

# SUOMI - FINLAND

*Patentti No 115863*

## PATENTTI- JA REKISTERIHALLITUS

*on tänään myöntänyt 15 päivänä joulukuuta 1967 annetun patenttilain siihen myöhemmin tehtyine muutoksineen nojalla oheisen patenttijulkaisun mukaisen patentin. Patentinhaltijan nimi, keksinnön nimitys ja patenttihakemuksen tekemispäivä käyvät ilmi patenttijulkaisun etusivulta.*

*Helsingissä, 29.07.2005*

*Yukta Laine*





SUOMI - FINLAND  
(FI)

PATENTTI- JA REKISTERIHALLITUS  
-PATENT- OCH REGISTERSTYRELSEN

(12) PATENTTIJULKAISU  
PATENTSKRIFT



(10) FI 115863 B

(45) Patenti myönnetty - Patent beviljats

29.07.2005

(51) Kv.Ik.7 - Int.kl.7

G06F 13/38, H04B 1/00, H04L 29/06

(21) Patentihakemus - Patentansökning

20011874

(22) Hakemispäivä - Ansökningsdag

24.09.2001

(24) Alkupäivä - Löpdag

24.09.2001

(41) Tullut julkiseksi - Blivt offentlig

25.03.2003

(73) Haltija - Innehavare

1 •Gurux Oy, Kaskimäenkatu 1, 33900 Tampere, SUOMI - FINLAND, (FI)

(72) Keksijä - Uppfinnare

1 •Kurunsaari, Mikko, Nikinkatu 24, 33580 Tampere, SUOMI - FINLAND, (FI)

2 •Kakkuri, Lasse, Teiskontie 8 D 44, 33540 Tampere, SUOMI - FINLAND, (FI)

(74) Asiamies - Ombud: Tampereen Patenttitoimisto Oy  
Hermiankatu 12 B, 33720 Tampere

(54) Keksinnön nimitys - Uppfinningens benämning

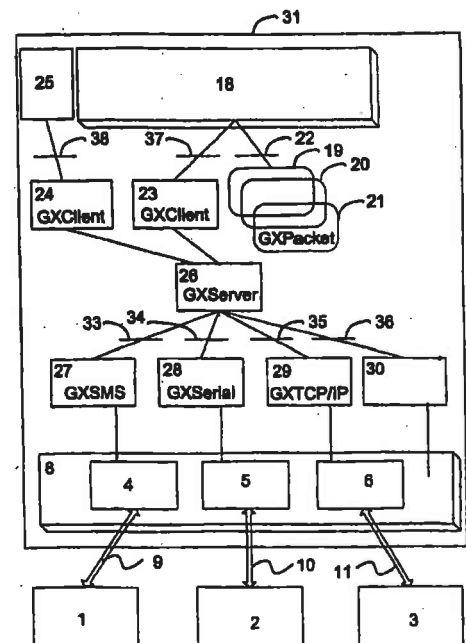
**Systemi ja menetelmä sovelluksen ja tietoliikenneväylän välisen datavirran käsittelemiseksi tietokoneohjatussa ohjausjärjestelmässä**  
**System och förfarande för behandling av en dataström mellan en tillämpning och en datatrafikbuss i ett datorstyrt kontrollsystem**

(56) Viitejulkaisut - Anförda publikationer

DE 20019138 U, WO 0021200 A,  
IEEE Transactions on Consumer Electronics, vol. 44, nro 3, elokuu 1988, P. Warriner et al.: "Proposal for domestic automation networking system", p. 612-616

(57) Tiivistelmä - Sammandrag

Tietokoneohjattu ohjausjärjestelmä ja menetelmä sovelluksen (18, 25) ja tietoliikenneväylän (4, 5, 6, 9, 10, 11) välisen datavirran käsittelemiseksi tietokoneohjatussa ohjausjärjestelmässä (31). Menetelmässä pakataan ohjausjärjestelmässä (31) vastaanoton ja lähetyksen yhteydessä välitettävää datavirtaa yhdeksi tai useammaksi pakettikomponentiksi (19, 20, 21), joka käsittää tarvittavan standardisoidun ohjelmointirajapinnan (22), jonka välityksellä datavirtaa luetaan pakettikomponentista (19, 20, 21) tai kirjoitetaan pakettikomponenttiin (19, 20, 21) kyseisen sovelluksen (18) toimesta. Ohjausjärjestelmä käsittää ainakin käyttöympäristön, joka on tarkoitettu yhden tai useamman ohjaussovelluksen ajamiseksi, ja ohjausvälineet sovelluksen (18, 25) ja tietoliikenneväylän (4, 5, 6, 9, 10, 11) välisen datavirran käsittelemiseksi.



115863

Datorstyrt kontrollsystem och förfarande för behandling av en dataströmning mellan en tillämpning (18, 25) och en kommunikationsbuss (4, 5, 6, 9, 10, 11) i det datorstyrda kontrollsystemet (31). I förfarandet packas i kontrollsystemet (31) dataströmningen som skall förmedlas mellan mottagning och sändning som en eller flera paketkomponent (19, 20, 21) som omfattar det nödvändiga standardiserade programmeringsgränssnittet (22), medelst vilket dataströmningen läses från paketkomponentet (19, 20, 21) eller skrivs i paketkomponentet (19, 20, 21) genom sagda tillämpning (18). Styrsystemet omfattar åtminstone en användningsmiljö som är avsedd för körning av en eller flera styrtillämpningar, och styrmedel för behandling av dataströmningen mellan tillämpningen (18, 25) och kommunikationsbussen (4, 5, 6, 9, 10, 11).

Systeemi ja menetelmä sovelluksen ja tietoliikenneväylän välisen datavirran käsittelemiseksi tietokoneohjatussa ohjausjärjestelmässä

Keksinnön tausta

5

Keksinnön kohteena on patenttivaatimuksen 1 johdanto-osan mukainen menetelmä sovelluksen ja tietoliikenneväylän välisen datavirran käsittelemiseksi tietokoneohjatussa ohjausjärjestelmässä.

10

Keksinnön kohteena on patenttivaatimuksen 10 johdanto-osan mukainen tietokoneohjattu ohjausjärjestelmä.

15

Tietokoneisiin liitettävät teollisuuden laitteistot ja oheislaitteet sekä näihin kiinteästi liittyvät etäkäyttö- ja kontrolliohjelmistot muodostavat merkittävän osan tietoliikennealan markkinoista. Erittäin yleinen ratkaisu sulautettujen järjestelmien ohjaamiseen on tarjota erillinen PC-tietokoneella toimiva ohjelmisto, joka kommunikoi laitteiden kanssa asiakkaan valitsemaa tiedonsiirtoväylää hyväksikäyttäen.

20

Tiedonsiirrossa käytetty protokolla voi vaihdella sovellusalueesta riippuen. Tietokoneohjatulla järjestelmällä saavutetaan merkittäviä etuja ja kustannussäästöjä, sillä vähänkään monimutkaisempi käyttöliittymä on huomattavasti helpompi ja halvempi tehdä PC-tietokoneelle kuin itse ohjattavaan laitteeseen. Lisäksi ohjattava laite voi näin sijaita fyysisesti

25

hyvinkin kaukana ohjauskeskuksesta ja useita laitteita voidaan kontrolloida haluttaessa samalla ohjelmistolla.

30

Kun täysin uutta laitetta aletaan kehittää, yleisin tapa on, että ohjausohjelmiston kehitystyö alkaa täysin alusta. Monet yritykset haluavat lisäksi käyttää laitteidensa ohjauksessa omaa protokollaa, jonka yksityiskohtia ei haluta paljastaa kilpailijoille. Viestirakenteet ja protokollapakettien sisältö saattaa vaihdella hyvinkin paljon jopa saman valmistajan eri tuotteiden välillä. Tämä aiheuttaa yleensä sen, että uusissa järjestelmissä hyödynnetään hyvin vähän valmista tai aikaisemmin

35

käytettyä ohjelmakoodia.

Kuvassa 1 on esitetty yksinkertaistettu malli siitä, miten nykyiset ohjausohjelmistot toimivat. Kuvassa ohjausjärjestelmään 63 on kytketty kolme erilaista laitetta 1, 2 ja 3, joista kukin käyttää erilaista tietoliikenneyhteyttä 4, 5 ja 6. Yhteydenotto ja datayhteyden ylläpito laitteiden 1 – 3 ja sovelluksen 7 välillä tapahtuu käyttöjärjestelmän ja sen tarjoamien tietoliikennepalveluiden 8 avulla. Käyttöjärjestelmän 8 tehtävänä on muuntaa lähetettävä data fyysiselle tiedonsiirtotielle 9 – 11 sopivaksi.

Merkittävää on huomata, että jokaiselle erilaiselle tietoliikenneyhteydelle 4 – 6 joudutaan tekemään sovellusohjelmaan 7 oma lohkonsa 12 – 14, joka osaa kommunikoida käyttöjärjestelmäpalveluiden 8 kanssa. Tähän ratkaisuun on yleensä päädytty, koska käyttöjärjestelmäpalveluiden 8 tarjoama rajapinta (tiedonsiirtoprotokollat 15 – 17) saattaa olla hyvinkin erilainen eri tietoliikenneyhteyksiä verrattaessa.

Rajapintana on tavallisesti ns. API (Application Programming Interface). Esimerkiksi sarjaportin käyttö on ohjelmoijan kannalta täysin eri asia kuin käyttää TCP/IP-yhteyttä, puhumattakaan siitä, että laitteiston valmistaja haluaa käyttää kontrollointiin jotain käyttöjärjestelmän 8 kannalta eksoottisempaa tapaa, kuten esimerkiksi tekstiviestejä (SMS, Short Message Service). Mikäli eroavaisuuksiin ei olla kiinnitetty riittävästi huomiota jo sovelluksen 7 suunnitteluvaiheessa, ratkaisuna käytetään yleisesti myös sitä, että uusia tietoliikenneyhteyksiä varten tehdään kokonaan uusi sovellusohjelma 7.

