gateway, wie oben angezeigt ist.

**[0183]** Die JSP-Implementierung sollte eine interne Tabelle beibehalten, die eine Abbildung von Gateway-Typen auf Gateway-Adressen enthält, so dass die Vorrichtung eine Verbindung mit einem Gateway gemäß Adresse herstellen kann.

**[0184]** Das JSP führt ein Rundsenden einer Gateway-Anforderung durch, durch Aussenden einer GWY-Meldung über einen Datagrammport. (Wenn das JSP über ein IP-Netzwerk läuft, wird dies ausgeführt durch Öffnen eines UDP-Ports, um die Gateway-Meldung zu senden/empfangen.) Nach dem Rundsenden der GWY-Meldung hört das JSP dann nach einer Antwort an. der selben lokalen Adresse (oder dem UDP-Port), durch den es die Rundsendung gesendet hat. Ein JetSend-Gateway hört nach Rundsendungen an seinem bekannten Datagrammtransportport. Wenn er eine Gateway-Rundsendung empfängt (GWY-Paket), stellt er sicher, dass er die Version in dem Versionsfeld unterstützen kann. Wenn die Version nicht unterstützt wird, wirft er das Paket weg. Wenn die Version unterstützt wird, versucht er zu sehen, ob der Zielort „alle Gateways“ (0 × FF) oder dieser bestimmte Gateway ist. Wenn eine Übereinstimmung vorliegt, dann prüft er, ob es der angeforderte Typ von Gateway ist. Wenn der angeforderte Typ NULL ist, dann sollten alle Gateways aller Typen antworten. Wenn er ungleich NULL ist, dann sollten nur die Gateways dieses Typs antworten. Wenn das Gateway bestimmt, es sollte antworten, antwortet es auf die Rundsendung unter Verwendung einer GACK-Meldung. Diese GACK wird über einen Datagrammport auf dieselbe Weise gesendet wie eine GWY-Meldung.

**[0185]** Wenn das JSP eine GACK-Antwort zu seinem GWY empfängt, stellt es sicher, dass es die Version in dem Versionsfeld unterstützen kann. Wenn nicht, ignoriert es das Paket. Wenn das JSP die Version unterstützen kann, speichert es die Gateway-Informationen in seiner internen Tabelle zwischen und benachrichtigt das JIP über den neuen Gateway. Das JIP ist nun in der Lage, eine Sitzung mit dem neuen Gateway zu öffnen.

**[0186]** Wenn eine GACK nicht rechtzeitig ankommt, nachdem das JSP eine GWY-Meldung rundsendet, ver-

wendet das JSP einen Zeitgeber zum Neuübertragen einer GWY, wie in [Fig. 10](#) gezeigt ist. Wenn das JSP die erste GWY sendet, setzt es einen Zeitgeber für sechs Sekunden. Wenn sechs Sekunden abgelaufen sind und das JSP keine GACK empfangen hat, verdoppelt es den Zeitgeber auf zwölf Sekunden und überträgt erneut. Jedes Mal, wenn es erneut überträgt, verdoppelt es den Zeitgeber. Wenn die abgelaufene Gesamtzeit neunzig Sekunden erreicht, gibt das JSP auf. Zu dieser Zeit antwortet es dem JIP, dass es keine Sitzung zu der entfernten Vorrichtung einrichten kann.

**[0187]** Üblicherweise wird die Gateway-Rundsendung ansprechend auf eine Anforderung von dem JIP ausgeführt. Es wird jedoch darauf hingewiesen, dass keine Möglichkeit für eine Gateway-Vorrichtung besteht, ihr Vorhandensein anzukündigen. Anders ausgedrückt wird das Senden einer unaufgeforderten GACK-Meldung von dem Protokoll bei dieser Implementierung nicht erlaubt.

**[0188]** JSP-Anfangsblockformate werden nun beschrieben, zusammen mit einer kurzen Beschreibung des Anfangsblocks für jede spezifische JSP-Meldung. Jede JSP-Meldung wird mit einem 20-Byte-JSP-Anfangsblock eingeleitet. Das „generische“ Format des Anfangsblocks wird nachfolgend in [Fig. 11](#) gezeigt, und eine allgemeine Beschreibung jedes Feldes folgt.

**[0189]** Einige der Felder werden bei jeder JSP-Meldung verwendet, während andere nur von einigen benötigt werden. Oktett(Byte-) Versätze von dem Anfang der Meldung sind von oben nach unten nummeriert. Bitpositionen sind von rechts nach links beginnend mit dem niedrigwertigsten Bit und endend mit dem höchstwertigen Bit nummeriert. Der numerische Wert eines gesetzten Bits ist die Bitposition hoch Zwei. Alle Mehrfachbytworte in einem Segment werden über das Netzwerk in einer Netzwerkbytereihenfolge übertragen – was in einer Big-Endian-Form erfolgt.

**[0190]** JSP-Anfangsblockfelder (Header Fields) werden nun beschrieben. Das 16-Bit-Steuerbit-Feld (Control Bit field) belegt Bytes 0 und 1 des Anfangsblocks. Es ist mit den nachfolgenden Bits codiert, die hier definiert sind:

1. SYN Einrichten von Sitzung oder Kanal.
2. ACK Akzeptieren von Sitzung oder Kanal.
3. NAK Ablehnen von Sitzung oder Kanal.
4. RST Schließen der Sitzung oder des Kanals.
5. NUL Lebenserhaltung.
6. CHN Die Meldung wird zu einem spezifischen Kanal gerichtet.
7. GWY Die Meldung ist eine Gateway-Rundsendung oder -Antwort.
8. EXT Erweiterter Anfangsblock. Byte 8 enthält zusätzliche Steuerbits (siehe CEXTLSN).
9. LF Fragment-Flag. Wenn es 0 ist, ist es nicht das letzte Fragment, wenn es 1 ist, ist es das letzte Fragment.
10. Unbenutzt – gesetzt auf 0.
11. Unbenutzt – gesetzt auf 0.
12. Unbenutzt – gesetzt auf 0.
13. Unbenutzt – gesetzt auf 0.
14. Unbenutzt – gesetzt auf 0.
15. Unbenutzt – gesetzt auf 0.
16. Unbenutzt – gesetzt auf 0.

**[0191]** Die Steuerbits werden in Kombinationen verwendet, um die Meldungstypen zu erzeugen, die in Tabelle 1 unten gezeigt sind (das Bit EXT ist 0 in all diesen Kombinationen).

SYN	ACK	NAK	RST	NUL	CHN	Name	Bedeutung
1	0	0	0	0	0	SYN	Anforderung nach Sitzung
1	1	0	0	0	0	ASYN	Sitzungsanforderung akzeptieren
1	0	1	0	0	0	NSYN	Sitzungsanforderung ablehnen
0	0	0	1	0	0	RST	Sitzung schließen
0	0	0	0	1	0	NUL	Lebenserhaltung
0	0	0	0	0	1	CHN	Daten für einen Kanal
1	0	0	0	0	1	CSYN	Anforderung für Kanal
1	1	0	0	0	1	CASYN	Kanal Anforderung akzeptieren
1	0	1	0	0	1	CNSYN	Kanal Anforderung ablehnen
0	0	0	1	0	1	CRST	Kanal schließen

TABELLE 1: JSP-Meldungstypen

ACK	CHN	Name	Bedeutung
0	1	GWY	Gateway-Rundsendung
1	1	GACK	Gateway-Rundsenderückantwort

TABELLE 2: JSP-Gateway-Meldungen

**[0192]** Das 16-Bit-Versionsnummernfeld belegt die Bytes 2 und 3 des Anfangsblocks. Es enthält die Version des JSP-Protokolls, die verhandelt wurde oder wird. Es sei angenommen, dass die Version eine signierte 16-Bit-Ganzzahl ist.

**[0193]** Das 16-Bit-Source-Feld (Source = Quelle) belegt Bytes 4 und 5 des Anfangsblocks. Es enthält den Source-Identifizierer des Pakets, und abhängig von dem JSP-Meldungstyp könnte dies entweder die JSP-Portnummer, Verbindungsidentifizierer oder Kanalidentifizierer sein. Es wird angenommen, dass die Source eine signierte 16-Bit-Ganzzahl ist.

**[0194]** Das 16-Bit-Zielortfeld (Destination Field) belegt Bytes 6 und 7 des Anfangsblocks. Es enthält einen Zielortidentifizierer für das JSP-Paket. Wiederum könnte dies abhängig von dem JSP-Meldungstyp entweder eine JSP-Portnummer, Verbindungsidentifizierer oder Kanalidentifizierer sein. Es wird angenommen, dass der Zielort eine 16-bit-signierte ganze Zahl ist. In dem Fall eines GWY-Pakets ist er auf  $0 \times FF$  eingestellt, als ein Zeichen für alle Gateways zu antworten.

**[0195]** Das 32-Bit-Zusatzdatenfeld (Additional Data field) belegt Bytes 8–11 des Anfangsblocks. Es wird durch das JSP verwendet, um zusätzliche Daten zu halten, die durch bestimmte JSP-Pakete benötigt werden. Ein weiteres solches Feld belegt Bytes 12–15 des Anfangsblocks. Zum Beispiel werden bei CEXTLSN die ersten 8 Bytes für spezifische Codes verwendet, um anzuzeigen, ob das Hören nach Meldungs- und/oder Strom-Kanälen begonnen oder geendet hat.

**[0196]** Das 16-Bit-Länge-Feld belegt Bytes 16 und 17 des Anfangsblocks. Es enthält die Länge (in Bytes) jeglicher Daten, die dem JSP-Anfangsblock folgen. Es wird angenommen, dass die Länge eine unsignierte 16-Bit-Ganzzahl ist.

**[0197]** Das letzte 16-Bit-Feld, das Auffüllfeld (Padding field), belegt Bytes 18 und 19 des Anfangsblocks. Dieses Auffüllen wird verwendet, um sicherzustellen, dass der JSP-Anfangsblock auf eine 4-Byte-Grenze fällt. Dies ist das Ende des JSP-Anfangsblocks.

**[0198]** Das JetSend-Wechselwirkungsprotokoll (JIP; JetSend Interaction Protocol) und die Meldungen, die das Protokoll bilden, werden nun beschrieben. Die Verwendung des JSP durch das JIP, um zu erlauben, dass Geräte Informationen austauschen und gemeinschaftlich verwenden, wird ebenfalls erörtert.

**[0199]** Grundlegend für das JIP ist das Konzept des Oberflächenaustauschs. Eine Weise, das Konzept darzustellen, ist sich „eine Oberfläche“ als die Oberfläche eines Blocks aus Modellierton vorzustellen. Dieser Block aus Ton kann durch seinen Eigentümer in jegliche Form geformt werden. Ein Beobachter sieht die Oberfläche dieses Objekts, wie es durch seine Oberfläche dargestellt ist: anders ausgedrückt formt der Eigentümer des Tons den Block in eine Form, deren Oberfläche beschreibt, was das Objekt ist.

**[0200]** Es sei angenommen, der Beobachter hat einen anderen, ungeformten Tonblock. Der Eigentümer des geformten Originalblocks aus Ton kann seinen geformten Block auf die Oberfläche des ungeformten Blocks des Beobachters abdrucken. Der Beobachter hat nun eine exakte Kopie der geformten Originalform, die auf seinen eigenen Tonblock abgedruckt ist (genau genommen bricht die Metapher an diesem Punkt zusammen: beim Oberflächenaustausch hat der Beobachter eine genaue Kopie des Originals, keine Umkehrung oder Spiegelbild). So hat der Eigentümer des Originaltonblocks (die Ausdrucks-Vorrichtung; expressive device) auf den Tonblock des Beobachters (die Abdrucks-Vorrichtung; impressive device) eine Kopie der Originaloberfläche abgedruckt.

**[0201]** Das JIP besteht aus einer geringen Anzahl von Meldungen, die jeglicher Anzahl von Vorrichtungen erlauben, Informationen gemeinschaftlich zu verwenden, die Oberflächen genannt und gemäß dem Oberflächenaustauschmodell ausgetauscht werden. Bei jeglicher Wechselwirkung besitzt eine Vorrichtung die Oberfläche. Die Kopie des Eigentümers wird als der Ausdruck der Oberfläche bezeichnet, und der Eigentümer selbst ist als die Ausdrucksvorrichtung bekannt. Alle anderen Kopien der Oberfläche werden als die Abdrucke bezeichnet und die Vorrichtungen, die dieselben halten, werden als die Abdrucksvorrichtungen bezeichnet. Die Meldungen, die durch das JIP geliefert werden, ermöglichen der Ausdrucksvorrichtung, Abdrucke zu erzeugen und zu zerstören, den Abdrucksvorrichtungen Abdrucke zu zerstören, die sie haben und jeglicher Vorrichtung, den Originaloberflächenausdruck zu modifizieren.

**[0202]** Um das Konzept von Oberflächen, Ausdrucken, Abdrucken usw. zu implementieren, wurde eine Liste aus Meldungen erzeugt. Durch die Verwendung dieser Meldungen finden alle „Oberflächenwechselwirkungen“ statt. Die nachfolgenden Meldungen bilden das Wechselwirkungsprotokoll:

- SurfaceMsg (Impress; Abdrucken) – erzeugt neue Abdrucke von Oberflächen auf der Zielvorrichtung, wird auch verwendet, um Anforderungen nach Abdrucken abzulehnen.
- SurfaceDeleteMsg (Delete; Löschen) – benachrichtigt Abdrucksvorrichtungen, dass die Ausdrucksvorrichtung den Originalausdruck gelöscht hat.
- SurfaceReleaseMsg (Unimpress; Abdruck entfernen) – benachrichtigt die Ausdrucksvorrichtung, dass eine Abdrucksvorrichtung einen Abdruck gelöscht hat.
- SurfaceRequestMsg (Surface Request; Oberflächenanforderung) – erlaubt einer Vorrichtung, einen Abdruck einer benannten Oberfläche anzufordern.
- DescriptionRequestMsg (Description Request; Beschreibungsanforderung) – ermöglicht einer Vorrichtung, die Beschreibung für eine Oberfläche anzufordern, von der sie einen Abdruck hat.
- DescriptionReplyMsg (Description Reply; Beschreibungsrückantwort) – überträgt die Beschreibung für einen Abdruck ansprechend auf eine Beschreibungsanforderung.
- ContentRequestMsg (Content Request; Inhaltsanforderung) – erlaubt einer Abdrucksvorrichtung, einige Inhaltsdaten von der Ausdrucksvorrichtung anzufordern.
- ContentReplyMsg (Content Data; Inhaltsdaten) – überträgt einige Inhaltsdaten von der Ausdrucksvorrichtung auf eine Abdrucksvorrichtung ansprechend auf eine Inhaltsanforderung: es kann eine Folge dieser Meldungen ansprechend auf eine Inhaltsanforderung vorliegen, und diese Meldung wird ebenfalls verwendet, um eine Inhaltsanforderung abzulehnen.
- SurfaceChangeMsg (Change; Änderung) – benachrichtigt eine Vorrichtung, dass sich die Informationen geändert haben (d. h. dadurch, dass Ausdrucksvorrichtungen Abdrucksvorrichtungen über eine Änderung benachrichtigen, und dass Abdrucksvorrichtungen eine Änderung eines Ausdrucks anfordern – auch Ablehnungen dieser Anforderungen).

**[0203]** Eine Oberfläche hat eine Anzahl von Attributen. Es gibt einen Namen, einen Identifizierer, eine Klasse, einen Satz aus Eigenschaften, eine Beschreibung, einige Inhaltsdaten und eine Version. Der Name ist eine mit NULL abschließende ASCII-Zeichenfolge. Der Identifizierer ist jeder Oberfläche zugeordnet und identifiziert sie eindeutig in dem JIP. Die Klasse wird verwendet, um den Zweck der Oberfläche zu bestimmen. Der Satz aus Eigenschaften steuert, auf welche JIP-Meldungen eine Ausdrucksvorrichtung antwortet. Die Beschreibung

enthält eine Beschreibung der Formate, in denen die Daten verfügbar sind, oder die die Ausdrucksvorrichtung bereit ist, zu liefern. Die Inhaltsdaten enthalten die tatsächlichen Bytes der Informationen selbst. Die Version wird durch den Änderungsmechanismus verwendet, so dass Ausdrucks- und Abdrucks-Vorrichtungen wissen, auf welche Version einer Oberfläche sich eine Änderung bezieht.

**[0204]** Eine typische Wechselwirkung geht wie folgt vonstatten. Zuerst erzeugt die Vorrichtung mit Informationen zum Übertragen, was die Ausdrucksvorrichtung ist, einen Ausdruck. Um dies zu tun, muss sie einen Namen erzeugen, einen eindeutigen Identifizierer zuordnen, einen Satz aus Eigenschaften erzeugen und eine Beschreibung erzeugen. An diesem Punkt muss sie keine Inhaltsdaten erzeugen, obwohl sie in der Lage sein muss, den Inhalt zu erzeugen, der in der Oberflächenbeschreibung beschrieben ist. Als Nächstes verwendet die Ausdrucksvorrichtung diese Attribute und versucht, Abdrucke dieser Oberfläche zu erzeugen, durch Senden einer SurfaceMsg zu der Ziel-Vorrichtung oder den – Vorrichtungen. Es wird darauf hingewiesen, dass solche SurfaceMsgs unaufgefordert ausgesendet werden können oder ansprechend auf eine frühere SurfaceRequestMsg ausgesendet werden können, die von einer anderen Vorrichtung empfangen wird. Es wird auch darauf hingewiesen, dass zum Erzeugen eines Abdrucks unter Verwendung der SurfaceMsg die Ausdrucksvorrichtung eine „Zielfläche“ haben muss, auf die der Ausdruck „abgedruckt“ wird. Wenn die SurfaceMsg auf eine frühere SurfaceRequestMsg antwortet, kann sich dieser Zielflächenidentifizierer in der SurfaceRequestMsg befinden. Wenn jedoch die Ausdrucksvorrichtung einen unaufgeforderten Abdruck erzeugt, kann der Zielflächenidentifizierer der eines existierenden Abdrucks sein, wobei in diesem Fall der Ausdruck bereits existieren muss oder auf den „Voreingestelltes Ziel“ Identifizierer eingestellt sein kann.

**[0205]** Der Voreingestelltes-Ziel-Identifizierer wird manchmal auch als die „Arbeitsfläche“ bezeichnet. Die Existenz einer solchen Voreinstellung ist wichtig für die ordnungsgemäße Implementierung des JIP. Ansonsten besteht ein Umladeproblem, wenn eine Ausdrucksvorrichtung erstmals eine Meldung zu einer Abdrucksvorrichtung sendet: die Ausdrucksvorrichtung weiß nicht, wo ein Abdruck auf der Abdrucksvorrichtung erzeugt werden soll (wovon sie an diesem Punkt nichts weiß), und die Abdrucksvorrichtung kann der Ausdrucksvorrichtung nicht ohne weiteres sagen (ohne Senden einer Art globaler Meldung), da sie sich nicht bewusst ist, dass die Ausdrucksvorrichtung einen Abdruck erzeugen will. Die Lösung ist, dass für alle Vorrichtungen, die Abdrucke akzeptieren, eine voreingestellte oder Arbeits-Oberfläche mit einem Voreingestelltes-Ziel-Identifizierer existiert (bei dieser Implementierung wurde der Voreingestelltes-Ziel-Identifizierer auf Eins eingestellt). Dies ermöglicht, dass jegliche Ausdrucksvorrichtung einen Abdruck auf einer Abdrucksvorrichtung erzeugt, durch Einstellen des Ziel-Identifizierfeldes auf Eins. Die Abdrucksvorrichtung kann dann in eine Kommunikation mit der Ausdrucksvorrichtung eintreten (z. B. mit einer SurfaceRequestMsg-Meldung, die Abdrucke auf eine neue Zielfläche anfordert).

**[0206]** Eine Reihe von Beispielen, die die Verwendung der Meldungen des JIP darstellen, wird nachfolgend gegeben, Bezug nehmend auf [Fig. 12a](#) bis [Fig. 12k](#). [Fig. 12a](#) ist im Wesentlichen ähnlich zu [Fig. 1](#), ist aber der Einfachheit halber als [Fig. 12a](#) vorgesehen.

Beispiel 1: [Fig. 12a](#)

**[0207]** Eine Ausdrucksvorrichtung möchte einen nicht angeforderten Abdruck erzeugen. Zuerst wird ein Oberflächenausdruck **121** erzeugt. Dieser wird dann auf die Abdrucksvorrichtung mit SurfaceMsg abgedruckt und ein Abdruck **122** der Oberfläche existiert an der Abdrucksvorrichtung.

Beispiel 2: [Fig. 12b](#)

**[0208]** Eine Ausdrucksvorrichtung erzeugt einen Oberflächenausdruck für Informationen, die sie mit anderen Geräten austauschen möchte. Bei diesem Beispiel existiert der Ausdruck bereits, bevor er angefordert wird, aber dies ist nicht notwendigerweise der Fall (z. B. werden Kindoberflächen in einigen Fällen möglicherweise nicht erzeugt, bis sie tatsächlich angefordert werden). Die Ausdrucksvorrichtung empfängt dann eine Anforderung nach einem Oberflächenabdruck in einer SurfaceRequestMsg von der Abdrucksvorrichtung, und versucht ansprechend darauf, den Abdruck mit einer SurfaceMsg zu erzeugen. Das Endergebnis ist wie bei Beispiel 1, ein Abdruck **122** wird an der Abdrucksvorrichtung erzeugt.

Beispiel 3: [Fig. 12c](#)

**[0209]** Eine Ausdrucksvorrichtung erzeugt einen Oberflächenausdruck und versucht, einen nicht angeforderten Abdruck auf einer Abdrucksvorrichtung zu erzeugen, wie bei Beispiel 1. Der Abdruck **122** wird erzeugt, wird aber dann direkt mit einer SurfaceReleaseMsg von der Abdrucksvorrichtung zu der Ausdrucksvorrichtung frei-

gegeben **129**. Der Endzustand ist ein Ausdruck **121** der Oberfläche an der Ausdrucksvorrichtung, aber kein Abdruck der Oberfläche an der Abdrucksvorrichtung.

Beispiel 4: [Fig. 12d](#)

**[0210]** Wie bei Beispiel 1 wird ein nicht angeforderter Abdruck **122** erfolgreich auf die Abdrucksvorrichtung abgedruckt. Die Abdrucksvorrichtung kann dann die Beschreibung in dem Abdruck **122** verwenden, um zu bestimmen, welche Aktion als Nächstes unternommen werden soll. In einigen Fällen, wie dem bei diesem Beispiel, ist die Oberflächenbeschreibung, die in der ursprünglichen SurfaceMsg enthalten ist, nicht vollständig. Die Abdrucksvorrichtung kann dann mehr Informationen von der Ausdrucksvorrichtung mit einer DescriptionRequestMsg-Meldung anfordern. Die Ausdrucksvorrichtung antwortet auf die DescriptionRequestMsg mit einer DescriptionReplyMsg, die die weitere Oberflächenbeschreibung enthält.

Beispiel 5: [Fig. 12e](#)

**[0211]** Eine Oberflächenbeschreibung kann eine Bezugnahme auf Teiloberflächen oder Kindoberflächen der Oberfläche oberer Ebene enthalten (z. B. E-Material, codiert als eine Zuordnung, enthält in der Praxis immer Kindoberflächen). Beispiel 5 bezieht sich auf eine Oberfläche A, die eine Kindoberfläche A1 aufweist. Ein Ausdruck **121**, **123** jeder Oberfläche wird an der Ausdrucksvorrichtung bereitgestellt (alternativ kann an diesem Punkt nur ein Ausdruck **121** der Oberfläche A vorgesehen sein). Oberfläche A wird dann auf die Abdrucksvorrichtung mit einer SurfaceMsg abgedruckt. Die Abdrucksvorrichtung kann dann einen Abdruck der Kindoberfläche A1 von der Ausdrucksvorrichtung mit einer SurfaceRequestMsg anfordern. Diese Anforderung kann abgelehnt oder angenommen werden, wobei in dem letzteren Fall die Ausdrucksvorrichtung eine weitere SurfaceMsg sendet (nach dem erstmaligen Erzeugen eines Ausdrucks einer Kindoberfläche A1, wenn ein solcher Ausdruck nicht bereits existiert). Der Endzustand ist mit einem Ausdruck **121** der Oberfläche A und einem Ausdruck **123** der Kindoberfläche A1 an der Ausdrucksvorrichtung und entsprechenden Abdrucken **122**, **124** der Oberfläche A und Kindoberfläche A1 an der Abdrucksvorrichtung.

Beispiel 6: [Fig. 12f](#)

**[0212]** Sobald ein Abdruck einer Oberfläche an einer Abdrucksvorrichtung bereitgestellt ist, kann die Abdrucksvorrichtung Inhalt mit einer ContentRequestMsg anfordern. Beim Empfangen einer ContentRequestMsg kann die Ausdrucksvorrichtung die Anforderung ablehnen oder Inhalt **125** in dem angeforderten Format liefern. Dieser Inhalt kann als eine ContentReplyMsg-Meldung (wie hier), eine Reihe von ContentReplyMsg-Meldungen oder durch eine andere Einrichtung gesendet werden, wie z. B. einen Strom.

Beispiel 7: [Fig. 12g](#)

**[0213]** Wenn eine Abdrucksvorrichtung entscheidet, dass sie einen Abdruck nicht mehr benötigt (z. B. ist es ein Drucker und er hat bestätigt, dass die Oberfläche ein Dokument darstellt, das nun erfolgreich gedruckt wurde), kann sie den Abdruck freigeben durch Senden einer SurfaceReleaseMsg zu der Ausdrucksvorrichtung. Diese Situation ist in [Fig. 12g](#) gezeigt, die der Situation von Beispiel 6 folgt nachdem Inhalt durch die Abdrucksvorrichtung angefordert wurde und empfangen wurde, wird eine SurfaceReleaseMsg zurück zu der Ausdrucksvorrichtung gesendet, um der Ausdrucksvorrichtung zu sagen, dass der Abdruck „entfernt bzw. aufgehoben“ wird. Die Ausdrucksvorrichtung ignoriert dann jegliche nachfolgenden Meldungen, die sich auf die vom Abdruck bereinigte Oberfläche beziehen.

Beispiel 8: [Fig. 12h](#)

**[0214]** Eine Ausdrucksvorrichtung kann selbst einen Ausdruck **128** löschen. Sie tut dies durch Senden einer SurfaceDeleteMsg zu allen Abdrucksvorrichtungen, die einen Abdruck **122** des Originalausdrucks **121** haben: die Meldung zeigt an, dass der Ausdruck gelöscht wurde, und die Ausdrucksvorrichtung ignoriert dann jegliche Meldungen, die sich auf die gelöschte Ausdrucksoberfläche beziehen.

Beispiel 9: [Fig. 12i](#)

**[0215]** Die Eigenschaften einer Ausdrucksoberfläche können so eingestellt sein, dass die Abdrucksvorrichtung in der Lage ist oder nicht, die Ausdrucksoberfläche zu ändern (die Ausdrucksvorrichtung kann dies immer tun). [Fig. 12i](#) zeigt eine Änderung der Ausdrucksoberfläche **126** durch die Ausdrucksvorrichtung. Die Änderung der Ausdrucksoberfläche wird reflektiert durch das Senden einer SurfaceChangeMsg-Meldung von der

Ausdrucksvorrichtung zu allen Abdrucksvorrichtungen, um anzuzeigen, dass eine Änderung an dem Originalausdruck ausgeführt wurde. Dem folgen üblicherweise neue Inhaltsanforderungen und möglicherweise sogar neue Beschreibungsanforderungen.

Beispiel 10: [Fig. 12j](#)

**[0216]** Bei diesem Beispiel fordert die Abdrucksvorrichtung eine Änderung an dem Originalausdruck an. Wiederum wird dies mit Hilfe einer SurfaceChangeMsg ausgeführt. Dies kann von der Ausdrucksvorrichtung entweder erlaubt oder abgelehnt werden. Wenn die Änderung akzeptiert wird, sendet die Ausdrucksvorrichtung eine weitere SurfaceChangeMsg zu allen Abdrucksvorrichtungen, die die Änderung für die anfordernde Abdrucksvorrichtung bestätigt und die verbleibenden Abdrucksvorrichtungen benachrichtigt. Wenn die Änderung abgelehnt wird, benachrichtigt die Ausdrucksvorrichtung die anfordernden Abdrucksvorrichtungen, dass die Anforderung fehlgeschlagen ist.

**[0217]** Wenn eine anfordernde Abdrucksvorrichtung erfolgreich eine Änderung an dem Ausdruck angefordert hat, muss sie im Allgemeinen keinen aktualisierten Inhalt anfordern (obwohl dies andere Abdrucksvorrichtungen tun müssen). Der Grund dafür ist, dass die Abdrucksvorrichtung normalerweise in der Lage ist, ihren eigenen Inhalt zu aktualisieren, basierend auf der Beschreibungsänderung, die sie von der Ausdrucksvorrichtung anforderte.

Beispiel 11: [Fig. 12k](#)

**[0218]** [Fig. 12k](#) zeigt explizit einen Fall, in dem zwei Abdrücke eines Ausdrucks vorliegen, und eine Änderungsanforderung durch eine der Abdrucksvorrichtungen ausgeführt und durch die Ausdrucksvorrichtung akzeptiert wird. Die selbe SurfaceChangeMsg-Meldung wird im Allgemeinen durch die Ausdrucksvorrichtung zu beiden Abdrucksvorrichtungen gesendet (obwohl, wenn die Ausdrucksvorrichtung sich ausreichend über die Präferenzen der Abdrucksvorrichtungen bewusst ist, es ebenfalls möglich ist, dass SurfaceChangeMsg-Meldungen, die spezifisch an die Anforderungen jeder Abdrucksvorrichtung angepasst sind, geliefert werden könnten), und der Endzustand ist, dass alle Oberflächen die Änderung reflektieren: die zweite Abdrucksvorrichtung (die die Änderung nicht angefordert hat) sendet wahrscheinlich eine ContentRequestMsg zu der Ausdrucksvorrichtung, um neuen Inhalt zu erhalten.

**[0219]** Das JIP läuft über das JetSend Session Protocol (JSP). Wie oben erörtert wurde, verwaltet das JSP alle Aspekte einer Sitzung zwischen zwei Vorrichtungen, einschließlich dem Erzeugen und Löschen von Sitzungen wie dem Entscheiden, wann eine Sitzung unverwendbar wurde. Das JSP liefert ferner Zugriff auf die Basisadressierung, zuverlässige Meldungstransportprotokolle und jegliche anderen Transportprotokolle, die durch das JIP verwendet werden.

**[0220]** Ein Gerät kann eine Sitzung aufweisen, die passiv gestartet wird, oder es kann aktiv eine Sitzung mit einem anderen starten. Damit das JIP eine Sitzung passiv startet, muss es zuerst das JSP anweisen, nach einer Sitzung auf einem spezifischen Transport zu hören. Sobald das JSP auf diesem Transport hört, kann ein anderes Gerät dann aktiv eine Sitzung starten, durch Anweisen des JSP, die Vorrichtung auf dem spezifischen Transport zu rufen. Wenn eine Verbindung hergestellt wird, verhandelt das entfernte und lokale JSP zu einer unterstützten Version des JSP, und an diesem Punkt sollten beide JIPs benachrichtigt werden, dass eine Sitzung eingerichtet wurde. Das JSP liefert dem JIP Sitzungs-Handhabungen, die auf diese Sitzung abbilden. Diese Sitzungs-Handhabungen sollten immer verwendet werden, wenn das JIP spezifisch Bezug auf die Sitzung nehmen muss, wie z. B. wenn das JIP die Sitzung beenden möchte.

**[0221]** Eine Implementierung des JIP behält potentiell relativ viel Zustand über Oberflächen bei, die sich auf eine gegebene Sitzung beziehen. Unter einigen Umständen kann das JSP eine Sitzung beenden, während Vorrichtungen immer noch ausstehende Abdrücke, Anforderungen und Inhaltsdaten haben. Dies kann z. B. auftreten, wenn eine Vorrichtung heruntergefahren wird oder wenn ein Netzwerkausfall vorliegt, der verursacht, dass das JSP die Sitzung löscht. Wenn das JIP unerwartet über das Ende einer Sitzung benachrichtigt wird, muss es dann seinen lokalen Zustand erledigen, der sich auf diese Sitzung bezieht, so dass keine Artefakte bleiben, die nur eine Bedeutung in dem Kontext der Sitzung haben. Zum Beispiel empfängt das lokale JIP keine SurfaceReleaseMsgs, die ihm sagen, dass seine ausstehenden Abdrucke freigegeben wurden und er somit seinen internen Zustand aufräumen bzw. erledigen muss. Dasselbe gilt für jegliche ausstehenden ContentRequestMsgs, DescriptionRequestMsgs, etc.

**[0222]** Das JIP tauscht seine Meldungen über Kanäle aus, die durch das JSP bereitgestellt und beibehalten

werden. Diese Meldungskanäle müssen zuverlässig und geordnet sein. Eine Implementierung eines JSP kann verschiedene andere Kanaltypen liefern, wie z. B. einen Strom-Kanal.

**[0223]** Die Meldungskanäle können erzeugt werden, sobald eine Sitzung zwischen zwei Geräten eingerichtet wurde. Sobald eine Sitzung eingerichtet wurde, ist es die Verantwortung des aktiven JIP, anzufordern, dass der erste Meldungskanal geöffnet wird. Entsprechend ist es die Verantwortung des passiven JIP, nach einem Kanal zu hören. Somit wird vorgeschlagen, dass das passive JIP das JSP anweist, nach einem Kanal zu hören, sobald eine Verbindung eingerichtet ist. Die Funktionalität des Rufens, Schließens und Hörens nach Kanälen wird durch das JSP bereitgestellt. Es wird darauf hingewiesen, dass ein aktiver Ruf zum Öffnen eines Kanals nicht durch das JIP ausgegeben werden sollte, bis das JSP eine Benachrichtigung gibt, dass die entfernte Vorrichtung passiv nach einem Kanal hört.

**[0224]** Sobald das JSP über einen Meldungskanal verhandelt und denselben geöffnet hat, liefert es eine Handhabung zu diesem Kanal zu dem JIP. Das JIP kann dann diesen Kanal verwenden, um Meldungen zu senden und zu empfangen. Dieser Kanal ist gültig, bis er geschlossen wurde, entweder durch einen Netzwerkausfall oder durch einen ausdrücklichen Ruf, dies zu tun. Sobald eine Sitzung eingerichtet wurde und ein Meldungskanal geöffnet wurde, kann jede Seite der Verbindung anfordern, dass zusätzliche Kanäle geöffnet werden. Diese zusätzlichen Kanäle können auf jedem der Transporte geöffnet werden, die durch das JSP unterstützt werden.

**[0225]** Zusätzlich zum Senden von Inhaltsdaten über den Meldungskanal ermöglicht das JIP, dass Inhaltsdaten über jeglichen spezifischen Kanal gesendet werden, der durch die zwei Geräte unterstützt wird. Sowohl die ContentRequestMsg als auch die ContentReplyMsg enthalten ein Feld, das sich auf die Auswahl eines Kanals bezieht, über den die Inhaltsdaten fließen. Die CEXTLSN-Meldung des JSP enthält Informationen, die spezifizieren, welche Kanaltypen zum Öffnen verfügbar sind.

**[0226]** Zur Darstellung wird das Beispiel des JIP/JSP genommen, das auf einem IP-Netzwerk läuft. Hier könnten zwei Kanaltypen durch das JSP unterstützt werden: Meldungskanäle, die RMTP verwenden, und Stromkanäle, die TCP verwenden. Nachdem die Sitzung eingerichtet ist und ein Meldungskanal erzeugt wurde, können verschiedene JIP-Meldungen rückwärts und vorwärts gesendet werden. Die ContentRequestMsg kann spezifizieren, dass die tatsächlichen Inhaltsdaten über einen spezifischen, existierenden Kanal, jeglichen Meldungskanal oder jeglichen Kanal eines beliebigen Typs gesendet werden.

**[0227]** Die Reihenfolge ist wie folgt. Zuerst gibt das Empfangsgerät eine ContentRequestMsg zu dem Quellgerät aus. Wenn die Empfangsvorrichtung Daten auf einem bereits bestehenden Kanal mit der Quellvorrichtung empfangen möchte, kann sie dies in der Anforderung spezifizieren, durch Integrieren des Kanalidentifizierers der Quellvorrichtung, der auf den spezifischen Kanal abbildet. (Dieser Identifizierer kann in der CA-SYN-Meldung des JSP enthalten sein, die sich auf die Erzeugung dieses Kanals bezieht.) Wenn sie statt dessen den Inhalt auf jeglichem Meldungskanal empfangen möchte, verwendet sie eine 0 als Kanalidentifizierer. Schließlich, wenn es ihr egal ist, über welchen Kanal oder Kanaltyp die Daten kommen sollen, verwendet sie einen Wert -1 als Kanalidentifizierer.

**[0228]** Die Quellvorrichtung empfängt die ContentRequestMsg. Wenn die Anforderung einen spezifischen Kanal spezifiziert, muss die Quellvorrichtung ihren Inhalt über diesen JSP-Kanal senden. Wenn die Anforderung spezifiziert, dass die Daten über einen Meldungskanal kommen, müssen die Daten über einen existierenden Meldungskanal gesendet werden. Schließlich, wenn die Anforderung anzeigt, dass es dem Empfänger egal ist, über welchen Kanal die Daten gesendet werden sollen, kann die Quellvorrichtung dann entweder einen existierenden Kanal (jeglichen Typs) verwenden oder einen neuen Kanal (jeglichen Typs) mit dem Empfänger öffnen und die Daten über denselben senden.

**[0229]** Die Art und Weise, wie das JIP Inhalt über die JSP-Stromkanäle sendet, ist unterschiedlich von der Weise, wie es Inhalt über Meldungskanäle sendet. Alle Inhaltsdaten, die durch das JSP über Meldungskanäle gesendet werden, werden in der Form von CHN-Meldungen gesendet. Wenn Inhalt als Meldungen gesendet wird, verwendet das JIP eine ContentReplyMsg, die die tatsächlichen Inhaltsdaten umfasst. Diese ContentReplyMsg wird zu dem JSP hinuntergeleitet. Das JSP seinerseits sendet sie in der Form von CHN-Meldungen aus. Basierend auf der Gesamtgröße des Inhalts kann das JIP eine oder mehrere dieser ContentReplyMsg-Meldungen senden, um die angeforderte Inhaltslieferung fertigzustellen.

**[0230]** In dem Fall, in dem das JIP den Inhalt über einen Stromkanal senden soll, wird jedoch keine ContentReplyMsg verwendet. Statt dessen gibt das JIP den rohen Inhalt weiter hinunter zu dem JSP, um ausgesendet

zu werden.

**[0231]** Dieser Mechanismus einer spezifischen Kanalauswahl ermöglicht Vorrichtungen die volle Kontrolle darüber, wann spezielle Kanäle, wie z. B. Stromkanäle, verwendet werden und wann sie eingerichtet werden. Vorrichtungen können jeglichen offenen Kanal für Inhaltsübertragungen verwenden und wieder verwenden, und sind nicht gezwungen, separate Kanäle für jede Anforderung, Seite, Auftrag oder ähnliches zu öffnen.

**[0232]** Um einen ordnungsgemäßen Datenaustausch zwischen Geräten zu ermöglichen, ist es erforderlich, dass alle JIP-Meldungen auf einer 4-Byte-Grenze gesendet und empfangen werden. Diese Einschränkung erzwingt, dass eine Null-Auffüllung am Ende bestimmter JIP-Meldungsanfangsblöcke hinzugefügt wird. Für bestimmte JIP-Meldungsanfangsblöcke variabler Länge, wie z. B. die SurfaceMsg, variiert diese Auffüllgröße. Die JIP-Meldungsfelder selbst müssen jedoch nicht auf 4-Bytes oder geradzahlige Byte-Grenzen innerhalb des Anfangsblocks fallen. Dies zwingt die Implementierung des JIP, sein Packen/Entpacken von Meldungen auf solche Weise auszuführen, um jegliche Probleme zu verhindern, die eine ungerade Ausrichtung darstellen könnte.

**[0233]** Bestimmte Meldungsanfangsblöcke enthalten ASCII-Zeichenfolgenwerte variabler Längen. Es gibt einen maximalen Wert, der für jedes dieser Felder definiert ist, und dieser Wert wird in der entsprechenden Meldungsbeschreibung unten gegeben. Für jedes Meldungsfeld, das einen numerischen Wert enthält, soll dieser Wert in einer Netzwerkbytereihenfolge (d. h. Big-Endian) sein. Schließlich gibt es in jedem Meldungsanfangsblock ein Feld Protocol Version (Protokollversion).

**[0234]** Jede Meldung des JetSend-Wechselwirkungsprotokolls wird nun detailliert spezifiziert.

#### SurfaceMsg (Impress; Abdrucken)

**[0235]** Diese Meldung wird in drei Situationen verwendet: erstens zum Initiieren einer Übertragung einer Oberfläche von der Ausdrucksvorrichtung zu einer anderen Vorrichtung. Zweitens wird sie als die Antwort auf eine SurfaceRequestMsg von einer anderen Vorrichtung verwendet. Drittens wird sie verwendet, um eine SurfaceMsg von einer Ausdrucksvorrichtung abzulehnen. Ein Statusfeld ist gesetzt, um anzuzeigen, welche Interpretation verwendet werden soll.

**[0236]** Wenn diese Meldung entweder zum Initiieren einer Oberflächenübertragung oder als eine Antwort auf eine Oberflächenanforderung verwendet wird, erzeugt die Sendevorrichtung einen Eintrag in ihrer Oberflächentabelle, so dass die Abdrucksvorrichtung über jegliche Änderungen benachrichtigt werden kann.

**[0237]** Beim Empfang der Meldung, wenn der Zielort auswählt, den Abdruck zu akzeptieren, erzeugt er einen Eintrag in seiner Oberflächentabelle, der den Abdruck dem Ausdruck zuordnet. Dies ermöglicht ihm, die Ausdrucksvorrichtung nachfolgend darüber zu informieren, wann er Änderungen an der Oberfläche anfordern möchte oder wann er dies abgeschlossen hat. Wenn der Zielort auswählt, den Abdruck nicht zu akzeptieren, dann sollte einer eine Freigabemeldung zurückschicken, um ihn abzulehnen und keinen Tabelleneintrag erzeugen. Jegliche nachfolgenden Meldungen, die sich auf den „Abdruck“ beziehen, sollten dann ignoriert werden.

**[0238]** Wenn eine Sendevorrichtung eine Freigabemeldung für einen Abdruck empfängt, sollte sie den Eintrag, der sich auf den Abdruck bezieht, aus ihrer Tabelle löschen. Dies stellt sicher, dass die Vorrichtung, die den Abdruck freigegeben hat, keine Meldungen empfängt, die sich auf den Abdruck beziehen.

**[0239]** Es gibt eine kurze Periode zwischen dem Senden eines Abdrucks und dem Empfang einer Freigabemeldung, die denselben ablehnt. Während dieser Periode kann die Ausdrucksvorrichtung erachten, dass der Abdruck existiert. Dies verursacht kein praktisches Problem, da die Frage an dem Empfangsende gelöst wird: die „Abdrucksvorrichtung“ muss die Meldungen ignorieren, die sich auf Abdrucke beziehen, die nicht akzeptiert wurden, oder die nicht mehr bestehen.

**[0240]** Es gibt die nachfolgenden Meldungsfelder:

Feld	Typ	Verwendung
Meldungstyp	unsignierte 8-Bit-Ganzzahl	Meldungstyp
Protokollversion	unsignierte 8-Bit-Ganzzahl	Wechselwirkungsprotokollversionsnummer
Quelloberflächen-identifizierer	unsignierte 32-Bit-Ganzzahl	Identifizierer des Ausdrucks
Quelloberflächen-klasse	unsignierte 32-Bit-Ganzzahl	Ausdrucksklasse
Quelloberflächen-version	unsignierte 32-Bit-Ganzzahl	Aktuelle Versionsnummer der Oberfläche
Quelloberflächen-eigenschaften	unsignierte 32-Bit-Ganzzahl	Eigenschaften der Oberfläche. Eine Bitmaske.
Zieloberflächen-identifizierer	unsignierte 32-Bit-Ganzzahl	Identifizierer des Zielausdrucks

Reserviert	2 Bytes	Reserviert für spätere Verwendung. Muss auf NULL gesetzt sein.
Status	unsignierte 16-Bit-Ganzzahl	Der Status für Abdrucke und Abdrucksanforderungen
Anforderungs- identifizierer	unsignierte 16-Bit-Ganzzahl	Der Anforderungsidentifizierer, wenn dieser Abdruck aus einer Anforderung resultierte
Reserviert	2 Bytes	Reserviert für spätere Verwendung. Muss auf NULL gesetzt sein.
Identifizierer abdrucken	unsignierte 32-Bit-Ganzzahl	Ein eindeutiger Identifizierer für diese Abdrucksmeldung
Quelloberflächen- namenlänge	unsignierte 16-Bit-Ganzzahl	Länge des Ausdrucksnamens in Byte
Quelloberflächen- name	Byte-Sequenz	Ausdrucksname
Zieladresslänge	unsignierte 16-Bit-Ganzzahl	Länge der Zieladresse in Byte
Zieladresse	Sitzungsprotokoll- adresse (Byte- Sequenz)	Adresse der Zielvorrichtung

TABELLE 3: SurfaceMsg-Meldungsfelder

**[0241]** Diese Felder werden nun kurz erörtert, wo es nicht offensichtlich ist oder vorangehend erörtert wurde. Meldungstyp: Der Meldungstyp identifiziert eindeutig diese Meldung von den anderen Meldungen in dem Protokoll. Das Meldungstypfeld für die Abdrucksmeldung sollte auf 0 × 00 gesetzt sein.

**[0242]** Quelloberflächenidentifizierer: Dies ist ein Identifizierer, der durch die Ausdrucksvorrichtung zu der Oberfläche zugeordnet ist, die eindeutig für die Zeit dazwischen ist, wann der erste Abdruck erzeugt wird und der letzte Abdruck beseitigt wird. Dieser Identifizierer wird durch das Protokoll verwendet, um eine Oberfläche eindeutig zu identifizieren. Die Werte 0 und 1 sind reserviert: 0 bedeutet eine NULL-Oberfläche und 1 wird verwendet, um den „Voreingestelltes-Ziel“-Oberflächenausdruck zu bezeichnen (verwendet für die Arbeitsoberfläche, wie oben erörtert wurde).

**[0243]** Quelloberflächenklasse: Dies ist die Klasse der Quelloberfläche. Klasse wird verwendet, um die Verwendung der Oberfläche zu bestimmen. Die legalen Klassenwerte sind in Tabelle 4 unten angezeigt.