Tietoliikennesovelluksia tehtäessä lisähaasteita luo myös mahdolliset datapakettien uudelleenlähetykset, vasteaikojen vaihtelu, virhetilanteiden hallinta ja datan puskurointi. Tuotekehityssykliä voidaankin lyhentää merkittävästi käyttämällä keksinnön mukaista järjestelmää, joka ratkaisee nämä ongelmat ja tarjoaa lisäksi rajapinnan, joka on tietoliikenneyhteydestä tai käytetystä protokollasta riippumaton, pysyy vakiona, on tarkoitettu kommunikointiin tietoliikenneyhteyden yli ja joka minimoi täten sovellusohjelman muutostarpeen yhteystyyppien muuttuessa.

35 Keksinnön lyhyt selostus

Keksinnön mukainen menetelmä on esitetty patenttivaatimuksessa 1.

Keksinnön mukainen tietokoneohjattu ohjausjärjestelmä on esitetty patenttivaatimuksessa 10.

5 Järjestelmän komponenttipohjaisen ratkaisun avulla teollisuuden tiedonsiirtojärjestelmien kehitystyö nopeutuu laitteiden ja tietokoneiden kommunikoidessa standardin rajapinnan kautta. Järjestelmä on toteutettavissa erityisesti Windows<sup>®</sup> -ympäristöön ja se tarjoaa helppokäyttöisen ohjelmointirajapinnan tietoliikennealan sovelluskehittäjille.

10 Erityistä etua keksinnöllä saavutetaan järjestelmässä datan vastaanottopuolella. Asiakassovellukselle voidaan tarjota kiinteänä pysyvä rajapinta datan rakenteesta tai käytetystä protokollasta huolimatta. Näin helpotetaan asiakassovelluksen tietoliikenneosion ohjelmointia ja  
15 sen pääsyä erityisesti datapaketin sisältämään otsikko- ja datakenttiin.

#### Kuvioiden lyhyt selostus

20 Keksintöä selostetaan seuraavassa tarkemmin käyttäen esimerkkinä keksinnön erästä edullista suoritusmuotoa, samalla viitaten oheisiin piirustuksiin, joissa:

kuva 1 esittää lohkoakaaviona erästä tunnetun tekniikan mukaista ohjausjärjestelmää ja ohjausohjelmiston toimintaa,

25 kuva 2 esittää lohkoakaaviona keksinnön erään edullisen suoritusmuodon mukaista ohjausjärjestelmää ja ohjausohjelmiston toimintaa,

30 kuva 3 esittää signaalikaaviona tietoliikenneyhteyden avaamista kuvan 2 mukaisessa järjestelmässä,

kuva 4 esittää signaalikaaviona datan lähettämistä kuvan 2 mukaisessa järjestelmässä, ja

35 kuva 5 esittää signaalikaaviona datan vastaanottamista kuvan 2 mukaisessa järjestelmässä.

## Keksinnön yksityiskohtainen selostus

5 Keksinnön mukainen järjestelmä koostuu ohjelmakomponenteista, jotka seuraavassa esimerkissä on toteutettu COM-teknologiaa hyväksikäyttäen, jonka on kehittänyt Microsoft Corporation. COM-teknologiaa (Component Object Model) ja sen tarjoamia etuja on kuvattu mm. EP-hakemusjulkaisussa 0 778 688 A2, sekä alan kirjallisuudessa. Keksintö on toteutettavissa COM-teknologian sijasta myös käyttäen CORBA-

10 komponentteja (Common Object Request Broker Architecture) tai olio-ohjelmointimenetelmiä. Järjestelmän toteuttaminen näitä muita menetelmiä käyttäen on alan ammattimiehelle selvää tämän selostuksen perusteella. Etuna on se, että COM-teknologia on CORBA-teknologiaa yleisemmin käytetty ja oliomenetelmiin verrattuna COM-teknologia tarjoaa ohjelmakoodin helpomman uudelleenkäytettävyyden ja tuen useammille ohjelmointikielille. Järjestelmä ei ole myöskään riippuvainen käyttöjärjestelmästä.

Järjestelmän eri komponentit on esitetty kuvassa 2. Järjestelmästä käytetään nimitystä GXCom. Sovellusohjelman 18 vastaanottama ja lähettämä data on pakattu komponentteihin 19 – 21, joista kustakin käytetään nimitystä GXPacket-komponentti. Datan lisääminen tai lukeminen GXPacket-komponentista 19 – 21 käy helposti komponentin rajapinnasta 22 löytyvien metodien avulla. GXPacket-komponentin 19 – 21 avulla varmistetaan, että sovellusohjelman 18 käyttämä protokolla ei vaikuta muun GXCom-järjestelmän toteutukseen. GXCom-järjestelmä ei siis tunne asiakkaan (ts. sovelluksen 18) käyttämää protokollaa vaan käsittelee kaiken tiedon GXPacket-komponentteina 19 - 21. Tämän ansiosta myös sovellusohjelman 18 rajapinta 37 järjestelmään pysyy muuttumattomana vaikka protokolla muuttuisikin. GXCom-järjestelmä luo GXPacket-komponentteja 19 – 21 ajon aikana yhden tai useampia sovellusohjelman 18 pyyntöjen mukaan.

35 Ns. GXClient-komponentti 23 tarjoaa sovellusohjelmalle 18 rajapinnan 37 järjestelmään. Pakettien 19 – 21 luonti, lähettäminen ja vastaanotto sekä järjestelmän asetusten määrittely tapahtuvat kaikki tämän komponentin 23 kautta. Jokainen asiakassovellus 18 voi täten luoda oman

instanssin kevytrakenteisesta GXClient-komponentista. Esimerkkinä on käytetty toista GXClient-komponenttia 24 ja sovellusta 25. Paljon suuremmasta ns. GXServer-komponentista 26 on tietokoneen muistissa vain yksi instanssi, johon tietokoneeseen 31 järjestelmä ohjelmakomponentteineen on toteutettu. Tämä on tärkeää mm. sen takia, että näin GXServer-komponentti 26 pystyy järjestämään keskitetyn tehokkaasti resurssien jaon mm. eri sovelluksien 18, 25 kesken. Toteutustapa mahdollistaa esimerkiksi sen, että kaksi asiakassovellusta voi olla yhteydessä samaan tietoliikennemediaan yhtä aikaa.

10

Joissakin tunnetuissa käyttöjärjestelmissä toimintaa on esimerkiksi rajoitettu siten, että sarjaportin saa varata vain yksi sovellus kerrallaan. GXCom-järjestelmän avulla sarjaportti on varattavissa keskitetysti järjestelmän toimesta kaikkien sovellusohjelmien käyttöön, eikä kukin sovellusohjelma tee sitä erikseen. Asiakasohjelma 18, 25 voi myös halutessaan järjestelmän asetusten avulla erikseen määrätä, että sarjaporttia ei saa käyttää kuin kyseinen sovellus yksinään, mikäli halutaan varmistua tietoliikennekapasiteetin riittävydestä.

20

Järjestelmän ytimenä on GXServer-komponentti 26. Sen päätehtävänä on lähetystilanteessa ottaa vastaan GXClient-komponentin 23 välittämiä GXPacket-komponentteja 19 – 21, hakea niiden sisältä asiakkaan syöttämä data ja lähettää data tavujonona edelleen mediakomponenteille 27 – 30, jotka huolehtivat datan välittämisestä käyttöjärjestelmäpalveluiden 8 avustuksella fyysiselle tietoliikenneväylälle (kuvan 1 mukaiset tietoliikenneyhteydet 4 – 6 ja fyysiset tiedonsiirtotiet 9 – 11). Vastaanottotilanteessa GXServer-komponentti 26 osaa muuntaa mediakomponenteilta 27 – 30 tulevan datan GXPacket-komponenteiksi 19 – 21 sille asetettujen parametrien ansiosta. Paketit 19 – 21 välitetään vastaanottotilanteessa edelleen GXClient-komponentille 23, joka taas välittää ne edelleen asiakassovellukselle 18.

30

35

Mediakomponenttien 27 – 30 tehtävänä on lähetystilanteessa välittää GXServer-komponentin 26 tarjoama datavirta fyysiselle siirtotielle, jolloin se käyttää hyväksi käyttöjärjestelmäpalveluita 8, ja vastaanottotilanteessa taas muuntaa käyttöjärjestelmäpalveluilta 8 tuleva data datavirraksi. Muunnosta varten jokainen mediakomponentti 27 – 30 sisältää

ns. Properties-metodin, jonka avulla tietoliikenneyhteyspesifisiä parametreja voidaan muokata suoraan asiakassovelluksesta 18. GXServer-komponentti 26 välittää yhteyttä muodostettaessa yhteysmuotoa vastaavan jonkin mediakomponentin 27 – 30 rajapinnan suoraan asiakassovellukselle 18. Esimerkiksi sarjaporttiyhteyttä muodostettaessa mediakomponentille 28 tulee kertoa mm. liikennöinnissä käytetty baudinopeus ja databittien lukumäärä. Datavirta koostuu bittimuotoisesta informaatiosta, joka on jaettavissa tavuihin niiden sisältämän informaation käsittelyä varten

Kuvassa 3 on tarkemmin kuvattu miten asiakassovellus 18 avaa tietoliikenneyhteyden. Tietoliikennemediaksi on tässä esimerkissä valittu sarjaporttiyhteys 28, mutta muut yhteydet muodostetaan aivan samalla tavalla vain tietoliikenneparametrien vaihdellessa.