Wert	Symbol	Bedeutung
1	OTHER	andere Oberfläche
2	SELF	Selbstoberfläche
3	IN	Innen-Oberfläche
4	STAT	Status-Oberfläche
5	ADDR	Adress-Oberfläche

TABELLE 4: Klassenwerte für Oberflächen

**[0244]** Die Verwendung jeder Klasse wird weiter unten in der Erörterung aus Wechselwirkungstaktiken adressiert. Zukünftige Ausführungsbeispiele (Versionen) des Protokolls können diesem Feld neue Werte hinzufügen. Es ist möglich, dass Vorrichtungen nicht in der Lage sind, spezifische Oberflächenklassen zu handhaben: solche Vorrichtungen können entweder konfiguriert sein, um alle solchen Oberflächen zu ignorieren oder alle Oberflächen dieser Klasse als Oberflächen einer unterschiedlichen Klasse zu behandeln, für die die Vorrichtung konfiguriert ist, um diese zu handhaben.

**[0245]** Quelloberflächenversion: Dies ist die aktuelle Version der Oberfläche. Das Protokoll behält eine Versionsnummer für jede Oberfläche in Verwendung bei. Diese wird jedes Mal aktualisiert, wenn die Ausdrucksvorrichtung die Oberfläche ändert.

**[0246]** Quelloberflächeneigenschaften: Dies sind Eigenschaften der Oberfläche, die abgedruckt wird. Die Werte für dieses Feld und ihre zugeordneten Bedeutungen sind in Tabelle 5 unten ausgeführt.

Wert	Bedeutung
1	Die Ausdrucksvorrichtung antwortet auf SurfaceMsg auf dieser Oberfläche
2	Die Ausdrucksvorrichtung akzeptiert eine Surface-ChangeMsg von einer Ausdrucksvorrichtung

TABELLE 5: Eigenschaften der Oberfläche, abgedruckt bzw. bedruckt mit SurfaceMsg

**[0247]** Dies könnte bei anderen Implementierungen ausgedehnt werden durch Hinzufügen eines Werts von 3, für den die Ausdrucksvorrichtung sowohl auf eine SurfaceMsg antwortet als auch eine SurfaceChangeMsg akzeptiert.

**[0248]** Zieloberflächenidentifizierer: Dieses Feld enthält den Oberflächenidentifizierer für die Zieloberfläche. Wenn dieser Wert auf 1 gesetzt ist, dann wird angenommen, dass das Ziel die Voreingestellte-Ziel-Oberfläche oder die Arbeitsoberfläche des Zielorts ist. Ansonsten muss dieses Feld den Oberflächenidentifizierer aus einem früheren Abdruck einer Zieloberfläche enthalten.

**[0249]** Reserviert: Bytes, reserviert für eine mögliche spätere Verwendung, müssen bei dieser Implementierung auf NULL gesetzt sein.

**[0250]** Status: Dieses Feld identifiziert den Status dieser SurfaceMsg. Die nachfolgenden Werte sind definiert:

Wert	Bedeutung
0	Dies ist eine nicht angeforderte SurfaceMsg
1	Dies ist ein Abdruck ansprechend auf eine SurfaceRequestMsg. Das Anforderungsidentifiziererfeld ist auf den entsprechenden Anforderungsidentifizierer gesetzt
2	Dies ist eine Ablehnung einer vorangehenden SurfaceRequestMsg. Das Anforderungsidentifiziererfeld ist auf den entsprechenden Anforderungsidentifizierer gesetzt

TABELLE 6: Definierte Werte für den SurfaceMsg-Status

**[0251]** Anforderungsidentifizierer: Für eine SurfaceMsg, die ein Ergebnis einer vorangehenden SurfaceRequestMsg ist, ist dieses Feld auf den Anforderungsidentifizierer gesetzt, der in der entsprechenden Oberfläche SurfaceRequestMsg enthalten ist. Für alle anderen Situationen sollte dieses Feld auf 0 gesetzt sein.

**[0252]** Abdrucksidentifizierer: Dies ist ein eindeutiger Identifizierer, der durch die Ausdrucksvorrichtung zugeordnet wird. Er kann verwendet werden, um zwischen unterschiedlichen Abdrucken der selben Oberfläche zu unterscheiden. Es wird darauf hingewiesen, dass dieser Identifizierer nur eindeutig für jeden Abdruck einer Oberfläche sein muss. Abdrucke anderer Oberflächen können den selben Identifizierer verwenden. Somit, während Ausdrucksidentifizierer über alle lokalen Oberflächen eindeutig sind, muss ein Abdrucksidentifizierer nur eindeutig innerhalb des Abdrucksatzes sein, der sich auf eine spezifische lokale Oberfläche bezieht.

**[0253]** Quelloberflächenennamenlänge: Dies ist die Länge in Bytes des mit Null abgeschlossenen ASCII-Zeichenfolgenquelloberflächenennamens (einschließlich dem NULL-Byte). Der Maximalwert für dieses Feld ist 64.

**[0254]** Quelloberflächenname: Dies ist der Oberflächenname der Oberfläche, die abgedruckt wird. Der Name muss eine mit NULL abgeschlossene Sequenz aus ASCII-Zeichen sein.

**[0255]** Zieladresslänge: Dies ist die Länge in Bytes der mit NULL abgeschlossenen Zieladresszeichenfolge (einschließlich dem NULL-Byte). Der Maximalwert für dieses Feld ist 256.

**[0256]** Zieladresse: Die Zieladresse ist die Sitzungsprotokolladresse der Zielvorrichtung. Abdruckoperationen haben immer eine Zielvorrichtung und eine Zieloberfläche. Bei den meisten Transaktionen ist nur eine Quelle und ein Ziel umfasst. Das Protokoll ermöglicht jedoch einer Vorrichtung, einen Abdruck von einer zweiten Vorrichtung auf einen Abdruck von einer dritten Vorrichtung abzudrucken. In diesem Fall kann die dritte Vorrichtung nur durch ihre JetSend-Sitzungsprotokolladresse identifiziert werden. Somit ermöglicht das Protokoll eine ausdrückliche Spezifikation der Zielvorrichtung-Sitzungsprotokolladresse. In dem Fall, in dem die Zieladresse dieselbe ist wie der Zielort der SurfaceMsg, sollte das Adresslängenfeld auf 0 gesetzt sein und keine Zieladresse sollte umfasst sein.

**[0257]** Die Verwendung von SurfaceMsg ansprechend auf andere Meldungen ist in Tabelle 7 unten zusammengefasst.

Ansprechend auf:	Zweck	Erklärung
Keine Meldung	Abdruck erzeugen	Eine Ausdrucksvorrichtung kann einen Abdruck auf einer anderen Vorrichtung erzeugen durch Senden einer nicht angeforderten SurfaceMsg. Die Abdrucksvorrichtung kann dann entweder den Abdruck behalten oder kann ihn ablehnen durch sofortiges Freigeben desselben mit einer Rückantwort SurfaceReleaseMsg.
SurfaceRequestMsg	Abdruck erzeugen	Eine Ausdrucksvorrichtung kann einen Abdruck erzeugen durch Senden einer SurfaceMsg zu einer Abdrucksvorrichtung ansprechend auf eine SurfaceRequestMsg von dieser Abdrucksvorrichtung.
SurfaceRequestMsg	Anforderung nach einem Abdruck ablehnen	Eine Ausdrucksvorrichtung kann eine SurfaceRequestMsg ablehnen durch Antworten mit einer SurfaceMsg, deren Status reflektiert, dass dies eine Ablehnung ist.

TABELLE 7: SurfaceMsg ansprechend auf andere JIP-Meldungen

SurfaceDeleteMsg (Delete; Löschen)

**[0258]** Diese Meldung wird durch eine Ausdrucksvorrichtung verwendet, um Abdrucksvorrichtungen zu benachrichtigen, dass der Ausdruck gelöscht wurde. Wenn eine Ausdrucksvorrichtung einen Ausdruck löscht, muss sie alle Abdrucksvorrichtungen benachrichtigen, die einen Abdruck des Gelöschten halten. Sie sollte ferner die Einträge für den Ausdruck und alle Abdrucke desselben aus ihrer Oberflächentabelle löschen. Sie muss jegliche nachfolgende Meldung ignorieren, die sich auf den Ausdruck oder jeglichen seiner Abdrucke beziehen.

**[0259]** Wenn eine Abdrucksvorrichtung eine Löschmeldung empfängt, sollte sie alle Einträge aus ihrer Oberflächentabelle löschen, die sich auf Abdrucke der gelöschten Oberfläche beziehen. Sie sollte keine Meldungen mehr erzeugen, die sich auf diese Abdrucke beziehen.

**[0260]** Es gibt eine kurze Periode zwischen dem Punkt, an dem die Ausdrucksvorrichtung diese Meldung ausgibt und dem, an dem die Abdrucksvorrichtung dieselbe empfängt und den Abdruck aus ihrer Oberflächentabelle löscht. Die Abdrucksvorrichtung kann daher Meldungen, die sich auf den Ausdruck beziehen, während dieser Periode senden, aber es resultieren keine praktischen Schwierigkeiten, da die Ausdrucksvorrichtung jegliche Meldungen ignoriert, die sich auf Ausdrücke beziehen, die sie gelöscht hat.

**[0261]** Die Felder der Meldung sind in Tabelle 8 unten ausgeführt.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit-Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit-Ganzzahl	Wechselwirkungs- protokollversions- nummer
Gelöschter Aus- drucksidentifi- zierer	Unsignierte 32-Bit-Ganzzahl	Der Identifizierer des gelöschten Ausdrucks

TABELLE 8: Meldungsfelder für SurfaceDeleteMsg

**[0262]** Meldungstyp: Das Meldungstypfeld für die SurfaceDeleteMsg sollte auf 0 × 01 gesetzt sein.

**[0263]** Gelöschter-Ausdruck-Identifizierer: Dies ist der Oberflächenidentifizierer der Oberfläche, die die Ausdrucksvorrichtung löscht. Die Ausdrucksvorrichtung ignoriert jegliche nachfolgenden Meldungen, die diesen Oberflächenidentifizierer enthalten. Da das Protokoll asynchron ist, können Meldungen vorliegen, die sich auf diese Oberfläche beziehen, die immer noch in dem Netz sind.

#### SurfaceReleaseMsg (Unimpress; Abdruck entfernen)

**[0264]** Diese Meldung wird durch eine Abdrucksvorrichtung verwendet, um die Ausdrucksvorrichtung zu benachrichtigen, dass der Abdruck entfernt wurde. Wenn eine Abdrucksvorrichtung einen Abdruck nicht mehr benötigt, löscht sie ihn aus ihrer Oberflächentabelle und sendet der Ausdrucksvorrichtung eine SurfaceReleaseMsg-Meldung. Die Abdrucksvorrichtung muss dann alle Meldungen ignorieren, die sich auf den gelöschten Abdruck beziehen. Es ist möglich, dass eine Vorrichtung mehrere Abdrucke der selben Oberfläche hat: in diesem Fall ignoriert die Abdrucksvorrichtung nur Meldungen, wo sie sich spezifisch auf den entfernten Abdruck beziehen.

**[0265]** Wenn eine Ausdrucksvorrichtung eine solche Meldung empfängt, sollte sie den Eintrag auf ihrer Oberflächentabelle löschen, der sich auf diesen spezifischen Abdruck bezieht. Sie sollte keine Meldungen zu der relevanten Abdrucksvorrichtung mehr senden, die sich auf diesen Abdruck beziehen.

**[0266]** Es besteht eine kurze Periode zwischen dem Punkt, an dem die Abdrucksvorrichtung diese Meldung ausgibt und dem, an dem die Ausdrucksvorrichtung dieselbe empfängt und den Eintrag aus ihrer Oberflächentabelle löscht. Die Ausdrucksvorrichtung kann daher während dieser Periode Meldungen senden, die sich auf den Abdruck beziehen, aber es resultiert keine praktische Schwierigkeit, da die Abdrucksvorrichtung jegliche Meldungen ignoriert, die sich auf Ausdrücke beziehen, die sie entfernt hat.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit- Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit- Ganzzahl	Wechselwirkungsproto- koll-Versionsnummer
Entfernter Aus- drucksidentifizie- rer	Unsignierte 32-Bit- Ganzzahl	Der Identifizierer des entfernten Ausdrucks
Entfernter Ab- drucksidenti- fizierer	Unsignierte 32-Bit- Ganzzahl	Der Abdrucksidentifizie- rer aus der ursprüngli- chen SurfaceMsg, die den Abdruck erzeugt hat.

TABELLE 9: Meldungsfelder für SurfaceReleaseMsg

**[0267]** Meldungstyp: Das Meldungstypfeld für die SurfaceReleaseMsg sollte auf 0 × 02 gesetzt sein.

**[0268]** Entfernter Ausdrucksidentifizierer: Dies ist der Oberflächenidentifizierer für den Ausdruck, von dem ein Abdruck durch die Abdrucksvorrichtung entfernt wurde. Wenn dies der letzte verbleibende Abdruck des Ausdrucks auf der Abdrucksvorrichtung ist, dann ignoriert diese Vorrichtung alle nachfolgenden Meldungen, die diesen Oberflächenidentifizierer betreffen.

**[0269]** Entfernter Abdrucksidentifizierer: Jeder SurfaceMsg ist ein eindeutiger Identifizierer durch die Ausdrucksvorrichtung zugeordnet. Dies ermöglicht, dass Vorrichtungen zwischen mehreren Abdrucken der selben Oberfläche unterscheiden.

**[0270]** Wenn ein Abdruck entfernt wird, ermöglicht die Verwendung des Abdrucksidentifizierers in der SurfaceReleaseMsg, dass die Ausdrucksvorrichtung bestimmt, welcher Abdruck entfernt wurde.

**[0271]** Die Verwendung einer SurfaceReleaseMsg ansprechend auf andere Meldungen ist in Tabelle 10 unten zusammengefasst.

Ansprechend auf:	Zweck	Erklärung
SurfaceMsg	Gibt einen Abdruck frei	Eine Abdrucksvorrichtung kann die Ausdrucksvorrichtung benachrichtigen, dass sie nicht mehr an dem Abdruck interessiert ist. Die Abdrucksvorrichtung kann diese Meldung jederzeit nach dem Empfang der entsprechenden SurfaceMsg senden, die den Abdruck „erzeugt“ hat.
SurfaceMsg	Lehnt einen Abdruck ab	Eine Abdrucksvorrichtung kann einen Abdruck „ablehnen“, durch sofortiges Antworten auf die SurfaceMsg mit einer SurfaceReleaseMsg.

TABELLE 10: SurfaceReleaseMsg als Antwort auf andere Meldungen

SurfaceRequestMsg (SurfaceRequest; Oberflächenanforderung)

**[0272]** Diese Meldung wird durch eine Vorrichtung verwendet, um einen Abdruck von einer anderen Vorrichtung anzufordern. Die Meldung fordert eine Vorrichtung auf, einen Abdruck auf dem Anforderer zu erzeugen. Der Name kann ein gültiger Oberflächenname auf der entfernt liegenden Vorrichtung sein oder nicht. Wenn der Name gültig ist, sollte die entfernt liegende Vorrichtung einen Abdruck auf dem Anforderer erzeugen: wenn der Name nicht gültig ist, sollte die Anforderung abgelehnt werden. Der Zieloberflächenidentifizierer muss ein gültiger Identifizierer für einen Ausdruck sein, von dem die entfernt liegende Vorrichtung einen Abdruck hat: ansonsten wird die Anforderung abgelehnt.

**[0273]** Wenn eine Ausdrucksvorrichtung eine Oberflächenanforderung empfängt, sollte sie wenn nötig die angeforderte Oberfläche erzeugen und die Abdrucksmeldung verwenden, um sie auf die anfordernde Vorrichtung abzu drucken. Alternativ, wenn die Anforderung ungültig ist, sollte die Ausdrucksvorrichtung die Anforderung ablehnen. Der Anforderungsidentifizierer in der Abdrucks- oder Ablehnungs-Meldung muss derselbe sein wie der Anforderungsidentifizierer in der ursprünglichen Anforderungsmeldung.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit- Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit- Ganzzahl	Wechselwirkungsprotokoll- Versionsnummer
Quelloberflächen- namenlänge	Unsignierte 16-Bit- Ganzzahl	Die Länge des nachfolgenden Namens
Quelloberflächen- name	Byte- Sequenz	Der Name des Ausdrucks, für den ein Abdruck angefordert wird
Quelloberflächen- klasse	Unsignierte 32-Bit- Ganzzahl	Die Klasse des Ausdrucks, für den ein Abdruck ange- fordert wird
Zieloberflächen- identifizierer	Unsignierte 32-Bit- Ganzzahl	Der Identifizierer der Zieloberfläche
Anforderungs- identifizierer	Unsignierte 16-Bit- Ganzzahl	Ein eindeutiger Identifi- zierer zum Identifizieren dieser Anforderung
Reserviert	2 Bytes	Reserviert für spätere Verwendung. Muss auf NULL gesetzt sein.

TABELLE 11: Meldungsfelder für SurfaceRequestMsg

**[0274]** Meldungstyp: Das Meldungstypfeld für die SurfaceRequestMsg sollte auf 0 × 03 gesetzt sein.

**[0275]** Quelloberflächennamenlänge: Dies ist die Länge in Bytes des mit NULL abgeschlossenen ASCII-Zeichenfolgenquelloberflächennamens (einschließlich dem NULL-Byte). Der Maximalwert für dieses Feld ist 64.

**[0276]** Quelloberflächenname: Dies ist der Name der Oberfläche, von der die Vorrichtung einen Abdruck möchte. Der Name ist eine mit NULL abgeschlossene ASCII-Zeichenfolge.

**[0277]** Quelloberflächenklasse: Dies ist die Klasse der Oberfläche, von der die Vorrichtung einen Abdruck möchte. Zwei Ausdrücke können den selben Namen haben und durch ihre entsprechenden Klassen unterschieden werden.

**[0278]** Zieloberflächenidentifizierer: Dies ist der Identifizierer der Oberfläche, die als Ziel für die SurfaceMsg oder den Abdruck, der angefordert wird, verwendet werden soll. Die Ausdrucksvorrichtung muss einen Abdruck dieser Oberfläche haben, um in der Lage zu sein, die angeforderte Oberfläche erfolgreich abzdrukken. Dieser Zieloberflächenidentifizierer kann auf die „voreingestellte“ Zieloberfläche (1) gesetzt sein, für die alle Vorrichtungen implizit einen Abdruck haben.

**[0279]** Anforderungsidentifizierer: Dieser Identifizierer ist durch 1 das Protokoll zugeordnet und identifiziert eindeutig diese SurfaceRequestMsg von jeglichen anderen Anforderungen für die selbe Oberfläche.

**[0280]** Verwendung in Beziehung zu anderen JIP-Meldungen ist in Tabelle 12 zusammengefasst.

Ansprechend auf:	Zweck	Erklärung
Keine Meldung	Anforderung eines Abdrucks	Eine Ausdrucksvorrichtung kann einen Abdruck anfordern durch Senden einer SurfaceRequestMsg zu einer anderen Vorrichtung. Die Vorrichtung, die diese Meldung empfängt, kann mit einer SurfaceMsg antworten, die die Anforderung entweder gewährt oder ablehnt.
SurfaceMsg	Anforderung eines Abdrucks einer „Kindoberfläche“	Die Oberflächenbeschreibung eines Abdrucks kann Bezugnahmen auf Teiloberflächen oder „Kindoberflächen“ enthalten. Die Ausdrucksvorrichtung kann durch Senden einer SurfaceRequestMsg zu der Ausdrucksvorrichtung anfordern, dass diese Kindoberflächen abgedruckt werden.

TABELLE 12: SurfaceRequestMsg als Antwort auf andere Meldungen

DescriptionRequestMsg (Description Request; Beschreibungsanforderung)

**[0281]** Diese Meldung wird durch eine Ausdrucksvorrichtung verwendet, um die Beschreibung für eine Oberfläche anzufordern, von der die Vorrichtung einen Abdruck hat. Der Abdrucksidentifizierer muss ein gültiger Abdruck sein, der durch die anfordernde Vorrichtung gehalten wird. Wenn die Ausdrucksvorrichtung eine Anforderung für eine weitere Beschreibung empfängt, sollte sie die Beschreibungsantwort verwenden, um die angeforderte Beschreibung zurückzusenden. Beschreibungsanforderungen können nicht abgelehnt werden, obwohl die resultierende Antwort auch keine Daten enthalten kann.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit-Ganzzahl	Meldungstyp
Protokoll- version	Unsignierte 8-Bit-Ganzzahl	Wechselwirkungsprotokoll- Versionsnummer
Abdrucksober- flächen- identifizierer	Unsignierte 32-Bit-Ganzzahl	Der Abdrucksidentifizie- rer der Oberfläche, für die eine Beschreibung angefordert wird
Anforderungs- identifizierer	Unsignierte 16-Bit-Ganzzahl	Ein eindeutiger Identifi- zierer, der dieser Anfor- derung durch die Vorrich- tung zugeordnet ist, die die DescriptionRequestMsg ausführt
Reserviert	2 Bytes	Reserviert für spätere Verwendung. Muss auf NULL gesetzt sein.

TABELLE 13: Meldungsfelder für DescriptionRequestMsg

**[0282]** Meldungstyp: Das Meldungstypfeld für die DescriptionRequestMsg sollte auf 0 × 04 gesetzt sein.

**[0283]** Abdrucksoberflächenidentifizierer: Dies ist der Abdrucksidentifizierer der Oberfläche, für die eine Beschreibung angefordert wird.

**[0284]** Anforderungsidentifizierer: Dies ist ein eindeutiger Identifizierer, zugeordnet durch die anfordernde Vorrichtung, um diese DescriptionRequestMsg von anderen Anforderungen für Beschreibungen zu identifizieren, die sie ausgeben kann.

Ansprechend auf:	Zweck	Erklärung
SurfaceMsg	Anfordern der Beschreibung der Oberfläche.	Sobald eine Vorrichtung einen Abdruck hat, kann sie durch Senden einer Description-RequestMsg anfordern, dass die Ausdrucksvorrichtung die Beschreibung dieser Oberfläche liefert. Diese Meldung wird allgemein durch die Abdrucksvorrichtung gesendet, in dem Fall, in dem die Beschreibung, die in der ursprünglichen SurfaceMsg enthalten ist, nicht vollständig war. Diese Meldung kann jedoch zu jeder Zeit nach Erzeugung eines Abdrucks gesendet werden.

TABELLE 14: DescriptionRequestMsg als Antwort auf andere Meldungen

DescriptionReplyMsg (DescriptionReply; Beschreibungsantwort)

**[0285]** Diese Meldung wird verwendet, um eine Beschreibung für eine Oberfläche zurückzusenden, ansprechend auf eine DescriptionRequestMsg-Meldung. Das Ergebnis einer schlechten Anforderung sollte keine Daten enthalten, was anzeigt, dass keine Beschreibung mehr für diese spezifische Anforderung vorliegt. Es gibt keine Möglichkeit, eine Beschreibungsanforderung „abzulehnen“, da eine Vorrichtung, die einen Abdruck einer Oberfläche liefert, bereit sein muss, eine vollständige Beschreibung dieser Oberfläche zu liefern. Die Beschreibungsrückantwort muss den Anforderungsidentifizierer aus der Beschreibungsanforderung enthalten, auf die sie eine Antwort ist.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit- Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit- Ganzzahl	Wechselwirkungsprotokoll- Versionsnummer
Abdrucksober- flächen- identifizierer	Unsignierte 32-Bit- Ganzzahl	Der Abdrucksidentifizierer für die Oberfläche, für die eine Beschreibung geliefert wird
Anforderungs- identifizierer	Unsignierte 16-Bit- Ganzzahl	Der Anforderungsidentifizie- rer aus der entsprechenden DescriptionRequestMsg
Reserviert	2-Byte	Reserviert für spätere Verwendung. Muss auf NULL gesetzt sein.

TABELLE 15: Meldungsfelder für DescriptionReplyMsg

**[0286]** Meldungstyp: Das Meldungstypfeld für die Oberflächenanforderungsmeldung sollte auf 0 × 05 gesetzt sein.

**[0287]** Abdrucksoberflächenidentifizierer: Dies ist der Abdrucksidentifizierer der Oberfläche, für die eine Beschreibung zurückgesendet wird.

**[0288]** Anforderungsidentifizierer: Dies ist der Anforderungsidentifizierer aus der DescriptionRequestMsg, auf die diese Meldung eine Antwort ist.

Ansprechend auf:	Zweck	Erklärung
DescriptionRequestMsg	Sendet die Beschreibung einer Oberfläche.	Eine Ausdrucksvorrichtung kann eine DescriptionRequestMsg von einer Vorrichtung empfangen, die einen Abdruck aufweist. Die DescriptionRequestMsg wird allgemein in dem Fall gesendet, in dem die Beschreibung, die in der original SurfaceMsg enthalten ist, nicht vollständig war. Auf den Empfang einer DescriptionRequestMsg hin antwortet die Ausdrucksvorrichtung mit einer DescriptionReplyMsg, die die Beschreibung der Oberfläche enthält, deren Beschreibung angefordert wurde.

TABELLE 16: DescriptionReplyMsg als Antwort auf andere Meldungen

## ContentRequestMsg (Inhaltsanforderung)

**[0289]** Diese Meldung wird durch eine Ausdrucksvorrichtung verwendet, um gewisse Inhaltsdaten von einer Ausdrucksvorrichtung anzufordern. Der Abdrucksidentifizierer muss ein gültiger Abdruck sein, der durch die anfordernde Vorrichtung gehalten wird.

**[0290]** Die Sendevorrichtung kann anfordern, dass die Inhaltsdaten über eine Stromverbindung ausgetauscht werden. Wenn eine Vorrichtung eine solche Inhaltsanforderung empfängt, sollte sie den neuen Strom erzeugen, falls angefordert, und dann den Inhalt auf diesem Strom zurücksenden. Wenn die Empfangsvorrichtung keine Ströme unterstützt, kann sie statt dessen zurückkehren zum Zurücksenden des Inhalts als eine Reihe aus Inhaltsantwortmeldungen. Entsprechend, wenn die anfordernde Vorrichtung die Übertragung über einen Strom anfordert, muss sie vorbereitet sein, um den Inhalt entweder über einen Strom oder als eine Reihe von Inhaltsantwortmeldungen zu empfangen.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit-Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit-Ganzzahl	Wechselwirkungs-Protokollversionsnummer
Quelloberflächen-Identifizierer	Unsignierte 32-Bit-Ganzzahl	Der Ausdrucks-Identifizierer der Oberfläche, für die Inhalt angefordert wird.
Anforderungs-Identifizierer	Unsignierte 16-Bit-Ganzzahl	Ein eindeutiger Identifizierer, der dieser Anforderung durch die anfordernde Vorrichtung zugeordnet ist.
Reserviert	2-Byte	Reserviert für spätere Verwendung. Muss auf NULL eingestellt sein.
Kanal Anforderungs-Identifizierer	Signierte 16-Bit-Ganzzahl	Eine Handhabung zur Verwendung für strombasierte Inhaltsübertragung.

TABELLE 17: Meldungsfelder für ContentRequestMsg

**[0291]** Meldungstyp: Das Meldungstypfeld für die ContentRequestMsg sollte eingestellt sein auf 0 × 06.

**[0292]** Quelloberflächen-Identifizierer: Dies ist der Ausdrucks-Identifizierer der Oberfläche, für die der Inhalt angefordert wird.

**[0293]** Anforderungs-Identifizierer: Dies ist ein eindeutiger Identifizierer, der durch die anfordernde Vorrichtung zugeordnet wird, um diese ContentRequestMsg aus anderen Anforderungen nach Inhalt zu identifizieren, die sie ausgeben kann.

**[0294]** Kanal Anforderungs-Identifizierer: Dies ist der Identifizierer für einen spezifischen Kanal, wie z. B. einen Stromkanal, auf der entfernten Vorrichtung, der verwendet werden sollte, um die Rückantwort auf diese Inhaltsanforderung zu schreiben. Eine positive Zahl zeigt einen gültigen Kanal an, der bereits existiert. Diese Zahl sollte der tatsächliche Kanal-Identifizierer der entfernten Vorrichtung sein. 0 zeigt an, dass die Inhaltsdaten über Inhaltsmeldungen über einen existierenden Meldungskanal gesendet werden sollten. Eine negative Zahl zeigt an, dass der Empfänger die Inhaltsdaten gern über einen neuen Strom senden lassen würde.

**[0295]** Die Verwendung dieser Meldung ansprechend auf andere JIP-Meldungen ist in Tabelle 18 unten zusammengefasst.

Ansprechend auf:	Zweck	Erklärung
SurfaceMsg	Fordert Inhalt von einer Oberfläche an	Sobald eine Vorrichtung einen Abdruck aufweist, kann sie anfordern, dass die Ausdrucksvorrichtung den Inhalt liefert, der durch diese Oberfläche „enthalten“ ist. Die Ausdrucksvorrichtung führt dies aus durch Senden einer ContentRequestMsg zu der Ausdrucksvorrichtung. Diese Meldung kann jederzeit nach dem Empfang einer SurfaceMsg gesendet werden, die einen Abdruck erzeugt.

TABELLE 18: ContentRequestMsg als Antwort auf andere Meldungen

## ContentReplyMsg (Inhaltsdaten)

**[0296]** Diese Meldung wird durch eine Ausdrucksvorrichtung verwendet, um gewisse Inhaltsdaten zu einer Ausdrucksvorrichtung zu senden. Es kann eine Sequenz aus Inhaltsdatenmeldungen ansprechend auf eine einzelne Inhaltsanforderung vorliegen. Diese Meldung wird ebenfalls verwendet, um eine Anforderung nach Inhaltsdaten abzulehnen.

**[0297]** Wenn der Inhaltsanbieter die Daten über einen Strom liefert, setzt er das Strom-Feld auf den Strom-Identifizierer und lässt die Inhaltslänge und Daten leer. Wenn der Inhaltsanbieter keine Ströme unterstützt oder nicht in der Lage ist, einen Strom zu diesem Zweck zu verwenden, stellt er den Strom-Identifizierer auf 0 und überträgt die Inhaltsdaten als eine Reihe aus Inhaltsrückantwortmeldungen: in diesem Fall wird die Inhaltslänge und Daten verwendet, um die Inhalte der Meldung zu übermitteln.

**[0298]** Üblicherweise wird die ContentReplyMsg nur verwendet, um eine Inhaltsanforderung abzulehnen, wenn der Inhaltsanbieter die Anforderung aus internen Gründen nicht mehr erfüllen kann oder wenn die Anforderung ungültig ist.

**[0299]** In allen Fällen muss der Anforderungs-Identifizierer in der Rückantwort auf den Anforderungs-Identifizierer eingestellt sein, der in der Originalanforderung enthalten ist.

**[0300]** Wenn eine Vorrichtung eine Inhaltsrückantwort empfängt, muss sie zuerst bestimmen, ob die Rückantwort eine Ablehnung ist. Wenn ja, muss die Vorrichtung eine Entscheidung im Hinblick auf ihre nächste Aktion treffen. Wenn die Rückantwort keine Ablehnung ist, dann muss die Vorrichtung bestimmen, ob der Inhalt über einen Strom geliefert wird oder nicht. Wenn der Inhalt über einen Strom geliefert wird, sollte er von dort gelesen werden: wenn nicht, dann sollte der Inhalt aus dieser und nachfolgenden Rückantworten gelesen werden, die denselben Anforderungs-Identifizierer aufweisen (da der Transport geordnet ist, sind solche Rückantworten in Reihenfolge und das Statusfeld identifiziert die letzte Rückantwort in der Reihe).

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit- Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit- Ganzzahl	Wechselwirkungs- Protokollversionsnummer
Quelloberflächen- Identifizierer	Unsignierte 32-Bit- Ganzzahl	Der Ausdrucks- Identifizierer der Ober- fläche, für die Inhalt gesendet wird
Anforderungs- Identifizierer	Unsignierte 16-Bit- Ganzzahl	Der Anforderungs- Identifizierer von der entsprechenden ContentRe- questMsg
Reserviert	2-Byte	Reserviert für spätere Verwendung. Muss auf NULL gesetzt sein.
Reserviert	2-Byte	Reserviert für spätere Verwendung. Muss auf Null gesetzt sein.
Status	Unsignierte 16-Bit- Ganzzahl	Der Status dieser Meldung
Kanal- Identifizierer	Signierte 16-Bit- Ganzzahl	Der Kanal-Identifizierer des <i>Empfängers</i> , auf dem die Daten gesendet wer- den.

TABELLE 19: Meldungsfelder für ContentReplyMsg

**[0301]** Meldungstyp: Das Meldungstypfeld für die ContentReplyMsg sollte auf 0 × 07 eingestellt sein.

**[0302]** Quelloberflächen-Identifizierer: Der Ausdrucksidentifizierer der Oberfläche, deren Inhaltsdaten es sind.

**[0303]** Anforderungs-Identifizierer: Dies ist der Anforderungs-Identifizierer aus der ursprünglichen Conten-

tRequestMsg, für die dies eine Rückantwort ist.

**[0304]** Status: Dieses Feld zeigt den Status der Meldung an. Die legalen Werte sind in Tabelle 20 unten ausgeführt.

Wert	Bedeutung
0	Dies ist eine ContentReplyMsg, die Inhaltsdaten enthält - es kommen weitere Blöcke für diese Anforderung
1	Dies ist die letzte ContentReplyMsg mit Daten für diese Anforderungs-ID.
2	Dies ist eine ContentReplyMsg, die die entsprechende Inhaltsanforderung ablehnt. Die Inhaltsdaten sind leer.

TABELLE 20: Legale Werte des Statusfeldes für ContentReplyMsg

**[0305]** Kanal-Identifizierer: Der Kanal-Identifizierer des Empfängers, auf dem die Inhaltsdaten gesendet werden. Eine positive Zahl zeigt einen spezifischen Kanal an, während eine 0 oder eine negative Zahl anzeigt, dass Inhaltsdaten als Meldungen über einen beliebig ausgewählten Meldungskanal gesendet werden.

**[0306]** Verwendung von ContentReplyMsg ansprechend auf andere JIP-Meldungen ist in Tabelle 21 unten ausgeführt.

Ansprechend auf:	Zweck	Erklärung
ContentRequestMsg	Sendet Inhalt zu anfordern-der Vorrichtung.	Eine Abdrucksvorrichtung, die Inhalt anfordert, sendet eine ContentRequestMsg zu der Ausdrucksvorrichtung. Wenn die ContentRequestMsg signalisiert, dass die Daten über einen Meldungskanal kommen sollten, kann die Ausdrucksvorrichtung dann den angeforderten Inhalt senden durch die Verwendung der ContentReplyMsg. Die Ausdrucksvorrichtung kann eine oder mehrere ContentReplyMsgs senden, um die Anforderung zu erfüllen.
ContentRequestMsg	Lehnt die Anforderung nach Inhalt ab	Eine Ausdrucksvorrichtung kann die Anforderung nach Inhalt ablehnen durch Senden einer ContentReplyMsg wenn das Statuswertfeld des Anfangsblocks über den Meldungskanal auf 2 eingestellt ist. Es wird darauf hingewiesen, dass dies das Verfahren ist, das zum Ablehnen aller Inhaltsanforderungen verwendet wird, einschließlich jener, die anfordern, dass die Daten über einen Stromkanal gesendet werden.

TABELLE 21: ContentReplyMsg als Antwort auf andere Meldungen

**[0307]** Wenn das JIP Inhalt über JSP-Stromkanäle sendet, wird ContentReplyMsg nicht zum Senden des Inhalts verwendet. Statt dessen leitet das JIP den Rohinhalt zu dem JSP zum Senden über den spezifizierten Stromkanal.

**[0308]** Bei einer Strominhaltsübertragung, wenn das JIP den letzten Datenblock gesendet hat, der der Inhaltsanforderung entspricht, schließt es dann den Strom, um das Ende der Inhaltsübertragung für diese Anforderung zu signalisieren.

**[0309]** Eine Inhaltsbereitstellung auf einem Meldungskanal ist in [Fig. 13](#) dargestellt. Die Bereitstellung von Inhalt auf einem Meldungskanal wird durch die Empfangsvorrichtung angefordert und wird durch eine Reihe von ContentReplyMsg-Meldungen geliefert, wobei das Statusfeld der Letzten auf 1 eingestellt ist. Eine Inhaltsbereitstellung auf einem Stromkanal ist in [Fig. 14](#) gezeigt. Die Bereitstellung von Inhalt auf einem Stromkanal wird durch die Empfangsvorrichtung angefordert und wird als Rohinhalt über den Stromkanal geliefert. Das Ende des Inhalts wird durch das Sende-Ende signalisiert, das den Strom schließt.

#### SurfaceChangeMsg (Change; Änderung)

**[0310]** Diese Meldung wird zu drei Zwecken verwendet: Erstens durch eine Ausdrucksvorrichtung, um Ausdrucksvorrichtungen über eine Änderung an einer Oberfläche zu benachrichtigen; zweitens durch Ausdrucksvorrichtungen, um eine Änderung an einem Ausdruck anzufordern; und drittens zum Benachrichtigen einer Ausdrucksvorrichtung über eine fehlgeschlagene Anforderung zum Ändern eines Ausdrucks.

**[0311]** Wenn eine Ausdrucksvorrichtung eine Änderung an einem ihrer Ausdrücke ausführt, muss sie alle Ausdrucksvorrichtungen über die Änderung benachrichtigen. Sie tut dies durch Nachschlagen der Abdrucke in ihrer Oberflächentabelle und Senden einer Änderungsmeldung zu jeder Ausdrucksvorrichtung. Wenn eine Vorrichtung mehrere Abdrucke aufweist, ist es nicht notwendig, dass eine Änderungsmeldung für jeden Abdruck gesendet wird: die Ausdrucksvorrichtung behält eine Verknüpfung zwischen dem Ausdruck und den Abdrucken bei.

**[0312]** Wenn eine Ausdrucksvorrichtung eine Änderungsmeldung empfängt, muss sie die Änderung an jedem ihrer Abdrucke des geänderten Ausdrucks ausführen. In einigen Fällen enthält die Änderungsmeldung die Änderung selbst, wohingegen zu anderen Zeiten die Meldung möglicherweise nur eine Benachrichtigung enthält und die Ausdrucksvorrichtung den Inhalt für die geänderte Oberfläche erneut abrufen muss. Wenn die Ausdrucksvorrichtung die Codierung des Inhalts kennt, der von jeder Ausdrucksvorrichtung benötigt wird, dann kann die Ausdrucksvorrichtung konfiguriert sein, um separate Änderungsmeldungen zu liefern, die jeweils den Inhalt in der Form enthalten, der von der relevanten Ausdrucksvorrichtung benötigt wird.

**[0313]** Wenn eine Ausdrucksvorrichtung eine Änderung an einem ihrer Abdrucke ausführen möchte, muss sie die Änderungsmeldung verwenden, um eine Anforderung nach der Änderung zu der Ausdrucksvorrichtung zu senden. Die Ausdrucksvorrichtung muss in der Meldung die Version des Ausdrucks umfassen, auf die die angeforderte Änderung zutrifft.

**[0314]** Beim Empfangen einer Änderungsanforderungsmeldung muss eine Ausdrucksvorrichtung entscheiden, ob die Änderung angenommen oder abgelehnt wird. Die Entscheidung kann auf der Version und dem Wesen der Anforderung basieren. Eine Ausdrucksvorrichtung benachrichtigt den Anforderer dahingehend, ob die Änderung akzeptiert oder abgelehnt wird, durch eine Änderungsmeldung mit entsprechendem eingestellten Status. Die Ausdrucksvorrichtung benachrichtigt ferner alle Ausdrucksvorrichtungen über eine akzeptierte Änderung unter Verwendung der Änderungsmeldung, wie vorangehend beschrieben wurde.

**[0315]** Eine Ausdrucksvorrichtung, die eine erfolgreiche Änderungsanforderung ausgibt, empfängt somit zwei Benachrichtigungen über die Änderung, wobei eine die Änderungsannahme ist (die spezifisch zu derselben gesendet wird), und die andere die Änderungsbenachrichtigung ist, die zu allen Vorrichtungen mit Abdrucken des Ausdrucks gesendet wird. Diese Meldungen können derart identifiziert werden, dass sie sich auf die selbe Änderung beziehen, entweder von dem Anforderungs-Identifizierer oder der Versionsinformation.

Feld	Typ	Verwendung
Meldungstyp	Unsignierte 8-Bit- Ganzzahl	Meldungstyp
Protokollversion	Unsignierte 8-Bit- Ganzzahl	Wechselwirkungs- Protokollversionsnummer
Geänderter Oberflächen- Identifizierer	Unsignierte 32-Bit- Ganzzahl	Der Ausdrucks- Identifizierer der Ober- fläche, die geändert wird
Original- Oberflächenversion	Unsignierte 32-Bit- Ganzzahl	Die Versionsnummer der Oberfläche zu der Zeit, zu der die Änderung ausgeführt wurde
Neue Oberflächen- version	Unsignierte 32-Bit- Ganzzahl	Die Versionsnummer der Oberfläche, nachdem die Änderung ausgeführt wurde
Reserviert	2-Byte	Reserviert für eine  spätere Verwendung. Muss auf NULL einge- stellt sein.
Status	Unsignierte 16-Bit- Ganzzahl	Der Status der Ände- rungsmeldung

TABELLE 22: Meldungsfelder für SurfaceChangeMsg

**[0316]** Meldungstyp: Das Meldungstypfeld für die SurfaceChangeMsg sollte auf 0 × 08 eingestellt sein.

**[0317]** Geänderter Oberflächen-Identifizierer: Dies ist der Ausdrucks-Identifizierer der Oberfläche, die geändert wird.

**[0318]** Original-Oberflächenversion: Jede Oberfläche hat eine Versionsnummer, um einer Ausdrucksvorrichtung zu ermöglichen, zu bestimmen, welche Version einer Oberfläche geändert wird und in welchem Zustand die Abdrucke einer Oberfläche sind. Dieses Feld enthält die Versionsnummer der Oberfläche, bevor diese Än-

derung ausgeübt wurde.

**[0319]** Neue Oberflächenversion: Wenn eine Ausdrucksvorrichtung eine SurfaceChangeMsg ausgibt, um Ausdrucksvorrichtungen über eine Änderung an einer Oberfläche zu benachrichtigen, enthält dieses Feld die neue Versionsnummer der Oberfläche, nachdem die Änderung ausgeübt wurde. Dies ermöglicht, dass die Ausdrucksvorrichtung Versionsnummern überspringt. Ansonsten sollte dieses Feld auf 0 eingestellt sein.

**[0320]** Status: Dieses Feld spezifiziert den Status dieser Änderungsmeldung. Legale Werte und ihre Bedeutungen sind in Tabelle 23 ausgeführt.

Wert	Bedeutung
0	Dies ist eine Benachrichtigung von einer Ausdrucksvorrichtung über eine Änderung an der Oberfläche.
1	Dies ist eine Anforderung von einem Abdruck nach einer Änderung an einer Oberfläche.
2	Dies ist eine Ablehnung einer vorangehenden SurfaceChangeMsg. Die Änderungsanforderung kann aus dem Oberflächen-Identifizierer und der Originalversionsnummer identifiziert werden.

TABELLE 23: Legale Werte des Status für SurfaceChangeMsg

**[0321]** Die Verwendung von SurfaceChangeMsg ansprechend auf andere JIP-Meldungen ist in Tabelle 24 unten zusammengefasst.

Ansprechend auf:	Zweck	Erklärung
Keine Mel- dung/ SurfaceChan- geMsg	Benachrichtigt Ab- drucksvorrichtungen, dass ein Ausdruck geändert wurde.	Wenn eine Ausdrucks- vorrichtung einen lokalen Ausdruck ändert, muss sie alle Abdrucksvorrichtungen benachrichtigen, die gegenwärtig Abdrucke der Oberfläche halten, dass eine Änderung stattfand. Sie tut dies durch Aussenden einer SurfaceChangeMsg zu jeder Abdrucksvor- richtung. (Es wird darauf hingewiesen, dass die Ausdrucksvor- richtung die Oberflä- che aufgrund einer früheren Anforderung nach einer Änderung ändern kann, die in einer SurfaceChangeMsg beschrieben ist, die von einer Abdrucksvor- richtung gesendet wird.)

SurfaceMsg	Benachrichtigt die Ausdrucksvorrichtung, dass eine Änderung angefordert wird.	Die Eigenschaften eines Abdrucks können derart eingestellt sein, dass der Halter des Abdrucks anfordern kann, dass der Originalausdruck geändert wird. Die Ausdrucksvorrichtung tätigt diese Anforderung durch Senden einer SurfaceChangeMsg zu der Ausdrucksvorrichtung.
SurfaceChangeMsg	Lehnt eine Änderungsanforderung einer Ausdrucksvorrichtung ab	Eine Ausdrucksvorrichtung möchte vielleicht die angeforderte Änderung nicht ausführen, die in einer SurfaceChangeMsg beschrieben ist, die sie empfangen hat. Wenn dies der Fall ist, lehnt sie die Änderungsanforderung ab durch Antworten auf die Ausdrucksvorrichtung mit einer eigenen SurfaceChangeMsg, deren Statuswertfeld auf 2 eingestellt wurde.

TABELLE 24: SurfaceChangeMsg als Antwort auf andere Meldungen

**[0322]** Wechselwirkungstaktiken sind für bestimmte allgemeine Verwendungen des JIP definiert. Diese Wechselwirkungstaktiken sind nicht wesentlich für die Funktion des JIP, sind jedoch insofern vorteilhaft, dass sie eine konsistente und gut verständliche Handhabung einer Anzahl von Prozessen ermöglichen, die ein Schlüssel für eine nützliche Anwendung des Informationsaustauschs sind, der durch das JIP möglich gemacht wird.

**[0323]** Die nachfolgenden Taktiken (Policies) sind in der vorliegenden Implementierung definiert.

- Session Policy (Sitzungstaktik) – Wann Sitzungen zwischen Vorrichtungen eingerichtet und getrennt werden.
- Job Policy (Auftragstaktik) – Wie Dokumente zwischen Sendern und Empfängern gesendet werden.
- Self Policy (Selbsttaktik) – Wie globale Informationen über eine Vorrichtung ausgetauscht werden, wie z.

B. Etikett, Icon und Passwörter.

- Status Policy (Statustaktik) – Wie ein Status über eine Vorrichtung und über Aufträge gegeben wird.
- Address Policy (Adresstaktik) – Wie Vorrichtungen mit neuen Zielortadressen programmiert werden.

**[0324]** Nicht alle Taktiken ergeben für alle Vorrichtungen Sinn. Die Erörterung von individuellen Taktiken zeigt die Vorrichtungstypen an, für die sie zutreffen. Taktiken sind im Allgemeinen optional (mit der Ausnahme, dass eine Form einer Session Policy erforderlich ist, um sicherzustellen, dass zwei Vorrichtungen sich verbinden und überhaupt Informationen austauschen können). Wenn zwei Vorrichtungen eine bestimmte Taktik unterstützen, tauschen sie Informationen aus, die diese Taktik implementiert. Wenn nur eine Vorrichtung, die an einer Sitzung beteiligt ist, eine Taktik unterstützt, wird diese Taktik ignoriert.

**[0325]** Eine bevorzugte, minimale Vorrichtungsimplementierung umfasst eine Vorrichtung, die zumindest die Job Policy unterstützt. Es ist allgemein vorteilhaft für Vorrichtungen, zumindest die Self Policy und auch den Vorrichtungsstatustyp der Status Policy zu unterstützen.