Aluksi asiakassovellus 18 kutsuu GXClient-komponentin 23 ns. SelectMedia-metodia 32, jolle parametrina annetaan haluttu yhteystyyppi. Esimerkkitapauksessa käytetty yhteystyyppi on COM1 (SelectMedia(COM1)), joka tarkoittaa sitä, että halutaan käyttää sarjaporttia numero yksi. Järjestelmässä on määriteltynä joukko eri yhteystyyppejä, joita jokaista vastaa tietty mediakomponentti 27 - 30. Sarjaporttiyhteystyyppejä vastaa ns. GXSerial-mediakomponentti 28, kun taas TCP/IP-yhteydelle löytyy ns. GXTCP-IP-mediakomponentti 29. Mediakomponentteja ja uusia yhteystyyppejä pystytään helposti lisäämään järjestelmään jälkikäteen asiakkaiden vaatimusten mukaan. Tämä on mahdollistettu sillä, että kaikissa mediakomponenteissa on identtiset rajapinnat 33 - 36. Näin voidaan minimoida uuden mediakomponentin lisäyksen aiheuttamat muutostarpeet järjestelmän muihin komponentteihin. Asiakassovellukset 18, 25 voivat kysyä järjestelmän sen hetkisen version tukemat yhteystyypit GXClient-komponentin 23, 24 rajapinnasta 37, 38 löytyvillä metodeilla. Järjestelmän arkkitehtuurin etuna on laajennettavuuden lisäksi se, että eri asiakasryhmille voidaan tarjota erilaisia asennuspaketteja, joista löytyy mediakomponenttien 27 – 30 muodossa tuki eri yhteystyypeille.

Sen jälkeen kun SelectMedia-metodia 32 on kutsuttu, GXClient-komponentti 23 välittää mediakomponentin luontipyyynnön 39 GXServer-

komponentille 26 (GetMedia(COM1)). GXServer-komponentin 26 tehtävänä on luoda (vaihe 40) varsinainen mediakomponentti eli tässä tapauksessa GXSerial-komponentti 28, ja palauttaa osoitin (vaihe 41) sen rajapintaan 34 GXClient-komponentille 23, joka taas välittää (vaihe 5 42) sen edelleen asiakassovellukselle 18. Saamansa mediakomponentin 23 rajapintaosoittimen kautta asiakassovellus 18 pystyy asettamaan (vaihe 43) yhteystyyppin vaatimat asetukset Properties-metodilla, joka löytyy jokaisesta mediakomponentista 27 – 30. Metodi on toteutettu siten, että sille ei anneta parametreja lainkaan, vaan se 10 avaa asetussivun, jonka avulla yhteystyyppikohtaiset asetukset voidaan asettaa ja tallettaa mediakomponenttiin. Tähän ratkaisuun on päädytty, koska jokaisella mediakomponentilla on erilainen, joskus suurikin joukko asetuksia. Ratkaisun avulla mediakomponenttien rajapinnat voidaan pitää identtisinä myös parametrien osalta, jotta 15 myös asiakassovelluksen muutostarpeiden määrä olisi mahdollisimman pieni, kun uusi yhteystyyppi lisätään järjestelmään jälkikäteen.

Kun yhteystyyppin haluamat asetukset on asetettu, asiakassovellus 18 kutsuu ns. AssignMedia-metodia 44, jonka ansiosta GXServer-komponentti 26 luo mediakomponentista 28 kopion (AssignMedia(GXSerial) 20 itselleen (vaiheet 45 ja 46) yhteydenmuodostusta ja datan siirtoa varten. Toteutus on järjestetty siten, että metodin kutsumisen jälkeen median asetuksia ei voi enää muokata sulkematta ensin kaikkia mediakomponentin 28 kautta kulkevia yhteyksiä. Näin varmistetaan se, 25 että vain yksi sovellusohjelmista pääsee tekemään tietoliikenneasetukset eikä muut sovellusohjelmat pääse muokkaamaan niitä yhteyden ollessa päällä, kun usea asiakasohjelma on käynnissä yhtä aikaa. Yhteys suljetaan lopuksi GXClient-komponentin 23 ns. Close-metodilla.

30 Kun dataa lähetetään tietoliikenneyhteyden yli, järjestelmä muuntaa datapaketit tietoliikenneyhteyden vaatimaan muotoon ja huolehtii datan puskuroinnista, uudelleenlähetyksistä ja virheiden raportoinnista. Asiakassovellus 18 voi hallita kaikkia näitä tehtäviä GXClient-komponentin 23 asetuksilla.

35 Kuvassa 4 on pääpiirteittäin esitetty yhden datapaketin lähetyksen järjestelmän välityksellä. Aluksi sovellusohjelma 18 asettaa lähetyksen

varten tarvittavat asetukset 47 GXClient-komponentille 23. Uudelleenlähetyksen lukumäärä ja aika, joka odotetaan vastausta ennen uudelleenlähetystä, ovat perusasetuksia, joiden avulla voidaan hallita uudelleenlähetyksmekanismi. Mikäli pakettiin ei tule vastausta kyllin nopeasti, lähetys suoritetaan automaattisesti uudestaan, mikäli uudelleenlähetyksen lukumääräksi on asetettu yksi tai enemmän. Jos vastausta ei tule ja kaikki uudelleenlähetykset on käytetty, GXClient-komponentti 23 informoi asiakassovellusta virhetilanteesta lähettämällä tälle ns. Event-viestin.

10

GXCom-järjestelmä ei itse voi tunnistaa, mikä vastaanotetuista paketeista on vastaus millekin lähetetylle paketille tai onko tämä vastaus virheellinen, sillä tämä toiminnallisuus vaatisi asiakkaan 18 käyttämän protokollan tuntemista. Asia on GXCom-järjestelmässä ratkaistu välittämällä kaikki vastauspaketit asiakassovellukselle 18 Event-viestillä. Asiakassovellus 18 voi Event-viestin paluuparametreissa määritellä oliko kyseessä vastauspaketti vai virheviesti ja järjestelmä käyttää tätä tietoa hyväkseen määritellessään suoritetaanko uudelleenlähetystä.

20

Muita asetuksia, joita asiakassovellus 18 voi käyttää, ovat mm. automaattinen tarkistussumman laskeminen lähteviin paketteihin, käytössä oleva tavujärjestys ja käytettävien lähetys- ja vastaanottopuskureiden maksimikoot, sekä mahdollinen paketin alku- ja loppumerkki. Alku- ja loppumerkki lisätään jokaisen lähetettävän paketin alkuun ja loppuun samalla kun ne muutetaan mediakomponentille 28 lähteväksi tavuvirraksi. Alku- ja loppumerkkiä käytetään joissakin protokollissa pakettirajojen tunnistamiseen vastaanottotilanteissa. Niiden asetusten osalta, joita ei aseteta asiakassovelluksesta 18, käytetään oletusarvoja.

25

30

GXClient- 23 ja GXPacket-komponenteista 19 löytyy osittain myös samoja asetuksia. Ideana on se, että GXClient-komponentin 23 asetuksia käytetään kaikkien pakettien lähetyksessä, mutta asiakassovellus 18 voi halutessaan lähettää yksittäisen paketin eri lähetysasetuksin. Paketin asetukset ajavat siis GXClient-komponentin 23 asetusten yli.

35

Tällä mahdollistetaan se, että sovellusohjelma 18 voi helposti lähettää useita paketteja, jotka esimerkiksi lähetetään virhetilanteissa uudestaan viisi kertaa, mutta tämän lisäksi esimerkiksi yhden paketin, jota ei

lähetetä uudestaan kertaakaan vaikka lähetysvirhe tapahtuisikin. Mikäli asetuksia ei aseteta GXClient-komponentille 23, niin asiakassovellus 18 joutuu asettamaan ne jokaiseen GXPacket-komponenttiin 19 erikseen, mikäli oletusasetukset eivät miellytä. GXClient 23 kopioi (vaihe 5 48) kaikki saamansa asetukset suoraan GXServer-komponentille 26, joka varsinaisesti huolehtii uudelleenlähetyksestä ja pakettien pusku-roinnista.

Kun GXClient-komponentin 23 asetukset ovat kohdallaan, asiakassovellus 18 luo (vaihe 49) GXPacket-komponentin 19 saaden samalla 10 osoittimen kyseisen paketin rajapintaan. Rajapinnan kautta asiakassovellus 18 voi asettaa jo edellisessä kappaleessa kuvattuja pakettikoh- taisia asetuksia. Tämän lisäksi pakettikomponenttiin 19 täytetään ot- sikko- ja datakenttien sisältämä data (vaihe 50). Molemmille kentille 15 (paketin otsikkokenttä ja datakenttä) on omat metodinsa, joiden avulla lähetettävä tavujono voidaan välittää pakettikomponentille 19.

Kun GXPacket-komponentin 19 tarvitsemat tiedot on täytetty, asiakas- sovellus 18 voi lähettää paketin osoittimen GXClient-komponentille 23 20 ns. Send-metodin 51 parametrina (Send()). GXClient-komponentti 23 kopioi pakettiin lähetysasetukset ja lähettää (vaihe 52) paketin edelleen GXServer-komponentille 26. GXServer-komponentin 26 tehtävänä on tallentaa pakettikomponentti puskuun 53 odottamaan lähetystä (vaihe 54). GXServer-komponentin tehtävänä lähetystilanteessa kerätä 25 lähetettävä otsake- ja datatieto pakettikomponentista 19 ja välittää se mediakomponentille 28 tavuvirtana. GXServer-komponentti 26 huolehtii myös mahdollisesta uudelleenlähetyksestä ja tarkkailee, missä vai- heessa paketin lähettämisestä luovutaan virhetilanteessa.