**[0326]** Üblicherweise bezieht sich jede Taktik auf eine Gruppe von Oberflächen einer gegebenen Klasse (z. B. sind die Oberflächen, die in der Self Policy beschrieben sind, von der Klasse SELF (Wert: 2). Innerhalb einer gegebenen Taktik werden Oberflächen einer gegebenen Klasse durch Namen oder durch den Kontext unterschieden, in dem sie erscheinen. Solche Oberflächen werden hier „gut bekannte Oberflächen“ genannt, da sie eine bekannte Bedeutung für Vorrichtungen aufweisen, die dieselben austauschen.

**[0327]** Für eine gegebene, gut bekannte Oberfläche definieren die Taktiken Regeln über die Codierung, die sie enthalten muss (z. B. muss die „Etikett“-Oberfläche der Self Policy die vText-Codierung enthalten). Weitere Details solcher Codierungen sind in der Erörterung von E-Material weiter unten gegeben. Jede gut bekannte Oberfläche kann graphisch in zusammenfassender Form dargestellt werden. Eine solche Darstellung für eine Selbst-Oberfläche ist in [Fig. 15](#) gegeben. Die Darstellung zeigt an, dass der Oberflächenname „Self“ sein muss, seine Klasse SELF und die Oberflächenencodierung vPlane. Offensichtlich ist dies einfach eine Zusammenfassung aus Schlüsselattributen: die SurfaceMsg, die zum Abdrucken einer solchen Oberfläche verwendet wird, enthält weit mehr Daten.

**[0328]** Individuelle Wechselwirkungstaktiken werden nun beschrieben.

#### Session Policy (Sitzungstaktik)

**[0329]** Vorrichtungen öffnen und schließen Sitzungen unter Verwendung des JetSend Session Protocols. Die Session Policy spezifiziert, wann Vorrichtungen Sitzungen öffnen und schließen. Das Grundprinzip ist, dass die Vorrichtung, die die Sitzung öffnet, dieselbe auch schließt. Üblicherweise wird eine Sitzung durch eine Vorrichtung geöffnet, die eine Übertragung initiiert.

**[0330]** JetSend kann verwendet werden, um Sitzungen über eine Vielzahl von Transportprotokollen einzurichten. Die angenommene Session Policy ist in hohem Maße abhängig von der darunter liegenden Transportschicht und den Charakteristika ihres Protokolls. Direktverbindungsprotokolle (wie z. B. IrDA und IEEE1284) sind einfach, da Sitzungen durch eine Nähe oder physische Verbindung auf automatische Weise eingerichtet werden und geschlossen werden, wenn die Vorrichtungen nicht mehr in der Nähe zueinander sind. Ähnliche Betrachtungen gelten für kleine Netzwerke oder Busse (wie z. B. IEEE1394 oder SCSI): Alle Vorrichtungen auf dem Bus lassen automatisch eine Sitzung miteinander einrichten, und ein ausdrückliches Öffnen und Schließen der Sitzungen ist unnötig. Auf großen Netzwerken, wie z. B. LANs und WANs, wo IP als Transportprotokoll verwendet wird, ist es notwendig, dass Sitzungen ausdrücklich nach Bedarf geöffnet und geschlossen werden. Die nachfolgend erörterten Betrachtungen gelten hauptsächlich für solche Netzwerke.

**[0331]** Sender Üblicherweise öffnen Sender von Informationen (wie z. B. ein Scanner, der z. B. Informationen zu einem Drucker aussenden möchte) Sitzungen zu Empfängern. Dies kann als Ergebnis einer Benutzeraktion oder aus anderen Gründen sein. Das typische Szenario ist, dass der Sender eine Sitzung mit einem Empfänger öffnet, Anfangsinformationen austauscht, einen Auftrag startet, und wenn Daten übertragen wurden, die Sitzung schließt. Während Sender das Öffnen und Schließen von Sitzungen auf diese Weise steuern, unterliegt dies üblicherweise der Steuerung eines Benutzers. Das Standardsitzungsverhalten ist, dass ein Sender eine Sitzung öffnet, direkt bevor er einen Auftrag beginnt. Die Sitzung wird entweder geschlossen, wenn der Auftrag abgeschlossen ist oder wenn der Sender das Beobachten des Auftragsstatus beendet hat, der durch den Empfänger geliefert wird. In einigen Fällen können Sitzungen offen gelassen werden, um zu ermöglichen, dass der Sender den Vorrichtungsstatus einer entfernten Vorrichtung überwacht.

**[0332]** Die meisten Sender hören auch nach Sitzungen und akzeptieren somit Sitzungen von entfernten Vorrichtungen. Zum Beispiel können entfernte Vorrichtungen eine Verbindung wünschen, um den Status des Senders zu beobachten. In dem Fall einer Zieh-Operation (wird nachfolgend weiter unter Job Policy erörtert) öffnet der Empfänger die Sitzung, fordert ein Dokument an und schließt die Sitzung; folglich müssen Sender, die ermöglichen, dass Vorrichtungen Informationen ziehen, nach Sitzungen hören, um dieses Verhalten zu ermöglichen.

**[0333]** Empfänger Üblicherweise hören Empfänger nach Sitzungen, die durch Sendevorrichtungen initiiert werden. Sie tauschen Informationen mit dem Sender aus, wenn die Sitzung eingerichtet ist, und warten darauf, dass die entfernte Vorrichtung einen Auftrag beginnt und die Sitzung endet. In einigen Fällen wird kein Auftrag gestartet (z. B. überwacht der Sender vielleicht nur den Empfängervorrichtungsstatus), wobei in diesem Fall die Empfangsvorrichtung wartet, bis der Sender entweder einen Auftrag sendet oder die Sitzung schließt. Empfänger, die Informationen ziehen, öffnen und schließen Sitzungen nach Bedarf, wie oben angezeigt ist.

**[0334]** Es gibt möglicherweise Ressourcenverwaltungsfragen bei Annahme von Sitzungen von entfernten Vorrichtungen Empfänger können Einschränkungen betreffend die Anzahl von Sitzungen aufweisen, die akzeptiert werden können. Empfänger müssen somit allgemein sicherstellen, dass ein Zugriff zu anderen Sendern nicht blockiert ist: z. B. könnte die Anzahl von offenen Sitzungen beschränkt sein. Wenn ein Ressourcenverwaltungsproblem besteht, wenn eine neue Vorrichtung anfordert, eine Sitzung zu öffnen, und eine gegenwärtig angeschlossene Vorrichtung nicht dabei ist, einen Auftrag zu senden, kann der Empfänger die existierenden Sitzungen schließen und die Neue initiieren – wenn jedoch die aktuell angeschlossene Vorrichtung einen Auftrag sendet, kann der Empfänger die Sitzungsanforderung von der neuen Vorrichtung ablehnen.

**[0335]** Einige Vorrichtungen wirken sowohl als Sender als auch Empfänger, und hören nach Sitzungen und initiieren Sitzungen, wie oben beschrieben ist. Einige Vorrichtungen können weder senden noch empfangen, liefern oder beobachten aber einfach Status, tauschen Adressen aus oder ähnliches: solche Vorrichtungen öffnen und schließen Sitzungen nach Bedarf durch ihre Operation.

#### Job Policy (Auftragstaktik)

**[0336]** Die Job Policy ist vorteilhaft beim Ermöglichen einer effizienten Sendung eines Dokuments zwischen Vorrichtungen. Die Job Policy unterstützt zwei Arten, einen Auftrag zwischen einem Sender und einem Empfänger zu initiieren: "Drücken", ein vom Sender initiiertes Auftrag, und "Ziehen", ein vom Empfänger initiiertes Auftrag. Abgesehen von der Initiierung eines Auftrags steuert der Empfänger in jedem Fall im Wesentlichen alle nachfolgenden Aspekte der Übertragung und der Sender erfüllt einfach Anforderungen des Empfängers.

**[0337]** Eine Sitzung muss bereits eingerichtet worden sein, bevor die Job Policy verwendet werden kann. Die Job Policy umfasst eine gut bekannte Oberfläche, die die „Innen-Oberfläche“ genannt wird, die der Empfänger zu dem Sender liefert. Egal ob der Auftrag durch „Drücken“ oder „Ziehen“ initiiert wird, alle Empfänger müssen an einem bestimmten Punkt eine Innen-Oberfläche auf Vorrichtungen abdrucken, die eine Verbindung zu denselben herstellen, was signalisiert, dass sie in der Lage sind, Dokumente zu empfangen. Vorrichtungen, die Dokumente zu Empfängern senden möchten, müssen warten, bis sie die Innen-Oberfläche (in-surface) empfangen. Sobald sie die Innen-Oberfläche empfangen haben, verwenden sie die Oberflächenhandhabung der Innen-Oberfläche als ein Ziel für die Oberfläche oberster Ebene des Dokuments, das gesendet werden muss. Diese Operation ist auch als das Starten eines Auftrags bekannt. Der Auftrag wird als beendet betrachtet, wenn der Sender oder der Empfänger die Oberfläche oberster Ebene freigibt, oder wenn die Sitzung zwischen den Zwei geschlossen wird.

**[0338]** [Fig. 17](#) zeigt eine bildliche Darstellung einer Innen-Oberfläche. Die Klasse einer Innen-Oberfläche muss IN (3) sein. Ihr Name und ihre Codierung werden bestimmt durch den Transmitter bzw. Sender der Innen-Oberfläche (der Transmitter der Innen-Oberfläche ist der Empfänger des Dokuments). Die Innen-Oberfläche wird mit einer SurfaceMsg gesendet, und Vorrichtungen, die diese SurfaceMsg verwenden, verwenden die Oberflächen-ID, die darin angezeigt ist als eine Ziel-ID, wenn Aufträge gestartet werden. Vorrichtungen, die Innen-Oberflächen unterstützen, müssen die SurfaceRequestMsg-Meldung unterstützen. Wenn eine Vorrichtung eine Innen-Oberfläche anfordert, ist es nicht notwendig, einen Namen in der Anforderung zu spezifizieren, aber wenn dies der Fall ist, sollte der Besitzer der Innen-Oberfläche die Anforderung nur gewähren, wenn der Name erkannt wird. Die Oberflächenbeschreibung für die Innen-Oberfläche kann alles sein und ist üblicherweise NULL.

**[0339]** Eine Probedokumentübertragung gemäß der Job Policy ist in [Fig. 16](#) angezeigt. Die beteiligten Schritt-

te sind wie folgt:

1. Der Empfänger druckt seine Innen-Oberfläche auf den Sender mit SurfaceMsg ab. Die Oberfläche hat eine Oberflächen-ID von id1, einen Namen „in“ und eine Oberflächenklasse von IN (0 × 03). Das Oberflächeneigenschaftenfeld ist 1, das Ziel ist NoTarget (kein Ziel). Der Sender erkennt dies als eine Innen-Oberfläche, aufgrund der Oberflächenklasse.
2. Dies gilt nur für ein „Ziehen“. Der Empfänger fordert die Oberfläche an, an der er interessiert ist. Die Zieloberflächen-ID der SurfaceRequestMsg muss auf die Innen-Oberfläche eingestellt sein, und der Quelloberflächenname (und/oder Klasse) kann entsprechend dem eingestellt sein, das empfangen werden soll: der Name und/oder die Klasse der angeforderten Oberfläche muss bereits bekannt sein. Wenn ein leerer Name spezifiziert ist, sendet der Empfänger seine Standarddaten.
3. Bei einem „Drücken“ ist dieser Schritt das Starten eines Auftrags durch den Sender durch das Abdrucken der Oberfläche oberster Ebene des Auftrags auf die Innen-Oberfläche des Empfängers. Bei einem „Ziehen“ ist dieser Schritt ebenfalls das Starten eines Auftrags durch das selbe Mittel, aber in diesem Fall ansprechend auf die Oberflächenanforderung von dem Empfänger. Das Oberflächenziel ist auf die Oberflächen-ID der Innen-Oberfläche des Empfängers eingestellt.
4. Bei diesem Beispiel enthält die obere Ebene Bezugnahmen auf andere Oberflächen und der Empfänger startet mit dem Anfordern einer dieser Bezugnahmen nach Namen. Der Empfänger umfasst eine Anforderungs-ID (reg1), so dass er die Rückantwort mit dieser Anforderung in Übereinstimmung bringen kann.
5. Der Sender druckt die angeforderte Oberfläche ab. Seine Oberflächen-ID ist id3.
6. Der Empfänger fordert nun den Inhalt für die Oberfläche an.
7. Eine oder mehrere Inhaltsrückantworten werden durch den Sender gesendet.
8. Der Empfänger gibt die angeforderte Oberfläche frei (dies zeigt an, dass kein Inhalt mehr gebraucht wird).
9. Der Empfänger gibt die Oberfläche oberster Ebene frei, was anzeigt, dass der Auftrag beendet ist. Der Empfänger hat die Daten vielleicht noch nicht verarbeitet, die an diesem Punkt empfangen wurden, aber es steht dem Sender trotzdem frei, die Oberfläche freizugeben.

**[0340]** Das Beispiel oben beschreibt eine Einzelauftragsübertragung. Die Job Policy ermöglicht mehrere fortlaufende Aufträge von einem gegebenen Sender oder zu einem gegebenen Empfänger. Dies ist möglich, da Auftrags Elemente durch ihre Oberflächen-IDs unterschieden werden. Sender können einfach mehrere Oberflächen auf die Innen-Oberfläche von Empfängern abdrucken, mit denen sie verbunden sind.

**[0341]** Es ist eine Vorrichtungswahl, wie mehrere Aufträge gehandhabt werden. Einige Sendevorrichtungen sind nur in der Lage, einen Auftrag gleichzeitig zu senden, und einige Empfangsvorrichtungen sind nur in der Lage, einen einzelnen Auftrag oder eine begrenzte Anzahl von Aufträgen zu verarbeiten. Es ist eine Empfängerwahl, wie neue eingehende Aufträge gehandhabt werden, wenn bereits Aufträge verarbeitet werden. Auswahlmöglichkeiten sind:

das Verarbeiten des neuen Auftrags parallel zu bestehenden Aufträgen;  
den neuen Auftrag in Warteschlange setzen und ihn verarbeiten, wenn es die Ressourcen erlauben (den Auftragsstatus, falls implementiert, auf anstehender Auftrag (JobPending) einstellen); und  
den neuen Auftrag ablehnen (die Oberfläche oberster Ebene freigeben und den Auftragsstatus, falls implementiert, auf Auftrag fehlgeschlagen (JobFailed) einstellen).

**[0342]** Der Auftragsstatus wird nachfolgend weiter unter Status Policy (Statustaktik) erörtert.

**[0343]** Vorrichtungen können entweder ein Empfänger, ein Sender oder beides sein. Vorrichtungen, die weder noch sind, implementieren die Job Policy nicht. Sendevorrichtungen, wie z. B. Scanner, PCs und Kameras, warten auf Innen-Oberflächen von den Empfängern und verwenden dieselben als Ziele zum Starten neuer Aufträge. Empfangsvorrichtungen, wie z. B. Drucker, PCs und TVs senden Innen-Oberflächen zu Vorrichtungen, die mit denselben verbunden sind und warten darauf, dass entfernte Vorrichtungen neue Aufträge starten.

**[0344]** Einige Vorrichtungen wirken als drückende Vorrichtungen – d. h., entweder als das sendende oder empfangende Ende einer vom Sender initiierten Übertragung. Andere Vorrichtungen wirken als ziehende Vorrichtungen. Einige Vorrichtungen wirken als beides.

#### Self Policy (Selbsttaktik)

**[0345]** Die Self Policy wird verwendet, um Vorrichtungen zu ermöglichen, allgemeine Informationen über sich selbst auszutauschen. Die Gesamtstruktur der Selbst-Oberfläche einer Vorrichtung ist in [Fig. 18](#) ausgeführt.

[0346] **Fig. 18** zeigt fünf Oberflächen: eine Oberfläche oberster Ebene **181** (die Selbst-Oberfläche) und eine Anzahl von Kindoberflächen **182, 183, 184, 185**: Die Selbst-Oberfläche ist eine vPlane-Codierung mit Bezugnahmen bzw. Referenzen auf Teil- bzw. Unter-Oberflächen. Jede Unteroberfläche enthält eine spezifische Information, wie nachfolgend weiter erörtert wird. Eine Vorrichtung kann jegliche Untervorrichtungen umfassen, die sie denkt, man sollte sie unterstützen: es ist wünschenswert, dass zumindest die Etikett-Unteroberfläche (label) unterstützt wird. Die Pin-Unteroberfläche **185** ist bei dieser Implementierung ein Kind der Selbst-Oberfläche, muss dies aber nicht sein. Dies wird nachfolgend weiter erörtert.

[0347] Die Selbst-Oberfläche soll als eine „Geschäftskarte“ für Vorrichtungen wirken. Wenn sich zwei Vorrichtungen „treffen“, tauschen sie Selbst-Oberflächen aus, die bestimmte Informationen oberster Ebene enthalten, im Wesentlichen auf die selbe Weise, auf die Personen Geschäftskarten austauschen. Es ist ferner möglich, dass eine Vorrichtung ihre Selbst-Oberfläche zu der Innen-Oberfläche einer anderen Vorrichtung sendet (z. B. zum Betrachten, Speichern oder Drucken). Eine vorteilhafte Implementierung ist, wenn Vorrichtungen die Selbst-Oberfläche oberster Ebene austauschen und dann die Kind-Oberflächen anfordern, die sie benötigen.

[0348] Für die meisten Vorrichtungen ist es angemessen, die Selbsttaktik zu unterstützen. Vorrichtungen, die begrenzte Ressourcen zum Speichern oder Kommunizieren der Selbst-Oberflächenkomponenten aufweisen, können entscheiden, sie nicht zu implementieren. Wenn jedoch die Ressourcen vorhanden sind, sollten alle Vorrichtungen vorzugsweise Adresse, Etikett und Icon-Komponenten implementieren. Abhängig von Vorrichtungsbenutzerschnittstellenfähigkeiten fordern einige Vorrichtungen keine, einige oder alle der Komponenten von einer entfernten Vorrichtung. Zum Beispiel fordern Vorrichtungen mit nur einer Textanzeige nur das Etikett, nicht das Icon, von einer entfernten Vorrichtung. Empfangsvorrichtungen, die teure Ressourcen verwenden, können entscheiden, Unterstützung zu implementieren zum Anfordern der Pin-Komponente von entfernten Vorrichtungen. Sendevorrichtungen, die die Benutzerschnittstellenfähigkeit aufweisen, um PIN-Nummern einzugeben, sollten die Pin-Oberfläche auf ihrer Selbst-Oberfläche implementieren. Ein weiteres, wichtiges praktisches Merkmal ist, dass die Vorrichtungen in der Lage sein sollten, Unteroberflächen der Selbst-Oberfläche zu ignorieren, die sie nicht verstehen: dies gibt die Möglichkeit, die Selbsttaktik in Zukunft durch Hinzufügen weiterer Felder zu entwickeln.

[0349] Die Komponentenoberflächen werden nachfolgend weiter erörtert.

#### Selbst-Oberfläche (**181**)

[0350] Diese ist schematisch in **Fig. 19a** dargestellt. Sie enthält Bezugnahmen auf eine Anzahl von Unteroberflächen, die globale Informationen über eine Vorrichtung enthalten. Der Name der Oberfläche muss „Self“ (Selbst) sein und die Klasse SELF. Der Eigentümer einer Selbst-Oberfläche muss sie abdrucken, wenn neue Sitzungen mit Empfängern beginnen. Sie muss ebenfalls vorbereitet sein, um SurfaceReleaseMsg-Meldungen von Empfängern und SurfaceRequestMsg-Meldungen zu empfangen, so dass Kind-Oberflächen bereitgestellt werden können. Wenn ein Empfänger eine Selbst-Oberfläche anfordert, bei der der Name auf NULL gesetzt ist, aber die Klasse auf SELF gesetzt ist, muss der Sender mit seiner Selbst-Oberfläche antworten (sogar wenn sie keinen NULL-Namen aufweist): dies ist offensichtlich erwünscht im Hinblick auf die frühe Stufe des Kommunikationsprozesses, für die die Selbsttaktik zutrifft.

[0351] Die Oberfläche muss eine vPlane-Codierung sein. Die Selbstkomponenten müssen Kinder an der Front der Ebene sein und müssen beliebig zugreifbar sein (dies wird nachfolgend in der Erörterung von E-Material weiter erörtert). Sie kann einen Bezug auf verschieden benannte Oberflächen enthalten (gegenwärtig „Etikett“, „Icon“ und „Pin“, auch „Adresse“, obwohl andere umfasst sein könnten, z. B. zum Austausch von proprietären Informationen), und diese Unteroberflächen müssen separat angefordert werden. Es kann nur erwartet werden, dass eine Vorrichtung eine Anforderung nach einer Unteroberfläche gewährt, wenn sie diese Unteroberfläche als Kind aufgelistet hat.

[0352] Eine gültige E-Material-Codierung für die Selbst-Oberfläche ist die Nachfolgende. Die Bedeutung dieser Codierung kann bewertet werden Bezug nehmend auf die Erörterung von E-Material und insbesondere der vPlane-Codierung, die zu einem späteren Punkt folgt. Der Empfänger dieser Informationen (in einer SurfaceMsg-Meldung), kann entscheiden, welche Selbst-Oberflächenkomponenten angefordert werden sollen.

<i>Ebene</i>	<i>Attribut</i>	<i>Wert</i>
vPlane	vEncoding	vPlane
	vChildFront	3 vArbitrary
		"label" "address"
		"icon"
vPlane	vSize	(612000, 792000)
vPlane. "label"	vPosition	(90000, 90000)
vPlane. "label"	vAttachment	vFixed
vPlane. "address"	vPosition	(288000, 72000)
vPlane. "address"	vAttachment	vFixed
vPlane. "icon"	vPosition	(900000, 360000)
vPlane. "icon"	vAttachment	vFixed

## Etiketten-Oberfläche (183)

**[0353]** Die Etiketten-Oberfläche ist in [Fig. 19b](#) dargestellt. Ihr Name ist „Etikett“, die Klasse SELF und die Codierung vText. Da Etiketten Unteroberflächen sind, die durch Vorrichtungen angefordert werden, die sie wollen, müssen Vorrichtungen, die diese Oberfläche liefern, auf SurfaceRequestMsg und SurfaceReleaseMsg antworten. Es besteht kein Bedarf, DescriptionRequestMsg, DescriptionReplyMsg, ContentRequestMsg oder ContentReplyMsg zu unterstützen, wenn die Textzeichenfolge eingereiht platziert ist (dies wird später in der E-Material-Beschreibung erörtert). SurfaceChangeMsg kann verwendet werden, um eine entfernte Vorrichtung über eine Änderung eines Etiketts zu informieren (und muss eine vollständig aktualisierte Oberflächenbeschreibung enthalten). Was das Etikett darstellt und ob es vom Benutzer konfigurierbar ist, ist eine Vorrichtungsauswahl. Eine gültige beispielhafte E-Material-Codierung ist unten ausgeführt.

<i>Ebene</i>	<i>Attribut</i>	<i>Wert</i>
	vEncoding	vText
vText	vSize	(7200, 108000)
vText	vSymbolSet	vAscii vUnicode
VText. vAscii	vData	„Soeren`s Printer“
VText. vUnicode	vCodeSet	0x0000..0x01FF
VText. vUnicode	vData	„Soeren's Printer“

## Icon-Oberfläche (184)

**[0354]** Die Icon-Oberfläche, dargestellt in [Fig. 19c](#), ist eine Bilddarstellung der Vorrichtung. Der Name ist „Icon“, die Klasse SELF und die Codierung vImage. Da dies eine Unter Oberfläche ist, gelten ähnliche Betrachtungen wie für die Etiketten-Oberfläche bei der Nachrichtenvermittlung. Die E-Material-Codierung muss bei der vorliegenden Implementierung ein unkomprimiertes 32 × 32-Bit-Farbbild mit eingereihten Daten he aufweisen, obwohl andere Möglichkeiten ebenfalls gegeben werden können.

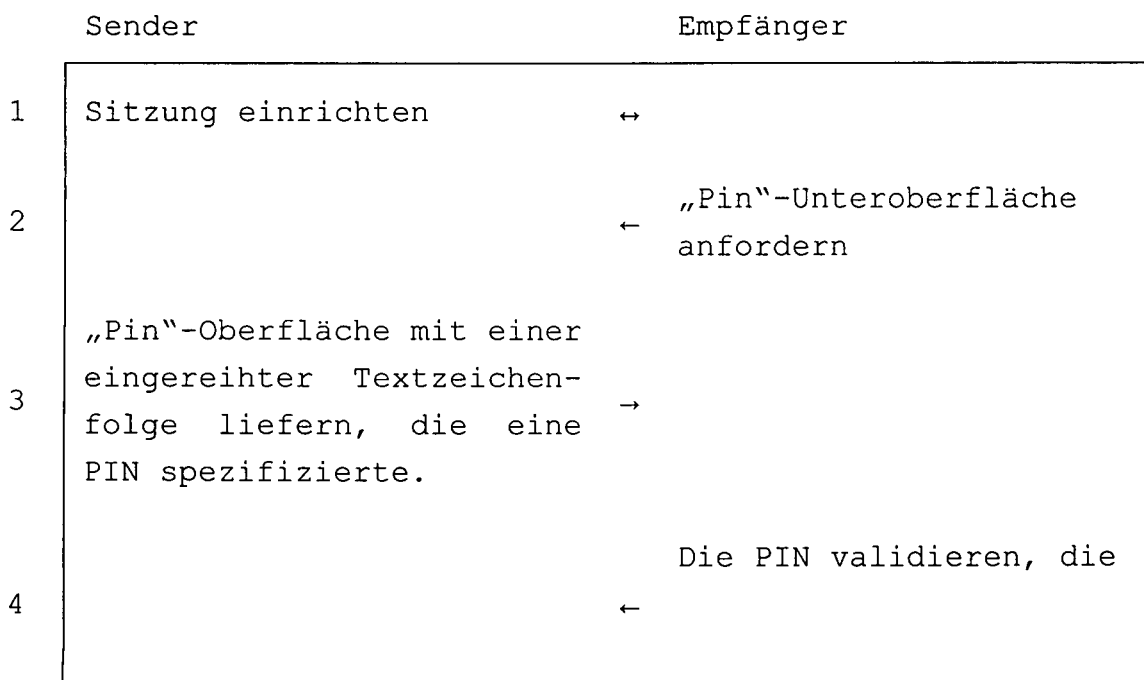
## PIN-Oberfläche (185)

**[0355]** Eine Vorrichtung umfasst eine Bezugnahme auf eine Pin-Oberfläche, wenn die Vorrichtung eine Passwortfunktionalität unterstützt. Die Pin-Oberfläche enthält eine PIN-Nummer (persönliche Identifikationsnummer oder Passwort). Eine verriegelte Vorrichtung fordert die Pin-Oberfläche von einer entfernten Vorrichtung an, die versucht, auf dieselbe zuzugreifen. Die verriegelte Vorrichtung validiert die Inhalte der entfernten Pin-Oberfläche, bevor ein Zugriff auf ihre Ressourcen erlaubt wird. [Fig. 19d](#) zeigt eine Darstellung einer Pin-Oberfläche. Ihr Name ist „Pin“, die Klasse SELF und die Codierung vText.

**[0356]** Wenn ein Pin gesendet wurde, aber die anfordernde Vorrichtung ihn für ungültig erklärt, muss die anfordernde Vorrichtung eine SurfaceReleaseMsg senden, gefolgt von einer weiteren SurfaceRequestMsg, um anzuzeigen, dass die Pin ungültig war. Der Eigentümer der Pin-Oberfläche muss vorsichtig sein, um zu vermeiden, dass er nicht immer und immer wieder die selbe Pin sendet, sondern sollte die zweite Anforderung nach der Pin ablehnen, wenn er keine neue liefern kann. Es ist nicht erforderlich, dass DescriptionRequestMsg-ReplyMsg und ContentRequestMsg-ReplyMsg unterstützt werden, wenn die Pin eingereiht mit der SurfaceMsg-Message platziert wird.

**[0357]** Die Pin-Oberfläche enthält eine Textcodierung in dem ASCII-Symbolsatz. Es ist bei der spezifischen, angenommenen Implementierung erforderlich, dass die Textdaten aus numerischen Werten bestehen (d. h. aus Zeichen im Bereich von „0“ .. „9“ bestehen), da es einfache Vorrichtungen dem Benutzer vielleicht nicht erlauben, alphanumerische Zeichen einzugeben. Vorrichtungen, die PIN-Nummern validieren, müssen dann die Validierung auf numerische Werte einschränken. Um sicherzustellen, dass alle Vorrichtungen in der Lage sind, die längste verfügbare PIN einzugeben, ist es bei der spezifischen angenommenen Implementierung ebenfalls erforderlich, dass die Zeichenfolge auf ein Maximum von 10 Stellen beschränkt ist.

**[0358]** Eine Basisereignissequenz, die sich auf eine PIN-Validierung bezieht, ist in Tabelle 25 unten ausgeführt. Der Empfänger ist verriegelt und fragt den Sender nach einer gültigen PIN. Der Empfänger liefert seine Innen-Oberfläche nur, nachdem der Sender validiert wurde.



		der Sender in der „Pin“-Oberfläche spezifiziert hat. Die „Pin“-Oberfläche freigeben.
5a	←	Wenn gültig, Innen-Oberfläche auf Sender abdrucken und auf Aufträge warten.
5b	←	Wenn ungültig, die „Pin“-Oberfläche wieder anfordern oder die Sitzung schließen.

TABELLE 25: PIN-Validierung unter Verwendung einer Pin-Oberfläche

**[0359]** Es unterliegt der Wahl des Empfängers, wie er sich verhält, wenn eine PIN gültig oder ungültig ist. Wenn sie gültig ist, kann der Empfänger das Öffnen seiner Ressourcen, wie z. B. seiner Innen-Oberfläche und Status-Oberflächen für den Sender starten. Er kann Sitzungen auf Sendern schließen, die keine korrekte PIN innerhalb einer bestimmten Zeitsperre liefern. Es unterliegt ebenfalls der Wahl des Empfängers, ob er den Sender nach einem Passwort ein zweites oder drittes Mal fragt, in dem Fall, dass der Sender eine ungültige PIN liefert. Andere Möglichkeiten zum Anfordern einer PIN können ebenfalls eingesetzt werden: z. B. kann es Vorrichtungen erlaubt werden, anzufordern, dass eine PIN am Start jedes Auftrags geliefert wird oder bei spezifischen Arten von Aufträgen. Der PIN-Mechanismus, der hier beschrieben ist, ist nicht die einzige Möglichkeit zum Validieren entfernter JetSend-Vorrichtungen. Eine Empfangs- oder Sende-) Vorrichtung kann ferner entfernte Vorrichtungen basierend auf ihren Adressen validieren. Zum Beispiel kann eine JetSend-Internetvorrichtung ein IP-Filtern implementieren, so dass nur Vorrichtungen mit bestimmten IP-Adressen verbunden werden können.

#### Adress-Oberfläche (182)

**[0360]** Diese Oberfläche, dargestellt in [Fig. 19e](#), enthält die Adresse der Vorrichtung: Dies ist nützlich zum Weiterleiten zu Drittparteivorrichtungen unter Verwendung der Adress-Taktik, die nachfolgend erörtert wird. Ähnliche Betrachtungen, wie z. B. für Etikett-Oberfläche und Icon-Oberfläche, gelten im Hinblick auf die Nachrichtenübermittlung. Diese Oberfläche ist am nützlichsten in dem Kontext der Adress-Taktik (da offensichtlich eine Vorrichtung, die verbunden ist und in der Lage ist, die Adress-Oberfläche anzufordern, diese Informationen ebenfalls aus den Transportschichten kennt, die zum Verbinden mit der Vorrichtung verwendet werden).

#### Status-Taktik (Status Policy)

**[0361]** Oberflächen der Klasse STATUS werden zum Austauschen von Status-Informationen verwendet. Es gibt zwei Status-Typen: Vorrichtungs-Status und Auftrags-Status: es gibt nur einen Vorrichtungs-Status für jede gegebene Vorrichtung, aber es kann eine Mehrzahl von Auftrags-Status geben. Jeder Auftrags-Status ist einem Auftrag zugeordnet, der unter Verwendung der Auftrags-Taktik gestartet wird.

**[0362]** Ein Status-Anbieter ist hier als eine Vorrichtung definiert, die eine Status-Oberfläche besitzt. Ein Status-Beobachter ist eine Vorrichtung mit einem oder mehreren Abdrucken von Status-Oberflächenkomponenten einer anderen Vorrichtung. Eine Vorrichtung kann sowohl ein Status-Anbieter als auch ein Status-Beobachter sein.

**[0363]** [Fig. 20](#) zeigt die Gesamtstruktur der Status-Oberfläche. Die Oberfläche oberster Ebene **201** ist eine

Ebene mit Bezugnahmen auf Status-Unteroberflächen, die die aktuellen Status-Informationen in Textform enthalten. Für einen gegebenen Vorrichtungs- oder Auftrags-Status besteht eine „t“ und eine „n“ (z. B. „Dev“t- und „Dev“n-) Oberfläche. „t“ zeigt Text-Status an und „n“ zeigt numerischen Status an, und sie sind beide Textcodierungen. Der Text-Status ist eine vom Menschen lesbare Zeichenfolge, die lokalisiert werden kann, wie nachfolgend erörtert wird. Der numerische Status ist eine Zahl, die Erfolg, Fortschritt oder Ausfall-Status anzeigt.

**[0364]** Der Text-Status wird durch den Status-Anbieter definiert und ist spezifisch für die Funktion dieser Vorrichtung. Die numerischen Status-Werte sind ein fester Satz, der vorrichtungs-unabhängig ist und zwischen allen Vorrichtungen gemeinschaftlich verwendet wird. Die Definitionen der numerischen Vorrichtungs- und Auftrags-Status-Oberflächen folgen unten.

**[0365]** Eine Vorrichtung, die Status-Oberflächen empfängt, sollte die Text-Status-Zeichenfolgen dem Benutzer anzeigen, aber den numerischen Status interpretieren und entweder den Benutzer informieren oder automatisch eine entsprechende Aktion ausführen. Wenn z. B. ein Auftrag nicht erfolgreich verarbeitet wurde, kann eine Vorrichtung entscheiden, den Auftrag erneut zu senden, zu piepen, ein rotes Licht blinken zu lassen oder den Benutzer mit einer höher entwickelten Fehlermeldung zu benachrichtigen.

**[0366]** Text-Status-Meldungen können in der Sprache des Benutzers lokalisiert werden. Dies wird erreicht unter Verwendung des Lokalisierungsmechanismus, der beschrieben ist in der mitanhängigen Patentanmeldung Nr. ... mit dem Titel ..., eingereicht beim U.S.-Patentamt am ..., deren Inhalt hierin durch Bezugnahme aufgenommen ist. Eine Lokalisierung, die optional ist, ist nur effektiv, wenn beide Vorrichtungen in einer Kommunikation eine gegebene Sprache und ihren Symbolsatz unterstützen. Eine Unterstützung für Englisch ist in dem ASCII-Symbolsatz erforderlich.

**[0367]** Obwohl es für die meisten Vorrichtungen vorteilhaft ist, einen Vorrichtungs-Status zu implementieren, ist dies nicht erforderlich. Alle Vorrichtungen, die einen Status unterstützen, müssen einen numerischen Vorrichtungs-Status implementieren und die meisten implementieren ebenfalls einen Textvorrichtungs-Status. Nur Empfangsvorrichtungen implementieren einen Auftrags-Status: Sendevorrichtungen sind Beobachter des Auftrags-Status von den Empfängern.

**[0368]** Die individuellen Status-Oberflächenkomponenten werden nachfolgend erörtert.

#### Status-Oberfläche

**[0369]** Diese Oberfläche, gezeigt in [Fig. 21a](#), enthält Bezugnahmen auf Status-Meldungen, die den Gesamt-Status einer Vorrichtung oder den Status von Aufträgen anzeigen. Der Name ist „Status“, Klasse ist STAT (4) und die verwendete Codierung muss vPlane sein.

**[0370]** Status-Anbieter (egal ob von Auftrags-Status oder Vorrichtungs-Status) drucken ihre Status-Oberfläche oberster Ebene auf andere Vorrichtungen ab, die mit derselben verbunden sind, durch Senden einer SurfaceMsg-Meldung. Es liegt nun an der entfernten Vorrichtung (dem Status-Beobachter), die Komponenten der Status-Oberfläche anzufordern, die von Interesse sind. Nur wenn ein Status-Anbieter eine gegebene Unteroberfläche aufweist (z. B. „dev“t), die als Kind in der Status-Oberfläche aufgelistet ist, kann erwartet werden, dass er eine Anforderung nach den Daten für diese Oberfläche gewährt. Der Status-Anbieter muss eine SurfaceChangeMsg für die Status-Oberfläche oberster Ebene senden, wenn eine neue Komponente hinzugefügt oder entfernt wird.

**[0371]** Wenn z. B. ein neuer Auftrag gestartet wird und ein Auftrag-Status erzeugt wird, wird die Status-Oberfläche mit einer neuen Komponente in der Liste aus Komponenten aktualisiert. StatusChangeMsgs müssen nur zu Status-Beobachtern gesendet werden, die die Status-Oberfläche nicht mit einer SurfaceReleaseMsg freigegeben haben.

**[0372]** Die Oberflächencodierung muss eine Ebenencodierung sein, und allgemein gelten die selben Betrachtungen wie für die Selbst-Oberfläche. Es kann jedoch für Status-Anbieter wünschenswert sein, Komponenten so zu positionieren, dass sie zum Aufbereiten geeignet sind, und möglicherweise auch für Status-Anbieter, Nicht-Status-Elemente hinzuzufügen (wie z. B. Überschriften oder Logos), um die Lesbarkeit zu verbessern.

**[0373]** Eine gültige Probecodierung ist die Folgende:

<i>Ebene</i>	<i>Attribut</i>	<i>Wert</i>
	vEncoding	vPlane
vPlane	vChildFront	3 vArbitrary
		„devn“ „devt“ „jobn-44“
vPlane	vSize	(612000, 792000)
vPlane. „devn“	vPosition	(90000, 90000)
vPlane. „devn“	vAttachment	vFixed
vPlane. „devt“	vPosition	(288000, 72000)
vPlane. „devt“	vAttachment	vFixed
vPlane. „jobn-44“	vPosition	(900000, 360000)
vPlane. „jobn-44“	vAttachment	vFixed

#### Devt-Oberfläche (202)

**[0374]** Diese Oberfläche ist in [Fig. 21b](#) gezeigt. Wenn eine Lokalisierung der Textzeichenfolge unterstützt wird, müssen auch Inhaltsanforderungen und Rückantworten unterstützt werden: ansonsten können Informationen eingereicht geliefert werden. Die Textzeichenfolge darf kein Zeilenvorschub- oder ein anderes Steuer-Zeichen enthalten.

**[0375]** Die meisten Vorrichtungen erfordern, dass das Status-Feld aktualisiert wird, wenn sich der Status der Vorrichtung ändert. Dies wird ausgeführt durch Senden der SurfaceChangeMsg zu Vorrichtungen, die vorangehend eines SurfaceMsg für diese Oberfläche empfangen haben. Wenn eine Vorrichtung nicht interessiert ist, weiterhin Aktualisierungen für diese Oberfläche zu empfangen, sendet sie eine SurfaceReleaseMsg.

#### Devn-Oberfläche (203)

**[0376]** Diese Oberfläche ist in [Fig. 21c](#) gezeigt. Ähnliche Nachrichtenvermittlungsanforderungen für die Devt-Oberfläche gelten. Die Oberfläche muss in Text codiert sein und in dem ASCII-Symbolsatz geliefert werden. Ihr muss einer der folgenden Werte gegeben sein:

„1“ – Vorrichtung ist in Ordnung (und bereit, Aufträge zu senden/empfangen).

„2“ – Vorrichtung ist beschäftigt (verarbeitet gerade Aufträge).

„3“ – Vorrichtung benötigt Aufmerksamkeit (ist gegenwärtig nicht aktiv oder benötigt eine Bedienperson, um Verbrauchsgüter nachzuladen).

„4“ – Vorrichtung funktioniert nicht (interner Fehler, Vorrichtung ist nicht in der Lage, ihre Funktion auszuführen).

#### Jobt-Oberfläche (205, 207)

**[0377]** Diese Oberfläche ist in [Fig. 21d](#) gezeigt. Die Anforderungen für diese Oberfläche sind im Allgemeinen ähnlich zu der Devn-Oberfläche. Der wichtigste Unterschied liegt am Namen. Das #-Zeichen in dem Namen muss durch die Abdrucks-ID aus der SurfaceMsg ersetzt werden, die den Auftrag gestartet hat. Dies ist die

Abdrucks-ID der Oberfläche oberster Ebene, die zuerst auf die Innen-Oberfläche der Vorrichtung abgedruckt wurde, die den Auftrag empfängt. Die Sendevorrichtung verwendet diese Zahl zum Verknüpfen des Auftrags-Status mit dem entsprechenden Auftrag. Eine Empfangsvorrichtung könnte mehrere Aufträge aufweisen, die gleichzeitig mit verschiedenen Vorrichtungen ablaufen, und die Abdrucks-ID wird verwendet, um die unterschiedlichen Aufträge und Auftrags-Status zu unterscheiden.

#### Jobn-Oberfläche (204, 206)

**[0378]** Diese Oberfläche ist in [Fig. 21e](#) gezeigt. Anforderungen sind im Allgemeinen ähnlich zu der Jobt-Oberfläche, die Oberfläche muss jedoch als Text codiert sein und in dem ASCII-Symbolsatz geliefert werden. Die Oberfläche muss einen der folgenden Werte enthalten:

- „1" – Auftrag ist abgeschlossen und wurde erfolgreich verarbeitet.
- „2" – Auftrag ist abgeschlossen, wurde aber nicht erfolgreich verarbeitet.
- „3" – Auftrag wird gehalten und wird zu einem späteren Zeitpunkt verarbeitet.
- „4" – Auftrag wird jetzt verarbeitet.
- „5" – Auftrag wird jetzt verarbeitet und 0 % ist durchgeführt.
- „6" – Auftrag wird jetzt verarbeitet und 1 % ist ausgeführt.
- (etc.)
- „105" – Auftrag wird jetzt verarbeitet und 100 % ist ausgeführt.

**[0379]** Wert „4" ist gleich zu den Werten „5" bis „105", außer dass er keine Informationen über den Auftragsfortschritt enthält. Vorrichtungen, die diese Status-Werte empfangen, können dieselben auf jegliche Weise anzeigen, die sie als geeignet empfinden.

#### Adress-Taktik

**[0380]** Die Adress-Taktik ermöglicht es Vorrichtungen, Adressen von Drittparteivorrichtungen auszutauschen. Die Taktik verwendet zwei Oberflächentypen, eine Adressprogrammierungsoberfläche (oder Adress-Innen-Oberfläche) und eine Adresskartenoberfläche. Vorrichtungen, die Adressen von entfernten Vorrichtungen akzeptieren, haben eine Adress-Innen-Oberfläche, wohingegen Vorrichtungen, die Adressen zu anderen Vorrichtungen aussenden, Adresskartenoberflächen aufweisen, die sie auf entfernte Adress-Innen-Oberflächen abdrucken.

**[0381]** Üblicherweise haben nur Vorrichtungen, die Sitzungen initiieren, ein Interesse, mit Adressen von entfernten Vorrichtungen programmiert zu werden. Initiatoren von Sitzungen, die keine angemessenen Benutzerschnittstellenfähigkeiten zum Eingeben von Adressen aufweisen, haben keine vernünftige Möglichkeit, abgesehen davon, die Adresstaktik zu implementieren, um höher entwickelten Vorrichtungen zu ermöglichen, denselben Adressen hinzuzufügen. Sie implementieren die Adress-Innen-Oberfläche.

**[0382]** Vorrichtungen, die Adresskarten empfangen, können im Prinzip mit ihnen tun, was sie möchten. Üblicherweise fügen sie eingehende Karten zu ihrem Adressbuch hinzu. Einfachere Vorrichtungen können Adressen auch weiterleiten, die sie von anderen empfangen. Vorrichtungen, die andere mit Adressen programmieren möchten, implementieren Adresskartenoberflächen.

**[0383]** Eine Adress-Innen-Oberfläche **226** ist in [Fig. 22a](#) gezeigt. Vorrichtungen, die Adressen von entfernten Vorrichtungen akzeptieren, sollten die Adress-Innen-Oberfläche auf jegliche andere Vorrichtung abdrucken, die mit derselben verbunden ist. Vorrichtungen, die Adressen senden möchten, müssen das Abdrucks-Ziel auf den Oberflächen-Identifizierer der Adress-Innen-Oberfläche der entfernten Vorrichtung einstellen. Die Adress-Innen-Oberfläche ist sehr ähnlich zu der Innen-Oberfläche, die bei der Auftrags-Taktik verwendet wird, außer dass Oberflächen, die auf dieselbe abgedruckt sind, unterschiedlich verarbeitet werden. Die Oberflächeneigenschaften für diese Oberfläche müssen auf 1 eingestellt sein, um anzuzeigen, dass sie als Ziel für neue Abdrucke verwendet werden kann.

**[0384]** Eine Adresskartenoberfläche ist in [Fig. 22b](#) gezeigt. Diese Oberfläche **221** ist im Format im Wesentlichen dieselbe wie eine Selbst-Oberfläche, und das Format und Anforderungen für jede Unter-Oberfläche **222**, **223**, **224**, **225** sind ebenfalls wie bei der Selbst-Taktik. Der einzige Unterschied ist, dass das Klassenfeld der Adresskartenoberfläche und ihrer Unter-Oberfläche nicht auf SELF eingestellt sein müssen, sondern jegliche Klasse sein können (einschließlich SELF und OTHER). Adresskartenoberflächen werden auf Adress-Innen-Oberflächen abgedruckt. Offensichtlich muss die Adresskartenoberfläche zumindest die „Addr"-Oberfläche als eine Kindoberfläche aufweisen: andere Unter-Oberflächen sind optional.

**[0385]** Es gibt weitere nützliche Übereinkünfte, die nicht Teil einer spezifischen Wechselwirkungs-Taktik sind. Diese sind wie folgt.

#### Unerwartete Oberflächenmeldungen

**[0386]** Eine Vorrichtung empfängt Oberflächenmeldungen, die sie nicht erwartet hat. Wenn z. B. eine Vorrichtung keinen Träger für Selbst-Oberflächen aufweist, senden andere Vorrichtungen trotzdem eine SurfaceMsg mit einer Selbst-Oberfläche oberster Ebene, direkt nachdem eine Sitzung eingerichtet wurde. Zusätzlich dazu könnte eine Vorrichtung Anforderungen nach Oberflächen erhalten, die sie nicht liefern kann.

**[0387]** Die Antwort auf unerwartete Oberflächenmeldungen ist relativ einfach: sie müssen abgelehnt werden. Die Meldung für die Ablehnung hängt von der Originalmeldung ab:

- SurfaceRequestMsg – muss abgelehnt werden mit einer SurfaceMsg, deren Status auf 3 gesetzt ist (Ablehnung anfordern).
- SurfaceMsg – muss mit einer SurfaceReleaseMsg freigegeben werden.
- ContentRequestMsg – muss mit einer ContentReplyMsg mit dem Status abgelehnt werden, der auf 2 eingestellt ist (Anforderung ablehnen), und mit dem Datenfeld, das auf NULL gesetzt ist.
- SurfaceChangeMsg – nur wenn die Meldung den Status-Wert 1 aufweist (Änderungsanforderung), muss dies mit einer SurfaceChangeMsg mit einem Status-Wert von 2 abgelehnt werden (Änderung ablehnen).
- DescriptionRequestMsg – muss mit einer Description-ReplyMsg mit einem Datenfeld von NULL abgelehnt werden.