30 Mediakomponentti 28 ottaa GXServer-komponentilta 26 saapuvan ta- vuvirran ja lähettää sen edelleen kyseistä tietoyhteyttä kontrolloivalle käyttöjärjestelmäpalvelulle (kuvan 2 mukainen palvelu 8), joka huolehtii tiedon siirtämisestä fyysiselle siirtotielle (kuvan 1 mukainen siirtotie 9, 10 tai 11).

35

Tarkastellaan seuraavaksi järjestelmää datan vastaanottopuolelta. Ku- vassa 5 on kuvattu pääpiirteittäin yhden paketin vastaanotto.

Vastaanottotilanteessa asiakkaalle 18 halutaan tavuvirran sijaan välittää samoja GXPacket-komponentteja 19 – 21 kuin lähetettäessäkin. Koska GXPacket-komponentin 19 – 21 rajapinta 22 on kiinteästi määritetty, asiakassovellukselle 18 voidaan tarjota kiinteänä pysyvä rajapinta tavuvirran rakenteesta tai käytetystä protokollasta huolimatta. Näin helpotetaan asiakassovelluksen 18 tietoliikenneosion ohjelmointia. Pakettikomponentti 19 – 21 muodostaa loogisen kokonaisuuden ja mahdollistaa helpon pääsyn paketin sisältämiin otsikko- ja datakenttiin.

Jotta saapuvasta datavirrasta voidaan muodostaa pakettikomponentteja 19 – 21, asiakassovelluksen 18 on kuvailtava käyttämänsä protokolla GXServer-komponentille 26, sillä sen tehtävänä on koota saapuvasta datavirrasta GXPacket-komponentteja 19 – 21. GXServer-komponentin 26 asetuksista löytyy tähän tarkoitukseen ainakin neljä parametriä: BOP (Beginning of packet eli paketin alkumerkki), EOP (End of packet eli paketin alkumerkki), HeaderLength (otsakedatan kentän pituus) ja DataLength (datakentän pituus). Näistä parametreista voidaan asettaa (vaihe 55 kuvassa 5) joko kaikki tai vain sopiva osa. Alla on lueteltu eri protokollavariaatiot A – I, jotka systeemi osaa tunnistaa sekä niiden vaatimat asetukset. Huomattavaa on, että esimerkeissä yksinkertaistetaan muodostettavien pakettikomponenttien rakennetta huomattavasti. Käytännössä kyse on kuitenkin siitä, että palvelinkomponentti 26 luo aina GXPacket COM-komponentin, esimerkiksi komponentti 19, jonka otsaketiedot voi kysyä pakettikomponentin rajapinnan ns. ExtractHeader-metodilla ja datatiedot ns. ExtractData-metodilla.

Protokollavariaatio A1: paketit tunnistetaan pelkästä datapaketin alkumerkistä (BOP). Oletetaan esimerkiksi, että GXServer-komponentille 26 kerrotaan, että alkumerkki on tavu FF ja GXServer vastaanottaa datajonon FF E1 02 FF AA C3 C5 FF 00. Alkumerkin avulla GXServer havaitsee, että se on vastaanottanut kolme pakettia: E1 02, AA C3 C5 ja 00. Mikäli otsakekentän pituutta ei määritellä, talletetaan kaikki saapuvat tavut paketin datakenttään.

Protokollavariaatio A2: paketit tunnistetaan datapaketin alkumerkistä (BOP) ja lisäksi on määritely otsakekentän pituus (HeaderLength).

Näin data pystytään lisäksi jaottelemaan otsake- ja datakentiksi. Jos edellisen kohdan esimerkissä A1 otsakekentän pituus olisi yksi tavu, näyttäisivät vastaanotetut paketit seuraavilta: E1 / 02, AA / C3 C5 ja 00 /, jolloin kunkin paketin otsake- ja datakentän erottamiseksi on tässä  
5 käytetty /-merkkiä.

Protokollavariaatio A3: alkumerkin (BOP) lisäksi on määritelty paketin loppumerkki (EOP). Variaatio ei käytännössä juurikaan eroa kohdan A1 esimerkistä. Jos esimerkin A1 saapuvassa datassa olisi loppumerkkinä  
10 käytetty esimerkiksi tavua CC, niin se näyttäisi seuraavalta: FF E1 02 CC FF AA C3 C5 CC FF 00 CC.

Protokollavariaatio A4: pelkkä loppumerkki (EOP) on määritelty. Käytännössä kuten esimerkki A1, mutta paketit tunnistetaan alkumerkin sijaan loppumerkistä.  
15

Protokollavariaatio A5: loppumerkki (EOP) ja otsakekentän pituus (HeaderLength) on määritelty. Käytännössä tämä variaatio E on kuten  
20 esimerkki A2.

Protokollavariaatio B1: pelkkä datakentän pituus (DataLength) on määritelty. Tällöin kaikki saapuva data oletetaan kuuluvaksi datakenttään. Jos datakentän pituudeksi on valittu esimerkiksi kolme tavua ja vastaanotetaan seuraava data: 01 02 03 04 05 06 07 08 09, niin  
25 GXServer 26 pystyy kokoamaan siitä kolme pakettia (01 02 03, 04 05 06 ja 07 08 09).

Protokollavariaatio B2: lisäksi määritelty otsakekentän pituus (HeaderLength). Tällöin osa datasta halutaan sijoittaa otsakekenttään. Jos vastaanotetaan esimerkin B1 data ja ollaan määritelty otsakekentän pituudeksi yksi tavu ja datakentän pituudeksi kaksi tavua, niin muodostettavat paketit näyttäisivät seuraavalta: 01 / 02 03, 04 / 05 06 ja 07 / 08 09, jolloin otsake- ja dataosuudet on erotettu /-merkillä.  
30

Protokollavariaatio B3: pelkkä otsakekentän pituus (HeaderLength) on määritelty. Kuten esimerkki B1, mutta kaikki saapuva data sijoitetaan muodostettavan paketin otsakekenttään.  
35

Protokollavariaatio C1: pelkästään otsakekentän pituus (Header-Length) on määritelty ja tämän lisäksi kerrottu kohta, mistä otsakekentän alusta laskien datakentän pituus löytyy ja kuinka pitkä pituuden osoittava kenttä on. Vastaanotetaan esimerkiksi data **00 00 01 AA EF 00 02 AA BB**. Otsakekentän pituudeksi on annettu kolme tavua ja tämän lisäksi on määritelty, että datakentän pituus löytyy toisesta tavusta alusta alkaen ja pituuskentän mitta on kaksi tavua. Nyt GXServer -purkaa dataa alusta lähtien. Kolme ensimmäistä tavua eli **00 00 01** katsotaan otsakekentäksi. Datakentän pituuden kertoo näistä toinen ja kolmas tavu eli datakentän pituus on **00 01** tavua eli yksi tavu. Datakenttään saadaan tavu **AA**. Samalla logiikalla palvelinkomponentti purkaa seuraavan paketin muotoon **EF 00 02 / AA BB**.

15 Näillä säännöillä GXServer-komponentti 26 osaa siis luoda (vaihe 57) saapuvasta tavuvirrasta 56 COM-komponentteja (GXPacket-komponentteja), joilla on vakiorajapinnat. Palvelimen 26 luomat komponentit välitetään (vaihe 58) GXClient-komponentille 23, joka kysyy ns. IsReplyPacket-event -metodilla 59 (IsReplyPacket()) asiakassovellukselta 18, onko vastaanotettu pakettikomponentti vastauspaketti johonkin aiemmin lähetettyyn pakettiin. Mikäli on, niin asiakassovellus 18 kertoo, mihin pakettiin vastaanotettu pakettikomponentti on vastaus ja GXClient 23 lähettää GXServer-komponentille 26 tiedon siitä, että lähetys on onnistunut, jolloin pakettia, johon vastaus tuli, ei tarvitse lähettää enää uudelleen, vaan sen voi poistaa palvelimen 26 lähetyspuskurista. Tässä tapauksessa GXClient 23 kysyy seuraavaksi IsReplyOK-event -metodilla, mikä on vastauspaketin status (vaihe 60). Jos esimerkiksi ollaan lähetetty komento, jonka pitäisi kirjoittaa jotain vastaanottavan laitteen muistiin, vastauspaketti saattaa sisältää tiedon siitä onnistuiko muistiin kirjoittaminen vai epäonnistuiko se siitä huolimatta, että komento tuli perille. Mikäli IsReplyOK-metodiin annetaan kieltävä vastaus, niin palvelin voi vielä kokeilla uudelleenlähetystä tai asiakassovellus voi hoitaa virheenkäsittelyn itse. Mikäli vastaanotettu paketti ei ollut vastaus mihinkään aiemmin lähetettyyn pakettiin (kielteinen vastaus IsReplyPacket-metodiin), mikä tilanne on esitetty esimerkkinä kuvassa 5, niin palvelin 26 välittää paketin asiakassovellukselle 18 ns. Received-event -metodilla (Received()), vaiheet 61 ja 62).

Keksintöä ei ole rajoitettu ainoastaan edellä esitettyyn suoritusmuotoon, vaan sitä voidaan muunnella oheisten patenttivaatimusten puitteissa. Käyttöympäristönä (Execution Environment) voi olla erilaiset tietokonevälineet käyttöjärjestelmineen (Operating System), jotka on 5 tarkoitettu sovellusten ja ohjelmistojärjestelmien ajamiseen, jolloin kyseeseen tulee erityisesti henkilökohtainen tietokone (PC, Personal Computer) tai sellaisena toimiva työasema, joka käsittää tarkoitukseen soveltuvaa käyttöjärjestelmää. Laitteistossa ja käyttöjärjestelmässä on 10 tarvittavat sovellukset ja protokollavälineet tiedonsiirtoon muiden laitteiden kanssa. Käyttöjärjestelmä on sopivimmin tunnetun tekniikan mukainen valmis järjestelmä, joka tarjoaa valmiit palvelut tiedonsiirron datavirran siirtoon.

15

## Patenttivaatimukset

1. Menetelmä sovelluksen (18, 25) ja tietoliikenneväylän (4, 5, 6, 9, 10, 11) välisen datavirran käsittelemiseksi tietokoneohjatussa ohjausjärjestelmässä (31), tunnettu siitä, että menetelmässä:

5

– pakataan ohjausjärjestelmässä (31) vastaanoton ja lähetyksen yhteydessä välitettävää datavirtaa yhdeksi tai useammaksi pakettikomponentiksi (19, 20, 21), joka on eri sovelluksien (18, 25) kannalta standardimuotoinen ja riippumaton käyttöön valitusta tietoliikenneyhteydestä (4, 5, 6, 9, 10, 11), jolloin kyseinen pakettikomponentti (19, 20, 21) käsittää tarvittavan standardisoidun ohjelmointirajapinnan (22), jonka välityksellä datavirtaa luetaan pakettikomponentista (19, 20, 21) tai kirjoitetaan pakettikomponenttiin (19, 20, 21) kyseisen sovelluksen (18) toimesta,

10

15

– ohjataan standardin ohjelmointirajapinnan välityksellä käyttöjärjestelmää (8), joka käsittää tietoliikennepalveluja, joilla toteutetaan datavirtaa välittävä yksi tai useampi tietoliikenneyhteys (4, 5, 6, 9, 10, 11), joka on tarkoitettu ohjausjärjestelmän (31) ohjaustietokoneen ja yhden tai useamman ohjattavan laitteen (1, 2, 3) väliseen tiedonsiirtoon, jolloin mainittujen tietoliikenneyhteyksien (4, 5, 6, 9, 10, 11) käyttämät tiedonsiirtoprotokollat voivat myös poiketa toisistaan,

20

25

– muunnetaan lähetyksen yhteydessä datavirtaa, joka on luettu mainitusta pakettikomponentista (19, 20, 21), sellaiseen muotoon, joka on käyttöön valitun tietoliikenneyhteyden (4, 5, 6, 9, 10, 11) tiedonsiirtoprotokollan mukainen ja samalla mainitusta tietoliikenneyhteydestä (4, 5, 6, 9, 10, 11) riippuvainen, ja syötetään se tietoliikennepalveluille (8), sekä

30

35

– muunnetaan vastaanoton yhteydessä datavirtaa, joka on vastaanotettu tietoliikennepalveluilta (8) ja joka on muodol-

5 taan riippuvainen käyttöön valitun tietoliikenneyhteyden (4, 5, 6, 9, 10, 11) tiedonsiirtoprotokollasta, ja kirjoitetaan se mainittuun pakettikomponenttiin (19, 20, 21) mainitun standardisoidun ohjelmointirajapinnan (22) kautta, sekä välitetään se sovellukselle (18, 25), joka tulkitsee datavirran informaation.

10 2. Patenttivaatimuksen 1 mukainen menetelmä, **tunnettu** siitä, että muodostetaan datavirrasta mainittuja pakettikomponentteja (19, 20, 21) perustuen kyseisen sovelluksen (18) ennakolta asettamiin ja tallennettuihin yksilöllisiin arvoihin koskien parametreja, jotka määrittävät säännöt jatkuvan datavirran jakamiseksi erillisiksi pakettikomponenteiksi (19, 20, 21) ja sopivimmin myös pakettikomponentin (19, 20, 21) sisäisen rakenteen, erityisesti otsakekentän ja/tai datakentän osalta.

15 3. Patenttivaatimuksen 2 mukainen menetelmä, **tunnettu** siitä, että määritetään kyseisten yksilöllisten parametrien avulla ainakin: datavirrasta löytyvä alkumerkki pakettikomponentille; tai pakettikomponentin otsakekentän pituus tavuina; tai datavirrasta löytyvä loppumerkki pakettikomponentille; tai pakettikomponentin datakentän pituus tavuina; 20 tieto siitä, missä otsakekentän alueella on kerrottu datakentän pituus; tai jokin edellä lueteltujen yhdistelmä.

25 4. Patenttivaatimuksen 1 mukainen menetelmä, **tunnettu** siitä, että asetetaan sovelluksen (18, 25) toimesta valitun tietoliikenneyhteyden (4, 5, 6, 9, 10, 11) yhteysparametreja yhden tai useamman mediakomponentin (27, 28, 29) välityksellä, jotka käsittävät standardisoidut ohjelmointirajapinnat (33, 34, 35), ja valitaan käyttöön sovelluksen (18) valitseman yhteystyyppin mukainen yksilöllinen mediakomponentti (27, 30 28, 29), joka puolestaan kutsuu tietoliikennepalveluja (8) datavirran lähettämiseksi.

35 5. Patenttivaatimuksen 4 mukainen menetelmä, **tunnettu** siitä, että asetetaan ja tallennetaan tietoliikenneyhteyttä (4, 5, 6, 9, 10, 11) koskevat yhteystyyppin parametrit valittuun mediakomponenttiin (27, 28, 29) sovelluksen (18, 25) ja valitun mediakomponentin (27, 28, 29) välisen asetusdialogin välityksellä.

6. Jonkin patenttivaatimuksen 1 – 5 mukainen menetelmä, **tunnettu** siitä, että luodaan sovelluksen (18, 25) datavirrasta lähetystä varten pakettikomponentteja (19, 20, 21) kullekin sovellukselle (18, 25) yksilöllisen asiakaskomponentin (23, 24) välityksellä, joka asiakaskomponentti käsittää standardin ohjelmointirajapinnan (37, 38), jonka välityksellä kunkin sovelluksen (18, 25) pakettikomponentteja (19, 20, 21) koskevat yleisasetukset ovat asetettavissa ja tallennettavissa yksilölliseen asiakaskomponenttiin (23, 24), ja jonka välityksellä pakettikohtaiset asetukset ovat tarvittaessa asetettavissa kuhunkin pakettikomponenttiin (19, 20, 21).

7. Patenttivaatimuksen 6 mukainen menetelmä, **tunnettu** siitä, että välitetään pakettikomponentteja (19, 20, 21) kustakin asiakaskomponentista (23, 24) palvelinkomponentille (26), joka keskitetysti huolehtii pakettikomponenttien (19, 20, 21) muuntamisesta lähettäväksi datavirraksi ja kunkin tiedonsiirtoyhteyden (4, 5, 6, 9, 10, 11) tiedonsiirto-protokollan mukaiseen muotoon ja datavirran muuntamisesta pakettikomponentiksi (19, 20, 21) sovellusta (18, 25) varten.

8. Patenttivaatimuksen 6 tai 7 mukainen menetelmä, **tunnettu** siitä, että kutsutaan asiakaskomponenttia (23, 24) sovelluksen (18, 25) toimesta lähetyksen yhteydessä ja tietoliikenneyhteyden (4, 5, 6, 9, 10, 11) avaamiseksi, jolloin haluttu yhteystyyppi määritellään kutsun parametreissa, ja palautetaan sovellukselle (18) osoitintietoja koskien sitä ohjelmointirajapintaa (34, 35, 36), jonka välityksellä yhteystyyppin parametrit ovat asetettavissa.

9. Jonkin patenttivaatimuksen 1 – 9 mukainen menetelmä, **tunnettu** siitä, että jaetaan tietoliikenneyhteyksien (4, 5, 6, 9, 10, 11) resursseja eri sovellusten (18, 25) kesken keskitetysti palvelinkomponentin (26) välityksellä, joka myös keskitetysti huolehtii pakettikomponenttien (19, 20, 21) muuntamisesta datavirraksi ja päinvastoin.

10. Tietokoneohjattu ohjausjärjestelmä, joka käsittää:

- käyttöympäristön, joka on tarkoitettu yhden tai useamman ohjaussovelluksen ajamiseksi,
- 5 - ohjausvälineet sovelluksen (18, 25) ja tietoliikenneväylän (4, 5, 6, 9, 10, 11) välisen datavirran käsittelemiseksi,

**tunnettu siitä, että ohjausvälineet käsittävät:**

- 10 - välineet vastaanoton ja lähetyksen yhteydessä välitettävän datavirran pakkaamiseksi yhdeksi tai useammaksi pakettikomponentiksi (19, 20, 21), joka on eri sovelluksien (18, 25) kannalta standardimuotoinen ja riippumaton käyttöön valitusta tietoliikenneyhteydestä (4, 5, 6, 9, 10, 11), jolloin kyseinen pakettikomponentti (19, 20, 21) käsittää tarvittavan standardisoidun ohjelmointirajapinnan (22), jonka välityksellä datavirta on luettavissa pakettikomponentista (19, 20, 21) tai kirjoitettavissa pakettikomponenttiin (19, 20, 21) kyseisen sovelluksen (18) toimesta,

20 jolloin ohjausjärjestelmä käsittää lisäksi:

- käyttöjärjestelmävälineet (8), jotka on tarkoitettu tietoliikenneväylän (4, 5, 6, 9, 10, 11) muodostamiseksi ohjausjärjestelmän (31) ohjaustietokoneen ja yhden tai useamman ohjattavan laitteen (1, 2, 3) välille ja datavirran välittämiseksi, jolloin mainittujen tietoliikenneyhteyksien (4, 5, 6, 9, 10, 11) käyttämät tiedonsiirtoprotokollat voivat myös poiketa toisistaan, ja jotka käyttöjärjestelmävälineet käsittävät standardin ohjelmointirajapinnan välityksellä ohjattavia tietoliikennepalveluja, ja

jolloin ohjausvälineet käsittävät lisäksi:

- 35 - välineet tietoliikennepalvelujen ohjaamiseksi standardin ohjelmointirajapinnan välityksellä ja datavirran välittämiseksi vastaanoton ja lähetyksen yhteydessä,

5 – välineet datavirran lukemiseksi mainitusta pakettikomponentista (19, 20, 21), muuntamiseksi sellaiseen muotoon, joka on lähetyksen yhteydessä käyttöön valitun tietoliikenneyhteyden (4, 5, 6, 9, 10, 11) tiedonsiirtoprotokollan mukainen ja samalla mainitusta tietoliikenneyhteydestä (4, 5, 6, 9, 10, 11) riippuvainen, ja syöttämiseksi tietoliikennepalveluille (8),

10 – välineet tietoliikennepalveluilta (8) vastaanotetun datavirran muuntamiseksi vastaanoton yhteydessä, jolloin datavirta on muodoltaan riippuvainen käyttöön valitun tietoliikenneyhteyden (4, 5, 6, 9, 10, 11) tiedonsiirtoprotokollasta, datavirran kirjoittamiseksi mainittuun pakettikomponenttiin (19, 20, 21) mainitun standardisoidun ohjelmointirajapinnan (22) kautta, ja mainitun pakettikomponentin välittämiseksi sovellukselle (18, 25), joka on järjestetty tulkitsemaan datavirran informaation.

20 11. Patenttivaatimuksen 10 mukainen tietokoneohjattu ohjausjärjestelmä, **tunnettu** siitä, että ohjausvälineet on järjestetty muodostamaan datavirrasta mainittuja pakettikomponentteja (19, 20, 21) perustuen kyseisen sovelluksen (18) ennakolta asettamiin ja tallennettuihin yksilöllisiin arvoihin koskien parametreja, jotka määrittävät säännöt jatkuvan datavirran jakamiseksi erillisiksi pakettikomponenteiksi (19, 20, 25 21) ja sopivimmin myös pakettikomponentin (19, 20, 21) sisäisen rakenteen, erityisesti otsakekentän ja/tai datakentän osalta.

30 12. Patenttivaatimuksen 11 mukainen tietokoneohjattu ohjausjärjestelmä, **tunnettu** siitä, että kyseiset yksilölliset parametrit käsittävät ainakin seuraavan informaation: datavirrasta löytyvä alkumerkki pakettikomponentille; tai pakettikomponentin otsakekentän pituus tavuina; tai datavirrasta löytyvä loppumerkki pakettikomponentille; tai pakettikomponentin datakentän pituus tavuina; tieto siitä, missä otsakekentän alueella on kerrottu datakentän pituus; tai jokin edellä lueteltujen yhdistelmä.  
35

13. Tietokoneohjelma, joka käsittää ohjelmakoodivälineet, **tunnettu** siitä, että se on järjestetty suorittamaan jonkin patenttivaatimuksien 1 – 9 mukaisen menetelmän vaiheet, kun kyseinen tietokoneohjelma ajetaan ohjaustietokoneessa.

5

14. Tietokoneohjelmatuote, joka käsittää ohjelmakoodivälineet, jotka on tallennettu tietokoneen luettavaan mediaan, **tunnettu** siitä, että se on järjestetty suorittamaan jonkin patenttivaatimuksien 1 – 9 mukaisen menetelmän vaiheet, kun kyseinen tietokoneohjelma ajetaan ohjaustietokoneessa.

10

Patentkrav:

1. Förfarande för behandling av en dataström mellan en tillämpning (18, 25) och en kommunikationsbuss (4, 5, 6, 9, 10, 11) i ett datorstyrt kontrollsystem (31), **kännetecknat** av, att i förfarandet

5  
10  
15  
– packas i kontrollsystemet (31) dataströmmen som skall förmedlas i samband med mottagning och sändning till en eller flera paketkomponent (19, 20, 21) som har en standardform för olika tillämpningar (18, 25) och är oberoende av den för användning valda kommunikationsförbindelsen (4, 5, 6, 9, 10, 11) varvid sagda paketkomponent omfattar det nödvändiga standardiserade programmeringsgränssnittet (22), medelst vilket dataströmmen läses från paketkomponenten (19, 20, 21) eller skrivs i paketkomponenten (19, 20, 21) genom sagda tillämpning (18),

20  
25  
– genom det standardprogrammeringsgränssnittet styrs ett operativsystem (8) som omfattar kommunikationstjänster med vilka förverkligas en eller flera kommunikationsförbildelse (4, 5, 6, 9, 10, 11) som förmedlar dataströmmen och som är avsedd för dataöverföring mellan kontrollsystemets (31) styrdator och en eller flera styrbara anordningar (1, 2, 3), varvid kommunikationsprotokoll som används av sagda kommunikationsförbindelser (4, 5, 6, 9, 10, 11) kan också avvika från varandra,

30  
35  
– i samband med sändningen omvandlas dataströmmen som är läst från sagda paketkomponent (19, 20, 21) till en sådan form som är förenligt med kommunikationsprotokollet av den för användningen valda kommunikationsförbindelsen (4, 5, 6, 9, 10, 11) och samtidigt beroende av sagda kommunikationsförbindelse (4, 5, 6, 9, 10, 11), och den matas till kommunikationstjänster (8), och

– i samband med mottagningen omvandlas dataströmmen som är mottagen från sagda kommunikationstjänster (8)

5 och vars form är beroende av kommunikationsprotokollet av den för användningen valda kommunikationsförbindelsen (4, 5, 6, 9, 10, 11), och den skrivs i sagda paketkomponent (19, 20, 21) medelst sagda programmeringsgränssnitt (22) och förmedlas till tillämpningen (18, 25) som tolkar dataströmmens information.

10 2. Förfarande enligt patentkrav 1, **kännetecknat** av, att sagda paketkomponenter (19, 20, 21) bildas från dataströmmen på basis av individuella lagrade värden som sätts av den ifrågavarande tillämpningen (18) och som gäller parametrar som bestämmer regler för att fördela den kontinuerliga dataströmmen till separata paketkomponenter (19, 20, 21) och företrädesvis också för paketkomponentens (19, 20, 21) inre struktur, speciellt för rubrikfältet och/eller datafältet.

15 3. Förfarande enligt patentkrav 2, **kännetecknat** av, att med sagda individuella parametrar bestäms åtminstone: en startpunkt för paketkomponenten som kan hittas i dataströmmen; eller längden av paketkomponentens rubrikfält i bitgrupper; eller sluttecken för paketkomponenten som kan hittas i dataströmmen; eller längden av paketkomponentens datafält i bitgrupper; information i vilket område av rubrikfältet längden av datafältet är beskriven, eller en kombination av de ovan uppräknade.

25 4. Förfarande enligt patentkrav 1, **kännetecknat** av, att genom tillämpningen (18, 25) sätts förbindelseparametrar av den valda kommunikationsförbindelsen (4, 5, 6, 9, 10, 11) medelst en eller flera mediakomponenter (27, 28, 29), vilka omfattar standardiserade programmeringsgränssnitt (33, 34, 35), och en individuell mediakomponent (27, 28, 29) enligt den av tillämpningen valda förbindelse-  
30 typen väljs för användning, som för sin tur anropar för kommunikationstjänster (8) för att sända dataströmmen.

35 5. Förfarande enligt patentkrav 4, **kännetecknat** av, att parametrar av den kommunikationsförbindelsen (4, 5, 6, 9, 10, 11) gällande förbindelse-  
setypen sätts och lagras till den valda mediakomponenten (27, 28, 29)

genom sättningsdialogen mellan tillämpningen (18, 25) och den valda mediakomponenten (27, 28, 29).

5 6. Förfarande enligt patentkrav 1 – 5, **kännetecknat** av, att paketkomponent bildas från tillämpningens (18, 25) dataström för sändning genom en för varje tillämpning individuell klientkomponent (23, 24), vilken klientkomponent omfattar ett standardprogrammeringsgränssnitt (37, 38) medelst vilket allmänna inställningar gällande paketkomponent av var och en tillämpning (18, 25) kan sättas och lagras till den individuella klientkomponenten (23, 24) och medelst vilket de paketspecifika inställningarna kan sättas till varje paketkomponent (19, 20, 21) vid behov.

15 7. Förfarande enligt patentkrav 6, **kännetecknat** av, att paketkomponent (19, 20, 21) förmedlas från varje klientkomponent (23, 24) till serverkomponenten (26) som på ett centraliserat sätt tar hand om omvandling av paketkomponent (19, 20, 21) till en dataström som skall sändas och till en form enligt kommunikationsprotokollet av var och en kommunikationsförbindelse (4, 5, 6, 9, 10, 11) och omvandling av dataströmmen till en paketkomponent (19, 20, 21) för tillämpningen (18, 25).

∴ 25 8. Förfarande enligt patentkrav 6 eller 7, **kännetecknat** av, att klientkomponenten (23, 24) anropas genom tillämpningen (18, 25) i samband med sändningen och för att öppna kommunikationsförbindelsen (4, 5, 6, 9, 10, 11), varvid den önskade kommunikationstypen bestäms i anropets parametrar, och indikatorinformation som gäller programmeringsgränssnittet (34, 35, 36) genom vilket parametrarna av kommunikationstypen kan sättas returneras till tillämpningen (18).

30 ∴ 9. Förfarande enligt något av patentkraven 1 – 9, **kännetecknat** av, att resurser av kommunikationsförbindelser (4, 5, 6, 9, 10, 11) fördelas på ett centraliserat sätt mellan olika tillämpningar (18, 25) medelst serverkomponenten (26), som också på ett centraliserat sätt tar hand om omvandling av serverkomponenter (19, 20, 21) till en dataström och tvärtom.