**[0388]** Wenn diese Meldungen nicht als eine Antwort auf unerwartete Anforderungen gesendet werden, muss der Sender derselben das Warten auf die erwartete Antwort zeitlich begrenzen. Es ist effizienter, wenn der Empfänger explizit signalisiert, dass keine Rückantwort kommen wird, unter Verwendung der obigen Ablehnungsmechanismen.

#### Oberflächenanforderungen mit leerem Oberflächennamen

**[0389]** Diese Übereinkunft wurde teilweise unter jeder Taktik beschrieben. Wenn Oberflächenanforderungen getätigt werden, spezifiziert der Anforderer einen Namen und eine Klasse. Wenn der Name NULL ist (Länge Feld = 0), zeigt dies keine Anforderung nach einer Oberfläche ohne Namen an, sondern bedeutet, dass die Vorrichtung die voreingestellte Oberfläche innerhalb der spezifizierten Klasse liefern sollte. Zum Beispiel sollte eine Anforderung nach einer Oberfläche innerhalb der Klasse SELF und ein leerer Name eine Rückantwort ergeben, die die voreingestellte Selbst-Oberfläche oberster Ebene enthält.

#### Erfinden vorrichtungsspezifischer Taktiken

**[0390]** Eine Vorrichtung kann ihre eigenen Taktiken erzeugen durch Verwenden von Oberflächen mit der Klasse NOCLASS (1). Eine Vorrichtung kann spezielle Oberflächen unterscheiden durch Betrachten des Oberflächennamens. Vorrichtungen, die nichts von dieser speziellen Taktik wissen, lehnen einfach SurfaceMsgs und Oberflächenanforderungen für die Oberflächen mit diesen speziellen Namen ab.

**[0391]** Das Wesen und Format von E-Material wird nun beschrieben, zusammen mit Beispielen seiner Verwendung für unterschiedliche Dokumenttypen.

**[0392]** Wie vorangehend angezeigt wurde, ist E-Material die Form, in der eine Oberfläche ausgedruckt wird. E-Material weist sowohl Beschreibung als auch Inhalt auf. Die Beschreibung wird durch JetSend verwendet, um den Austausch von Inhalt zu verhandeln. Die Beschreibung erläutert die Attribute der Oberfläche. Diese Attribute bilden üblicherweise eine Hierarchie von Wahlmöglichkeiten, die eine Anzahl von Attributebenen umfassen. Diese Attribute zeigen an, wie Informationen übermittelt werden können: der Inhalt selbst sind die wahrnehmbaren Informationen, die übertragen werden sollen.

**[0393]** Ein erfolgreicher Austausch einer Oberfläche zwischen Geräten erfordert, dass jede Attributebene, die die Oberfläche beschreibt, identifiziert und verarbeitet wird. Der Prozess des Identifizierens und Verarbeitens dieser Ebenen zwischen den Geräten wird Verhandlung genannt. Sobald die vollständige Oberflächenbeschreibung verhandelt wurde, wird der Oberflächeninhalt ausgetauscht.

**[0394]** Der Austausch einer Oberfläche zwischen zwei JetSend-Geräten umfasst Oberflächenwechselwirkungsmeldungen, wie in dem JIP definiert ist. Eine Oberflächenbeschreibung kann ausgeführt werden unter

Verwendung von Beschreibungsanforderungen und Beschreibungsrückantworten. Der Oberflächeninhaltsaustausch wird fertiggestellt unter Verwendung von Inhaltsanforderungen und Inhaltsrückantworten. Unter eingeschränkten Umständen kann Oberflächeninhalt in der Oberflächenbeschreibung umfasst sein; dies wird eingereicherter Inhalt genannt. Bei eingereichertem Inhalt und einer kurzen Beschreibung erfordert ein Austausch vielleicht nur eine einzelne Meldung. Üblicherweise werden aber mehrere Meldungen ausgetauscht.

**[0395]** E-Material wird in der Form von E-Material-Blöcken geliefert. Das Byte-Format für solche Blöcke wird nachfolgend weiter erörtert, aber auf einer Ebene kann ein E-Material-Block in einer Zwei-Spalten-Tabelle dargestellt sein mit Attributnamen in der ersten Spalte und Attributwerten in der zweiten Spalte. Ein Attributname ist ein einzelnes, spezifisches Schlüsselwort, wie z. B. *vResolution* (Resolution = Auflösung) oder *vCompression* (Compression = Komprimierung). Ein Attributwert ist einer oder mehrere Werte, die einem Attribut zugeordnet sind, wie z. B. (300, 300) Pixel pro Zoll oder *vRLE* (run length encoding; Lauflängencodierung). Wenn ein Attributwert als eine Liste aus Werten spezifiziert ist, erscheinen diese als eine durch Abstand getrennte Liste in dem Wertefeld.

**[0396]** Die Attributwertpaare, die in dem E-Material-Block erscheinen, sind an einen Abschnitt einer JetSend-Oberfläche anwendbar. Diese Attributwertpaare können für die gesamte Oberfläche oder für einen spezifischen Teil der Oberfläche gelten. Alle Attribute und einige Werte sind aus einem eingeschränkten Satz vordefinierter Schlüsselwörter gezogen. Diese werden in dieser Beschreibung durch ein Präfix „v“ bezeichnet, wie z. B. *vEncoding* (Encoding = Codierung). Einige Werte sind aus einem Satz aus Datentypen gezogen. Diese werden bei dieser Beschreibung durch ein Präfix „em“ bezeichnet, wie z. B. *emListType* (ListType = Listentyp). In dieser Beschreibung gibt es eine Anzahl von darstellenden Typen, die definiert sind, um die Erörterung der grammatikartigen Struktur von JetSend-Oberflächen zu vereinfachen: diese werden durch entsprechende Identifizierer angezeigt, allgemein in dem Wertefeld.

**[0397]** Bestimmte Attribute sind einem Satz aus Werten zugeordnet, die eine Auswahl bieten. Jeder Wertesatz wird eine Auswahlliste genannt. Auswahllisten erscheinen als eine durch Abstand getrennte Liste in der Wertespalte einer zweispaltigen Tabelle. Zum Beispiel kann ein Attribut *vResolution* eine Auswahlliste aufweisen, die Werte enthält, wie z. B. (300, 300) Pixel pro Zoll oder (150, 150) Pixel pro Zoll. Während des Verhandlungsprozesses nimmt jedes Attribut einen Wert aus der Auswahlliste an. Die Auswahl jedes Werts führt zu der Identifizierung und Verarbeitung jeder Attributebene der Oberflächenbeschreibung. In dem E-Material-Block, gezeigt in Tabelle 26, nehmen die Attribute *vEncoding*, *vColorSpace*, *vPixelDepth*, *vResolution* und *vCompression* Auswahllisten als Werte. Das Attribut *vSize* nicht. Der Einfachheit halber ist es bei dieser Implementierung eine Regel, dass Attributdaten, die Listen sein können, als Listen codiert sein müssen – auch wenn sie nur einen Wert enthalten.

**[0398]** Das bedeutet, dass das Attribut „*vEncoding* = *vText*“ als eine Liste mit einem Element codiert sein muss, *vText*. Dies macht das Interpretieren von Attributdaten etwas einfacher, aber das Zusammensetzen etwas schwerer.

<i>Attribut</i>	<i>Wert</i>
<i>vEncoding</i>	<i>vImage vText vFile</i>
<i>vColorSpace</i>	<i>vGray vSRGE</i>
<i>vSize</i>	(72000, 72000)
<i>vPixelDepth</i>	1 8 24
<i>vResolution</i>	(300, 300) (150, 150)
<i>vCompression</i>	<i>vRLE vNone</i>

TABELLE 26: E-Material-Block-Beispiel

**[0399]** Wenn die Ebenen identifiziert und verarbeitet werden, werden die Attribute, die Auswahllisten annehmen, und die ausgewählten Werte, in einem E-Material-Block aufgezählt. Attribute, die keine Auswahllisten annehmen, werden weggelassen. Dieser E-Material-Block wird ein Entscheidungsblock genannt, da er die Entscheidungen im Hinblick auf die spezifischen Attribute des E-Materials darstellt. Die Beschreibungsanforderung und Inhaltsanforderung umfassen einen E-Material-Block, der die Attribute des E-Materials anzeigt, das angefordert wird. Tabelle 27 zeigt einen möglichen Entscheidungsblock, der aus der obigen Tabelle ausge-

wählt ist.

<i>Codierungs-Hierarchie</i>	<i>Attribut</i>	<i>Wert</i>
""	<i>vEncoding</i>	<i>vImage</i>
<i>vImage</i>	<i>vSize</i>	(576000, 756000)
<i>vImage</i>	<i>vColorSpace</i>	<i>vGray</i>
<i>vImage.vGray</i>	<i>vPixelDepth</i>	1 8
<i>vImage.vGray.1</i>	<i>vResolution</i>	(300, 300)
<i>vImage.vGray.1.(300,300)</i>	<i>vPixels</i>	(2400, 3150)
<i>vImage.vGray.1.(300,300)</i>	<i>vCompression</i>	<i>vRLE</i>
<i>vImage.vGray.8</i>	<i>vResolution</i>	(150, 150)
<i>vImage.vGray.8.(150,150)</i>	<i>vPixels</i>	(1200, 1575)
<i>vImage.vGray.8.(150,150)</i>	<i>vCompression</i>	<i>vRLE</i>

TABELLE 27: Probeentscheidungsblock für Tabelle 26 E-Material-Block

**[0400]** Ein Attributblock ist ein E-Material-Block, der alle Attributwertpaare erläutert, die sich auf eine spezifische Attributebene der Oberflächenbeschreibung beziehen. Ein Oberflächenbeschreibungsblock ist ein geordneter Satz aus Attributblöcken. Ein Oberflächenbeschreibungsblock kann als eine Drei-Spalten-Tabelle mit Attributblöcken dargestellt sein, die die letzten zwei Spalten belegen. Die erste Spalte der Tabelle enthält einen Entscheidungsweg, der an jeden Attributblock anwendbar ist. Ein Entscheidungsweg wird erzeugt unter Verwendung der Werte aus dem Entscheidungsblock. Diese sind miteinander in einer Punktschreibweise verketten, in der selben Reihenfolge wie die Attributblöcke. Somit kann ein Attributblock, der an ein Bild anwendbar ist, durch einen Entscheidungsweg qualifiziert sein, wie z. B. *vImage.vGray.8.(300, 300).vRLE*, was anzeigt, dass es ein lauffängencodiertes Bild ist mit einer Auflösung von 300 × 300 dpi und 8 Bits pro Pixel Grauskala. Diese Schreibweise beschreibt die Zusammensetzung der Oberfläche, was die Codierungshierarchie genannt wird. Die Wurzel dieser Hierarchie wird die Nullebene genannt.

<i>Codierungs-Hierarchie</i>	<i>Attribut</i>	<i>Wert</i>
""	<i>vEncoding</i>	<i>vImage</i>
<i>vImage</i>	<i>vSize</i>	(576000, 756000)
<i>vImage</i>	<i>vColorSpace</i>	<i>vGray</i>
<i>vImage.vGray</i>	<i>vPixelDepth</i>	1 8
<i>vImage.vGray.1</i>	<i>vResolution</i>	(300, 300)
<i>vImage.vGray.1.(300,300)</i>	<i>vPixels</i>	(2400, 3150)
<i>vImage.vGray.1.(300,300)</i>	<i>vCompression</i>	<i>vRLE</i>
<i>vImage.vGray.8</i>	<i>vResolution</i>	(150, 150)
<i>vImage.vGray.8.(150,150)</i>	<i>vPixels</i>	(1200, 1575)
<i>vImage.vGray.8.(150,150)</i>	<i>vCompression</i>	<i>vRLE</i>

TABELLE 28: Probenoberflächenbeschreibungsblock

**[0401]** Zum Beispiel ist in dem Fall, der in Tabelle 28 gezeigt ist, die Nullebene der Codierungs-Hierarchie das Attributwertpaar, das zeigt, dass der Wert des Attributs *vEncoding* *vImage* ist. Die Codierungs-Hierarchie bildet eine baumartige Struktur. Jeder Knoten in dieser Struktur ist eine Attributebene. Auf der nächsten Attributebene zeigt der Attributblock an, dass die Attributwertpaare, die für diese Ebene gelten, sind, dass *vSize* den Wert (576000, 756000) annehmen muss, wohingegen *vColorSpace* den Wert *vGray* annehmen muss. Eine Ebene ist abgeschlossen, wenn keines der Attribute, die sie enthält, Auswahllisten nimmt oder eine weitere

Beschreibung erfordert. Diese Ebene ist keine Abschließende, da vGray selbst ein Attribut aufweist, vPixel-Depth, und dies eine Auswahlwerteliste erfordert. Der Entscheidungsweg gibt den Entscheidungssatz an dem Baum hinunter zu der Ebene, die in dem Codierungs-Hierarchiefeld angegeben ist. Jeder Wert in dem Entscheidungsweg zeigt einen Abstieg zu einer nächsten Ebene hinunter an den Codierungs-Hierarchiebaum an, und der letzte Weg in dem Entscheidungsweg ist der Name der Attributebene. Eine Oberflächenbeschreibung ist als vollständig bekannt, wenn die Ebene in der Codierungs-Hierarchie als abschließend bekannt ist. Diese Hierarchieform ist in [Fig. 33](#) gezeigt. Ebenen, wie z. B. X.A, X.B.M und Y sind als abschließend ersichtlich: keine weitere Beschreibung über diese Ebenen hinaus ist verfügbar.

**[0402]** Ein Dokument besteht aus einer oder mehreren Oberflächen. Jeder Oberfläche kann ein Name gegeben sein, wie z. B. „mySurface“ oder „doc/plane3/image2“. Ein Dokument kann vollständig beschrieben werden unter Verwendung einer vierspaltigen Tabelle, wobei die letzten drei Spalten durch Oberflächenbeschreibungstabellen belegt sind. Die erste Spalte enthält den Namen der Oberfläche, die beschrieben wird. Ein Beispiel einer solchen Tabelle ist in [Fig. 29](#) gezeigt.

Surface	Codierungs-Hierarchie	Attribut	Wert
„Top“	""	vEncoding	vPlane
	vPlane	vChildFront	2 vSequential"A""B"
	vPlane	vSize	(612000, 792000)
	vPlane.A	vPosition	(18000, 18000)
	vPlane.A	vAttachment	vFixed
	vPlane.B	vPosition	(288000, 72000)
	vPlane.B	vAttachment	vFloat

„A“	""	vEncoding	vImage
	vImage	vSize	(576000, 756000)
	vImage	vColorSpace	vGray
	vImage.vGray	vPixelDepth	1 8
	vImage.vGray.1	vResolution	(300, 300)
	vImage.vGray.1.(300,300)	vPixels	(2400, 3150)
	vImage.vGray.1.(300,300)	vCompression	vRLE
	vImage.vGray.8	vResolution	(150, 150)
	vImage.vGray.8.(150,150)	vPixels	(1200, 1575)
	vImage.vGray.8.(150,150)	vCompression	vRLE

„B“	""	vLanguage	„en“
	en	vEncoding	vText
	en.vText	vSize	(72000, 18000)
	en.vText	vSymbol.Set	vAscii
	en.vText	vData	Hello

TABELLE 29: E-Material-Dokumentbeschreibung

**[0403]** Die Oberflächen in einem Dokument sind in einer baumartigen Struktur organisiert. Ein Dokument be-

steht aus Seiten; die Seiten bestehen aus Blöcken aus Text oder Bildern; und so weiter. Bestimmte Attribute einer gegebenen Oberfläche beschreiben die Anordnung von Kind-Oberflächen der Ebene darunter. Diese Attribute verursachen die Oberflächen-Hierarchie. Die Wurzel dieser Hierarchie wird die Oberfläche oberster Ebene genannt.

**[0404]** Einige Oberflächen innerhalb der Oberflächen-Hierarchie haben eine oder mehrere Kind-Oberflächen. Diese Kind-Oberflächen sind in einer oder mehreren geordneten Kindlisten organisiert, die sich für jede Eltern-Oberfläche unterscheiden. Eine Kindliste kann gut gekennzeichnet sein, wie z. B. eine Liste aus Bildern und Textblöcken auf einer Seite einer Seite. Sie ist jedoch vielleicht nicht vollständig zu jeder gegebenen Zeit verfügbar. Es wird ein Stapel aus Seiten betrachtet, die durch einen Mehrseitenscanner eingescannt werden. Während eine Anzahl, Bezeichnung und Inhalt für jede Seite existiert, ist die Kindliste, die die Zusammensetzung des Stapels reflektiert, nicht so ohne weiteres zu jeder gegebenen Zeit verfügbar. Solche Oberflächen werden vielleicht nicht erzeugt, bis sie angefordert werden, oder zu einer anderen zukünftigen Zeit, die für die relevante Vorrichtungsoption geeignet ist. Die Kindliste, wenn sie gekennzeichnet ist, gibt die Namen der Kindoberflächen, die sich auf eine Eltern-Oberfläche beziehen. In der Oberflächenbeschreibungstabelle wird dies eine Referenzliste genannt.

**[0405]** Die formelle Struktur für E-Material-Oberflächenbeschreibungen ist nachfolgend detailliert ausgeführt. E-Material ist in Tabellen spezifiziert, wobei die Elemente jeder Zelle auf grammatikartige Weise behandelt werden, um den Aufbau einer Beschreibung aus Attributen und Werten zu bestimmen. Datentypen niedriger Ebene, wie z. B. vorbestimmte Schlüsselwörter oder mit Null abgeschlossene ASCII-Zeichenfolgen, wären Abschlussymbole in einer Grammatik: hier beziehen sie sich auf Datentypen in einem E-Material-Byteformat (wird nachfolgend weiter erörtert).

**[0406]** E-Material-Schreibweisen-Übereinkünfte sind in Tabelle 30 unten ausgeführt.

<i>Symbol</i>	<i>Interpretation</i>
<code>::=</code>	kann ersetzt werden durch
<code> </code>	Ein Operator, der EXCLUSIVE OR bezeichnet, was bedeutet, einen unter den vertikalen Strichen auszuwählen.
<code>[]</code>	Die Inhalte der Klammern sind optional
<code>{ }</code>	Die Inhalte der Klammern bezeichnen einen Satz, aus dem Auswahlen getroffen werden, wenn die Operatoren, die oben definiert sind, verwendet werden und der Umfang zweideutig ist.

<b>Bold</b>	Jedes Symbol, das in Fettdruck ausgedruckt ist, ist ein bestimmtes Symbol.
<i>Italic</i>	Jegliches Symbol, das in kursiver Schrift ausgedruckt ist, ist ein vordefiniertes Schlüsselwort, definiert für die Verwendung bei diesem Dokument.
<b><i>BoldItalic</i></b>	Jegliches Symbol, das in Fettdruck/Kursivdruck ausgedruckt ist, ist ein explizites, vordefiniertes Schlüsselwort.
null	Ein Entscheidungsweg der Länge Null. Eine Referenz auf die Wurzelebene der Codierungshierarchie.
.	Punktzeichen
#	Nummernzeichen
()	Klammernzeichen
""	Anführungszeichen

TABELLE 30: E-Material-Schreibweisen-Übereinkünfte

[0407] E-Material-Datentypen (entsprechend E-Material-Byteformat-Datentypen, die nachfolgend weiter erörtert werden) sind in Tabelle 31 ausgeführt.

Zuordnung	Beispiel
emIntType ::= signierte 32 Bit Ganzzahl	10
emIntPairType ::= (emIntType, emIntType)	(100, 200)
emRangeType ::= (emIntType..emIntType)	(100..600)
emRangePairType ::= (emIntType..emIntType, emIntType..emIntType)	(100..600, 200..600)
emStringType ::= mit Null abgeschlossene ASCII-Zeichenfolge	„string“
emKeyType ::= 16-Bit-Ganzzahl, die vordefinierte Schlüsselwörter darstellt, die spezifische Attribute und Werte definieren.	<b><i>vImage</i></b>

emListType ::= emKeyType   emIntType	10
emIntPairType   emRangeType   em-	(100,200) (100..600)
RangePairType   emStringType	(100,600,200..600)
[[ emListType]...]	„string“ <b>vImage</b>
emDataType ::= 8-Bit-Binärdaten	0xdeadbeef

TABELLE 31: E-Material-Datentypen

[0408] Attributblöcke sind von dem Typ, der in Tabelle 32 ausgeführt ist.

<i>Zuordnung</i>
<i>Attribute</i> ::= emKeyType
<i>Value</i> ::= emListType   emKeyType
<i>keyword</i> ::= emKeyType

TABELLE 32: Attributblocktypen

[0409] Viele Werte sind als E-Material-Listen dargestellt. „Allgemeine Listen“ sind einfach Werte, die am einfachsten zusammen gehandhabt werden: Die Anzahl der Elemente in der Liste ist variabel. „Auswahllisten“ enthalten individuelle Werte, die während des Verhandlungsprozesses ausgewählt werden. „Referenzlisten“ enthalten Oberflächennamen, die miteinander verknüpft sind, um Seiten oder Dokumente zu erzeugen.

[0410] Allgemeine E-Material-Listenspezifikationstypen sind in Tabelle 33 unten ausgeführt.

<i>Zuordnung</i>	<i>Beispiel</i>
<i>keywordList</i> ::= emKeyType [[emKeyType]...]	<b>vGray vSRGB vRGB</b>
<i>intTripleListType</i> ::= emListType, die ein emIntType-Tripel enthält	255 255 255
<i>rangePairListType</i> ::= emListType von einem oder mehreren emRangeType	(100..600) (200..600) (500..600)

TABELLE 33: Allgemeine E-Material-Listen-Spezifikationstypen

[0411] Die Auswahlliste kann eine große Anzahl von Token-Kombinationen sein, wie definiert wird durch emListType. Legale Kombinationen sind eingeschränkt durch die spezifischen Attribute, mit denen die Auswahlliste gepaart ist. Während individuelle Werte in der Auswahlliste mit dem Attribut übereinstimmen müssen, besteht keine praktische Einschränkung für die Anzahl von Werten in der Liste. Üblicherweise auftretende Spezifikationstypen sind in Tabelle 34 gezeigt.

<i>Zuordnung</i>	<i>Beispiel</i>
<i>selection::=emKeyType emIntType  emIntPairType emStringType</i>	<b>vImage vText vFile</b>
<i>selectionList::=emListType</i>	
<i>keySelectionList::=emKeyType [[emKeyType]...]</i>	<b>vPlane vAssociation</b>
<i>stringSelectionList::=emStringType [[emStringType]...]</i>	„special1“ „spe- cial2“
<i>keyStringSelectionList::={emKeyType  emStringType}[[{emKeyType  emStringType}]...]</i>	<b>vPlane</b> „spezial“
<i>rangeSelectionList::=emRangeType [[emRangeType]...]</i>	(100..600, 200..600) (200..1200, 200..1200)
<i>intPairRangeTypeSelectionList::= {emIntPairType emRangeType} [[{emIntPairType}]...]</i>	(300,300) (100..600, 200..600)
<i>languageSelectionList::= stringSelectionList</i>	„en“ „en-us“ „en-gb“ „en-um“ „fr“ „ge“ „it“ „sp“ „da“ „ja“
<i>encodingSelectionList::= keyStringSelectionList</i>	<b>vPlane vImage</b> „spe- zial“

TABELLE 34: E-Material-Auswahllisten-Spezifikationstypen

**[0412]** Das Attribut vLanguage kann in jeder Codierungs-Hierarchie umfasst sein, wenn eine Bezugnahme auf Sprache und Land erwünscht ist. Dies ist erforderlich für die Codierung vText und optional für andere Codierungen. Das Attribut vLanguage fordert eine spezifische Auswahlliste an. Es identifiziert die Sprachen und optional die Länder, für die die anwendbaren Charakteristika verfügbar sind. Das erwartete Format ist languageCode[countryCode] (sprachCode[länderCode]). Es ist eine Anforderung, dass ein Sprachcode als ein eindeutiger Wert identifiziert ist, sogar wenn der Ländercode dem Sprachcode als einzelner Wert folgen kann. Es ist optional, dass der Ländercode zu einem eindeutigen Sprachcode hinzugefügt werden kann, wenn der Ländercode eine weitere Verdeutlichung zu dem Wert hinzufügen würde.

**[0413]** Eine Codierungsauswahlliste bietet im Allgemeinen nur einen oder mehrere emKeyType-Werte, wie z. B. vImage oder vText. Eine spezielle Codierung kann hinzugefügt werden durch Definieren eines emStringType in der Auswahlliste, der dem Attribut vEncoding zugeordnet ist, und einheitliches Verwenden desselben. Sie wird unabhängig von den Schlüsselwörtern betrachtet, die als emKeyType definiert sind, so dass die emKeyType vAssociation nicht dieselbe ist wie die spezielle Codierung emStringType "vAssociation".

**[0414]** Die Reihenfolge von Kombinationen innerhalb der Auswahlliste zeigt die Präferenz des Sendegeräts an.

**[0415]** Ein Entscheidungsweg wird aus den Werten in einem Entscheidungsblock aufgebaut. Diese Werte werden wiederum aus Auswahllisten entnommen. Die Codierungs-Hierarchie wird im Hinblick auf diese Entscheidungswege ausgedrückt. Die Entscheidungswege werden verwendet, um die Ebene jedes Attributblocks zu identifizieren. Entscheidungswegspezifikationstypen sind in Tabelle 35 unten ausgeführt.

Zuordnung	Beispiel
<code>slKeyword::=emKeyType</code> aus einer <code>selectionList</code> .	<b>vColorSpace</b>
<code>slInt::=emIntType</code> from a <code>selectionList</code> .	8
<code>slPair::=emIntPairType</code> from a <code>selectionList</code> .	(300,300)
<code>slString::=emStringType</code> from a <code>selectionList</code> .	„special“
<code>selectionHierarchy::=</code> <code>slKeyword slInt slPair slString</code>	(100..600), (200..600), (500..600)
<code>decisionPath::="" null </code> <code>selectionHierarchy</code> <code>[[selectionHierarchy]...]</code>	<b>vImage.vGray.8.</b> (300,300). <b>vRLE</b>

TABELLE 35: E-Material-Entscheidungsweg-Spezifikationstypen

**[0416]** Die Null-Ebene liefert einen Eingangspunkt für die Codierungs-Hierarchie einer JetSend-Oberfläche. Auf der Null-Ebene ist der Entscheidungsweg Null. Die Null-Ebene nimmt eine von drei unterschiedlichen Formen an, abhängig von der Situation:

- Die Null-Ebene kann nur das Attribut `vEncoding` enthalten. Dies ist der allgemeinste Typ einer Null-Ebene. Die meisten Oberflächen beginnen mit einer solchen Null-Ebene.
- Die Null-Ebene kann nur das Attribut `vLanguage` enthalten. Das Attribut `vLanguage` ist für Oberflächen `vText` erforderlich. Es wird gelegentlich auch mit anderen Oberflächenkodierungen gesehen.
- Wenn eine Oberflächenbeschreibung unvollständig ist, müssen eine Beschreibungsanforderung und eine Beschreibungsrückantwort ausgeführt werden, um den Austausch der Oberflächenbeschreibung fertigzustellen.

Die Beschreibungsanforderung und Beschreibungsrückantwort müssen zusammen auf jeder Attributebene wirken, um die Oberflächenbeschreibung fertigzustellen. Jene Attribute, die Auswahllisten zusammen mit den Werten annehmen, die bereits ausgewählt sind, erscheinen in dem Entscheidungsblock der Beschreibungsanforderung. Jene Attribute, die Auswahllisten annehmen, die nicht adressiert wurden, erscheinen in dem Entscheidungsweg der Beschreibungsrückantwort. Da die Unterbrechung in der geordneten Liste aus Attributen, die Auswahllisten annehmen, dynamisch eingerichtet ist, kann jede Ebene die Null-Ebene in der Beschreibungsrückantwort werden.

**[0417]** Die Oberflächen-Hierarchie entsteht aus der Verwendung von Referenzlisten. Eine Referenzliste zeigt die Struktur der Kindoberflächen in der Kindliste an, die mit dieser Eltern-Oberfläche verknüpft sind. Wenn eine Kindliste verfügbar ist, entsprechen die Kindoberflächen in dieser Liste direkt der Anzahl `childCount` (Kind-Zählwert), und den Kindoberflächennamen in der Referenzliste. Zu bestimmten Zeiten ist die Kindliste nicht verfügbar, so dass die Referenzliste ein unvollständiges Bild anzeigt.

**[0418]** Die Referenzliste ist eine einfache E-Material-Liste, `emListType`. Am häufigsten nimmt die Liste die Form von `childCount` an, das Schlüsselwort ist `vSequential` oder `vArbitrary`, und eine Liste aus Kindoberflächennamen. Eine Referenzliste kann Elemente auf einer einzelnen Seite (in dem Fall der Codierung `vPlane`) oder andere Seiten innerhalb eines Dokuments (in dem Fall der Codierung `vAssociation`) miteinander verknüpfen. Jede Oberfläche ist ein separat verhandeltes Auftragsselement.

**[0419]** Eine Referenzliste ist nur der Oberflächenkodierung `vPlane` oder `vAssociation` zugeordnet. Die anderen Codierungen (`vImage`, `vText`, `vFile`) können nur Anschlussebenen in der Oberflächen-Hierarchie sein. Eine Codierung `vPlane` kann zwei Kindoberflächenlisten (`vChildFront` für vorne und `vChildBack` für hinten), eine (`vChildFront` oder `vChildBack`) oder keine aufweisen. Eine Codierung `vAssociation` muss genau eine (`vChild`)

Kindoberflächenliste aufweisen. E-Material-Referenzlisten-Spezifikationstypen sind in Tabelle 36 ausgeführt.

<b>Zuordnung</b>	<b>Beispiel</b>
<i>childCount</i> ::= <i>emIntType</i>	1
<i>reference</i> ::= <i>emStringType</i>	„A“
<i>referenceList</i> ::= <i>reference</i> [[ <i>reference</i> ]...]	M3 M4 M7
<i>shortReferenceList</i> ::= <i>childCount</i> { <i>vArbitrary</i>   <i>vSequential</i> } <i>referenceList</i> Die positive ganze Zahl <i>childCount</i> stimmt mit der Anzahl von Referenzen in der <i>referenceList</i> überein.	3 <i>vArbitrary</i> M3 M4 M7
<i>longReferenceList</i> ::= <i>childCount</i> { <i>vArbitrary</i>   <i>vSequential</i> } Der <i>childCount</i> ist ein großer positiver Wert.	207 <i>vSequential</i>
<i>unknownReferenceList</i> ::= <i>childCount</i> { <i>vArbitrary</i>   <i>vSequential</i> } Der <i>childCount</i> ist ein negativer Wert.	-3 <i>vSequential</i>
<i>zeroReferenceList</i> ::= 0	0
<i>anyReferenceList</i> ::= <i>shortReferenceList</i>   <i>longReferenceList</i>   <i>unknownReferenceList</i>   <i>zeroReferenceList</i>	
<i>planeReferenceList</i> ::= <i>referenceList</i>   <i>vFrontFirst</i>   <i>vFrontLast</i>   <i>vBackFirst</i>   <i>vBackLast</i>	<i>vFrontLast</i>
<i>associationReferenceList</i> ::= <i>referenceList</i>   <i>vFirst</i>   <i>vLast</i>	M3 M4 M7

TABELLE 36: E-Material-Referenzlisten-Spezifikationstypen

**[0420]** Der Wert *childCount* kann negativ, Null oder positiv sein. Kleine positive Werte für *childCount* umfassen eine entsprechende Liste aus Referenzen, die die vollen Inhalte der Kindliste aufzählt. Große positive Werte von *childCount* zeigen an, dass die Kindliste bekannt ist, aber die Referenzliste zu lang ist, um umfasst zu sein. Andere *childCount*-Werte zeigen an, dass die Kindliste unbekannt oder von Null-Länge ist. In jedem Fall sind keine Referenzen umfasst.

**[0421]** Die Schlüsselwörter *vSequential* und *vArbitrary* werden verwendet, um anzuzeigen, ob Einschränkungen auf die Reihenfolge vorliegen, in der das Sendegerät Oberflächen senden kann. Dies ermöglicht, dass das Empfangsgerät vollständig die Reihe von angeforderten Oberflächen steuert. Es lässt das Empfangsgerät wissen, in welcher Reihenfolge Informationen, sowohl Inhalt als auch Kindoberflächen, angefordert werden können. Inhaltsinformationen über ein Kind können eingereiht sein oder die Inhaltsinformationen können für jeweils gleichzeitig ein oder mehrere Kinder angefordert werden; die Oberflächenbeschreibung eines Kindes muss jedoch eine nach der anderen angefordert werden.

**[0422]** Der Wert *vArbitrary* zeigt an, dass das Sendegerät auf Inhaltsinformationen zugreifen und dieselben senden kann, und die Kindoberflächen auf den Empfänger in jeder Reihenfolge abdrucken kann, die angefordert wird. Der Wert *vSequential* bedeutet, dass die Kinder aufgrund von Einschränkungen der Sendevorrich-

tung nur in einer vorbestimmten Reihenfolge angefordert werden können.

**[0423]** Der Rest der Referenzliste gibt Namen anderer Oberflächen. Diese Namen sind vorhanden, wenn die vollständige Kindliste bekannt und relativ kurz ist. Wenn Referenzen umfasst sind, gilt die nachfolgende Interpretation:

- Die Referenznamen müssen eindeutig sein.
- Das # (Nummernzeichen) ist ein reserviertes Symbol.
- Die Referenzen zeigen die Kinder an, die auf der Ebene aufbereitet werden sollen.
- Die Reihenfolge der Kinder ist die Reihenfolge (in der Z-Richtung – wird nachfolgend Bezug nehmend auf vPlane erörtert), in der sie aufbereitet werden sollten.
- Wenn Referenzen aufgelistet sind, dann muss die Anzahl von Referenzen mit der Anzahl von Kindern übereinstimmen, die unter dem Wert childCount spezifiziert sind. Eine Teilaufistung von Referenzen ist nicht zulässig. Wenn nicht alles aufgelistet werden kann oder die Referenzen unbekannt sind, dann müssen sie durch Mechanismen „entdeckt werden“, die in der Inhaltsanforderung verfügbar sind.

**[0424]** Es ist möglich, dass die Inhaltsinformationen für einige Kinder eingereicht sein können. Dies ist nur erlaubt, wenn ein Wert referenceList mit dem Attribut vChildFront, vChildBack oder vChild umfasst ist. Für Inhalt, der nicht eingereicht ist, wird eine Inhaltsanforderung verwendet.

**[0425]** Um sicherzustellen, dass JetSend-Geräte unter allen Umständen kommunizieren, müssen spezifische Entscheidungswege vorhanden sein. Diese werden default encodings (voreingestellte Codierungen) genannt. Sendegeräte müssen in der Lage sein, E-Material mit den Attributen zu erzeugen, die durch diese voreingestellten Entscheidungswege beschrieben werden. Auf ähnliche Weise müssen Empfangsgeräte in der Lage sein, E-Material mit diesen Charakteristika zu interpretieren. Ein Beispiel ist vImage.vGray.1.(300, 300).vRLE, was eine voreingestellte Codierung für vImage ist.

**[0426]** Voreingestellte Codierungen sind vorhanden, um sicherzustellen, dass Geräte E-Material unter normalen Bedingungen austauschen. Diese Codierungen sind die niedrigsten, gemeinsamen Charakteristika. Base encodings (Basiscodierungen) sind empfohlene Entscheidungswege, die ermöglichen, dass Geräte E-Material mit höherem Grad an Wiedergabetreue austauschen. Andere Attribute und Werte werden als optional betrachtet. Beispiele von Basiscodierungen für vImage sind vImage.vGray.8.(150, 150).vRLE, vImage.vGray.8.(150, 150).vNone, und vImage.vSRGB.24.(150,150).vNone.

**[0427]** Die voreingestellten Codierungen sind Entscheidungswege, wobei jedes Element derselben aus einer Auswahlliste kommt. Ein Attribut ist jeder Auswahlliste zugeordnet. Jedes der Attribute, dessen Werte eine voreingestellte Codierung bilden, muss in der Oberflächenbeschreibung vorhanden sein.

**[0428]** Codierungen für unterschiedliche Oberflächentypen werden nun erörtert und Beispiele gegeben.

#### vImage

**[0429]** Ein statisches Bild wird allgemein als Rasterinformationen betrachtet, eine Bitmap, die, wenn sie aufbereitet wird, als ein Bild betrachtet wird. Eine Abbildung kann als Punkte auf einer Seite aufbereitet werden, obwohl diese Punkte visuell als Text, Tabellen, Bilder, Zeichen usw. erscheinen. Eine Abbildung kann auch als Pixel auf einem Computerbildschirm aufbereitet werden. Eine Abbildung kann Pixel aus einer Kamera, einem Scanner oder einer ähnlichen Vorrichtung sein. Bilder werden allgemein nach Farbraum, Pixeltiefe, Auflösung und Komprimierungstechnik kategorisiert. Die Entscheidungswege innerhalb der Codierungs-Hierarchie werden verwendet, um diese Informationen zu handhaben.

**[0430]** Eine Hierarchie aus Attributen für vImage ist in [Fig. 23](#) gezeigt. Die Bedeutung jedes unterschiedlichen gezeigten Attributs wird kurz erörtert:

#### vLanguage:

**[0431]** Das Attribut vLanguage kann umfasst sein, wenn Referenz auf Sprache und Land erwünscht ist. Ein voreingestellter Sprachwert ist nicht spezifiziert.

#### vEncoding:

**[0432]** Dieser Abschnitt definiert spezifisch die Codierung vImage. Die anderen Codierungen, vText, vFile,

vPlane, vAssociation und alle eindeutigen emStringType-Codierungen können ebenfalls vorliegen. Ein vImage ist definiert als eine Sammlung aus Bits, die ein rasterisiertes, statisches Bild darstellen. Die Bildzusammensetzung ist durch die Attribute dieser Codierung definiert.

vSize:

**[0433]** Bezeichnet die Größe des Bildes relativ zu der Anzeige-Oberfläche. Der erste Wert ist die Abmessung entlang der X-Achse und der zweite Wert ist die Abmessung entlang der Y-Achse basierend auf 72.000 Einheiten pro Zoll. Die Empfangsvorrichtung sollte über die Anzeigeregion Bescheid wissen, die sie verwendet, um das Bild aufzubereiten, die Informationen über relative Größe und ihre eigenen Fähigkeiten. Dieses Attribut gibt der Empfangsvorrichtung Informationen, die beim Entscheiden über die beste Art und Weise verwendet werden können, die Bilddaten zu empfangen, wobei Bildgröße und Ressourcen bekannt sind, die zum Aufbereiten derselben erforderlich sein können.

vOpacity:

**[0434]** Bezeichnet durch einen unsignierten ganzzahligen Wert im Bereich von 0–255. Ein Wert von 0 wird als vollständig transparent interpretiert; 255 wird als vollständig opak interpretiert. Wenn vOpacity weggelassen ist, ist der Standardwert 255.

vColorSpace:

**[0435]** Wird verwendet, um die Farbräume anzubieten, die für das Bild verfügbar sind, das dieser Oberflächenbeschreibung zugeordnet ist. Voreinstellung ist vGray (Grauskala), Basis ist vSRGB und optional sind vRGB, vCIELAB, vCMY, vYCC, vYUV und vXYZ, wobei in jedem Fall der Name der Codierung einen relevanten (bekannten) Farbraum anzeigt; andere Farbräume könnten natürlich als optionale Codierungen bereitgestellt werden.

vPixelDepth:

**[0436]** Wird verwendet, um die Pixeltiefencodierung für den ausgewählten Farbraum anzubieten. Die verfügbaren Werte sind die Anzahl von Bits pro Pixel, die den Farbwert für dieses Pixel identifizieren: geeignete Voreinstellung und Basiscodierungen werden für die unterschiedlichen Farbräume bestimmt (z. B. 1 als Voreinstellung und 8 als Basis für vGray).

vResolution:

**[0437]** Bietet eine oder mehrere „Echtwelt“-Auflösungen, die zum Codieren der Pixeldaten der zugeordneten Bilder verwendet werden können.

vPixels:

**[0438]** Zeigt die Bildbreite und Bildhöhe in Pixeln an, spezifiziert bei der Auflösung, die durch den Attributwert vResolution angezeigt wird.

vCompression:

**[0439]** Stellt Kompressionsschemata dar, die mit verschiedenen ausgewählten Farbräumen verfügbar sind. In jedem Fall ist vNone eine Option, die keine Komprimierung bedeutet. Komprimierung ist optional, aber es ist vorteilhaft, wenn Standardkomprimierungsschemata als Optionen geliefert werden (z. B. vRLE, vTIFF, vJPEG). Wo eine Anzahl von Implementierungsoptionen für ein Komprimierungsschema ausgewählt werden kann, kann jede Implementierung, die mit dem E-Material-Format konsistent ist, angenommen werden. Ein Fachmann ist sich über Standardreferenzquellen für jedes erwünschte Komprimierungsschema bewusst (wie z. B. ISO 10918-1 JPEG Draft International Standard und CCITT Recommendation T.81 für JPEG).

vEndian:

**[0440]** Dies ist optional, kann aber umfasst sein, wenn eine Bezugnahme auf Endian erwünscht ist (Voreinstellung ist Big Endian).

**[0441]** Es ist möglich, dass die Anzahl von Optionen, die zum Darstellen einer gegebenen Oberfläche verfügbar sind, die verfügbare Beschreibungsgröße überschreiten. Wenn dies auftritt, muss die Beschreibung zwischen dem (anfänglichen) Oberflächenbeschreibungsblock und nachfolgenden Oberflächenbeschreibungsrückantworten aufgespaltet werden. Eine Beschreibungsanforderung kann durch das Empfangsgerät ausgegeben werden, um einen Austausch zu initiieren.

**[0442]** Das Empfangsgerät weiß, dass eine weitere Beschreibung verfügbar ist, wenn eine oder mehrere Auswahlen aus einer Auswahlliste nicht weiter in einem nachfolgenden Entscheidungsweg beschrieben sind. Beim Extrahieren der Informationen aus dem Beschreibungsblock kann sich herausstellen, dass keine weiteren Informationen unter einem bestimmten Punkt in dem Entscheidungsbaum vorliegen, der verfolgt wurde. Nach dem Herausfinden, dass keine weitere Beschreibung existiert (und keine eingereichten Daten verfügbar sind – eingereichte Daten werden nachfolgend weiter erörtert), wenn das Empfangsgerät vorbereitet ist, um eine weitere Beschreibung zu empfangen, wird eine Beschreibungsanforderung zu dem Sendegerät getätigt. Die Antwort auf die Anforderung wäre eine Fortsetzung des Beschreibungsblocks, die nachfolgende Auswahlen und Daten bietet.

**[0443]** Eine Beschreibungsanforderung wird aufgebaut durch Auflisten jedes Attributs aus der Beschreibung, das bereits empfangen wurde, zusammen mit der Auswahl aus jeder Auswahlliste. Die Codierungs-Hierarchie wird nicht verwendet. Somit identifiziert der Empfänger die Auswahlen, die durch den Empfänger getroffen werden, während die Oberflächenbeschreibung verarbeitet wird. Die Struktur einer Beschreibungsanforderung und der entsprechenden Beschreibungsrückantwort sind eng gekoppelt, da die Entscheidungswege, die die Codierungs-Hierarchie der Beschreibung bilden, abhängig von den Attributen antworten, die in der Beschreibungsanforderung umfasst sind.

**[0444]** Ein generischer Beschreibungsanforderungsblock für vImage ist unten in Tabelle 37 gezeigt.

## Beschreibungsanforderungsblock

Codierungs- Hierarchie	Attribut	Wert	Anmerkungen
(nicht verwen- det)	<u>(Optional):</u> vLanguage	slLanguage	Umfassen, falls verfügbar
	<u>(Erfoderlich):</u> vEncoding	vImage	Umfassen, falls verfügbar
	<u>(Erfoderlich):</u> vColorSpace	slColorSpace	Umfassen, falls verfügbar
	<u>(Erfoderlich):</u> vPixelDepth	slPixelDepth	Umfassen, falls verfügbar
	<u>(Erfoderlich):</u> vResolution	slResolution	Umfassen, falls verfügbar
	<u>(Erfoderlich):</u> Compression	slCompression	Umfassen, falls verfügbar
	<u>(Optional):</u> Endian	slEndian	Umfassen, falls verfügbar.

TABELLE 37: Generischer Beschreibungsanforderungsblock für vImage

**[0445]** Eine Beschreibungsrückantwort wird gebildet durch Verarbeiten der Anforderung, Erkennen der Auswahlmöglichkeiten und daher des Teilentscheidungswegs und Aufbauen einer Rückantwort basierend auf der Hierarchie unter der ausgewählten Ebene. Der Entscheidungsweg, der die Codierungs-Hierarchie bildet, ist abhängig von den Attributen, die in dem Beschreibungsanforderungsblock enthalten sind. Jede Ebene, die in der Beschreibungsanforderung umfasst ist, wird aus der Beschreibungsrückantwort weggelassen, aber ansonsten ist die Struktur des Beschreibungsrückantwortblocks identisch zu der Struktur des Originalbeschreibungsblocks. Ein generischer Beschreibungsrückantwortblock für vImage ist in Tabelle 38 gezeigt.

<b>Beschreibungsrückantwortblock</b>			
<i>Codierungs-Hierarchie</i>	<i>Attribut</i>	<i>Wert</i>	<i>Anmerkungen</i>
Null	(optional); vLanguage	stringSelection List	wie früher beschrieben
[vLanguage]	(Required): vEncoding	keyStringSelection List	wie früher beschrieben
[[vLanguage.]vImage]	(Required): vSize	emIntPairType	wie früher beschrieben
[[vLanguage.]vImage]	(Required): vColorSpace	keySelectionList	wie früher beschrieben
[[vLanguage.]vImage]	(Optional): vOpacity	emIntType	wie früher beschrieben
[[[vLanguage.]vImage.] slColorSpace]	(Required): vPixelDepth	intTripleListType	wie früher beschrieben
[[[[vLanguage.]vImage.] slColorSpace.] slPixelDepth]	(Required): vResolution	intPairRange SelectionList	wie früher beschrieben
[[[[[vLanguage.]vImage.] slColorSpace.]slPixel Depth.]slResolution]	(Required): vCompression	keySelectionList	wie früher beschrieben

[[[[[vLanguage.] vImage.]slColorSpace.] slPixelDepth.] slResolution.] slCompression]	(Optional): vEndian	keySelectionList	wie früher beschrieben
[[[[[[vLanguage.] vImage.]slColorSpace.] slPixelDepth.] slResolution.] slCompression.]slEndian]	(Optional): vData	emDataType	wie früher beschrieben

TABELLE 38: Generischer Beschreibungsrückantwortblock für vImage

**[0446]** Wie nachfolgend weiter erörtert wird, können Bildinhaltsdaten in der Oberflächenbeschreibung umfasst sein. In den meisten Fällen werden Inhaltsdaten nicht mit der Beschreibung geliefert, wobei sie in diesem Fall mit einer separaten Inhaltsanforderung angefordert werden müssen.