35

## 10. Datorstyrt kontrollsystem, som omfattar:

- 5           –           en användningsmiljö som är avsedd för körning av en eller flera styrtillämpningar,
- styrmedel för behandling av dataströmmen mellan tillämpningen (18, 25) och kommunikationsbussen (4, 5, 6, 9, 10, 11),

## kännetecknat av att styrmedlen omfattar:

- 10           –           medel för packning av dataströmmen som skall förmedlas i samband med mottagning och sändning till en eller flera paketkomponent (19, 20, 21) som har en standardform för olika tillämpningar (18, 25) och är oberoende av den för användning valda kommunikationsförbindelsen, varvid sagda paketkomponent omfattar det nödvändiga standardiserade programmeringsgränssnittet (22), medelst vilket dataströmmen kan läsas från paketkomponenten (19, 20, 21) eller skrivs i paketkomponenten (19, 20, 21) genom sagda tillämpning (18),

## varvid kontrollsystemet ytterligare omfattar:

- 25           –           operativsystemmedel (8) som är avsedda för att bilda en kommunikationsbuss (4, 5, 6, 9, 10, 11) mellan kontrollsystemets (31) styrdator och en eller flera styrbara anordningar (1, 2, 3) och för att förmedla dataströmmen, varvid kommunikationsprotokoll som används av sagda kommunikationsförbindelser (4, 5, 6, 9, 10, 11) kan också avvika från varandra, och vilka operativsystemmedlen omfattar kommunikationstjänster som kontrolleras genom ett standardprogrammeringsgränssnitt, och

## varvid styrmedlen ytterligare omfattar:

35

- medel för att kontrollera kommunikationstjänster genom ett standardprogrammeringsgränssnitt, och för att förmedla dataströmmen i samband med mottagning och sändning,
  - 5 – medel för att läsa dataströmmen från sagda paketkomponent (19, 20, 21), att omvandla den till en sådan form som är förenligt med kommunikationsprotokollet av densamma för användningen valda kommunikationsförbindelsen (4, 5, 6, 9, 10, 11) och samtidigt beroende av sagda  
10 kommunikationsförbindelse (4, 5, 6, 9, 10, 11), och matning av densamma till kommunikationstjänster (8),
  - medel för att omvandla dataströmmen som är mottagen från sagda kommunikationstjänster (8) i samband med  
15 mottagningen, varvid dataströmmen är beroende av kommunikationsprotokollet av den för användningen valda kommunikationsförbindelsen (4, 5, 6, 9, 10, 11), för skrivning av dataströmmen i sagda paketkomponent (19, 20, 21) medelst sagda programmeringsgränssnitt (22) och förmedling av sagda paketkomponent till tillämpningen (18,  
20 25) som är anordnat att tolka dataströmmens information.
11. Datorstyrt kontrollsystem enligt patentkrav 10, **kännetecknat** av, att styrmedlen är anordnade att bilda sagda paketkomponenter (19, 20,  
25 21) från dataströmmen på basis av de individuella lagrade värden som har sätts av den ifrågavarande tillämpningen (18) och som gäller parametrar som bestämmer regler för att uppdelar den kontinuerliga dataströmmen till separata paketkomponenter (19, 20, 21) och företrädesvis också för paketkomponentens (19, 20, 21) inre struktur, speciellt för  
30 rubrikfältet och/eller datafältet.
12. Datorstyrt kontrollsystem enligt patentkrav 11, **kännetecknat** av, att sagda individuella parametrar omfattar åtminstone följande information: en startpunkt för paketkomponenten som kan hittas i dataströmmen; eller längden av paketkomponentens rubrikfält i bitgrupper; eller sluttecken för paketkomponenten som kan hittas i dataströmmen; eller  
35 längden av paketkomponentens datafält i bitgrupper; information i vilket

område av rubrikfältet längden av datafältet är beskriven, eller en kombination av de ovan uppräknade.

- 5 13. Datorprogram som omfattar programkodmedel, **kännetecknad** av, att den är anordnad att utföra skeden av förfarandet enligt något av patentkraven 1 – 9, när sagda datorprogram körs i en styrdator.
- 10 14. Datorprogramprodukt som omfattar programkodmedel, som är lagrade i ett läsbart medium av en dator, **kännetecknad** av, att den är anordnad att utföra skeden av förfarandet enligt något av patentkraven 1 - 9, när sagda datorprogram körs i en styrdator.