**[0447]** Das Empfangsgerät liefert dem Sendegerät alle Auswahlen, die aus der Oberflächenbeschreibung getroffen wurden, um der Sendevorrichtung anzuzeigen, welche Auswahlen getroffen wurden. Zusätzlich dazu liefert das Empfangsgerät ferner dem Sendegerät den Startort zum Referenzieren der Bilddaten. Das Sendegerät kann dann die Bilddaten zurückbringen, die der Anforderung zugeordnet sind. Die Inhaltsrückantwort besteht aus der Anzahl von Elementen, die zu dem Empfänger gesendet werden, sowie aus dieser Anzahl von Bilddatenelementen.

**[0448]** Die Inhaltsanforderung ist in der Struktur ähnlich zu der Beschreibungsanforderung. Eine Inhaltsanforderung ist aus Attributen und Werten auf eine Weise ähnlich zu der der Beschreibungsanforderung aufgebaut. Dazu sind andere Informationen hinzugefügt, die es dem Empfangsgerät erlauben, zu spezifizieren, welche Abschnitte des gesamten Inhalts benötigt werden. Der Codierungs-Hierarchieabschnitt der Tabelle wird bei einer Inhaltsanforderung nicht verwendet.

**[0449]** Der erste Abschnitt wird verwendet, um die Auswahlen zu identifizieren, die durch den Empfänger getroffen werden, während die Oberflächenbeschreibung verarbeitet wird. Der Zweck ist derart, dass alle zurückgesendeten Daten den Auswahlen entsprechen. Der zweite Abschnitt wird verwendet, um die angeforderten Informationen zu spezifizieren. Die Attributwertpaare in der ersten der Inhaltsanforderungen werden basierend auf der Codierung, den ausgeführten Auswahlen und den Werten aus der Elternbeschreibung definiert. In einigen Fällen sind möglicherweise keine zusätzlichen Attribut-Wert-Paare definiert, so dass die Anforderung nur zum Identifizieren der getroffenen Auswahlen verwendet werden würde, und zum Anfordern der Daten, die zum Aufbereiten nötig sind, wenn sie nicht eingereicht bereitgestellt werden. Die Attribut-Wert-Paare in dem ersten Abschnitt sind als Teil der Inhaltsanforderung für jede Codierung definiert. Ein Beispiel eines generischen Inhaltsanforderungsblocks ist in Tabelle 39 unten gezeigt.

<b>Inhaltsanforderungsblock</b>			
Codierungs-Hierarchie	Attribut	Wert	Anmerkungen
(nicht verwendet)	(Optional): <i>vLanguage</i>	<b>slLanguage</b>	Umfasst, falls verfügbar.
	(Erforderlich): <b><i>vEncoding</i></b>	<b>vImage</b>	
	(Erforderlich): <b><i>vColorSpace</i></b>	<b>slColorSpace</b>	
	(Erforderlich): <b><i>vPixelFormat</i></b>	<b>slPixelFormat</b>	
	(Erforderlich): <b><i>vResolution</i></b>	<b>slResolution</b>	
	(Erforderlich): <b><i>vCompression</i></b>	<b>slCompression</b>	

	(Optional): <i>Endian</i>	slEndian	Umfasst, falls verfügbar.
	(Optional): <i>vStartByte</i>	emIntType	Wirkt als ein Index für das erste Byte der gewünschten Daten. Das erste Datenbyte in den Bilddaten hat einen Index 0. Entweder das Attribut <i>vStart-</i> <i>Byte</i> oder <i>vStartLine</i> muss spezifiziert sein.
	(Optional): <i>vStartLine</i>	emIntType	Wird als ein abtast- zeilenorientierter Index verwendet. Die zurückgesendeten Daten sind in Ab- tastzeilen. Der Index zu der ersten Zeile ist 0. Wird nur verwendet, wenn das ausgewählte Komprimierungsschema <i>vNone</i> ist. Entweder das Attribut <i>vStart-</i> <i>Byte</i> oder <i>vStartLine</i> muss spezifiziert sein.

	(Optional): vNumber	emIntType	Zeigt die Anzahl von angeforderten Bytes (oder Zeilen) an. Der Sender sendet nicht mehr als vNumber Bytes (oder Zeilen), aber er könnte weniger senden. Wenn das Attribut vNumber weggelassen ist, dann kann der Sender jegliche Anzahl von Bytes senden (oder Zeilen).
--	------------------------	-----------	---

TABELLE 39: Generischer Inhaltsanforderungsblock für vImage

**[0450]** Eine Inhaltsrückantwort wird verwendet, um die aufbereitbaren Daten zu dem Empfangsgerät zu liefern. Für vImage besteht eine Inhaltsrückantwort aus Rohinhaltsdaten. Die Daten sind als einer oder mehrere Datenblöcke formatiert.

**[0451]** Die allgemeine Beschreibung für eine Inhaltsrückantwort dieses Datenformats besteht aus zwei Teilen. Der erste Teil ist ein Anfangsblock variabler Länge, der als Teil des ersten Blocks (und nur des ersten Blocks) der Inhaltsdaten existiert. Der zweite Teil sind die tatsächlichen Daten, formatiert gemäß den verschiedenen anwendbaren Attributen.

**[0452]** Die Anfangsblockinformationen sind in dem ersten Datenblock für jede Inhaltsanforderung umfasst, auf die die Sendevorrichtung antwortet.

**[0453]** Das Format des Anfangsblocks ist wie folgt: Bytes 0 und 1 identifizieren die Länge des Anfangsblocks; Byte 2 identifiziert die Version (anfänglich 0); Byte 3 ist reserviert und initialisiert auf 0; und Bytes 4 bis 7 (4 Bytes) identifizieren die Gesamtanzahl der Bytes in dem Bild. Dies ist möglicherweise nicht das gleiche wie die Anzahl von Bytes in dem Block (oder die Anzahl von Bytes, die übertragen werden): wenn die Anzahl von Bytes bei der Bildgröße die Anzahl von Bytes bei dem Block überschreitet, dann zeigt das an, dass eine andere Übertragung erforderlich ist, um das Bild fertigzustellen. Die Anfangsblockinformationen sind in Big Endian. Der Datenabschnitt kann in Little Endian sein, wie in dem Attribut vEndian angezeigt ist.

**[0454]** Inhaltsdaten können jedoch „eingereiht“ geliefert werden: d. h., als Teil der Beschreibung ohne Bedarf nach einer separaten Inhaltsanforderung und Rückantwort. Dies ist möglicherweise nicht angemessen z. B. für ein großes Bild, das in einer Vielzahl von Codierungen erhältlich ist, kann aber sehr effizient sein, wenn die Menge an Inhaltsdaten gering ist. Daten, die eingereiht geliefert werden, werden mit dem Attribut vData bereitgestellt; der Wert, der ein Attributwertepaar mit vData bildet, sind einfach die Daten selbst. Bei dieser Implementierung muss der eingereihte Inhalt vollständig in einen Beschreibungsblock einpassen, so dass keine „teileingereichten“ Daten geliefert werden können.

**[0455]** Einige Beispiele von vImage E-Material und Verhandlung werden nachfolgend gegeben.

Beispiel 1 – Zwei-Zoll-Quadrat-Bild, angeboten nur als ein monochromes Bild, keine Komprimierung oder RLE-Komprimierung, unter Verwendung von Voreinstellungen womöglich ohne eingereichte Daten.

	vEncoding	vImage
vImage	vSize	(144000, 144000)
vImage	vColorSpace	vGray
vImage.vGray	vPixelDepth	1
vImage.vGray1	vResolution	(300,300) (150,150)
vImage.vGray1.(300,300)	vPixels	(600,600)
vImage.vGray.1.(300,300)	vCompression	vRLE vNone
vImage.vGray.1.(150,150)	vPixels	(300,300)
vImage.vGray.1.(150,150)	vCompression	vNone

Beispiel 2 – Ein Bild mit einem Zoll zum Quadrat angeboten als i) monochrom, zwei Auflösungen, keine Komprimierung oder RLE-Komprimierung; ii) Vier-Bit-Grauskala, nur eine Auflösung, zwei Komprimierungen; und iii) vSRGB mit 8 Bits pro Kanal (vSRGB hat drei Kanäle, so dass dies 24 Bits pro Pixel ist) in drei Auflösungen, keine komprimiert. Nur Vier-Bit-Grauskala mit 100 × 100 Auflösung und RLE-komprimiert ist eingereicht verfügbar.

	vEncoding	vImage
vImage	vSize	(72000, 72000)
vImage	vColorSpace	vGray vSRGB
vImage.vGray	vPixelDepth	1 4
vImage.vGray.1	vResolution	(200,100) (300,300)
vImage.vGray.1. (200,100)	vCompression	vNone vRLE
vImage.vGray.1. (300,300)	vCompression	vNone vRLE

vImage.vGray.4	vResolution	(100,100)
vImage.vGray.4. (100,100)	vPixels	(100,100)
vImage.vGray.4. (100,100)	vCompression	vNone vRLE
vImage.vGray.4. (100,100) vRLE	vData	(die Daten.....)
vImage.vSRGB	vPixelDepth	24
vImage.vSRGB.24	vResolution	(150,150) (300,300) (600,600)
vImage.vSRGB.24. (150,150)	vPixels	(150,150)
vImage.vSRGB.24. (150,150)	vCompression	vNone
vImage.vSRGB.24. (300,300)	vPixels	(300,300)
vImage.vSRGB.24. (300,300)	vCompression	vNone
vImage.vSRGB.24. (600,600)	vPixels	(600,600)
vImage.vSRGB.24. (600,600)	vCompression	vNone

Beispiel 3 – Verhandlung für ein Bild, angeboten in der vorgeschriebenen Codierung und 24-Bit-sRGB. Das sRGB ist verfügbar in einer Vielzahl von Formaten, von denen nur Pixeltiefe 24 Bits, Auflösung (150, 150) und JPEG-komprimiert eingereicht verfügbar ist.

→ Zur Empfangsvorrichtung

	vEncoding	vImage
vImage	vSize	(72000, 72000)
vImage	vColorSpace	vGray vSRGB
vImage.vGray	vPixelDepth	1
vImage.vGray.1	vResolution	(300, 300)
vImage.vSRGB	vPixelDepth	24
vImage.vSRGB.24	vResolution	(150, 150) (150..600, 150..1200)
vImage.vSRGB.24. (150, 150)	vPixels	(150, 150)
vImage.vSRGB.24. (150, 150)	vCompression	vNone vJPEG
vImage.vSRGB.24. (150, 150).vJPEG	vData	(die Daten....)

**[0456]** In diesem Fall ist vSRGB 24 Bit mit JPEG ausgewählt. Die voreingestellte Auflösung ist nicht ausgewählt, aber eine Auflösung innerhalb des angebotenen Bereichs ist ausgewählt. Angebote innerhalb eines Bereichs können nicht als eingereichte Daten angeboten werden (und wurden es in diesem Fall folglich nicht): eine Inhaltsanforderung muss daher ausgeführt werden. Der angeforderte Datenbetrag ist unbekannt, so dass der angeforderte Datenbetrag nicht spezifiziert ist: die Voreinstellung ist, so viel wie möglich zu senden. Die Inhaltsanforderung enthält das nachfolgende E-Material.

→ Zur Sendevorrichtung

vEncoding	vImage
vColorSpace	vSRGB
vPixelDepth	24
vResolution	(300, 600)
vCompression	vJPEG
vStartByte	0

**[0457]** Die Inhaltsrückantwort ist 5.500 Datenbytes, was in E-Material die folgende Formel aufweist.

→ Zur Empfangsvorrichtung

vData	(die Daten...)
-------	----------------

**[0458]** Die Empfangsvorrichtung weiß nicht, ob dies alle Daten sind, und sendet so eine weitere Inhaltsanforderung wie unten, und fragt nach Daten nach den ersten 5.500 Bytes.

→ Zur Sendevorrichtung

vEncoding	vImage
vColorSpace	vSRGB
vPixelDepth	24
vResolution	(300, 600)
vCompression	vJPEG
vStartByte	5500

**[0459]** Die Inhaltsrückantwort ist, dass keine Daten zu senden mehr vorhanden sind.

→ Zur Empfangsvorrichtung

vData
-------

**[0460]** Das Bild kann nun aufbereitet werden.

Beispiel 4 – Dies ist ähnlich zu Beispiel 3, aber nur eine Teilbeschreibung ist beim anfänglichen Senden verfügbar. Eine zusätzliche Beschreibungsanforderung und Beschreibungsrückantwort ist somit erforderlich, um die selbe Meldung wie das anfängliche Senden bei Beispiel 3 zu übermitteln. Die Öffnungssequenz der Meldungen ist wie folgt.

	vEncoding	vImage
vImage	vSize	(72000, 72000)
vImage	vColorSpace	vGray vSRGB
vImage.vGray	vPixelDepth	1
vImage.vGray.1	vResolution	(300,300)
vImage.vSRGB	vPixelDepth	24
vImage.vSRGB.24	vResolution	(150,150) (150..600, 150..1200)

**[0461]** Keine Informationen werden unter der Auflösungsebene geliefert. Wenn die Empfangsvorrichtung eine weitere Beschreibung für diese Oberfläche empfangen möchte, liefert sie nun eine Beschreibungsanforderung basierend auf den Attributen, die zusammen mit den Auswahlen aus der Auswahlliste geliefert wurden. Dies umfasst das Liefern einer Auflösung in dem Bereich.

→ Zur Sendevorrichtung

vEncoding	vImage
vColorSpace	vSRGB
vPixelDepth	24
vResolution	300,600)

**[0462]** Die Beschreibungsrückantwort ist der Rest der Beschreibung, die in [Fig. 3](#) gezeigt ist (oder eher der relevanten Teile dieser Beschreibung). Die Beschreibungswege, die in der Codierungs-Hierarchie erscheinen, sind abgeschnitten.

	vPixels	(150,150)
vJPEG	vCompression	vNone vJPEG
	vData	(die Daten ...)

**[0463]** Die nachfolgenden Faktoren sollten bei der Verwendung von vImage berücksichtigt werden.

1. Das Attribut vLanguage ist ein optionales Attribut, wo der Voreinstellungswert nicht spezifiziert wurde. Die Empfangsvorrichtung kann bestimmen, keinen Wert aus der selectionList (Auswahlliste) auszuwählen. In diesem Fall wählen die Bibliotheken den ersten Wert aus, der in der Ebene beschrieben ist, nicht in dem

emListType.

2. Für alle Bildcodierungen ist das Datenformat im Gegensatz zu dem, das in der Hierarchie spezifiziert ist, wort-gepackt (4 Bytes), wobei jede Abtastzeile zu einer 4-Byte-Grenze aufgefüllt ist. Komprimierungsmechanismen führen eine Komprimierung an den gepackten und ausgefüllten Daten aus. Dies gilt für mono-chrome, graue und Farb-Daten.

3. Bilddaten werden üblicherweise auf einem Medientyp aufbereitet. Bitte, siehe die Richtlinien für die Codierung vPlane. Dies mandatiert nicht, dass eine Vorrichtung eine vImage-Codierung eines Kindes einer vPlane ausführen muss, aber es wird sehr empfohlen, dass sie dies tut.

4. Ein Bild kann auch außerhalb der physischen Grenzen des Mediums einer Vorrichtung existieren. In solchen Fällen muss die Vorrichtung die Daten basierend auf den Einschränkungen der Vorrichtung aufbereiten. In einigen Fällen kann die Vorrichtung ferner in der Lage sein, die Bilddaten neu zu formatieren oder neu zu dimensionieren. Eine andere Vorrichtung kann in der Lage sein, ein Medium unterschiedlicher Größe auszuwählen. Eine andere Vorrichtung kann auf das Beschneiden von Daten beschränkt sein. Die Vorrichtung muss die geeignetste Aktion für die Vorrichtung und das Bild bestimmen.

#### vText

**[0464]** Die Codierung vText enthält Attributwertpaare, die dem Textinhalt zugeordnet sind. Grundlegende Attribute umfassen Zeichensatz und Sprache, die dem Textstrom zugeordnet sind. Üblicherweise bestehend aus einer Zeichendarstellung (Symbolsatz) und auf einem aufbereitbaren Format (Schriftart), kann der Text visuell auf einer Seite, einem Bildschirm oder einer Anzeige (Ebene) aufbereitet werden. Text weist ferner eine Größe auf, die als ein Bereich beschrieben werden kann, in dem er leben muss. Die „Größe“ oder der Bereich impliziert ferner eine Größe (Höhe und Breite) für den Text. Diese grundlegenden Attribute können verwendet werden, um eine Anzahl von Textdarstellungen zu beschreiben. Andere Attribute zum Beschreiben von Text können hinzugefügt werden.

**[0465]** Eine Hierarchie aus Attributen für vText ist in [Fig. 24](#) gezeigt. Die Bedeutung jedes unterschiedlichen gezeigten Attributs wird kurz erörtert:

#### vLanguage:

**[0466]** Das Attribut vLanguage muss in dieser Implementierung umfasst sein.

#### vEncoding:

**[0467]** Eine Oberfläche vText wird verwendet, um Textinformationen zu kommunizieren. Die Attribute von vText sind in dieser Codierung definiert.

#### vSize:

**[0468]** Bezeichnet die Größe des Texts relativ zu der Anzeigefläche. Der erste Wert ist die Abmessung entlang der X-Achse und der zweite Wert ist die Abmessung entlang der Y-Achse, basierend auf 72.000 Einheiten pro Zoll. Die Empfangsvorrichtung sollte über die Anzeigeregion Bescheid wissen, die sie zum Aufbereiten der Daten verwendet, die Relative-Größe-Informationen und ihre eigenen Fähigkeiten. Dieses Attribut gibt der Empfangsvorrichtung Informationen, die zum Entscheiden über die beste Art und Weise zum Aufbereiten der Textdaten verwendet werden können.

#### vSymbolSet:

**[0469]** Wird verwendet, um die geeignete Abbildung zwischen Textwerten und vom Menschen lesbaren Symbolen oder Glyphen zu bestimmen. Der Symbolsatz vAscii ist ein vorgeschriebener Wert. Dieser Schlüsselwortwert bedeutet, dass der Symbolsatz, der zum Interpretieren und Anzeigen eines 8-Bit-Textwerts verwendet werden soll, der voreingestellte Symbolsatz vAscii ist. vUnicode wird ebenfalls sehr empfohlen.

#### vCodeSet:

**[0470]** Erforderlich, wenn der Wert vUnicode dem Attribut vSymbolSet zugeordnet ist. Das Attribut vCodeSet gilt nur, wenn vSymbolSet den Wert von vUnicode annimmt. Ansonsten sollte es nicht umfasst sein. emRangeType-Werte definieren die Codesätze aus der Unicode-Definition, die die Vorrichtung unterstützt.

vData:

**[0471]** Liefert eingereichten Inhalt.

**[0472]** Beschreibungs-Anforderungen und -Rückantworten, Inhalts-Anforderungen und -Rückantworten und die Bereitstellung von eingereichten Daten sind auf die selbe Weise aufgebaut wie für vImage. Eingereichte Daten sind wahrscheinlich eine besonders bevorzugte Option für vText, wenn eine lesbare Meldung von einem Gerät zu einem anderen gesendet wird. Die nachfolgenden Faktoren sollten bei der Verwendung von vText berücksichtigt werden.

1. Wenn der Wert vText für das Attribut vEncoding spezifiziert ist, muss das Attribut vLanguage auch mit dem Wert „en“ spezifiziert sein, der verbindlich wird. Innerhalb der Codierung vText ist ein Attribut vSymbolSet vorhanden, bei dem der Symbolsatz vAscii auch erforderlich wird. Dies ist zum Sicherstellen einer Kommunikation zwischen Vorrichtungen. Text und Sprache sind Formen der Kommunikation, die schwierig zu spezifizieren sind, um unter allen Umständen einen Kommunikationsmechanismus sicherzustellen. Nicht alle Vorrichtungen (und die Personen, die die aufbereiteten Daten betrachten) sprechen die selbe Sprache. Das Spezifizieren einer spezifischen Sprache und eines Symbolsatzes ist das einzige Mittel, dieses Ziel zu erreichen.
2. Die Werte emRangeType, die dem Attribut vCodeSet zugeordnet sind, werden verwendet, um die Unicode-Codesätze zu definieren, die eine Vorrichtung unterstützt. Die Vorrichtungen sollten diese Bereiche als allgemeine Blocksätze spezifizieren und nicht als spezifische, individuelle Zeichencodes.
3. Textdaten werden üblicherweise auf einem bestimmten Medientyp aufbereitet. Üblicherweise bedeutet dies, dass vText-Codierungen Kindoberflächen einer vPlane-Codierung sind. Während nichts vorhanden ist, um zu erfordern, dass eine Vorrichtung eine vText-Codierung zu einem Kind einer vPlane machen muss, ist es höchst empfehlenswert, dass sie dies in den meisten Fällen tut.

**[0473]** Ein Beispiel von vText wird nun angezeigt.

Beispiel 5 – Einfacher Text, mehr als eine Codierung mit eingereichten Daten, mit eingefügtem Attribut vLanguage (Englisch, amerikanisches Englisch und Französisch sind verfügbar: en, en-us, fr).

	vLanguage	en en-us fr
en	vEncoding	vText
en.vText	vSize	(7200, 108000)
en.vText	vSymbolSet	vAscii vUnicode
en.vText.vAscii	vData	This□is□the□text.
en.vText.vUnicode	vCodeSet	(0000,007F)
en.vText.vUnicode	vData	This□is□the□text.
en-us	vEncoding	vText
en-us.vText	vSize	(7200, 108000)
en-us.vText	vSymbolSet	vAscii vUnicode
en-us.vText.vAscii	vData	This□is□the□text.

en- us.vText.vUnicode	vCodeSet	(0000, 007F)
en- us.vText.vUnicode	vData	This□is□the□text.
fr	vEncoding	vText
fr.vText	vSize	(7200, 108000)
fr.vText	vSymbolSet	vAscii vUnicode
fr.vText.vAscii	vData	Ceci□est□le□texte.
fr.vText.vUnicode	vCodeSet	(0000, 007F)
fr.vText.vUnicode	vData	Ceci□est□le□texte.
	vFile	

**[0474]** Dies ist eine Bereitstellung zum Übertragen von Informationen in Dateiformaten. vFile liefert einen Mechanismus zum Übertragen von Daten sowie jeglicher zugeordneter Icon- und Benennungs-Informationen. Eine Datei ist üblicherweise einem Speichermedium zugeordnet und wird üblicherweise in dem abstrakten Format aus Bits und Bytes betrachtet. Im Wesentlichen kann der Inhalt der Datei bekannt sein oder nicht. Das Speichermedium wendet ferner eine Benennungsübereinkunft an die Datei an, obwohl sie möglicherweise keine Bedeutung für den Inhalt der Datei gibt. Ferner ist ein Icon in einer computerisierten Umgebung häufig der Datei zugeordnet. Aufgrund des Wesens des Inhalts der Datei kann die Codierung vFile als ein Mechanismus verwendet werden zum Übertragen dieser Bits von einer Vorrichtung zu einer anderen, während die Beziehung des Namens, Icons und möglicher Inhaltsinformationen der Datei beibehalten wird.

**[0475]** Eine Hierarchie von Attributen für vFile ist in [Fig. 25](#) gezeigt. Die Bedeutung jedes unterschiedlichen Attributs (wo es nicht aus der vorangehenden Erörterung klar ist), das gezeigt ist, wird kurz erörtert:

vLanguage: Optional.

vEncoding:

**[0476]** vFile ist als eine rohe Sammlung aus Bits definiert, die jegliche erfassbare Information darstellen. Attribute der Datei sind in dieser Codierung definiert.

vMimeType:

**[0477]** Identifiziert die Charakteristika des Typs vFile. Das Etikett sollte gemäß normalen Richtlinien für ein Mime-Typ-Konstrukt sein: z. B. Anwendung/msWord, Audio/einfach, Bild/TIFF.

vDataSize:

**[0478]** eine unsigned ganze Zahl, die die Größe der binären Daten anzeigt.

vChildIcon:

**[0479]** Verknüpft die Datei mit einem Icon. Das Icon ist auf einer separaten Oberfläche, bezeichnet durch eine Bezugnahme auf diese separate Kindoberfläche: häufig ist die Icon-Kindoberfläche als vImage codiert.

vFilename:

**[0480]** Liefert einen Dateinamen als eine mit 0 abgeschlossene Textzeichenfolge, codiert in dem Ascii-Symbolsatz.

**[0481]** Beschreibungs-Anforderungen und -Rückantworten, Inhalts-Anforderungen und -Rückantworten und Bereitstellung von eingereichten Daten sind wie für andere Codierungen. Beispiele sind nachfolgend angezeigt.

## Beispiel 6 – Die einfachste Dateicodierung

	vEncoding	vFile
vFile	vDataSize	12345543

**[0482]** Die Datenmenge ist im Allgemeinen (wie hier) zu groß, um in eine Zeile zu passen, und muss so in einer Reihe von Inhaltsanforderungen angefordert werden.

Beispiel 7 – Wenn eine Datei ein Dokument darstellt, dann ist eine Möglichkeit, dass sie auch als eine vAssociation-Dokument-Codierung geliefert wird (siehe Erörterung weiter unten). Dieses Beispiel stellt diese Alternative für ein sechsseitiges Dokument dar.

	vEncoding	vFile vAssociation
vFile	vChildIcon	12
vFile	vDataSize	12345543
vFile	vFilename	"myfile"
vAssociation	child	6 vArbitrary 1 2 3 4 5 6

vPlane

**[0483]** Die Codierung vPlane ist entworfen, um bestimmte Informationstypen darzustellen, die relevant sind für räumliche Aspekte von physischen Objekten oder Aufbereitungen. Genauer gesagt kann sie Seiten (einschließlich doppelseitiger Seiten), die Beziehung zwischen zweidimensionalen Regionen auf einer Seite, einschließlich Regionen, die als Bild kategorisiert sind, Text und Anmerkung darstellen und kann auch das Medium darstellen, auf dem aufbereitet werden soll.

**[0484]** Der Wert vPlane zeigt eine Codierung oberster Ebene an und enthält Kinder zum Darstellen komplexer Informationstypen. Er wird weitgehend verwendet, um die Beziehung zwischen diesen verschiedenen Kindern zu beschreiben.

**[0485]** Eine Ebene könnte eine Seite darstellen, die aus einer Vorderseite und einer Rückseite besteht: die Seite weist Größe und Farbe auf; und Bilder und Text könnten Kinder darstellen, die auf der Ebene aufbereitet werden würden. Eine Ebene könnte ferner einen Computerbildschirm darstellen. Das Bildschirmlayout in diesem Fall besteht aus einer einzelnen Seite. Der Bildschirm weist Größe und Farbe auf. Bilder und Text könnten mehrere Kinder darstellen, die Fenster mit Text, Klebe-Nachrichten und Icons darstellen.

**[0486]** Eine graphische Ansicht einer Codierung vPlane ist in [Fig. 27](#) bereitgestellt. Dies zeigt eine Ebene **270** an, die eine Oberfläche darstellt. Die Oberfläche wird definiert, und dann werden Kinder, die Bilder darstellen, auf die Oberfläche platziert. Die Reihenfolge von Kindern entlang der Z-Achse zeigt die Reihenfolge an, in der die Kinder auf der Oberfläche platziert oder aufbereitet werden.

**[0487]** Eine Hierarchie aus Attributen für vPlane ist in [Fig. 26](#) gezeigt. Die Bedeutung jedes unterschiedlichen Attributs (wo es aus der vorangehenden Erörterung nicht deutlich ist), das gezeigt ist, wird kurz erörtert:

vEncoding:

**[0488]** Ein vPlane ist eine seitenartige Oberfläche. Sie kann zwei Seiten aufweisen und weist üblicherweise eine oder mehrere Unter-Oberflächen als Kinder auf. Die Attribute werden bei dieser Codierung definiert.

## vChildFront:

**[0489]** Liefert eine Referenzliste, die zu jeder Kindoberfläche auf der Vorderseite der Ebene verknüpft. Ein Wert ChildCount zeigt die Anzahl von bekannten Kindoberflächen an. Er kann negativ, Null oder positiv sein. Kleine positive Werte von ChildCount werden mit einer übereinstimmenden Anzahl von Referenzen gepaart. Andere Werte werden nicht mit Referenzen gepaart. Wenn Einschränkungen auf die Reihenfolge bestehen, in der das Sendegerät die Oberflächen senden kann, auf die Bezug genommen wird, wird das Schlüsselwort vSequential verwendet, ansonsten das Schlüsselwort vArbitrary. Dies erlaubt, dass das Empfangsgerät die Reihenfolge der angeforderten Oberflächen vollständig steuert.

## vChildBack:

**[0490]** Wie bei vChildFront, aber die Referenzliste verknüpft zu jeder Kindoberfläche auf der Rückseite der Ebene.

## vBackCoord:

**[0491]** Spezifiziert den Betrag einer Drehung des Rückebenen-Koordinatensystems. Die Rückebene muss die Abmessungen der Vorderseite reflektieren. Dieses Attribut ist optional, wenn das Attribut vChildBack weggelassen wird. Es kann auch weggelassen werden, wenn keine Änderungen an der Rückebenenkoordinate vorliegen.

**[0492]** Wenn die Werte X und Y für das Attribut vSize gleich sind (eine quadratische Ebene), sind die unterstützten Werte vBackCoord 0, 90, 180 und 270. Wenn die Werte X und Y für das Attribut vSize nicht gleich sind (eine rechteckige Ebene), sind die unterstützten Werte vBackCoord 0 und 180. Jedes Kind, das auf die Ebene platziert wird, sollte im Hinblick auf diese Koordinatentransformation aufbereitet werden.

**[0493]** Das vBackCoord wird wie folgt angewendet: die Rückebene kann betrachtet werden, als ob sie um die temporäre Y-Achse gedreht wäre (Referenz [Fig. 28a](#) und [Fig. 28b](#)). vBackCoord kann dann als Drehung, entgegen dem Uhrzeigersinn, um den X-, Y-Schnittpunkt angewendet werden (Referenz [Fig. 28c](#)). Die Ebene wird dann zu der voreingestellten Koordinatenposition bewegt (Referenz [Fig. 28d](#)).

## vSize:

**[0494]** Das Attribut vSize bezeichnet Aufbereitungsabmessungen für die Ebene, die beschrieben wird. [Fig. 27](#) zeigt das angenommene Koordinatensystem. Es sei ferner angenommen, dass die ganzzahligen verwendeten Werte in Echtweltkoordinaten von 1/72.000 Zoll sind (was bedeutet, dass 72.000 Einheiten pro Zoll vorhanden sind). Der erste ganzzahlige Wert ist entlang der X-Achse und der zweite ganzzahlige Wert ist entlang der Y-Achse.

## vOpacity:

**[0495]** Zeigt die Opazität der Ebene für darunter liegende Informationen an. Werte sind 0 bis 255, mit einer Voreinstellung von 255 (Opak).

## vColor:

**[0496]** Zeigt die Farbe der Ebene an. Eine Liste von drei unsignierten, ganzzahligen Werten, im Bereich von 0 bis 255. Der erste Wert ist die Komponente Rot, der zweite die Komponente Grün und der dritte die Komponente Blau. Ein Wert 0 zeigt keine Intensität an. 255 zeigt volle Intensität an. Wenn der Wert vOpacity 0 ist, haben Werte vColor keine Bedeutung.

## vPosition:

**[0497]** Stellt die Position des Kindes im Hinblick auf die Elternebene dar. Der erste Wert bei dem Paar ist die X-Koordinate und der zweite Wert ist die Y-Koordinate. Dieses Attribut kann für die Kinder umfasst sein, auf die durch die Attribute vChildFront und vChildBack Bezug genommen wird. Dieses Attribut muss aufgelistet sein, eindeutig, für jedes Kind, für das die Informationen geliefert werden sollen. Dieses Attribut kann für jegliche Anzahl von Kindern bereitgestellt werden, auf die Bezug genommen wird.

## vAttachment:

**[0498]** Beschreibt, wie das Kind relativ zu den Eltern manipuliert werden kann.

**[0499]** Der Wert vFixed zeigt an, dass das Kind fest an dem Punkt bleiben soll, der durch das Attribut vPosition spezifiziert ist. Der Wert vFloating zeigt an, dass das Kind relativ zu den Eltern durch Geräte neu positioniert werden kann, die zu einer solchen Manipulation in der Lage sind.

**[0500]** Das Attribut vAttachment kann für die Kinder umfasst sein, auf die durch die Attribute vChildFront und vChildBack Bezug genommen wird. Dieses Attribut muss aufgelistet sein, eindeutig, für jedes Kind, für das die Informationen geliefert werden sollen. Dieses Attribut kann für jegliche Anzahl der Kinder bereitgestellt werden, auf die Bezug genommen wird.

**[0501]** Beschreibungs-Anforderungen und -Rückantworten sind im Wesentlichen wie für andere Codierungen. Es gibt Unterschiede bei der Verbindung mit Inhalts-Anforderungen und -Rückantworten. Eine Inhalts-Anforderung wird durch eine Empfangsvorrichtung in zwei Situationen verwendet: zum Anfordern von Informationen über Kindoberflächen, wenn sie nicht umfasst sind, und wenn Beschreibungsinformationen, die einer Kind-Referenz zugeordnet sind, nicht verfügbar waren.

**[0502]** Ein Beispiel, wann alle Informationen in dem Beschreibungsblock nicht gefunden werden würden, wäre, wenn die Kindnamen-Referenzen nicht verfügbar sind. Diese Bezugnahmen (References; Referenzen) können durch die Inhalts-Anforderung unter Verwendung indirekter Kind-Referenzen angefordert werden (Werte von vFirst und vLast), um anzuzeigen, dass Kindinformationen angefordert werden. Eine zweite Instanz, wann eine Inhalts-Anforderung verwendet werden würde, wäre das Anfordern von Inhaltsinformationen in Bezug auf ein Kind, aber relevant für die Ebene. Inhaltsinformationen eines Kindes für die Codierung von vPlane bestehen aus vPosition- und vAttachment-Attributen und -Werten, die die Beziehung zwischen dem Kind und der Ebene anzeigen.

**[0503]** Die Empfangsvorrichtung benötigt eine Referenz für jedes Kind. Diese Kind-Referenzen sind üblicherweise Teil der referenceList, die den Attributen vChildFront oder vChildBack zugeordnet ist, oder wurde bereits durch eine Beschreibungsanforderung empfangen, wie oben erwähnt wurde. (Üblicherweise sind die Beschreibungsinformationen für ein referenziertes Kind „eingereiht“, da dem Kind auf dieser Ebene wenige Attribute zugeordnet sind). Wenn die Inhaltsinformationen nicht eingereiht sind, wird eine Inhaltsanforderung verwendet, um die Informationen anzufordern. Die Empfangsvorrichtung liefert der Sendevorrichtung alle Auswahlen, die aus der Oberflächenbeschreibung getätigt wurden, um der Sendevorrichtung anzuzeigen, welche Auswahlen getroffen wurden. Zusätzlich dazu liefert die Empfangsvorrichtung ferner der Sendevorrichtung eine explizite Liste aus Kindreferenzen, falls verfügbar, oder eine indirekte Referenz (die Werte von vFirst und vLast), um anzuzeigen, dass eine Anforderung nach Kindinformationen angefordert wird. Die Sendevorrichtung kann dann die Beschreibungsinformationen zurücksenden, die den Kindreferenzen zugeordnet sind.

**[0504]** Ein Inhaltsanforderungsblock weist optionale Attribute von vChild, vChildNext und vNumber auf.

**[0505]** vChild kann Werte annehmen, wie z. B. vFrontFirst, vBackLast und Ähnliche oder eine Referenzliste. Wenn ein Kind vorangehend unter Verwendung von vFrontFirst, vFrontLast, vBackFirst oder vBackLast referenziert wurde, können diese Schlüsselwortwerte nicht wieder verwendet werden. Wenn hier eine einzelne Referenz ausgeführt wird, dann spezifiziert sie einen Startpunkt für Inhaltsinformationen, die für ein oder mehrere Kinder zurückgesendet werden sollen: Die angeforderte Anzahl und die Sequenz, vorwärts oder rückwärts von der Referenz, ist unter dem Attribut vNumber spezifiziert. Wenn eine referenceList „Referenzliste“ spezifiziert ist, dann ist es eine explizite Anforderung nach dem Inhalt der spezifizierten Kinder: Das Attribut vNumber wird nicht verwendet.

**[0506]** Die Schlüsselwörter vFrontFirst und vBackFirst sind indirekte Referenzen auf das erste Kind in der Struktur. Dies gilt, wenn die Namensreferenzen spezifiziert sind sowie wenn die Namensreferenzen unbekannt sind. Die Schlüsselwörter vFrontLast und vBackLast werden verwendet, um auf das letzte Kind auf dieselbe Weise Bezug zu nehmen. Wenn die Anzahl der Kinder unbekannt ist oder der Zugriff sequentiell ist, kann das Ergebnis des indirekten Einstellens der Kindreferenz auf das Letzte möglicherweise nicht die erwarteten Ergebnisse erzeugen, da das Letzte möglicherweise nicht zugreifbar ist.

**[0507]** Die Sendevorrichtung behält keinen Status von vorangehend getätigten Anforderungen, was heißt, sobald vFrontFirst, vFrontLast, vBackFirst oder vBackLast verwendet werden, nachfolgende Anforderungen eine

explizite Referenz verwenden sollten, die in der vorangehenden Inhaltsrückantwort zurückgesendet wird, oder das Attribut vChildNext verwenden. Das Attribut vChild sollte nie mit dem Attribut vChildNext verwendet werden.

**[0508]** vChildNext kann verwendet werden, wo ein vorangehendes Kind referenziert wurde. vChildNext kann verwendet werden, um Inhalt für das nächste Kind in der Sequenz anzufordern, identifiziert durch die Sendevorrichtung. Dieses Attribut darf nicht verwendet werden, wenn ein vorangehendes Kind nicht referenziert wurde. Dieses Attribut sollte weggelassen werden, wenn das Attribut vChild mit einem Referenznamen verwendet wird.

**[0509]** Der Referenzwert ist der Kindname bevor die Kindinformationen angefordert werden. vChildNext kann verwendet werden, wenn der Name des nächsten Kindes unbekannt ist. Die angeforderte Anzahl und die Sequenz, vorwärts oder rückwärts von der Referenz, ist unter dem Attribut vNumber spezifiziert.

**[0510]** vNumber ist optional und wird nur verwendet, wenn eine einzelne Referenz unter dem Attribut vChild spezifiziert ist, oder wenn das Attribut vChild auf vFrontFirst, vFrontLast, vBackFirst, vBackLast eingestellt ist, oder wenn das Attribut vChildNext verwendet wird. vNumber ist einem Wert zugeordnet, der die Anzahl von Referenzen anzeigt, für die die Empfangsvorrichtung Inhalt anfordert. Wenn vNumber mit einer Referenz verwendet wird, die dem Attribut vChild zugeordnet ist, umfassen die zurückgesendeten Elemente das referenzierte Kind. Wenn vNumber mit einer Referenz verwendet wird, die dem Attribut vChildNext zugeordnet ist, umfasst die Anzahl von zurückgesendeten Elementen das referenzierte Kind nicht aber umfasst Kinder nachfolgend zu dem referenzierten Kind. Wenn möglich sendet der Sender die Inhaltsinformationen für die angeforderten Kinder, in der angeforderten Reihenfolge. Wenn dieses Attribut fehlt, oder wenn der Wert Null ist, dann sendet der Sender den Inhalt von so vielen Kindern wie möglich von der getätigten Referenz.

**[0511]** Die geeignete Inhaltsrückantwort ist relativ einfach. Das Attribut vNumber sendet die Anzahl von Kindern in der Inhaltsrückantwort zurück. vNames sendet Kindreferenzen in der angeforderten Reihenfolge zurück. vRemaining zeigt eine Anzahl von Kindern an, für die noch keine Inhaltsanforderung getätigt wurde. Werte von vPosition und vAttachment sollten für jede der zurückgesendeten Referenzen bereitgestellt werden.

**[0512]** Die nachfolgenden Faktoren erfordern eine Berücksichtigung bei der Verwendung von vPlane.

1. Die Codierung vPlane könnte in vielen Fällen als das Medium betrachtet werden, auf dem Bilder und Text aufbereitet werden. Seiten, Computeranzeigen, Statusanzeigen, Magnettafeln, Projektionsmedien, etc. sind Beispiele solcher Medien. Eine aufbereitbare Datencodierung, wie z. B. Bild oder Text, sollte an ein Medium wie vPlane angewendet werden. Eine explizite Beschreibung (aber nicht die Regel) wäre, dass vImage und vText Kinder von vPlane sein sollten.
2. Jede Referenz, die den Attributen vChildFront, vChildBack, vChild, vChildNext oder vNames zugeordnet ist, muss eindeutig sein. Es wurde ein Mechanismus bereitgestellt, bei dem eine einzelne Oberfläche mehrere Male referenziert werden kann. Wenn die Empfangsvorrichtung diesem Mechanismus implementieren möchte, hätte die Empfangsvorrichtung die Option, die Oberfläche mehrere Male anzufordern, oder die Referenzdaten intern beizubehalten. Eine „#reference“ (Nummernzeichen) (Referenz) kann umfasst sein, wenn die Oberflächeninformationen mehrere Male verwendet werden aber die Informationen des Kindinhalts (vPosition, vAttachment) unterschiedlich sind. Die „#reference“ wirkt als ein Index zum eindeutigen Identifizieren von Kind und Kindinhalt.
3. Wenn die referenceList die Namen der Kindoberflächen nicht enthält, müssen sie durch Abfrage erhalten werden.
4. Wenn der Wert vSequential erscheint, können die Kinder nur in einer vorbestimmten Reihenfolge angefordert werden, aufgrund von Einschränkungen der Sendevorrichtung. Dies impliziert eine Verarbeitungssequenz von:
  - Anfordern von Inhaltsinformationen (die den Anhang beschreiben) des nächsten Kindes (unter Verwendung einer Inhaltsanforderung und eines Rückantwortpaares auf der Elternoberfläche)
  - Anfordern der Oberflächenbeschreibung/des Abdrucks dieses Kindes (unter Verwendung eines Beschreibungs-Anforderungs- und -Rückantwort-Paares auf dieser Kindoberfläche)
  - Anfordern des Inhalts dieses Kindes (unter Verwendung eines Inhalts-Anforderungs- und -Rückantwort-Paares auf dieser Kindoberfläche)
  - Aufbereiten dieses Kindes
  - Anfordern von Inhalt-(Anhang-)Informationen des nächsten Kindes usw., bis keine weiteren Kinder mehr existieren.
5. Es gibt eine Anzahl von Möglichkeiten zum Anfordern von Inhaltsdaten für Kinder, eines oder mehrere gleichzeitig, egal ob die Kindliste bekannt oder unbekannt ist. Die nachfolgenden Verfahren sind anwend-

bar, um auf die Kindoberflächen in einer Kindliste zuzugreifen:

- Verwenden von `vChild` mit einem oder mehreren Referenznamen zum Zugreifen auf verschiedene Kindoberflächen.
- Verwenden von `vFrontFirst` oder `vBackFirst` zum Zugreifen auf die erste Kindoberfläche in der Liste.
- Verwenden von `vFrontLast` oder `vBackLast` zum Zugreifen auf die letzte Kindoberfläche in der Liste, falls angemessen.
- Verwenden von `vChildNext` mit `vNumber` gleich 1 oder mehr, um auf die nächste eine oder mehrere Kindoberflächen in der Liste zuzugreifen. Dies kann nicht der erste Zugriff auf eine Kindoberfläche in dieser Kindliste sein.
- Verwenden von `vChildNext` mit `vNumber` gleich Null oder undefiniert, um auf die verbleibenden Kindoberflächen in der Liste zuzugreifen. Dies kann ebenfalls nicht der erste Zugriff auf eine Kindoberfläche in dieser Kindliste sein.

6. Ein zufälliger Zugriff auf Oberflächen in der Kindliste wird durch die nachfolgende Sequenz erreicht:

- Verwenden des Oberflächennamens als den Wert für das Attribut `vChildFront` oder `vChildBack` in einer Inhaltsanforderung. Der Oberflächenname wird in der Inhaltsrückantwort zurückgesendet. In dem Fall von `vPlane` werden Anhanginformationen ebenfalls zurückgesendet.
- Verwenden des Kindoberflächennamens zum Aufbauen einer Beschreibungsanforderung. Die Oberflächenbeschreibung wird in einer Beschreibungsrückantwort zurückgesendet. Ein neuer Auftrag wird initiiert durch Ausgeben einer `SurfaceRequestMsg`, wie in dem Abschnitt Wechselwirkungstaktiken beschrieben ist.
- Verwenden einer Inhaltsanforderung zum Erhalten des Oberflächeninhalts der Kindoberfläche. Die Informationen werden in einer Inhaltsrückantwort zurückgesendet.
- Aufbereiten der Inhaltsinformationen.
- Dieser Prozess fährt nach Bedarf fort.

7. Serieller Zugriff auf Oberflächen in der Kindliste wird erreicht durch:

- Verwenden von `vFrontFirst` oder `vBackFirst` als den Wert für das Attribut `vChildFront` oder `vChildBack` in einer Inhaltsanforderung auf der Elternoberfläche. (Der Wert `vFrontLast` oder `vBackLast` kann als der Wert für diese Attribute verwendet werden, wenn `childCount` positiv ist und ein `vArbitrary`-Schlüssel gegeben ist.) Der Oberflächenname der spezifizierten Kindoberfläche in der Kindliste wird in der Inhaltsrückantwort zurückgesendet, zusammen mit der Anzahl von verbleibenden Kindoberflächen in der Kindliste. In dem Fall der Codierung `vPlane` werden Anhanginformationen ebenfalls zurückgesendet.
- Verwenden des Kindoberflächennamens zum Aufbauen einer Beschreibungsanforderung. Die Oberflächenbeschreibung wird in einer Beschreibungsrückantwort zurückgesendet. Ein neuer Auftrag wird initiiert durch Ausgeben einer `SurfaceRequestMsg`, wie in dem Abschnitt Wechselwirkungstaktiken beschrieben ist.
- Verwenden einer Inhaltsanforderung zum Erhalten des Oberflächeninhalts der Kindoberfläche. Die Informationen werden in einer Inhaltsrückantwort zurückgesendet.
- Aufbereiten der Kindoberflächeninhaltsinformationen. In dem Fall der Codierung `vPlane`, Verwenden der Anhanginformationen.
- Verwenden des Kindoberflächennamens als den Wert für das Attribut `vChildNext` in einer Inhaltsanforderung auf der Elternoberfläche. Dieser Oberflächenname der nächsten Kindoberfläche in der Kindliste wird in der Inhaltsrückantwort zurückgesendet, zusammen mit der Anzahl von verbleibenden Kindoberflächen in der Kindliste. In dem Fall der Codierung `vPlane` werden Anhanginformationen ebenfalls zurückgesendet.
- Dieser Prozess fährt fort bis keine Kindoberflächen mehr verbleiben.

8. Die Werte von `childCount` und `vArbitrary` oder der Schlüssel `vSequential` zeigen die Verfahren des Zugriffs auf die verschiedenen Oberfläche in der Kindliste an:

- Ein kleiner positiver Wert `childCount` und Schlüssel `vArbitrary` zeigt an, dass auf die Kindoberflächen auf jegliche Weisen zugegriffen werden kann: seriell vorwärts, seriell rückwärts oder zufällig. Ein zufälliger Zugriff wird möglich unter Verwendung der Oberflächennamen in der Referenzliste. Zusätzlich kann eingereihter Inhalt für eine Kindoberfläche umfasst sein.
- Ein kleiner positiver Wert `childCount` und Schlüssel `vSequential` zeigt an, dass die Kindoberflächen nur seriell vorwärts zugegriffen werden können. Die Oberflächennamen sind umfasst, erlauben aber keinen zufälligen Zugriff.
- Ein großer positiver Wert `childCount` und Schlüssel `vArbitrary` zeigt an, dass auf die Kindoberflächen entweder seriell vorwärts oder seriell rückwärts zugegriffen werden kann.
- Ein großer positiver Wert `childCount` und Schlüssel `vSequential` zeigt an, dass auf die Kindoberflächen nur seriell vorwärts zugegriffen werden kann.
- Ein negativer Wert `childCount` (unabhängig von der Einstellung des Schlüssels `vArbitrary` oder `vSequential`) zeigt an, dass die Kindoberflächen nur seriell vorwärts „entdeckt“ werden können.

– Ein Wert childCount von Null bedeutet, dass keine Kindoberflächen in der Kindliste existieren. In dem Fall von vPlane, wenn beide Kindlisten leer sind, ist die Oberfläche vPlane nur eine rudimentäre Oberfläche mit wenigen Attributen, wie z. B. vColor und vSize. In dem Fall von vAssociation, wenn die Kindliste leer ist, ist die Oberfläche nutzlos.

9. Der Zugriff auf die Kindoberflächen in der Liste kann direkt durch den Referenznamen erfolgen oder indirekt durch andere Mechanismen. Jede Kindoberfläche in der Kindliste hat einen Referenznamen und eine Reihenfolgenposition in der Liste. Eine Reihenfolge ist für alle Kindoberflächenlisten vorhanden, egal ob das Sendegerät darauf beschränkt ist, die Oberflächen in dieser Reihenfolge anzubieten (vSequential) oder in der Lage ist, die Oberflächen in jeder Reihenfolge anzubieten (vArbitrary). Alle Kindoberflächen in einer Oberfläche vPlane haben auch eine Position auf der Oberfläche (vPosition) und ein Verfahren zur Anbringung (vAttachement). Direkter Zugriff auf Beschreibungs- oder Inhalts-Informationen von Kindoberflächen erfolgt immer durch den Referenznamen. Indirekter Zugriff auf Beschreibungs- oder Inhalts-Informationen von Kindoberflächen erfolgt durch seriell Vorwärts- oder Rückwärts-Bewegen durch die Liste aus Kindern.

**[0513]** Beispiele der Verwendung der Codierung vPlane werden nachfolgend angezeigt.

Beispiel 8 – Eine Seitenebene, 8,5 Zoll mal 11 Zoll, zwei Kinder (wahrscheinlich Bilder, aber nicht umfasst). Kindinhaltsinformationen werden eingereiht bereitgestellt.

	vEncoding	vPlane
vPlane	vChildFront	2 vArbitrary 4 5
vPlane	vSize	(612000, 792000)
vPlane.4	vPosition	(72000, 360000)
vPlane.4	vAttachement	vFixed
vPlane.5	vPosition	(612000, 612000)
vPlane.5	vAttachement	vFixed

Beispiel 9 – Ein Blatt (eine Seite), zwei Seiten, die „Vorderseite“ mit zwei Kindern und die „Rückseite“ mit einem Kind. Kindinhaltsinformationen werden nicht eingereiht bereitgestellt.

	vEncoding	vPlane
vPlane	vChildFront	2 vArbitrary 6 7
vPlane	vChildBack	1 vArbitrary 8
vPlane	vBackCoord	0
vPlane	vSize	(8500, 11000)

**[0514]** Beispiel 10 – Ebene **290** mit einer gescannten Seite **291**, 8,5 Zoll mal 11 Zoll, mit einer Anmerkung **292**, einer gelben Hervorhebung **293** und einer roten Notiz vom „Klebe“-Typ **294** mit Text. Da die letzteren Drei offensichtlich nachträglich zu dem abgetasteten Bild hinzugefügt werden müssten, wird das Bild nicht dynamisch erzeugt und die Anzahl von Kindern ist bekannt. [Fig. 29](#) stellt dieses Beispiel bildhaft dar.

**[0515]** Die Ebenencodierung ohne Kinder ist

	vEncoding	vPlane
vPlane	vChildFront	4 vSequential 1 2 3 4
vPlane	vSize	(612000, 792000)
vPlane.1	vPosition	(90000, 90000)
vPlane.1	vAttachment	vFixed
vPlane.2	vPosition	(288000, 72000)
vPlane.2	vAttachment	vFixed
vPlane.3	vPosition	(900000, 360000)
vPlane.3	vAttachment	vFixed
vPlane.4	vPosition	(360000, 540000)
vPlane.4	vAttachment	vFloating

**[0516]** Das erste Kind stellt ein Bild dar (das Hintergrundbild).

	vEncoding	vImage
vImage	vSize	(612000, 792000)
vImage	...	
		.
		.
		.

**[0517]** Das zweite Kind stellt eine Textanmerkung dar.

	vEncoding	vText
vText	vSymbolSet	vAscii
vText	...	
		.
		.
		.

**[0518]** Das dritte Kind stellt den gelb hervorgehobenen Bereich dar: Dies ist in zwei Farbmodellen verfügbar.

	vEncoding	vPlane
vPlane	vSize	(144000, 14400)
vPlane	vColor	255 255 0
vPlane	vOpacity	110

**[0519]** Das vierte Kind stellt eine separate rote Notiz vom Klebetyp dar.

	vEncoding	vPlane
vPlane	vChildFront	1 vArbitrary 5
vPlane	vSize	(144000, 144000)
vPlane	vColor	255 0 0
vPlane.5	vPosition	(0, 7200)
vPlane.5	vAttachment	vFloating

**[0520]** vAssociation ist primär entworfen, um Dokumente und jegliche Korrelation zwischen ähnlichen und nichtähnlichen Codierungstypen darzustellen und zu beschreiben. vAssociation ist eine weitere Codierung oberster Ebene und ist entworfen, um Kinder zu enthalten, um komplexe Informationstypen darzustellen. Es wird primär verwendet, um Beziehungen zwischen Kindern zu beschreiben, die ebenfalls Codierungen oberster Ebene sind. Eine Zuordnung (Association) könnte verwendet werden, um eine Beziehung zwischen mehreren Seiten (einem Dokument) zu beschreiben. Sie könnte auch verwendet werden, um einen Ordner darzustellen, der eine Beziehung zwischen Dokumenten anwendet.

**[0521]** Eine Hierarchie von Attributen für vAssociation ist in [Fig. 30](#) gezeigt. Die Bedeutung von jedem unterschiedlichen Attribut ist offensichtlich aus der Erörterung anderer Codierungen oberster Ebene. vAssociation liefert einfach eine Zuordnungsbeziehung zwischen Kindoberflächen: Es hat keine eigenen Attribute.

**[0522]** Beschreibungs-Rückantworten und -Anforderungen sind technisch möglich, sind aber in der Praxis wahrscheinlich nicht erforderlich. Für Inhalts-Anforderungen und -Rückantworten gelten im Wesentlichen dieselben Berücksichtigungen wie für vPlane im Hinblick auf Kindoberflächen.

**[0523]** Viele der allgemeinen Berücksichtigungen, die sich auf vPlane beziehen, gelten ebenfalls. Die wichtigste zusätzliche Berücksichtigung, die Aufmerksamkeit erfordert, ist beim Verschachteln von Zuordnungen. Die Zuordnung kann viele unterschiedliche Dinge darstellen, und diese Zuordnungen wachsen, wenn die Anzahl von Codierungen wächst. Einige Vorrichtungen können verschiedene Ebenen von vAssociation als ein Kind einer Codierung vAssociation verstehen. Das Beispiel, wo eine Zuordnung einen Ordner darstellen könnte, der eine Beziehung zwischen Dokumenten anwendet, wobei die Dokumente eine Beziehung auf die Seiten der Dokumente anwenden, kann durch einige Vorrichtungen verstanden werden und durch andere nicht verstanden werden. Es sollte ein Versuch durch die Vorrichtungen unternommen werden, diesen Verschachtelungstyp einzuschränken, aber ihn auch zu verstehen, wenn er verwendet wird. Vorrichtungen, die nicht in der Lage sind, die Verschachtelung von nichtaufbereitbaren Ebenen anzuwenden, können auswählen, dieselben zu verwerfen und den letzten aufbereitbaren Abschnitt aufzubereiten. Vorrichtungen können auch auswählen, die Codierung als eine solche zu interpretieren, die sie nicht entziffern können. Verschachtelte Ebenen mit nicht aufbereitbaren Daten erfordern Speicher, und die Vorrichtungen müssen die Tiefe bestimmen, bis zu der sie arbeiten möchten. Es ist wünschenswert, zu erfordern, dass eine bestimmte Verschachtelungsebene unterstützt wird: Bei der vorliegenden bevorzugten Implementierung ist es erforderlich, dass bis zu vier Ebenen von Plane- (Ebene) oder Association-(Zuordnungs-)Verschachtelungen unterstützt werden.

Beispiel 11 – Eine Zuordnung, die ein dreiseitiges Dokument darstellt. Die Kinder können in jeder Reihenfolge abgedruckt werden, die der Empfänger auswählt, die logische Ordnung der Kinder als 1, 2, 3 sollte jedoch fest sein.

	vEncoding	vAssociation
vAssociation	Child	3 vAssociation 1 2 3

**[0524]** Eine kurze Beschreibung, wie ein E-Material-Block als Bytes codiert wird, ist nachfolgend gegeben.

**[0525]** Jeder Block besteht aus einer geordneten Sequenz aus Attributen. Ein Attribut besteht aus einem Namen und seinen Daten. Attributnamen und Attribut-Ebenenamen sind einfache Werte: Attributdaten können jedoch komplexe Werte annehmen (wie z. B. eine Liste aus einfachen Werten). Einfache Werte können Schlüsselwörter (kurze Schlüsselwörter aus einem Byte oder lange Schlüsselwörter aus zwei Bytes, die einen Werttypidentifizierer umfassen), Zeichenfolgen, ganze Zahlen, Paare (Sammlungen aus zwei Werten) und Ganzzahlenbereiche sein.

**[0526]** Das Byteformat zum Codieren von Attributen ist ähnlich zu dem Tabellenformat. Der Hauptunterschied liegt in der Codierung der Ebenen. Diese werden effizienter gespeichert, da keine Anforderung vorliegt, dass die Attributebene für jedes Attribut wiederholt wird. Stattdessen werden Ebenen wie Attribute codiert, mit einer Datentypenebene (einem komplexen Wert), der aus einer geordneten Sequenz aus anderen Attributen und Ebenen besteht. Ein E-Material-Block weist ferner einen Anfangsblock mit allgemeinen Informationen über den Block auf und ein Attribut, die Nullebene. Die Nullebene enthält Attribute von Unterebenen für den Block.

**[0527]** Ein schematische Diagramm eines E-Material-Blocks **310** ist in [Fig. 31](#) bereitgestellt. Die verschiedenen Elemente sind etikettiert und werden nachfolgend beschrieben.

**[0528]** Der Anfangsblock **311** enthält Informationen über den Block insgesamt. Er weist Versionsinformationen und Größeninformationen auf. Potentiell können mehr Felder hinzugefügt werden. Das Gesamtgrößenfeld in dem Anfangsblock kann größer sein als die Größe der Nullebene, die es enthält. Es ist für die Verwendung angemessen, wenn E-Material-Blöcke lokal aufgebaut werden: Das Programm ordnet einen Speicherblock zu, schreibt den Anfangsblock und die Nullebene in denselben und stellt die Gesamtgröße auf die Gesamtgröße ein, die in dem Speicherblock verfügbar ist. Dieser Wert kann dann verwendet werden, um verfügbaren Raum zu verfolgen, wenn der Block aufgebaut wird. Wenn jedoch ein Block für das Netzwerk gesendet wird, wird empfohlen, dass die Gesamtgröße so eingestellt wird, dass der Block keinen Puffer enthält.

**[0529]** Der erste Eintrag **312** in dem E-Material-Block muss die Nullebene sein. Die Nullebene ist die oberste Ebene, die alle Attribute und Unterebenen des E-Material-Blocks enthält. Nichts kann außerhalb der Nullebene existieren. Diese Ebene ist tatsächlich als ihre konstituierenden Ebenen codiert, nur ihr Name muss auf die Konstante vNULL (0 × 40) eingestellt sein. Der Anfangsblock für die Ebene enthält eine Größe der Ebene und den Namen. Der Wertetyp (LEVEL) folgt, um anzuzeigen, dass die Daten eine Ebene sind, d. h. eine Sammlung von anderen Attributen und Ebenen. Dann folgen die Attribute, die Teil der Nullebene sind, in Sequenz.

**[0530]** Attribute bestehen aus einer Größe, dem Namen des Attributs und einigen Daten. Sowohl Name als auch Wert sind auf dieselbe Weise codiert: als ein Werttyp gefolgt von einigen Daten. Die Größe ist die Anzahl von Bytes, die durch Namen und Daten zusammen eingenommen werden. Eine Unterebene **313** ist einfach ein Attribut dessen Datentyp auf LEVEL eingestellt ist. Eine Unterebene besteht aus den Ebenendaten, ihrem Namen, einem Ebenentypindikator und einer Sequenz aus Attributen innerhalb der Ebene.

**[0531]** Zum Beispiel kann der E-Material-Block, der unten gezeigt ist, bildhaft wie in [Fig. 32](#) dargestellt sein.

<i>Ebene</i>	<i>Attributname</i>	<i>Attributwert</i>
	•	
	vEncoding	{vText}
vText	vSize	{1000, 5000}
vText	vSymbolSet	{vAscii, vUnicode}
vText.vAscii	vData	„halloejsa“
vText.vUnicode	vCodeSet	{000..00FF}
vText.vUnicode	vData	„hallojsa“

**[0532]** Dies könnte zu dem nachfolgenden, angemerkten Hex-Speicherauszug führen.

<i>Bytewerte</i>	<i>Element/Name</i>	<i>Wert</i>
05 4D 00 00 05	Anfangsblock	Gesamtgröße 005D = 93 Byte
00 56 – 40 – 01		
00 05 – 82 – 05 00 01 D3	Nullebene	Größe 0056 = 86 Byte
00 4B – D3 – 01	Attribut vEncoding	[vText]
00 08 – 86 - 06 09 03 E8 09 13 88	Ebene vText	Größe 004B = 79 Byte
00 06 – D2 – 05 00 02 D0 D4	Attribut vSize	(1000,5000)
00 10 – D0 – 01	Attribut vSymbolSet	[vAscii, vUnicode]
00 0C – 81 – 04 68 61 6C 6C 6F	Ebene vText.vAscii	Größe = 16 Byte
65 6A 73 61 00	Attribut vString	„halloejsa“
00 23 – D4 – 01	vText.vUnicode	Größe = 35 Byte
00 09 – D5 – 05 00 01 0F 00 00 00 FF	Attribut vCodeSet	[00..FF]
00 14 – 81 – 03 00 68 00 61 00 6C	Attribut vData	Unicode („hallojsa“)
00 6C 00 6F 00 6A		
00 73 00 61 00 00		

**[0533]** Es ist ersichtlich, dass die Verwendung von E-Material eine extreme Flexibilität bei der Codierung erlaubt. Es sollte jedoch darauf hingewiesen werden, dass Standardcodierungen soweit wie möglich verwendet werden sollten, um eine tatsächlich allgemeine Kommunikation zu ermöglichen.

**[0534]** Vorrichtungen steht es frei, eine E-Material-Codierung auf jegliche Weise zu zersetzen, die geeignet ist.

**[0535]** Ein Attribut, dass als optional spezifiziert ist, kann einen Wert umfassen, der eine `seleccionList` ist, aus der der Entscheidungsweg aufgebaut ist. Einer dieser Werte in der `seleccionList` kann als der voreingestellte Wert spezifiziert sein. Die Reihenfolge, in der die Sendevorrichtung die Ebene aufbaut, bestimmt die Voreinstellung. Der voreingestellte Wert muss der erste Wert sein, der in der Auswahlliste beschrieben ist. Diese Ebenen müssen korrekt aufgebaut sein, wenn die Empfangsvorrichtung die Voreinstellung ordnungsgemäß auswählen soll.

**[0536]** Die Reihenfolge der Werte in einem Attributwertepaar kann verwendet werden, um Präferenz oder Priorität der Sendevorrichtung zu identifizieren. Es liegt an dem Ermessen der Empfangsvorrichtung, zu bestimmen, welcher Wert ausgewählt werden soll, unabhängig von der Präferenz der Sendevorrichtungen. Ein kluger Ansatz ist, dass das erste Element in einer Auswahlliste die bevorzugte Auswahl des Senders sein sollte.

**[0537]** Es besteht eine explizite Reihenfolge für Kindoberflächen. Bei einer `vAssociation`, sollte erwartet werden, Kindoberflächen eine nach der anderen fertigzustellen. Die Reihenfolge ist allgemein sehr stark an die Reihenfolge der physischen Medien gebunden. Bei der `vPlane`, spielt das Attribut `vPosition` eine bedeutendere Rolle beim Anzeigen einer physischen Reihenfolge. In beiden Fällen schränkt die Reihenfolge häufig Zugriff auf Kindoberflächen ein.

**[0538]** Eine gesamte Hierarchieebene muss in einer einzelnen Beschreibung oder Rückantwort enthalten sein, und kann sich nicht über Rückantworten erstrecken.

### Patentansprüche

1. Ein Verfahren zum Freigeben des Sendens von Daten von einer Sende-Informationshandhabungsvorrichtung (**343, 344**) zu einer Empfangs-Informationshandhabungsvorrichtung (**342, 343**) in einem Datenformat, das durch die Empfangs-Informationshandhabungsvorrichtung (**342, 343**) ausgewählt wird, wobei eine Einrichtung (**341, 345**) für eine Kommunikation von Informationen zwischen den Informationshandhabungsvorrichtungen vorliegt, wobei das Verfahren aufweist, dass die Sende-Informationshandhabungsvorrichtung (**343, 344**) folgende Schritte ausführt:

Bestimmen einer Mehrzahl von Datenformaten, in denen die Daten durch die Sende-Informationshandhabungsvorrichtung (**343, 344**) gesendet werden können, und Erzeugen einer Datenformathierarchie, bei der jedes der Mehrzahl von Datenformaten durch einen Weg durch die Datenformathierarchie dargestellt ist; und Senden der Datenformathierarchie zu der Empfangs-Informationshandhabungsvorrichtung (**342, 343**), so dass die Empfangs-Informationshandhabungsvorrichtung (**342, 343**) die Daten in einem erforderlichen Datenformat durch Navigieren entlang eines Wegs durch die Datenformathierarchie erhält.

2. Ein Verfahren gemäß Anspruch 1, das ferner das Empfangen einer Auswahl von der Empfangs-Informationshandhabungsvorrichtung (**342, 343**) aufweist, die einen Teilsatz der Mehrzahl von Datenformaten darstellt und eine weitere Datenformathierarchie liefert, die diesen Teilsatz darstellt.

3. Ein Verfahren gemäß Anspruch 1 oder Anspruch 2, das ferner das Empfangen einer Auswahl von der Empfangs-Informationshandhabungsvorrichtung (**342, 343**), die ein einzelnes Datenformat aus der Mehrzahl von Datenformaten darstellt, und das Senden dieses einzelnen Datenformats zu der Empfangs-Informationshandhabungsvorrichtung (**342, 343**) aufweist.

4. Ein Verfahren gemäß einem der vorhergehenden Ansprüche, das ferner das Senden von Inhaltsdaten in einem oder mehreren der Datenformate der Mehrzahl von Datenformaten zusammen mit einer Datenformathierarchie aufweist.

5. Ein Verfahren zum Freigeben des Empfangs von Daten von einer Sende-Informationshandhabungsvorrichtung (**343, 344**) zu einer Empfangs-Informationshandhabungsvorrichtung (**342, 343**) in einem Datenformat, das durch die Empfangs-Informationshandhabungsvorrichtung (**342, 343**) ausgewählt wird, wobei eine Einrichtung (**341, 345**) für eine Kommunikation von Informationen zwischen den Informationshandhabungsvorrichtungen

gen vorliegt, wobei das Verfahren aufweist, dass die Empfangs-Informationshandhabungsvorrichtung (**342, 343**) folgende Schritte ausführt:

Empfangen einer Datenformathierarchie mit jedem einer Mehrzahl von Datenformaten von der Sende-Informationshandhabungsvorrichtung (**343, 344**), in denen die Daten durch die Sende-Informationshandhabungsvorrichtung gesendet werden können, dargestellt durch einen Weg durch die Datenformathierarchie; und Navigieren entlang eines Wegs durch die Datenformathierarchie, um aus der Mehrzahl von Datenformaten ein einzelnes Datenformat oder einen Teilsatz der Mehrzahl von Datenformaten auszuwählen.

6. Ein Verfahren gemäß Anspruch 5, bei dem der Weg, entlang dem navigiert wird, einen Teilsatz der Mehrzahl von Datenformaten darstellt, und ferner das Anfordern einer weiteren Datenformathierarchie, die diesen Teilsatz darstellt, von der Sende-Informationshandhabungsvorrichtung (**343, 344**) aufweist.

7. Ein Verfahren gemäß Anspruch 6, das ferner das Empfangen der weiteren Datenformathierarchie, die diesen Teilsatz darstellt, von der Sende-Informationshandhabungsvorrichtung (**343, 344**) und das Navigieren entlang eines Wegs durch die weitere Datenformathierarchie aufweist, um aus der Mehrzahl von Datenformaten ein einzelnes Datenformat oder einen weiteren Teilsatz der Mehrzahl von Datenformaten auszuwählen.

8. Ein Verfahren gemäß Anspruch 5 oder Anspruch 7, bei dem der Weg, entlang dem navigiert wird, ein einzelnes Datenformat darstellt, und ferner das Anfordern von Inhaltsdaten in diesem einzelnen Datenformat von der Sende-Informationshandhabungsvorrichtung (**343, 344**) aufweist.

9. Ein Verfahren gemäß Anspruch 5 oder Anspruch 7, das ferner das Empfangen von Inhaltsdaten in einem oder mehreren der Datenformate der Mehrzahl von Datenformaten mit der Datenformathierarchie, das Auswählen eines einzelnen Datenformats, in dem Inhaltsdaten geliefert werden, und das Verwenden der Inhaltsdaten in diesem Datenformat aufweist.

10. Ein Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem alle Datenformate einen oder mehrere einer Mehrzahl von Datenformattypen aufweisen, und bei dem für jeden Datenformattyp ein Datenformat existiert, das durch alle Informationshandhabungsvorrichtungen empfangbar ist, die diesen Datenformattyp unterstützen.

11. Ein Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem die Datenformathierarchie in einer Beschreibung einer Oberfläche enthalten ist, wobei die Oberfläche eine Darstellung eines internen Zustands von einer der Informationshandhabungsvorrichtungen ist.

12. Ein Verfahren gemäß einem der vorhergehenden Ansprüche, bei dem eine Datenformathierarchie eine hierarchische Struktur, einen Satz aus Schlüsselwörtern, die sich auf Eigenschaften von gesendeten Informationen beziehen, und Werte, die jedem der Schlüsselwörter zugeordnet sind, aufweist.

13. Ein Verfahren gemäß Anspruch 12, bei dem jede Datenformathierarchie eine einzelne Codierung aufweist, wobei jede Codierung eine unterschiedliche Grundform darstellt, in der Informationen dargestellt sein können.

14. Ein Verfahren gemäß Anspruch 13, bei dem eine Auswahl einer Codierung ein Bild ist.

15. Ein Verfahren gemäß Anspruch 13 oder Anspruch 14, bei dem eine Auswahl einer Codierung ein Text ist.

16. Ein Verfahren gemäß einem der Ansprüche 13 bis 15, bei dem eine Auswahl einer Codierung eine binäre Datei ist.

17. Ein Verfahren gemäß einem der Ansprüche 13 bis 16, bei dem eine Auswahl einer Codierung eine Ebene ist, wobei eine Oberfläche mit einer Ebenencodierung Kind-Oberflächen aufweisen kann, die Informationen zum Aufbereiten auf dieser Ebene aufweisen.

18. Ein Verfahren gemäß Anspruch 17, bei dem die Ebene zwei Seiten besitzt und Informationen auf jeder Seite der Ebene aufbereitet werden können.

19. Ein Verfahren gemäß Anspruch 18, bei dem die Ebene eine variable Durchlässigkeit besitzt, wobei die Durchlässigkeit den Grad bestimmt, zu dem Informationen auf einer Seite der Ebene von der anderen Seite

der Ebene wahrgenommen werden können.

20. Ein Verfahren gemäß einem der Ansprüche 17 bis 19, bei dem die Datenformathierarchie Informationen enthält, die die Reihenfolge von Kind-Oberflächen-Informationen auf der Ebene bestimmen.

21. Ein Verfahren gemäß einem der Ansprüche 13 bis 20, bei dem eine Auswahl einer Codierung eine Zuordnung von Kind-Oberflächen ist, wobei jede Kind-Oberfläche der Zuordnung Informationen aufweist, die zusammen verarbeitet werden sollen.

22. Ein Verfahren gemäß Anspruch 21, bei dem Ebenen- oder Zuordnungscodierungen innerhalb anderer Ebenen- oder Zuordnungscodierungen verschachtelt sein können.

23. Ein Verfahren gemäß einem der Ansprüche 17 bis 22, bei dem die Datenformathierarchie bestimmt, ob die Position von Kind-Oberflächen in einer Ebenen- oder Zuordnungscodierung durch die Vorrichtung gesteuert wird, die die Datenformathierarchie liefert, oder durch die Vorrichtung modifiziert werden kann, die die Datenformathierarchie empfängt.

24. Ein Verfahren gemäß einem der Ansprüche 12 bis 23, bei dem, wo eine Datenformathierarchie eine Bezugnahme auf Kind-Oberflächen enthält, die Reihenfolge, in der die Kind-Oberflächen geliefert werden, als fest oder durch die Vorrichtung bestimmbar, die die Datenformathierarchie empfängt, ausgewählt sein kann.

25. Eine Informationshandhabungsvorrichtung, die angepasst ist, um das Senden von Daten von der Informationshandhabungsvorrichtung zu einer Empfangs-Informationshandhabungsvorrichtung (**342, 343**) in einem Datenformat freizugeben, das durch die Empfangs-Informationshandhabungsvorrichtung (**342, 343**) gemäß dem Verfahren ausgewählt wird, das in einem der Ansprüche 1 bis 4 oder einem der Ansprüche 10 bis 24, wo sie von einem der Ansprüche 1 bis 4 abhängig sind, beansprucht ist.

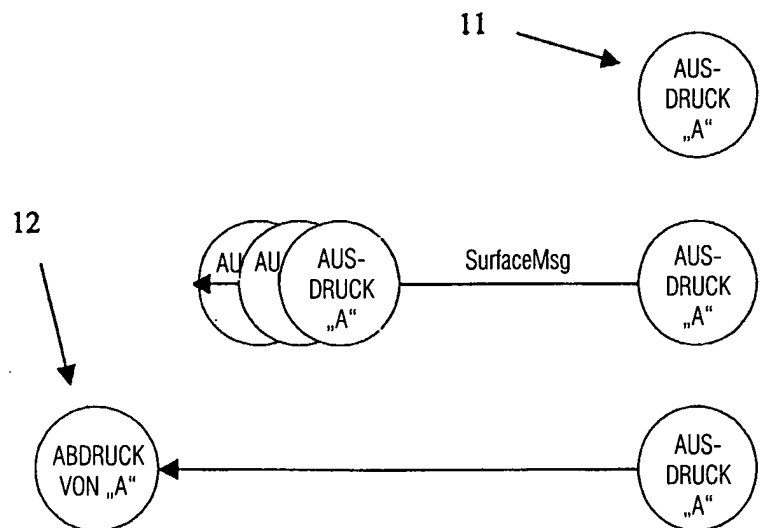
26. Eine Informationshandhabungsvorrichtung, die angepasst ist, um das Empfangen von Daten von einer Sende-Informationshandhabungsvorrichtung (**343, 344**) zu der Informationshandhabungsvorrichtung in einem Datenformat freizugeben, das durch die Informationshandhabungsvorrichtung gemäß dem Verfahren ausgewählt wird, das in einem der Ansprüche 5 bis 9 oder in einem der Ansprüche 10 bis 24, wo diese abhängig von einem der Ansprüche 5 bis 9 sind, beansprucht ist.