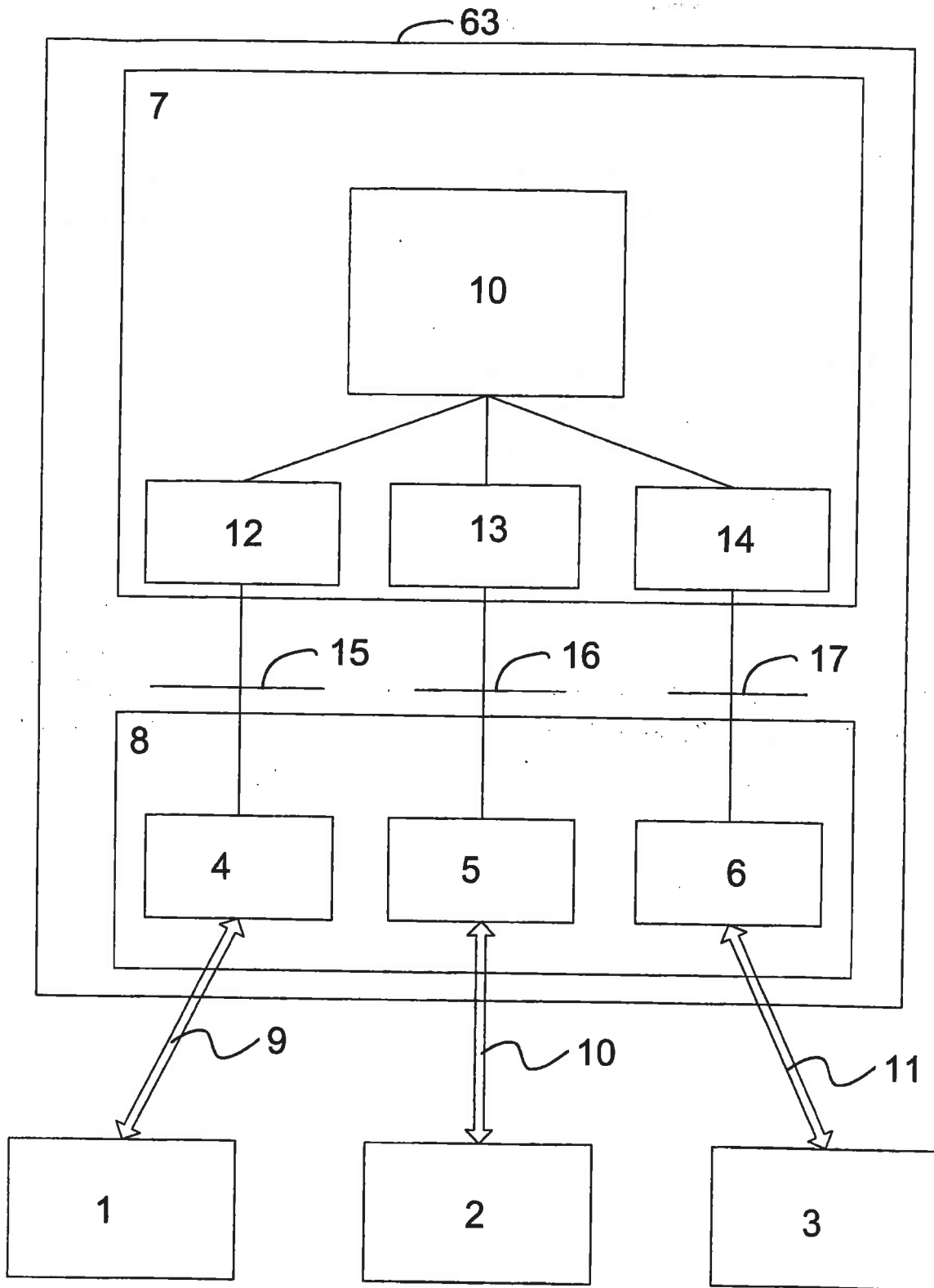


Fig. 1

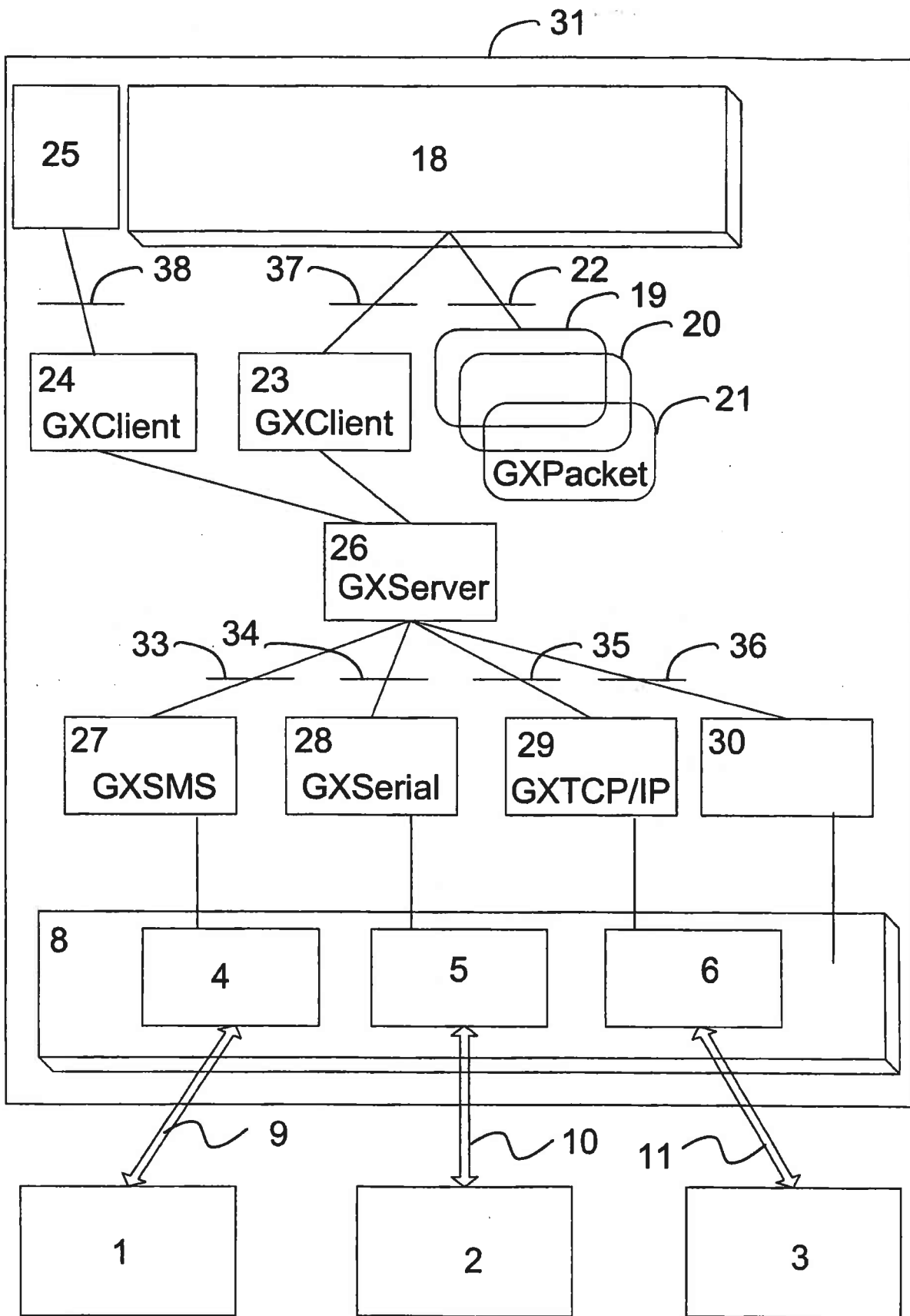


Fig. 2

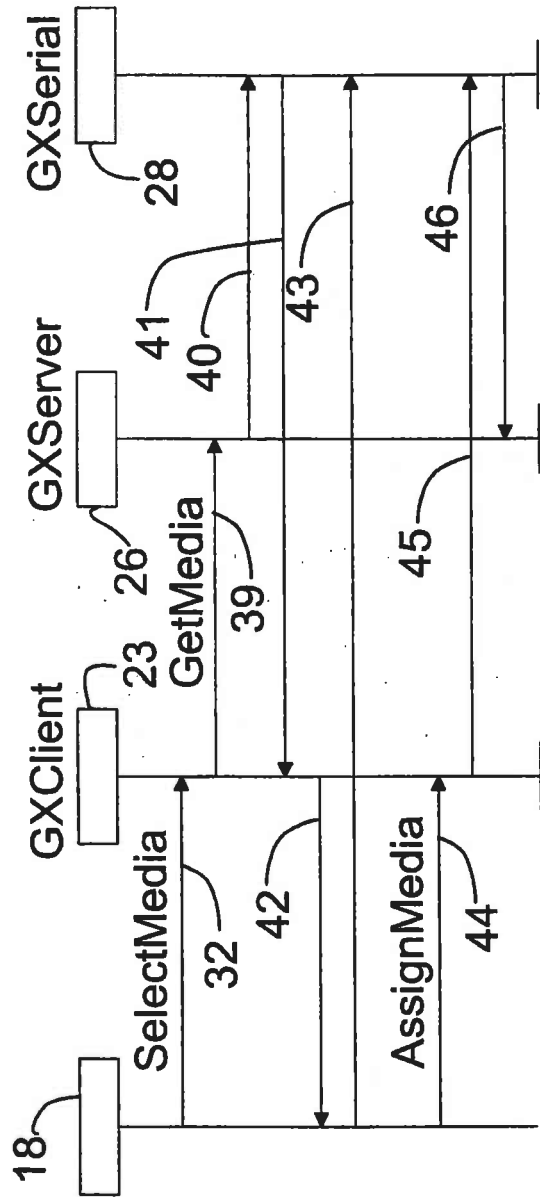


Fig. 3

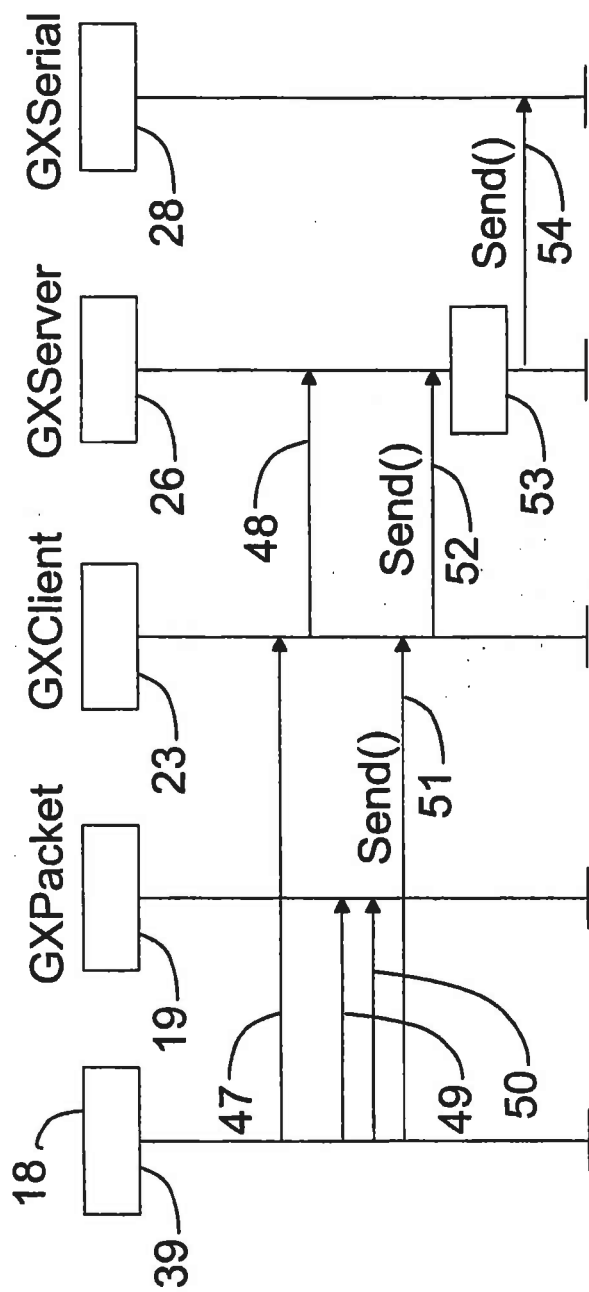


Fig. 4

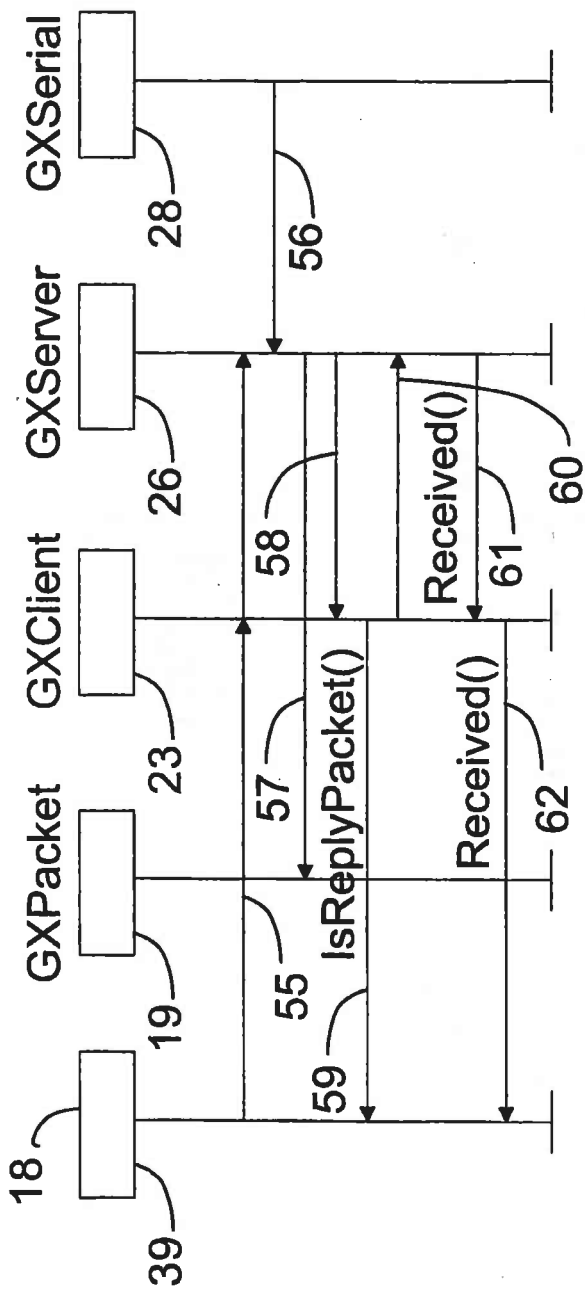


Fig. 5