Es folgen 34 Blatt Zeichnungen

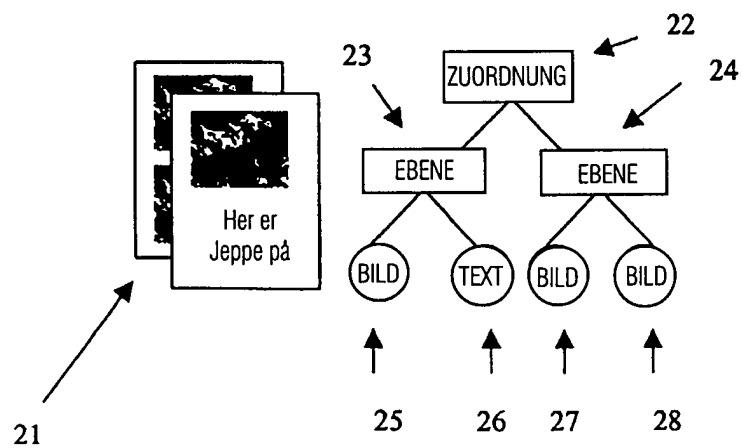
## Anhängende Zeichnungen

VORRICHTUNG B

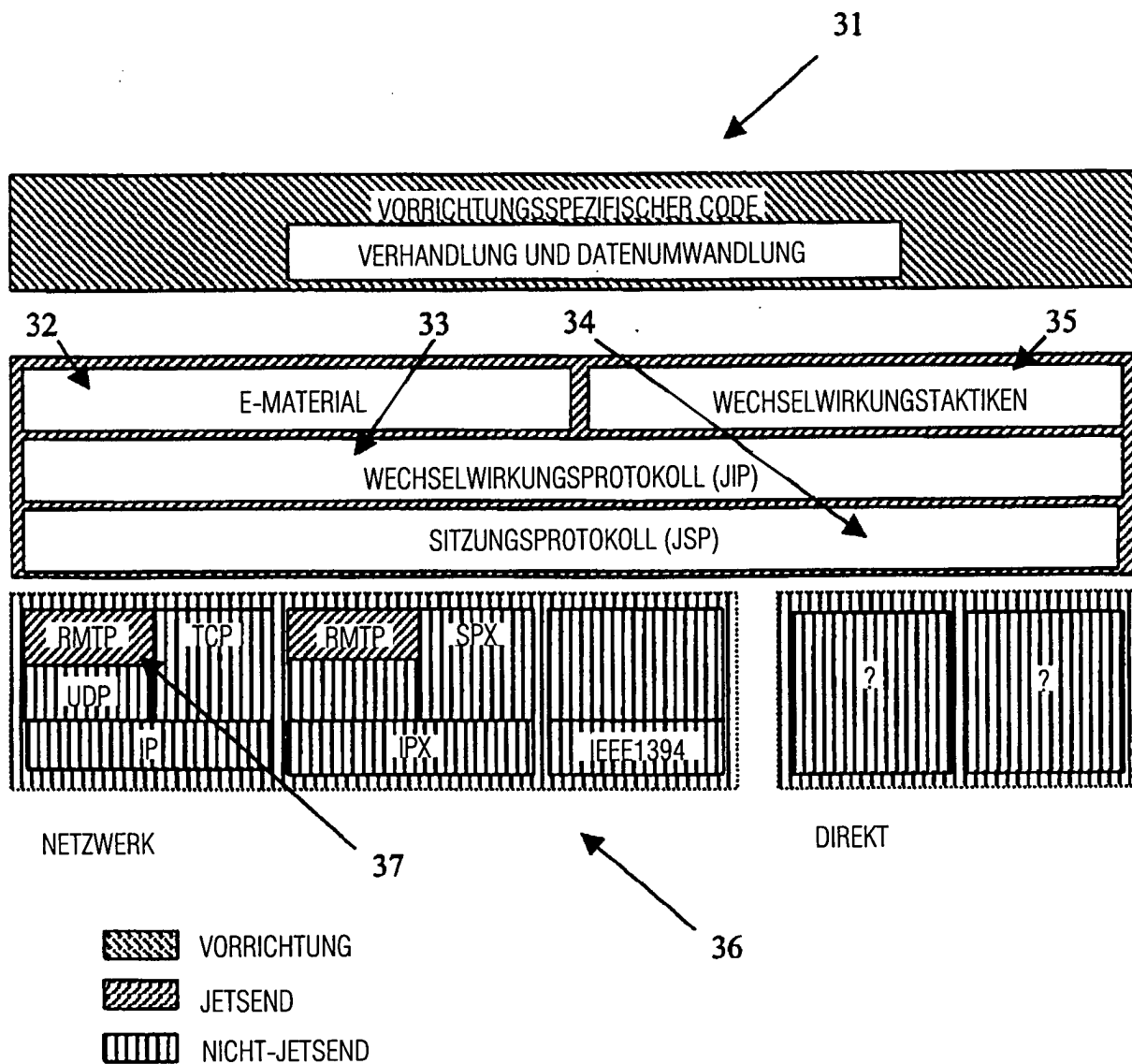
VORRICHTUNG A



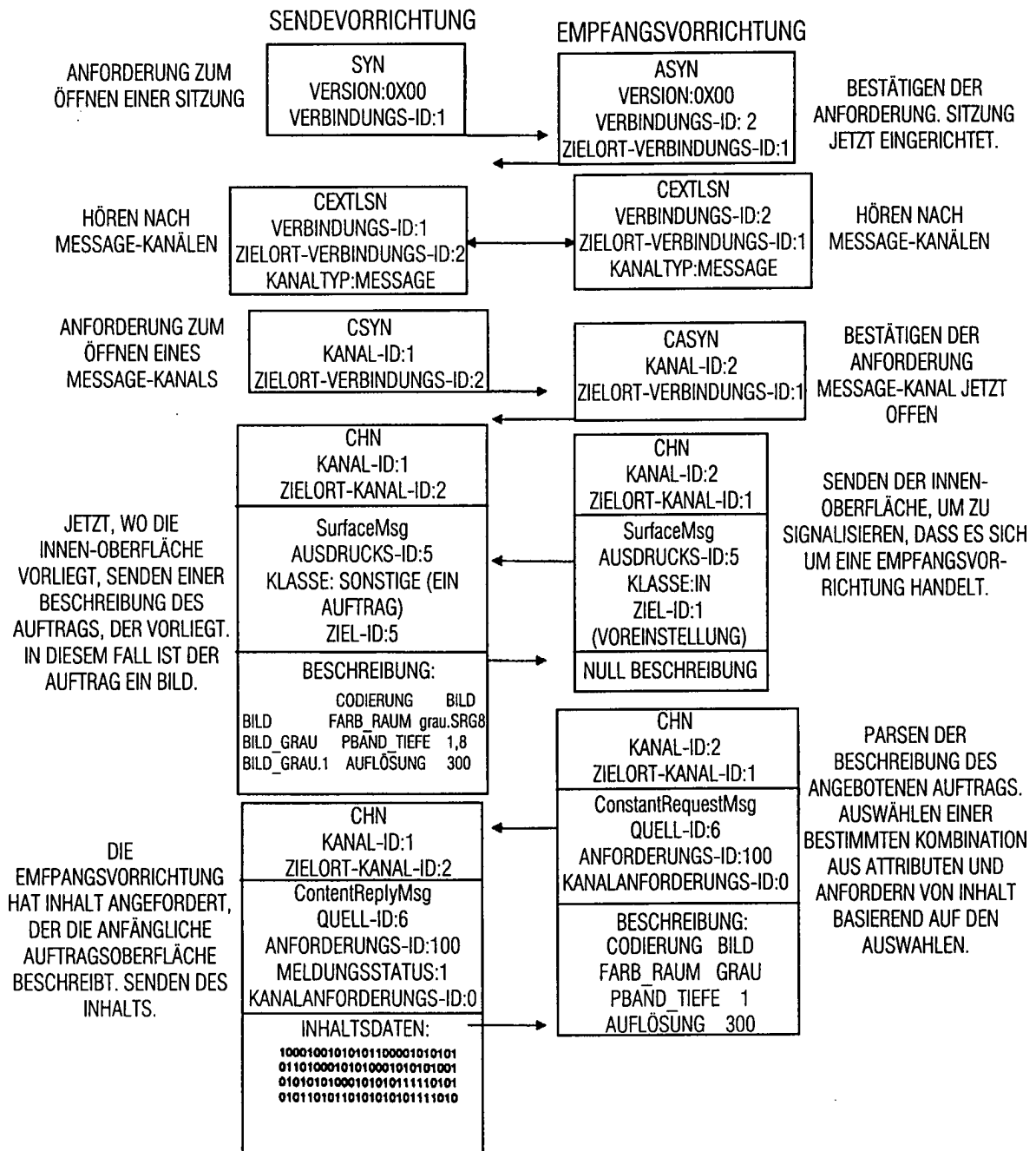
FIGUR 1



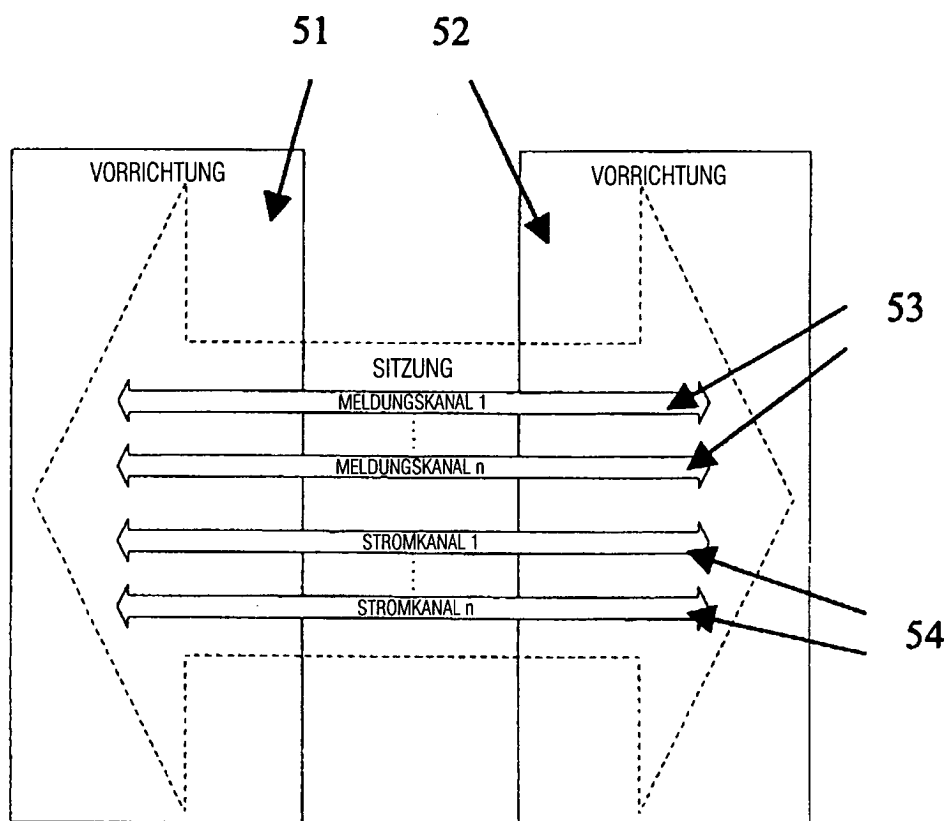
FIGUR 2



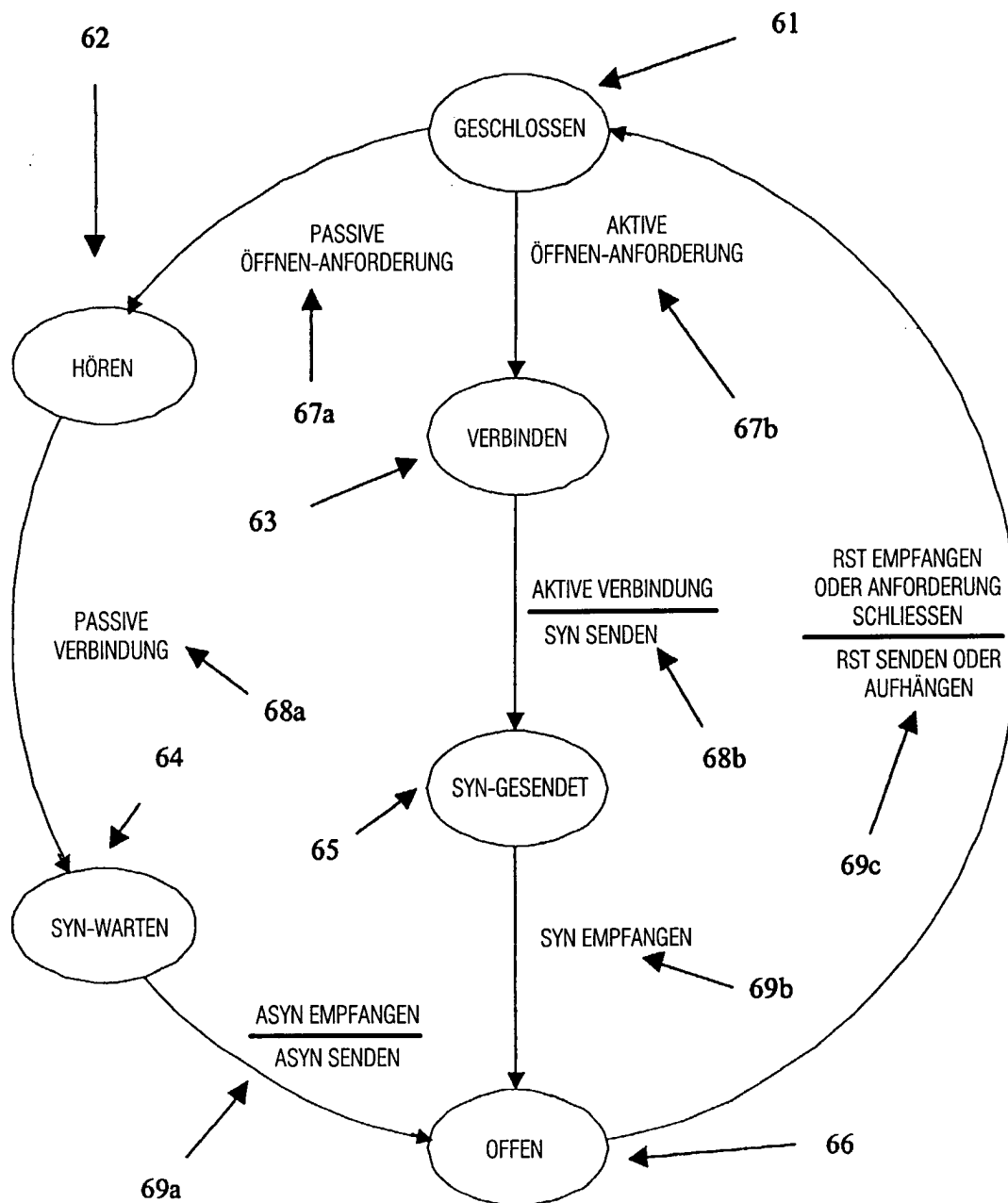
FIGUR 3



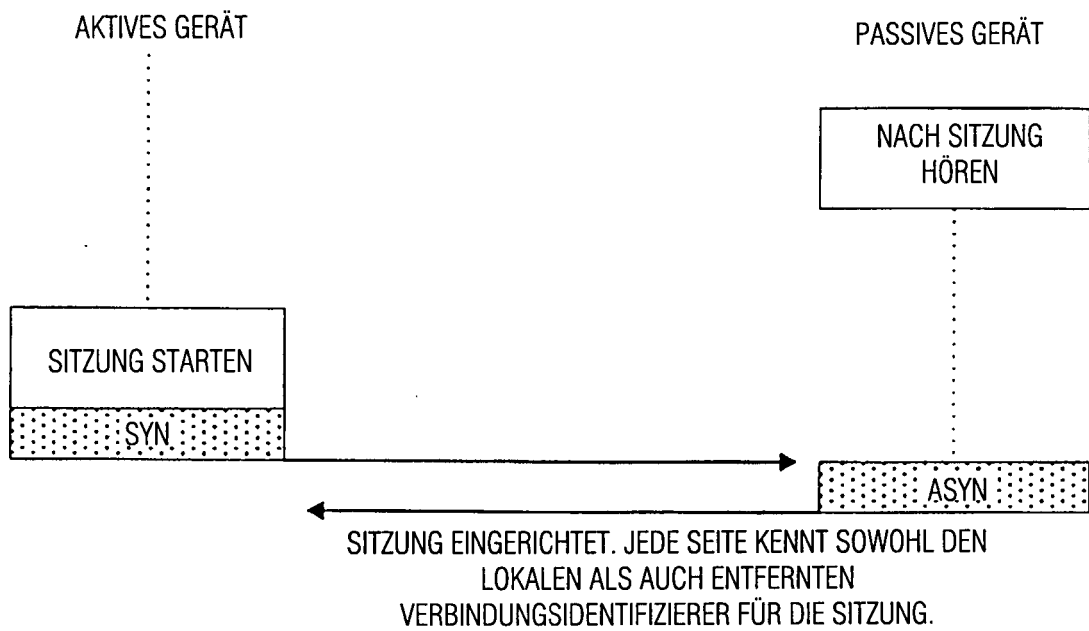
FIGUR 4



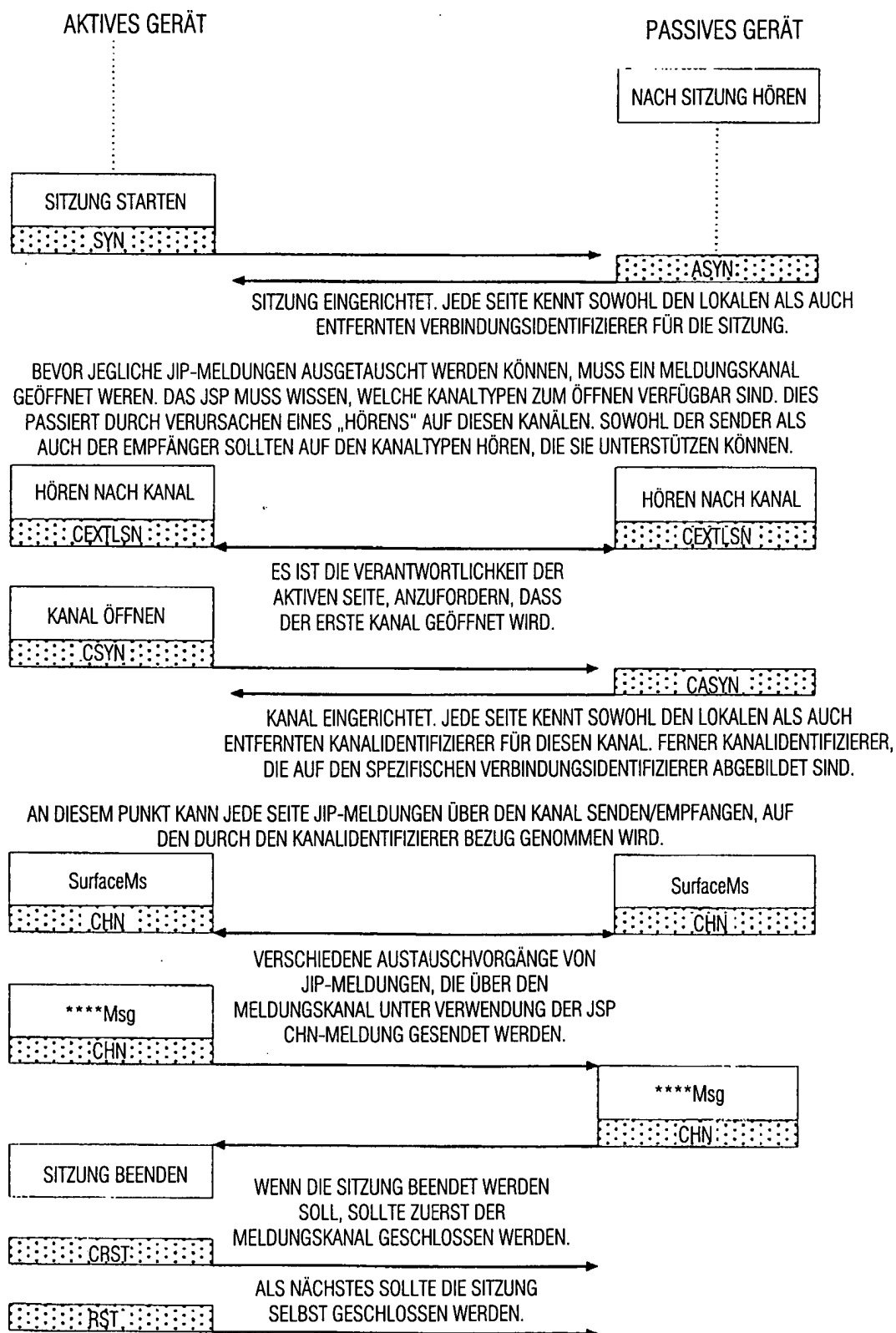
FIGUR 5



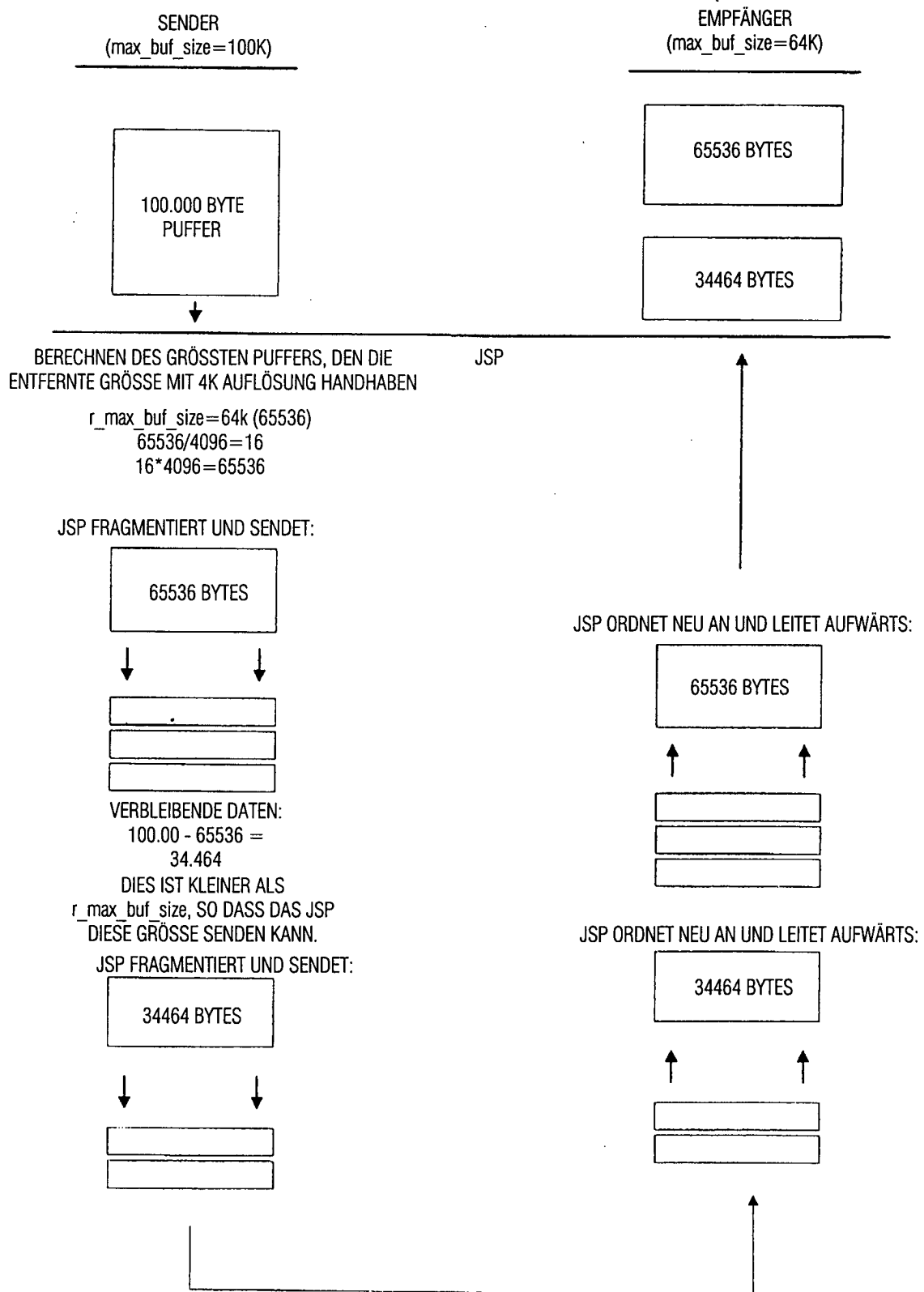
FIGUR 6



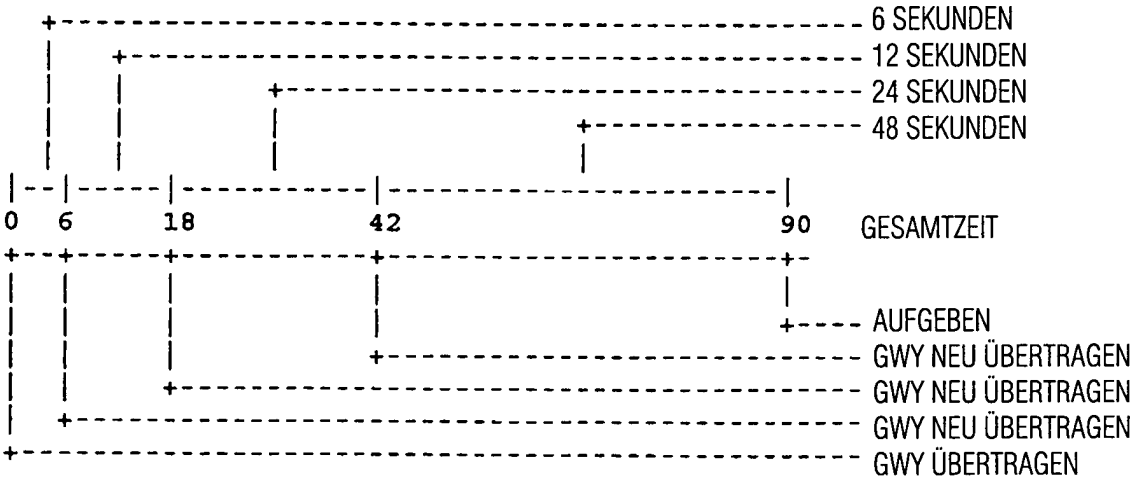
FIGUR 7



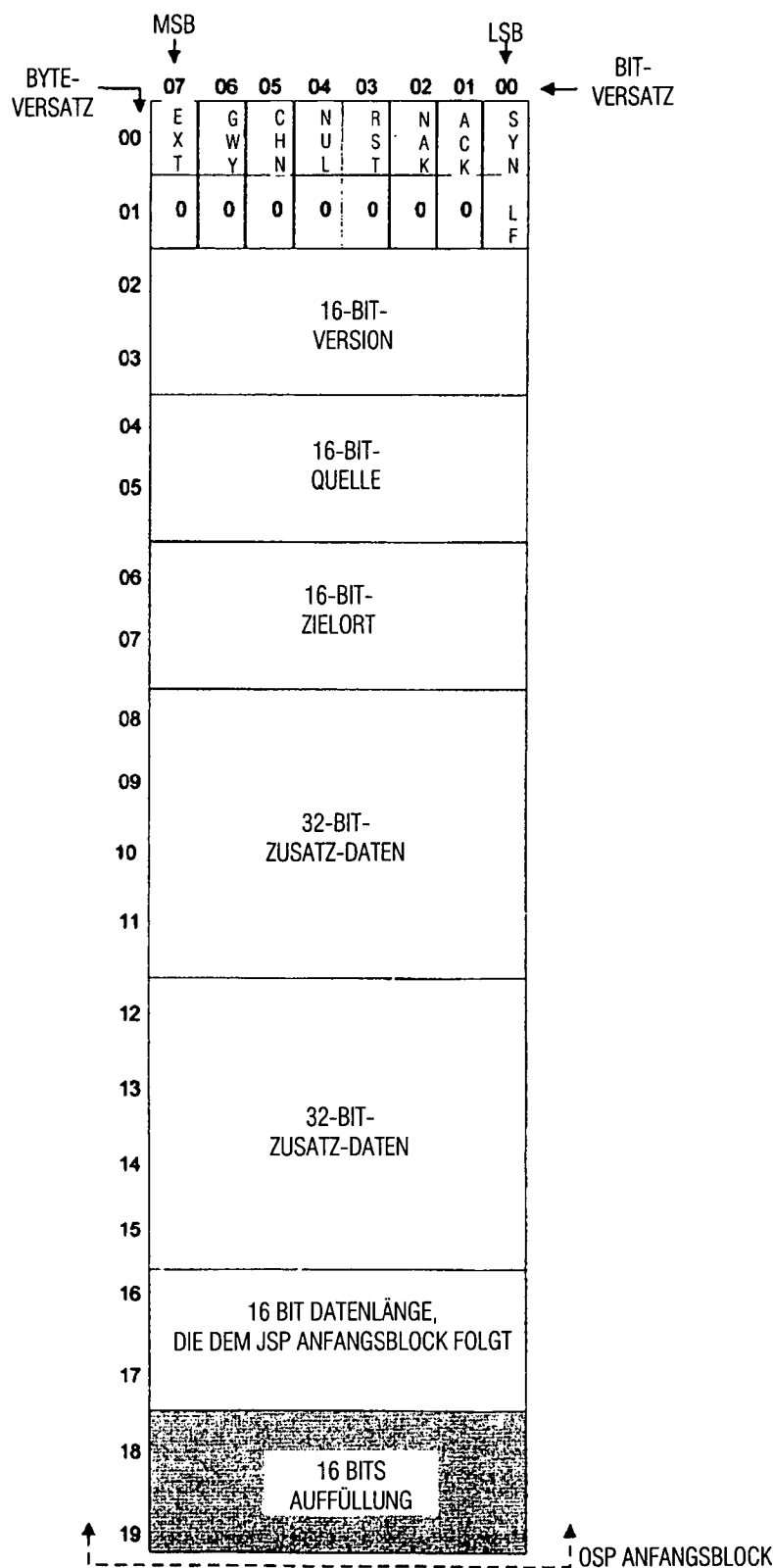
FIGUR 8



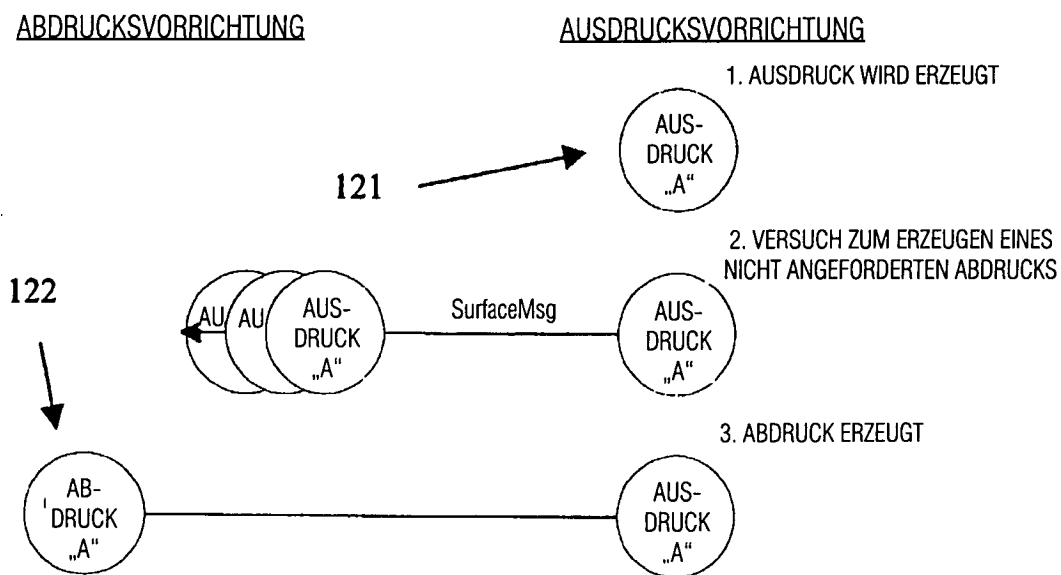
FIGUR 9



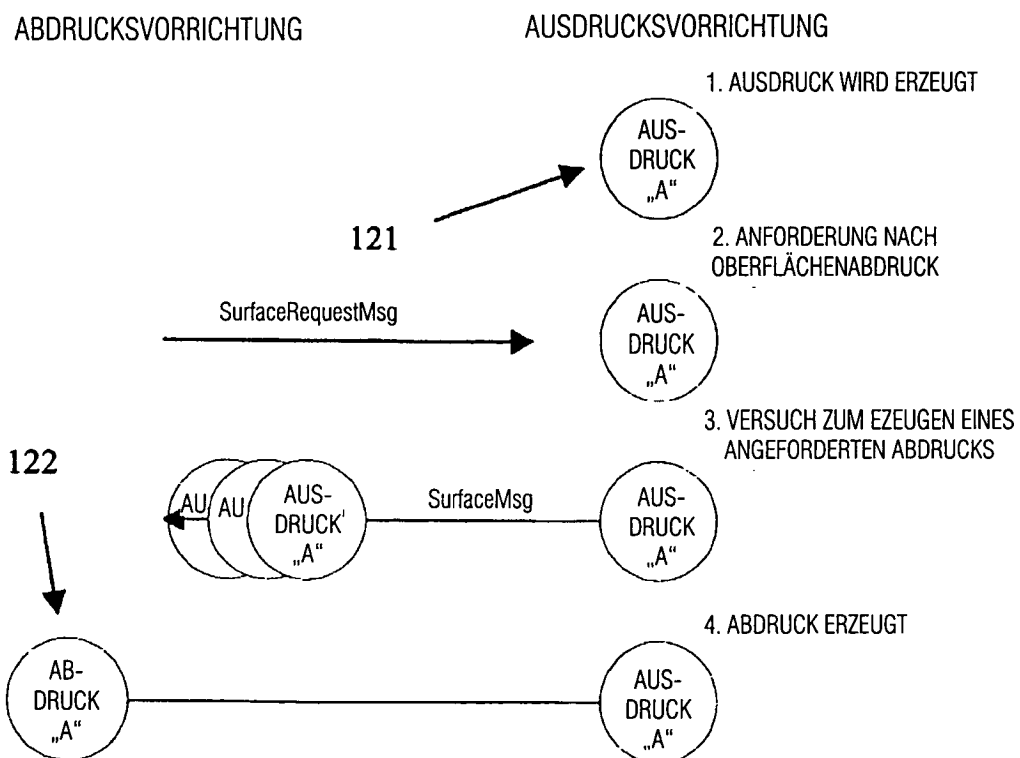
FIGUR 10



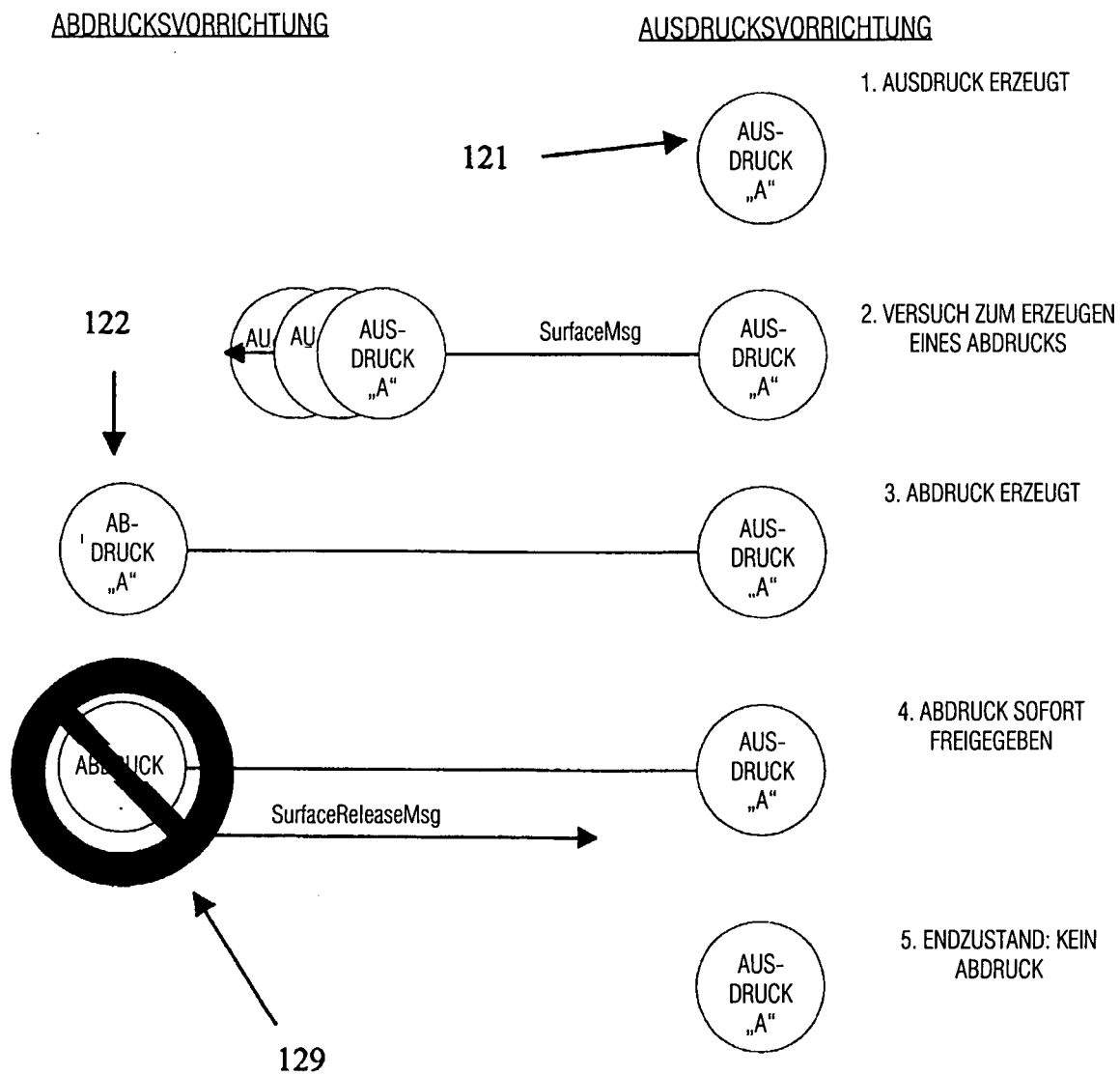
FIGUR 11



FIGUR 12A



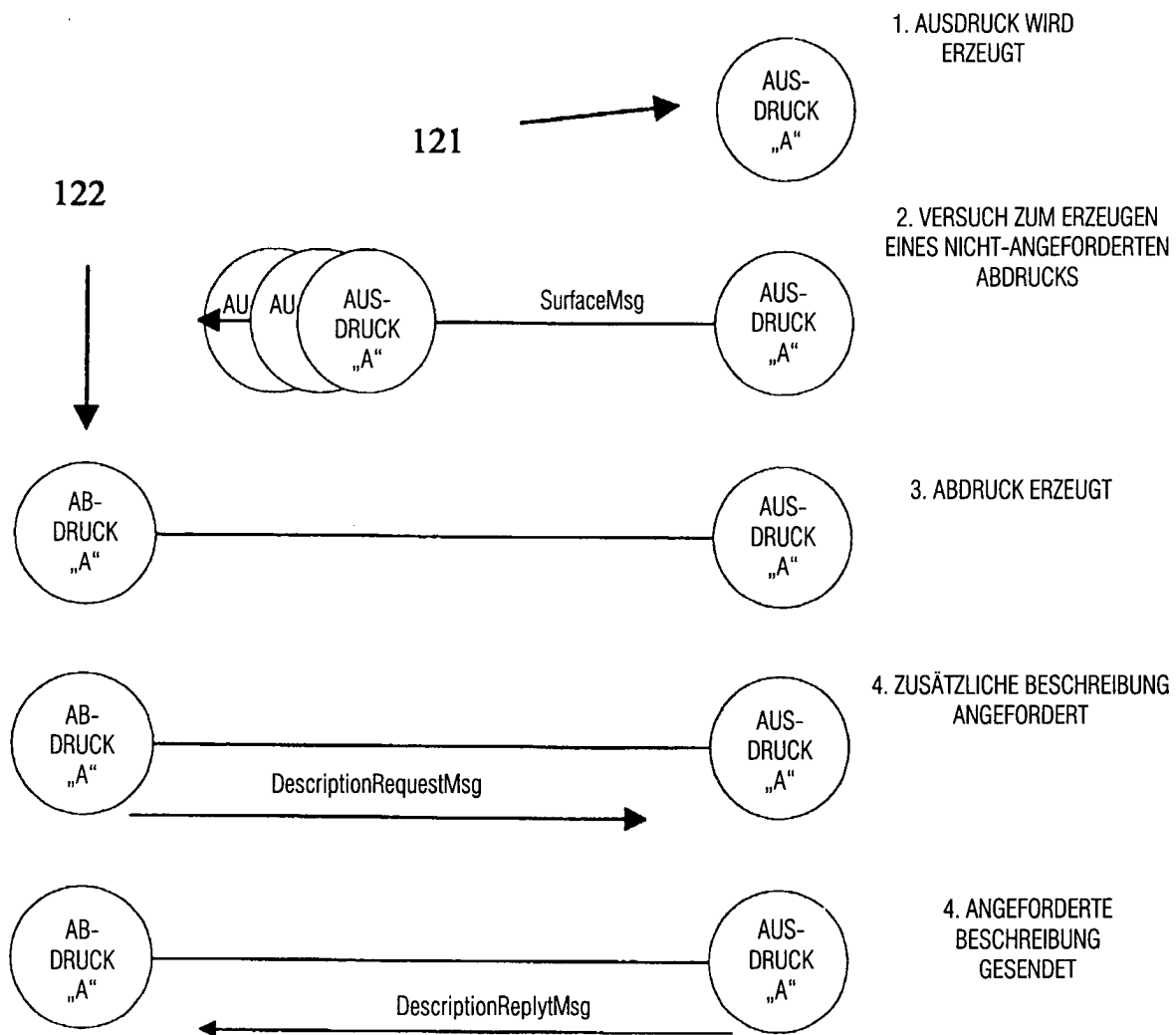
FIGUR 12B



FIGUR 12C

ABDRUCKSVORRICHTUNG

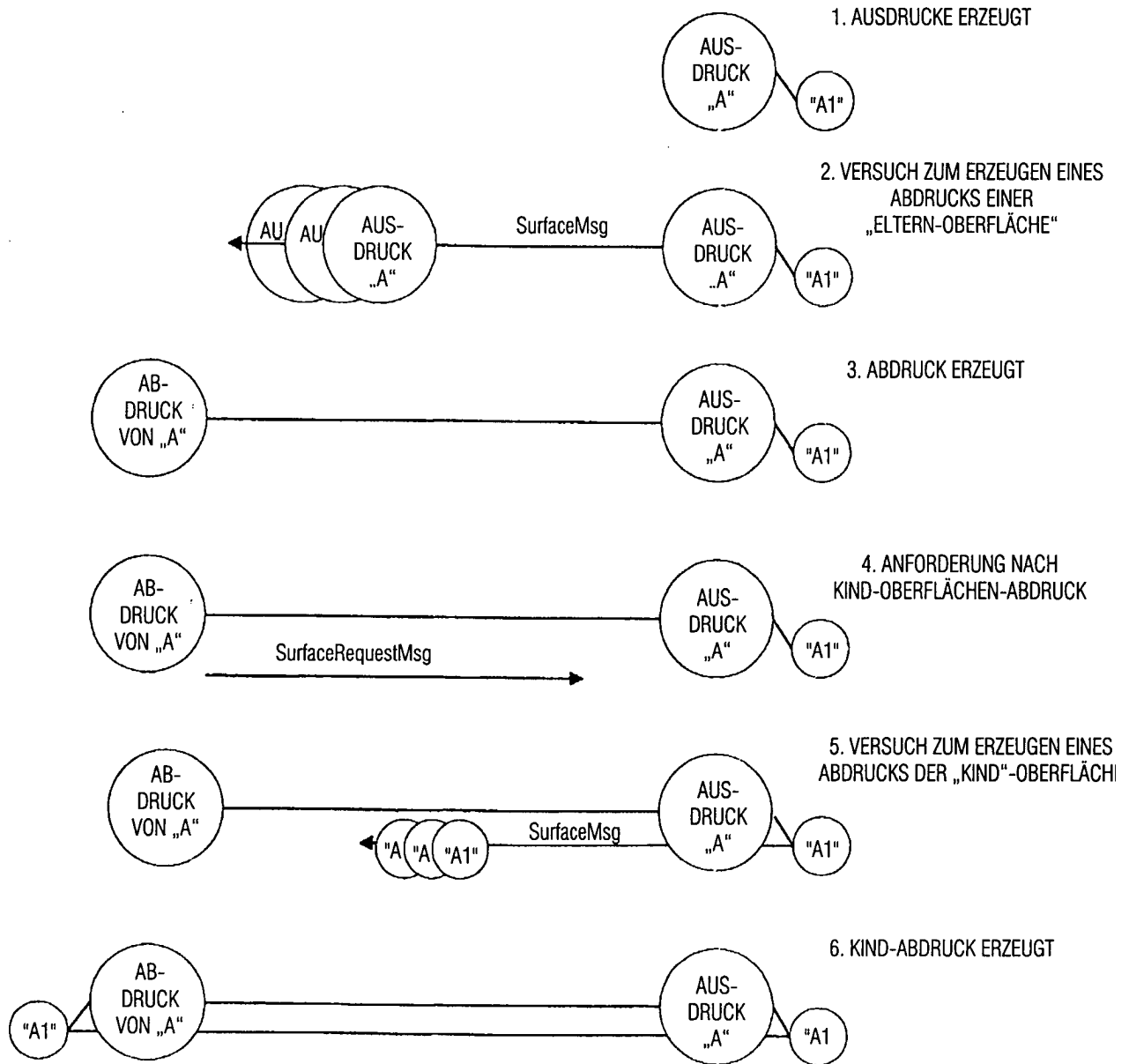
AUSDRUCKSVORRICHTUNG



FIGUR 12D

ABDRUCKSVORRICHTUNG

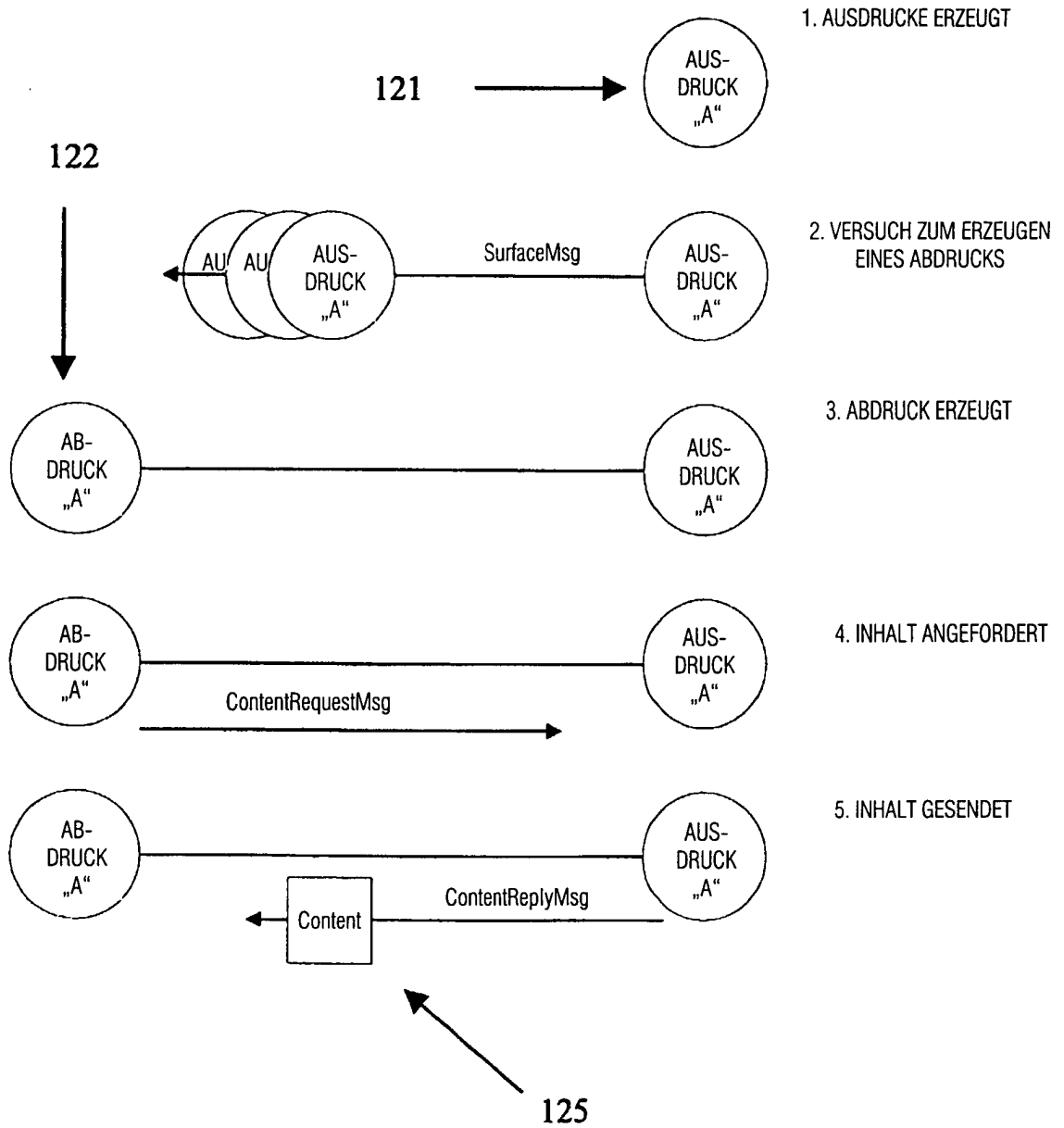
AUSDRUCKSVORRICHTUNG



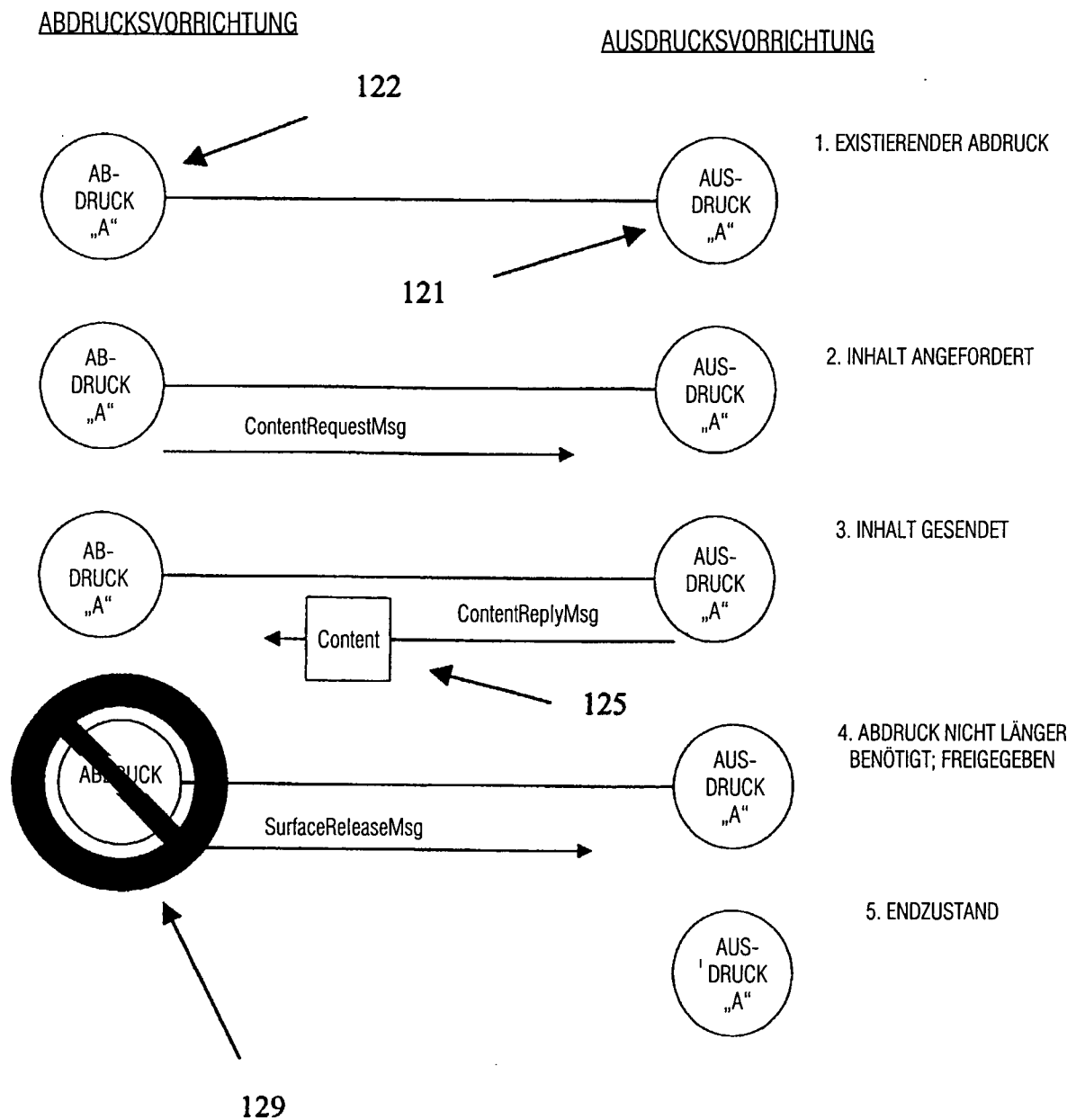
FIGUR 12E

ABDRUCKSVORRICHTUNG

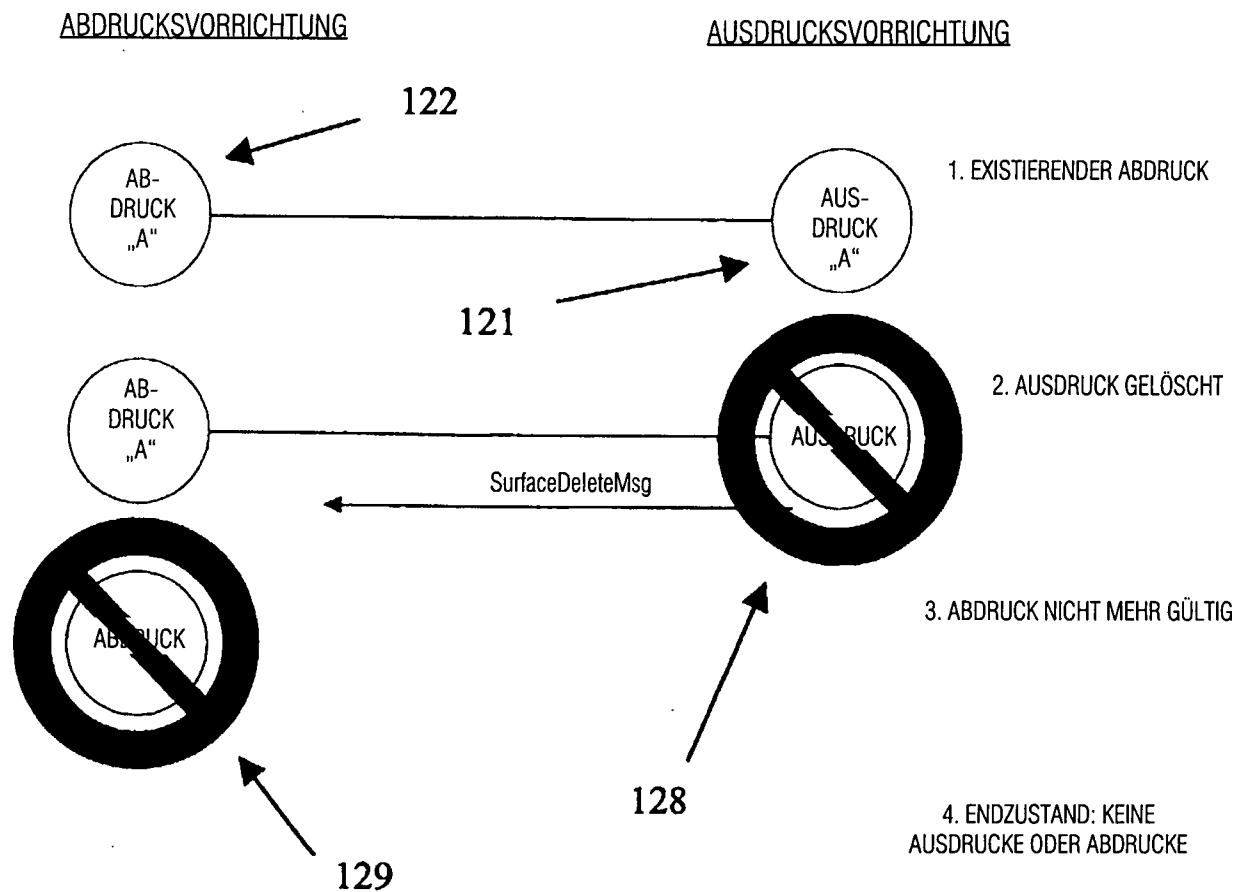
AUSDRUCKSVORRICHTUNG



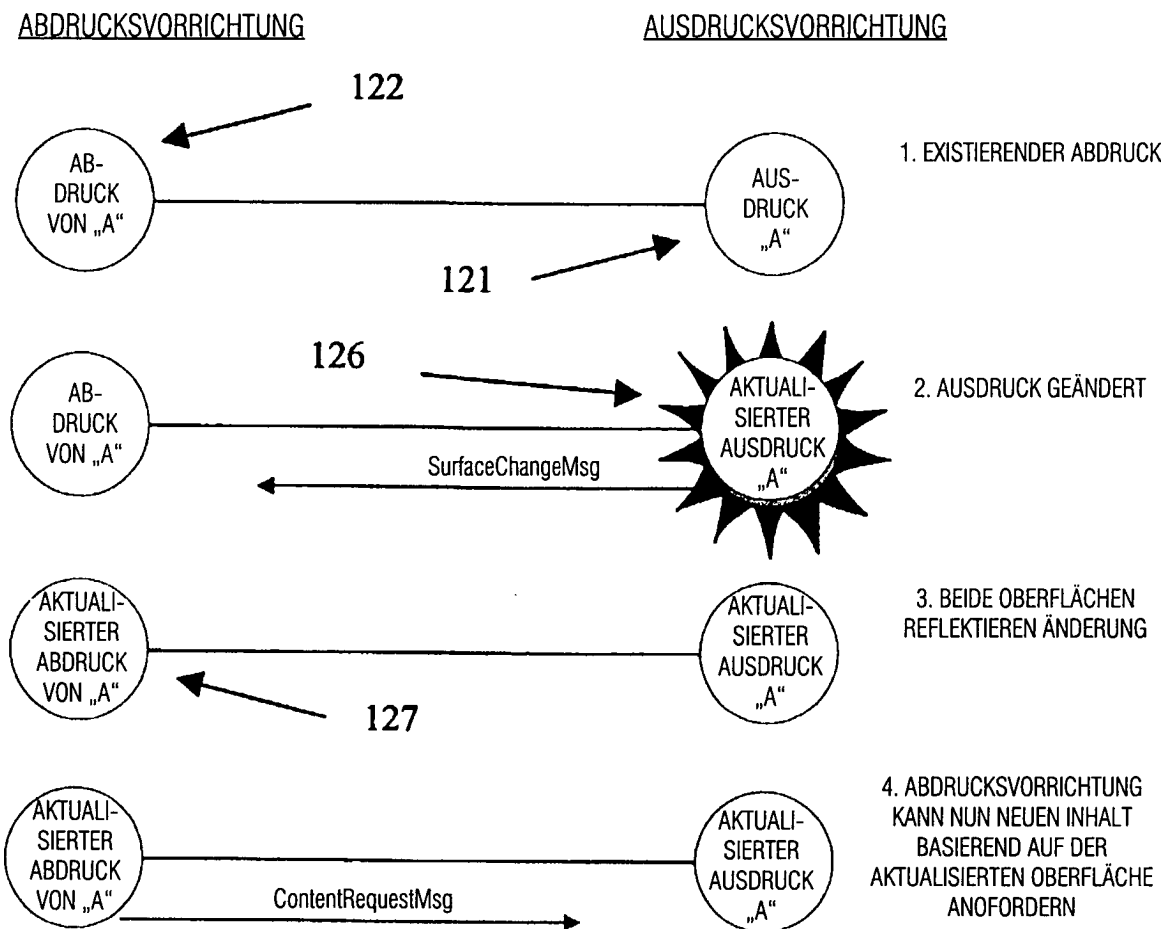
FIGUR 12F



FIGUR 12G



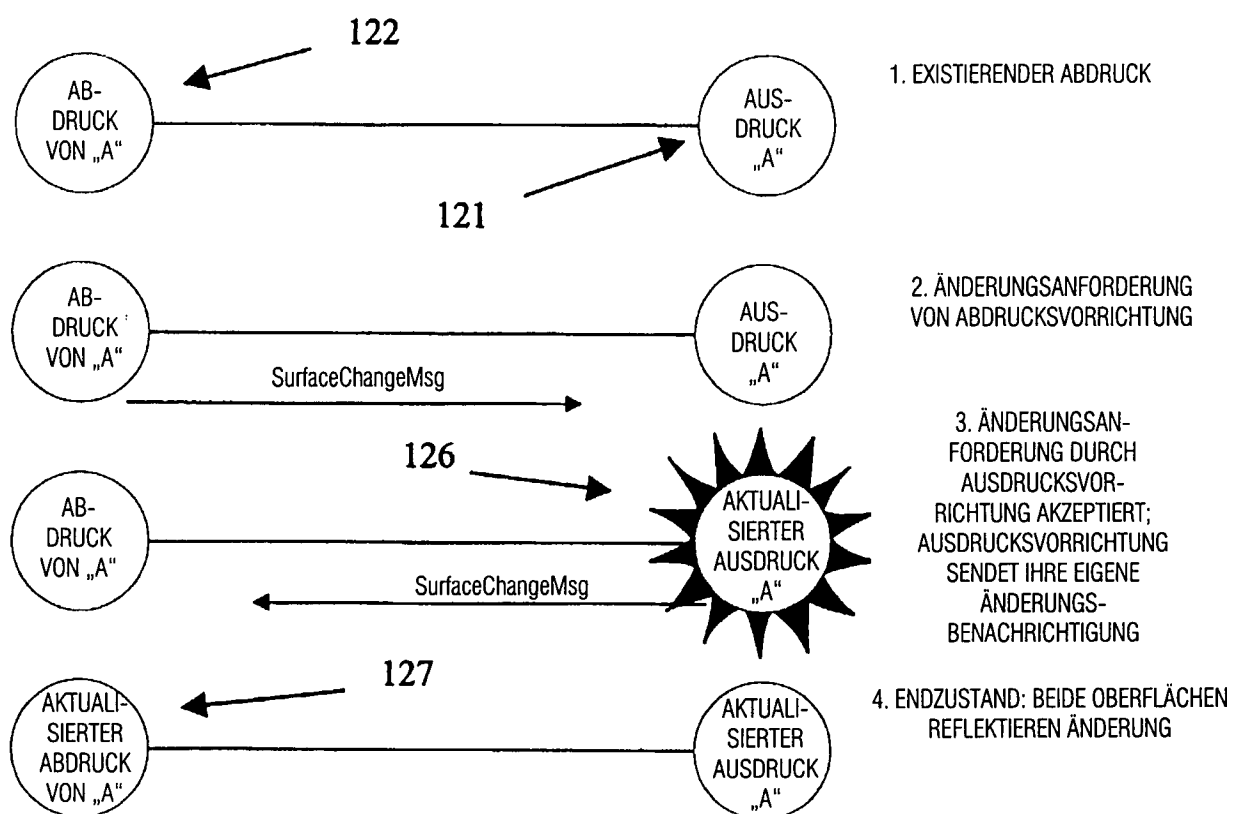
FIGUR 12H



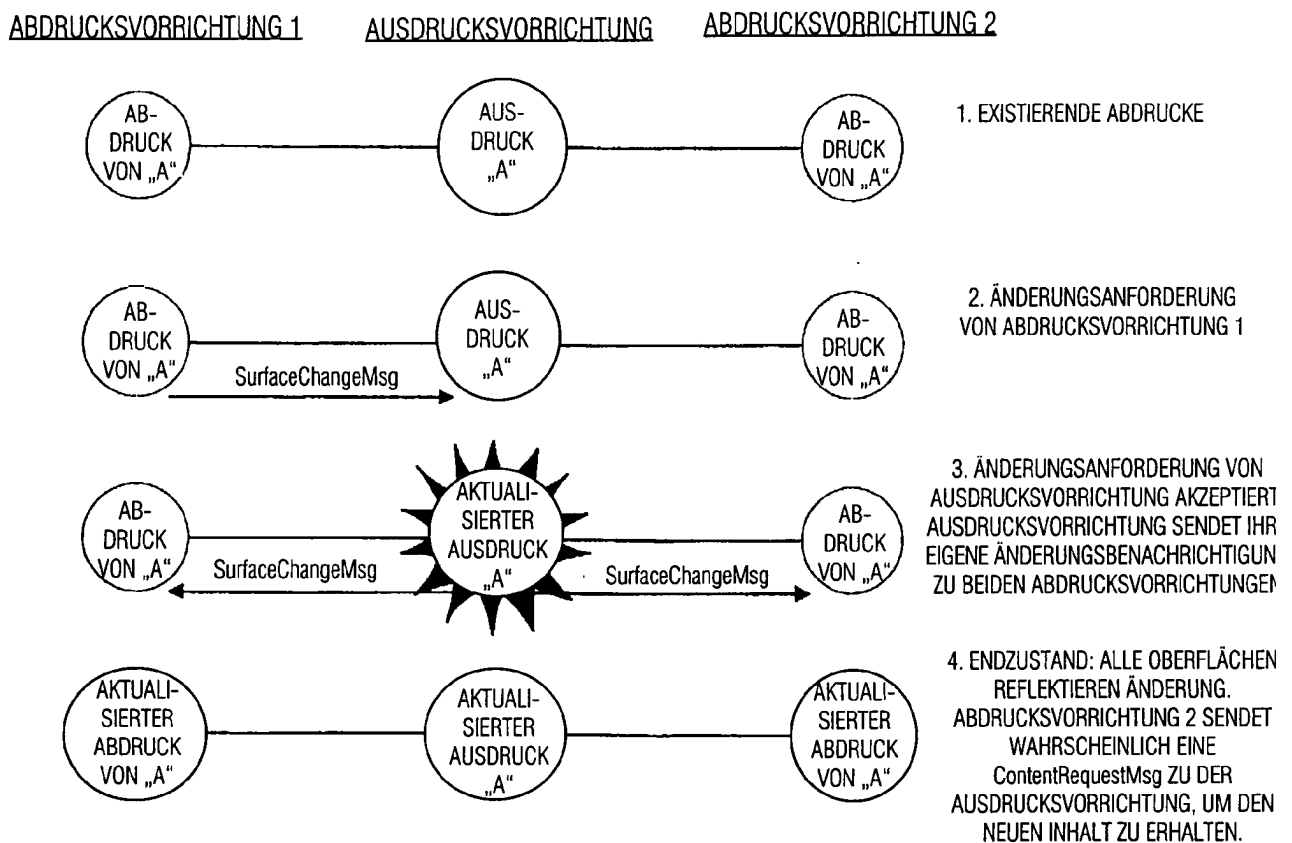
FIGUR 121

ABDRUCKSVORRICHTUNG

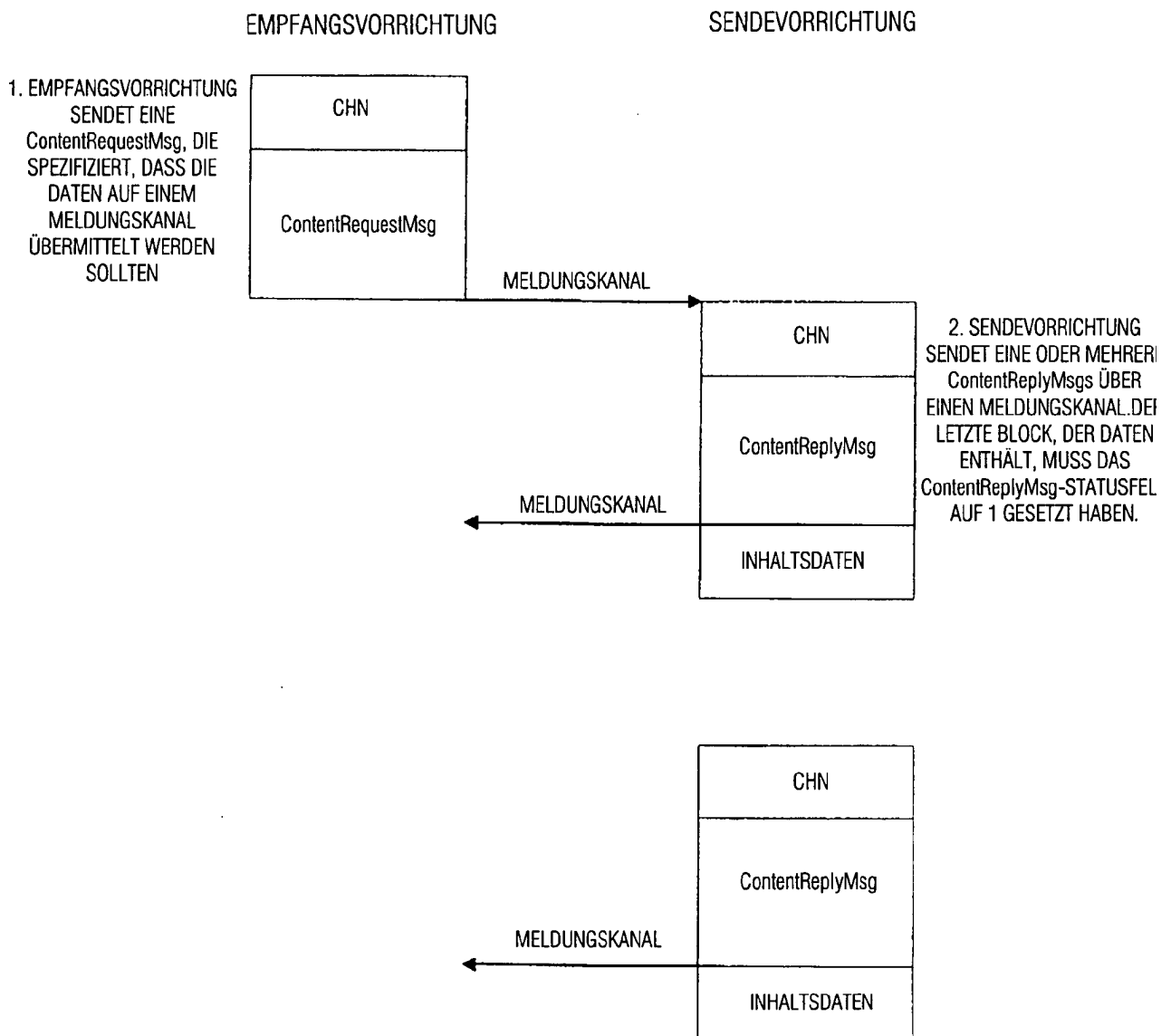
AUSDRUCKSVORRICHTUNG



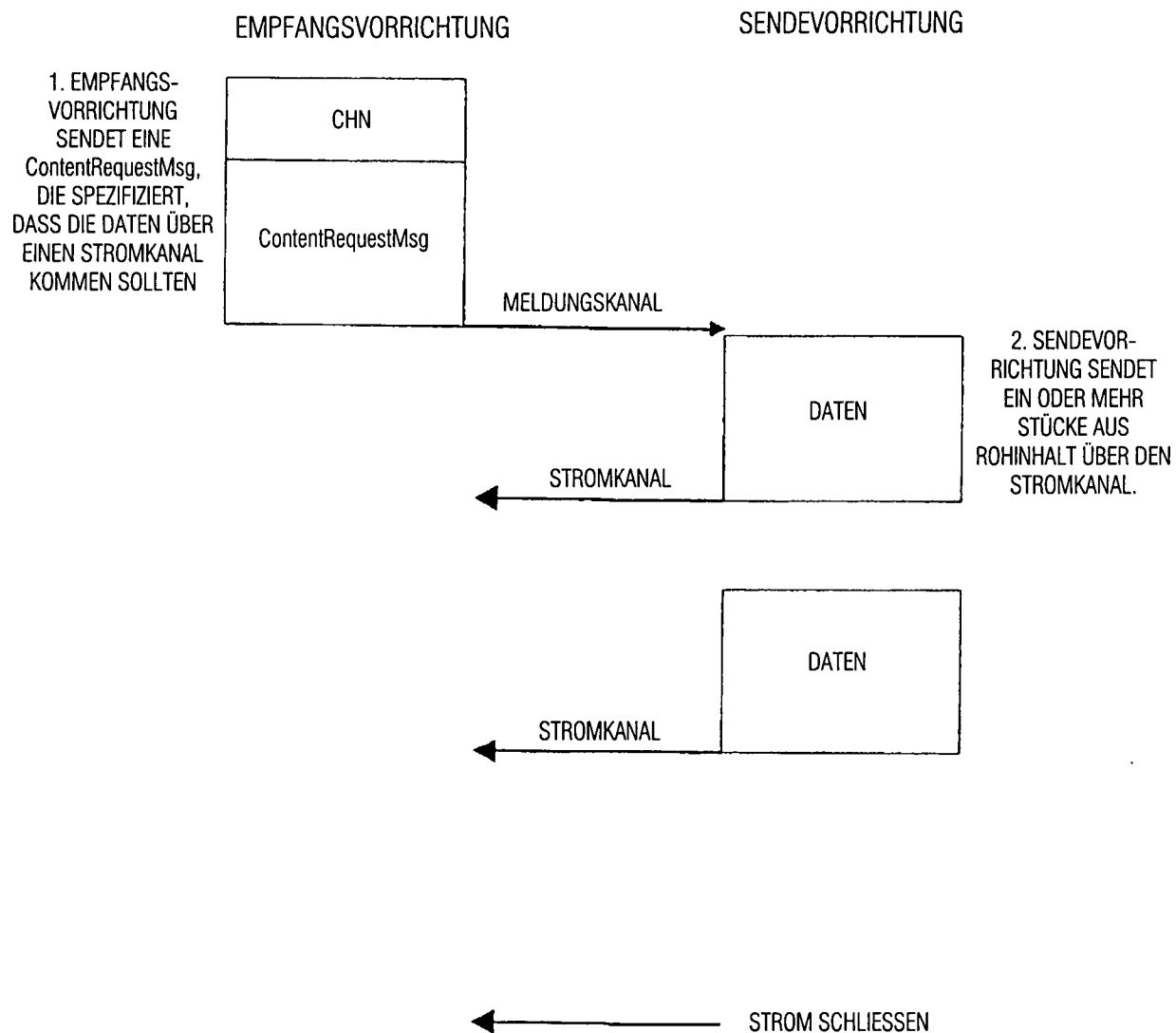
FIGUR 12J



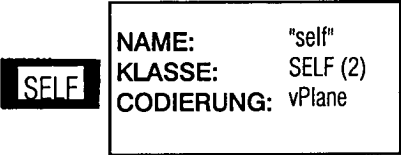
FIGUR 12K



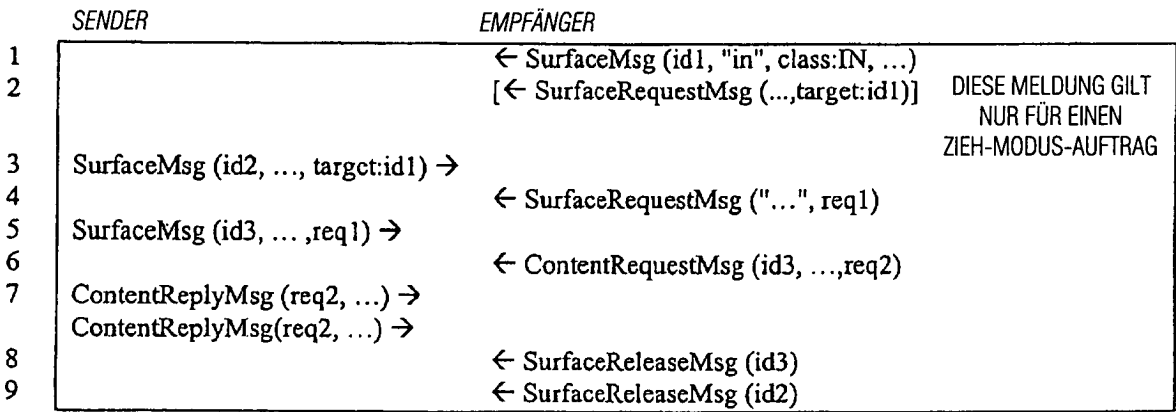
FIGUR 13



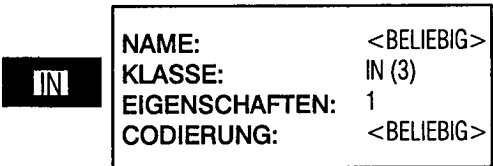
FIGUR 14



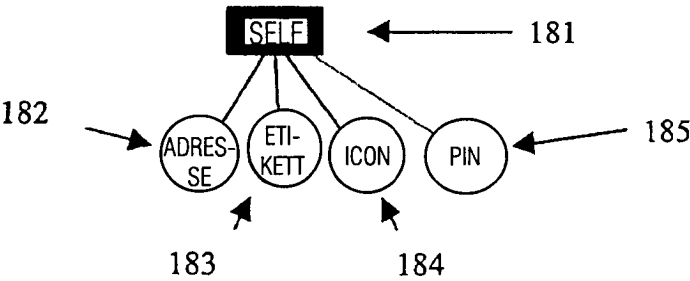
FIGUR 15



FIGUR 16



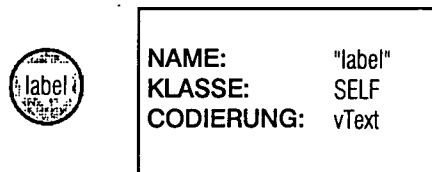
FIGUR 17



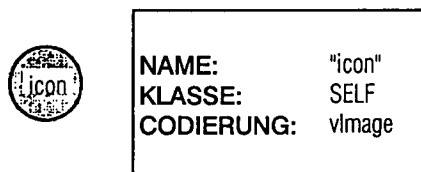
FIGUR 18



FIGUR 19A



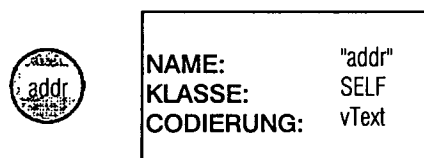
FIGUR 19B



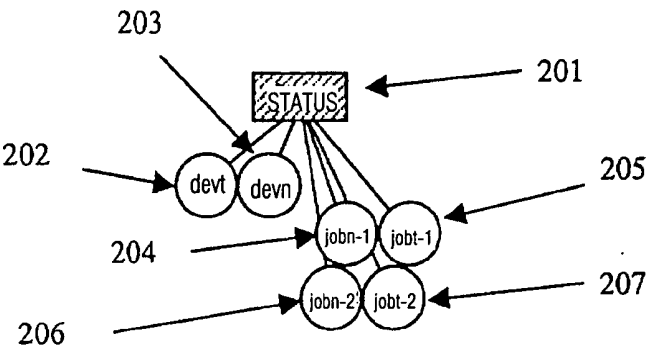
FIGUR 19C



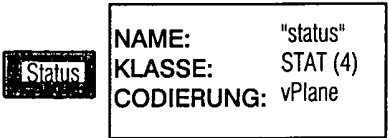
FIGUR 19D



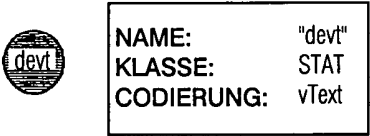
FIGUR 19E



FIGUR 20



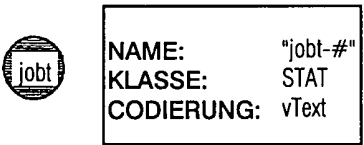
FIGUR 21A



FIGUR 21B



FIGUR 21C



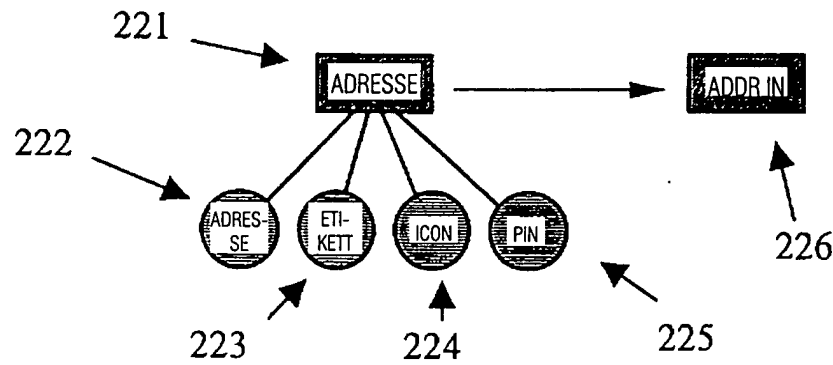
FIGUR 21D



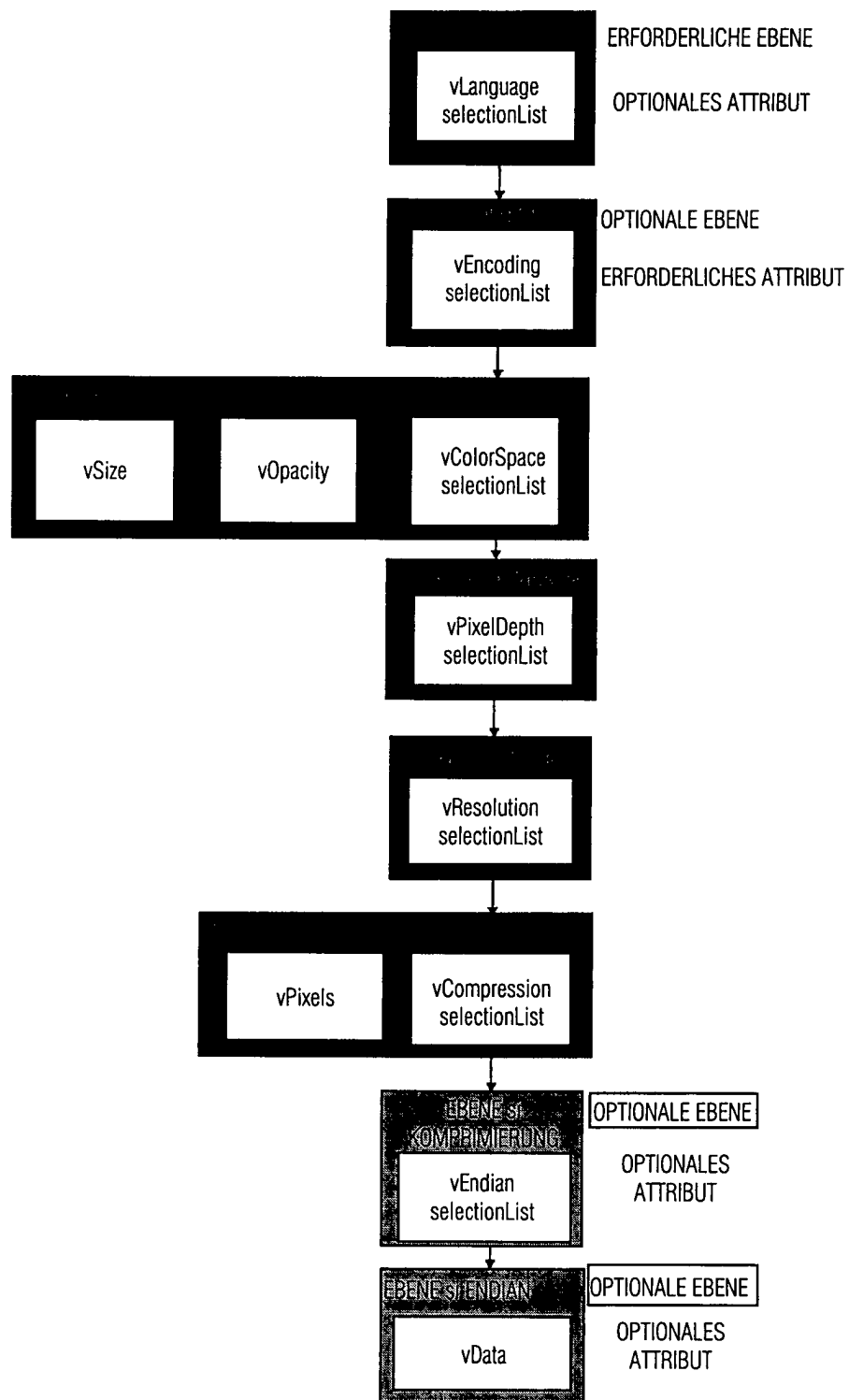
FIGUR 21E



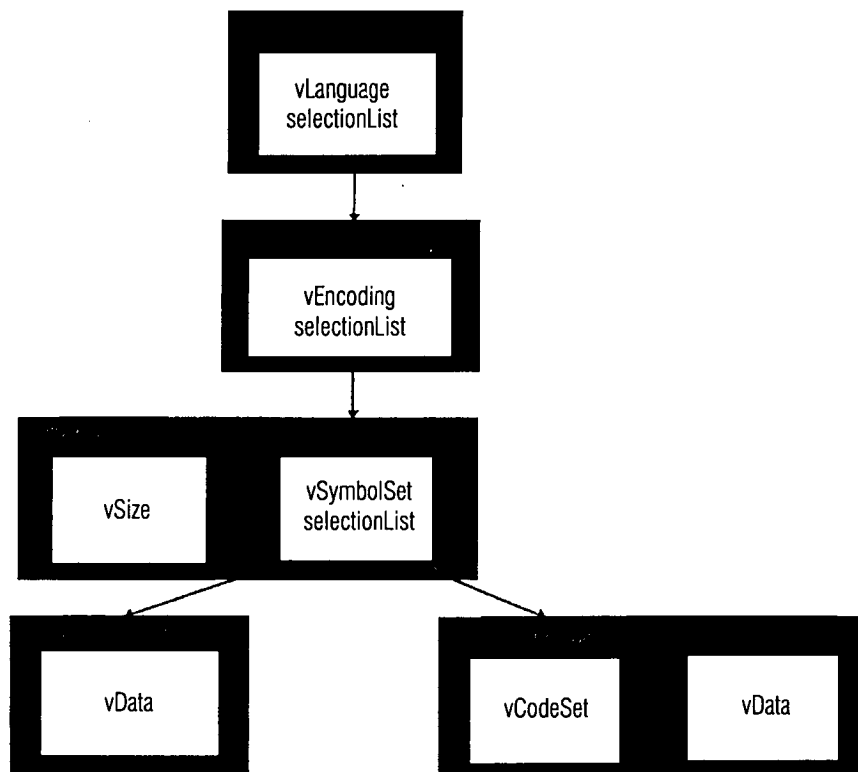
FIGUR 22A



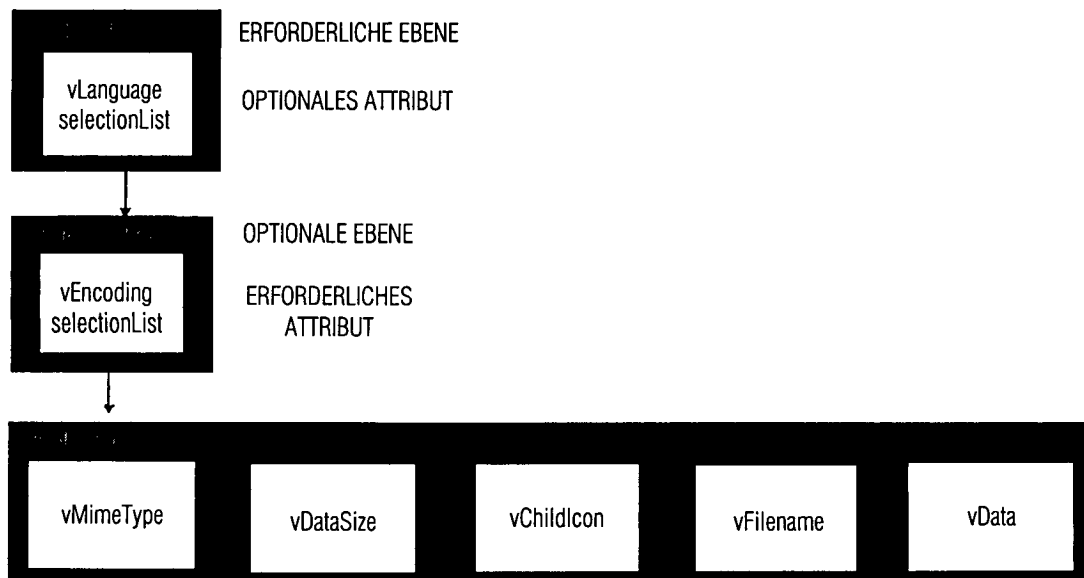
FIGUR 22B



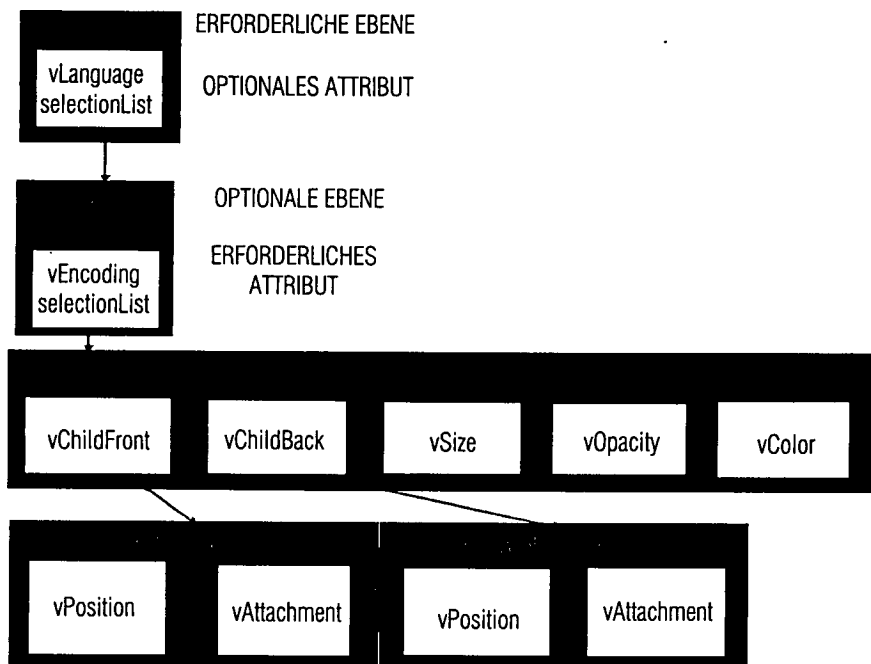
FIGUR 23



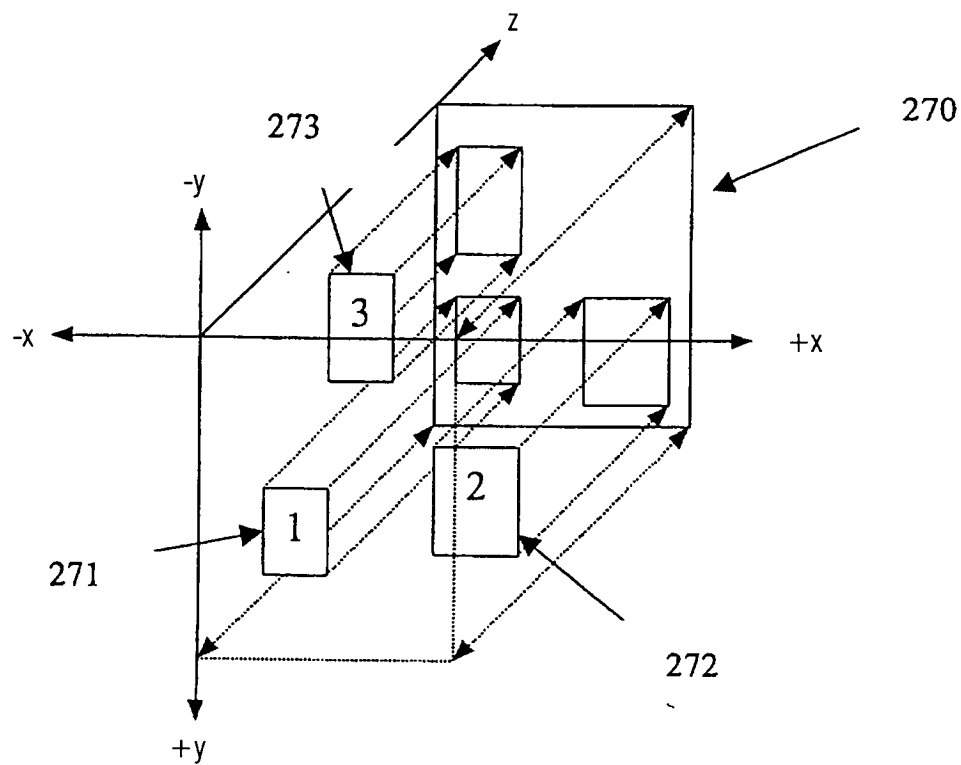
FIGUR 24



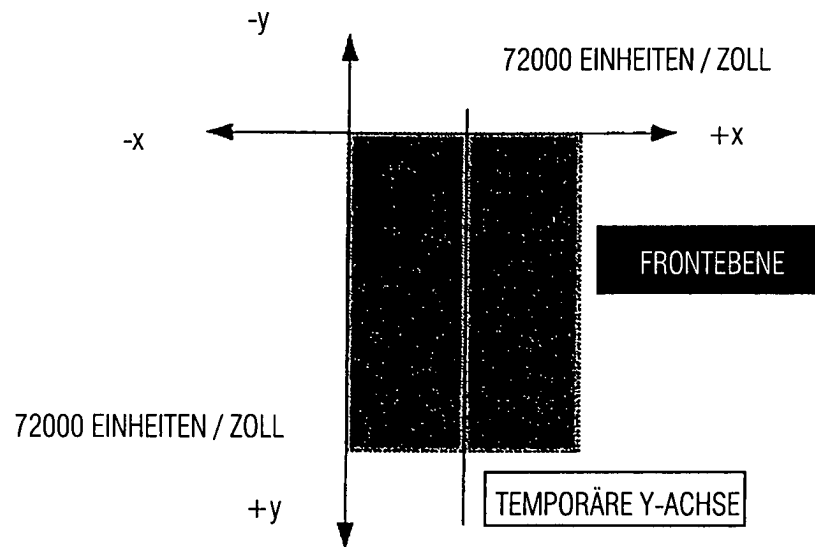
FIGUR 25



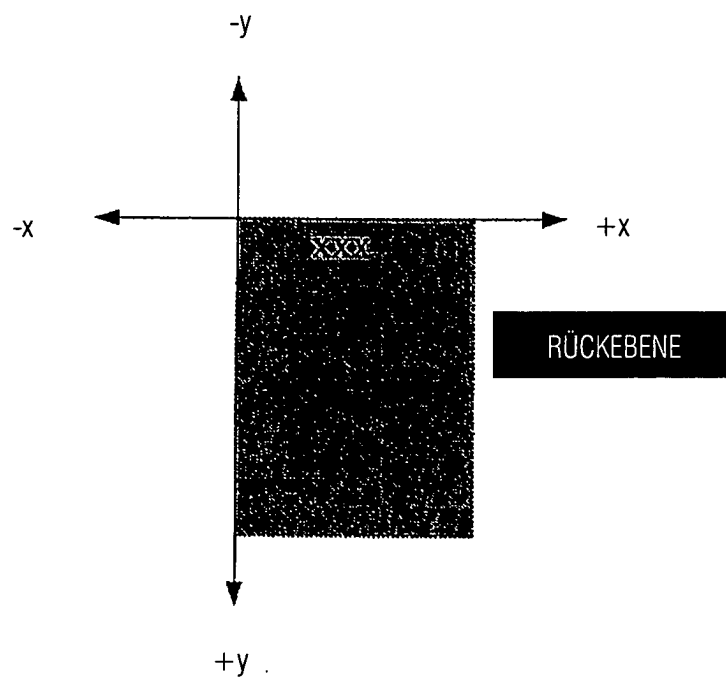
FIGUR 26



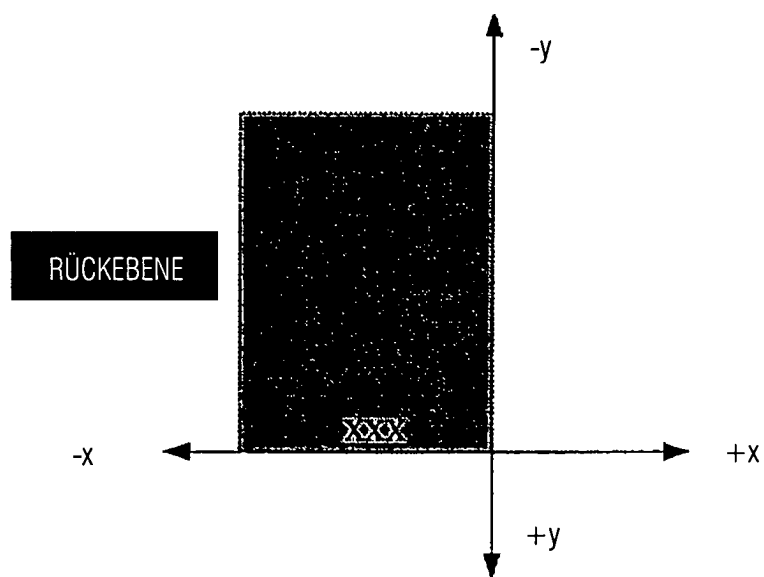
FIGUR 27



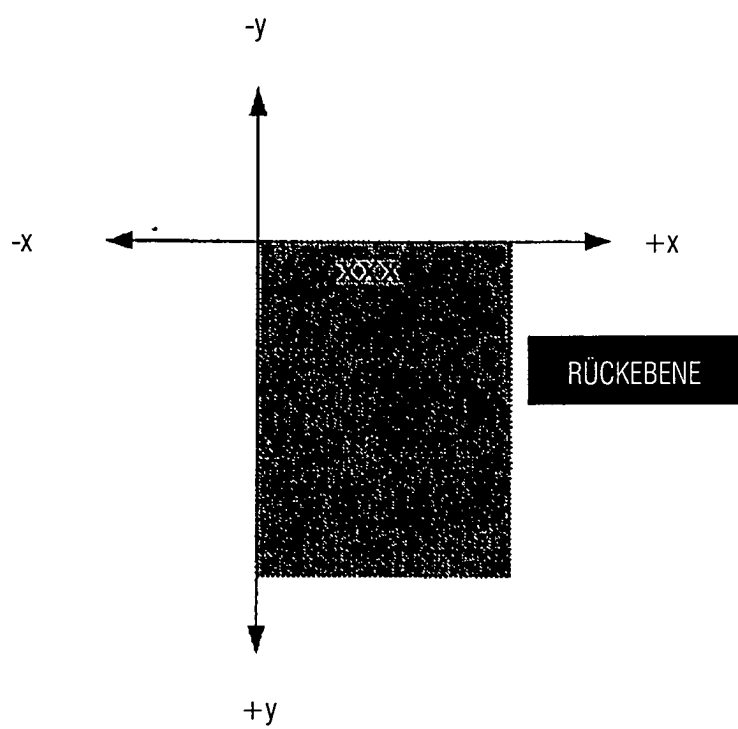
FIGUR 28A



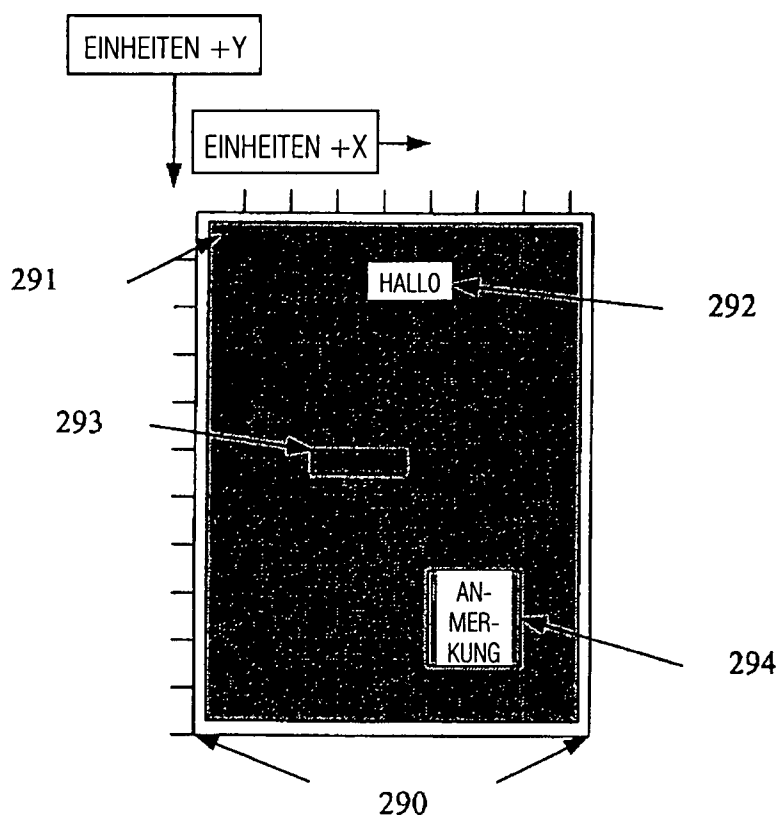
FIGUR 28B



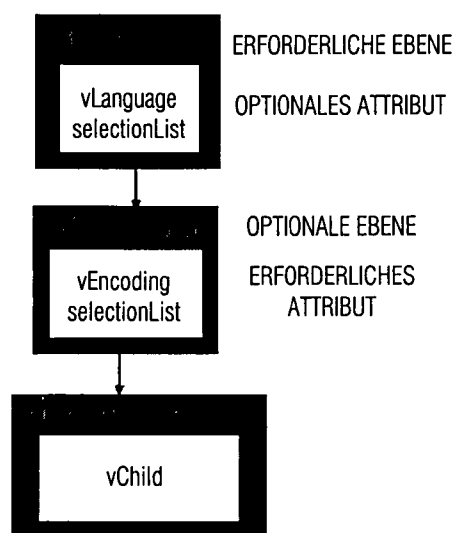
FIGUR 28C



FIGUR 28D'

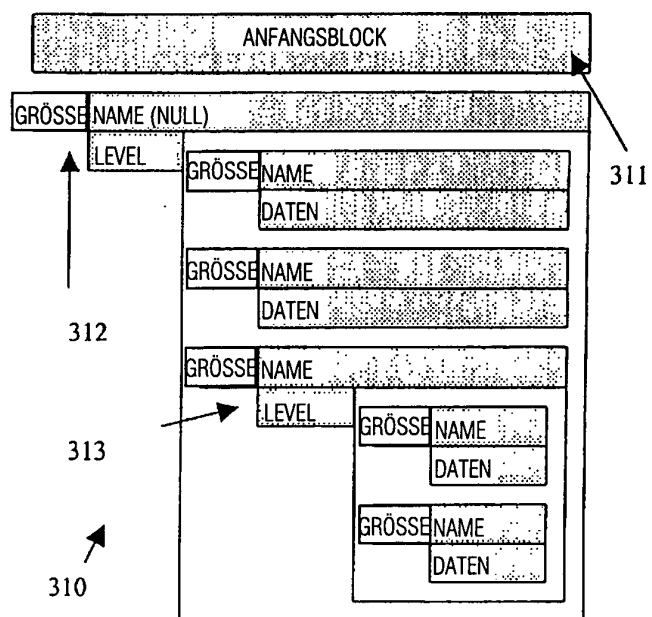


FIGUR 29

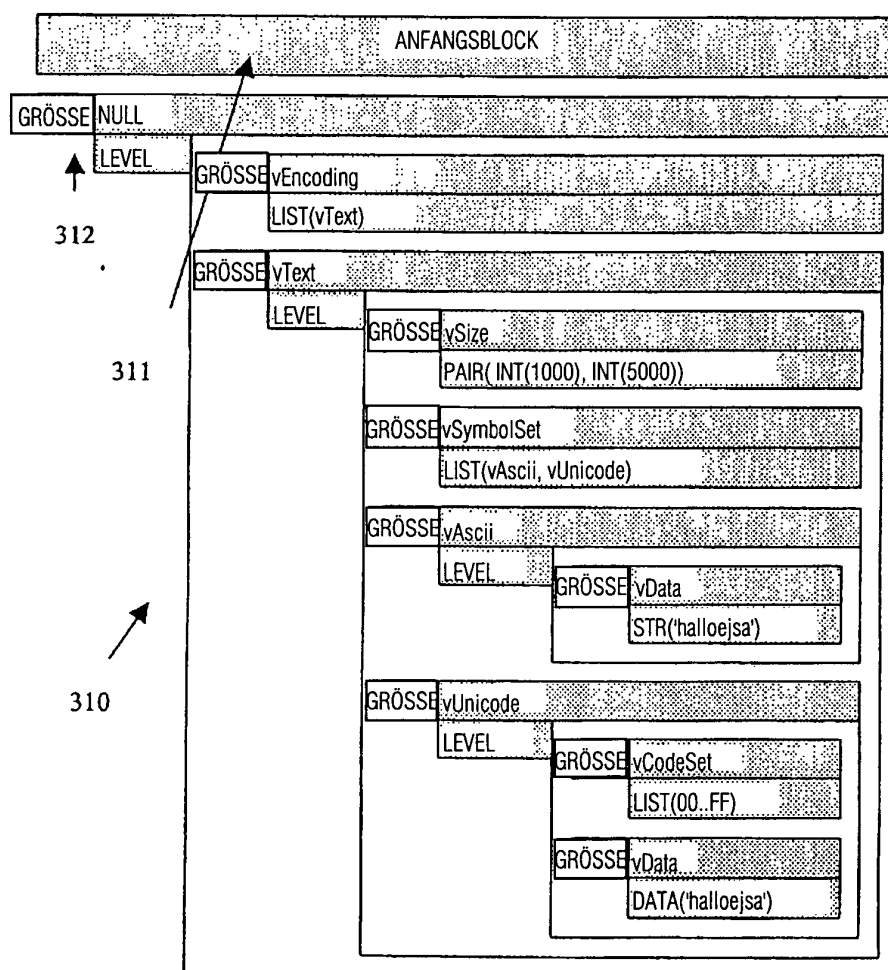


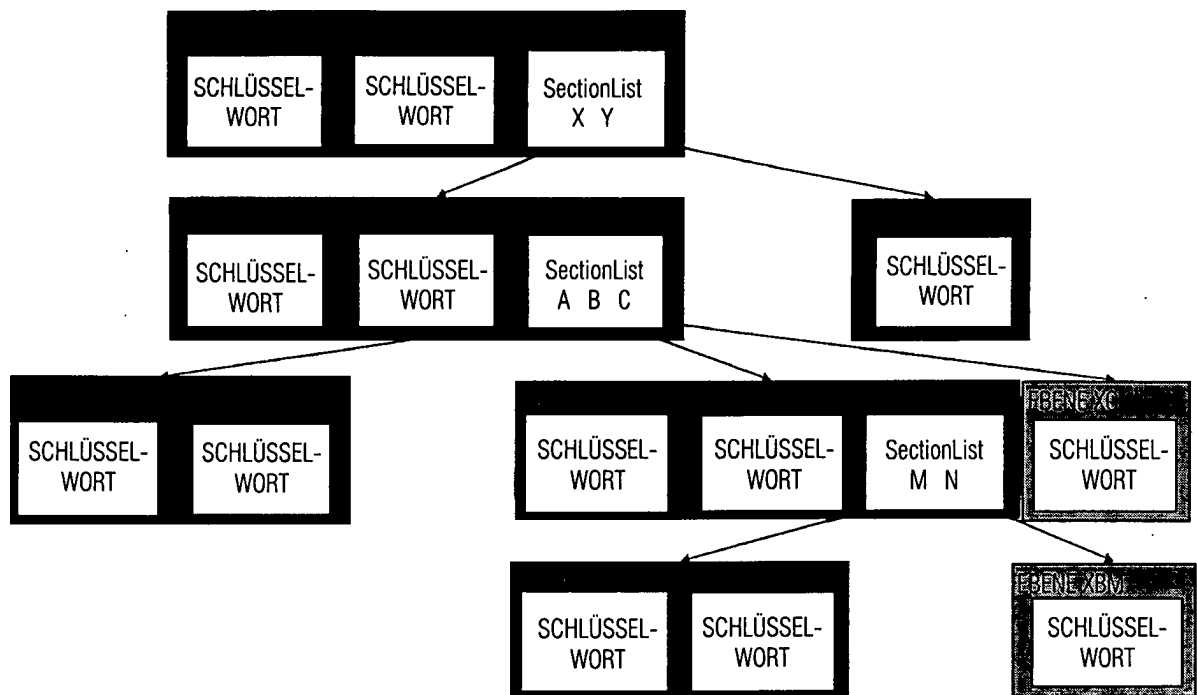
FIGUR 30

FIGUR 31

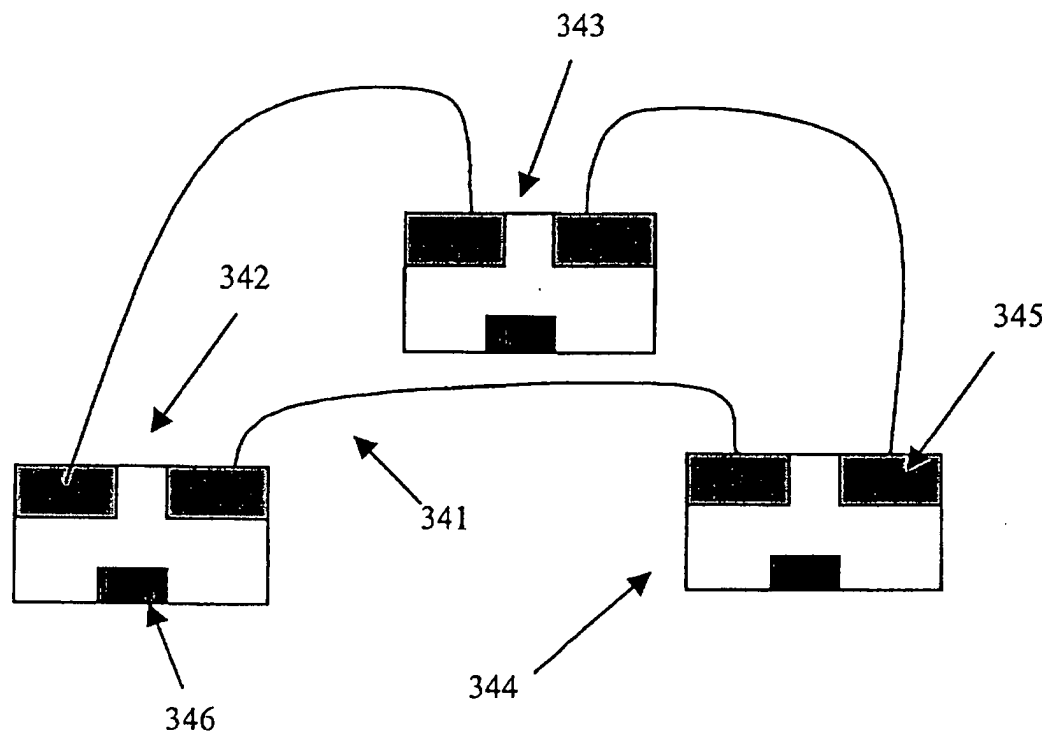


FIGUR 32





FIGUR 33



FIGUR 34