(12) **PATENT**

(19) NO (11) **324486** (13) **B1**

NORGE

(51) Int Cl.
G06F 9/445 (2006.01)
G06F 9/06 (2006.01)

Patentstyret

(21)	Søknadsnr	20023491	(86)	Int.inng.dag og søknadsnr	2001.11.21 PCT/JP01/10170
(22)	Inng.dag	2002.07.22	(85)	Videreføringsdag	2002.07.22
(24)	Løpedag	2001.11.21	(30)	Prioritet	2000.11.24, JP, 358046/00
(41)	Alm.tilgj	2002.09.19			
(45)	Meddelt	2007.10.29			

(73) Innehaver NTT DoCoMo Inc, 11-1, Nagatacho 2-chome, Chiyoda-ku, 100-6150 TOKYO , JP

(72) Oppfinner Masaaki Yamamoto, 2-10-107, Sugita 9-chome, Isogo-ku, Yokohama-shi, Kanagawa 235-0033, JP
Kazuhiro Yamada, 660, Mutsukawa 1-chome, Minami-ku, Yokohama-shi, Kanagawa 232-0066, JP
Yoshiaki Hiramatsu, 5-20, Funagura 2-chome, Yokosuka-shi, Kanagawa 239-0805, JP
Kyoko Inoue, 19-10-302, Ebisuminami 1-chome, Shibuya-ku, Tokyo 150-0022, JP
Dai Kamiya, 3-1-301, Tomihama 2-chome, Ichikawa-shi, Chiba 272-0115, JP
Motoki Tokuda, Chierukureru 2-202, 22-41, Tsukui 1-chome, Yokosuka-shi, Kanagawa 239-0834, JP
Tatsuro Ooi, 21-19-307, Mutsuura 1-chome, Kanazawa-ku, Yokohama-shi, Kanagawa 236-0031, JP
Yutaka Sumi, 23-8-502, Higashiazabu 1-chome, Minato-ku, Tokyo 106-0044, JP
Eriko Ooseki, 6-1-704, Hikarinooka, Yokosuka-shi, Kanagawa 239-0847, JP

(74) Fullmektig Protector Intellectual Property Consultants AS, Postboks 5074 Majorstua, 0301 OSLO

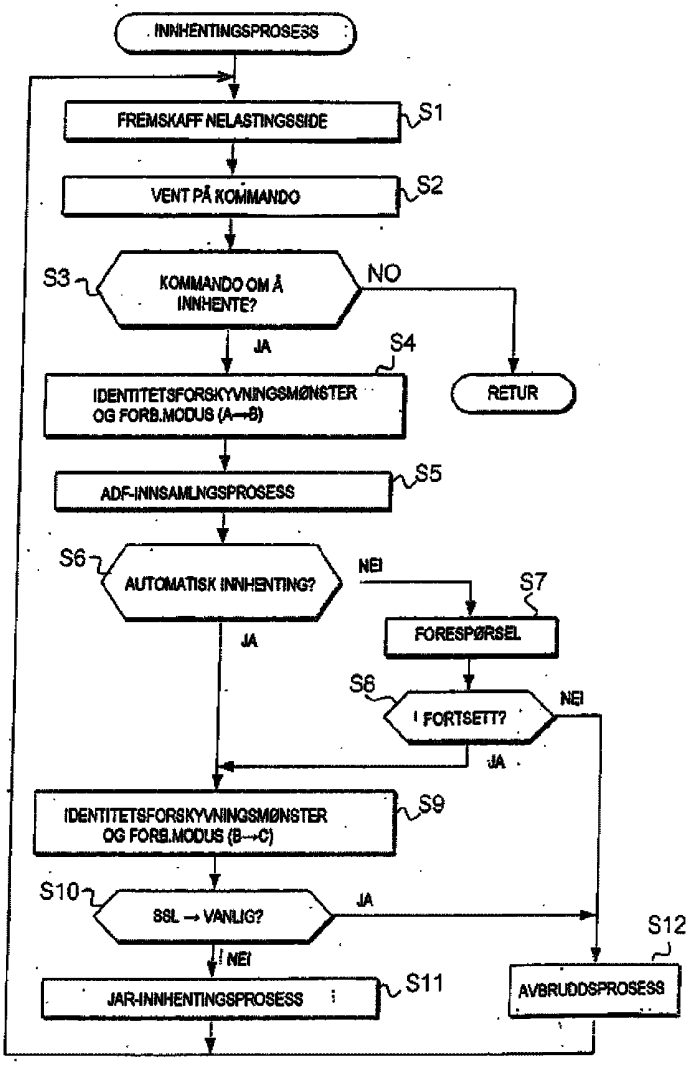
(54) Benevnelse **Fremgangsmåte og terminaler for tilveiebringing av data**

(56) Anførte publikasjoner

D1 JÖRG HEUER ET AL: "Adaptive Multimedia Messaging based on MPEG-7 - The M3-Box" PROCEEDINGS 2ND INTL SYMPOSIUM ON MOBILE MULTIMEDIA SYSTEMS & APPLICATIONS, 10 November 2000 (2000-11-10)
D2 CZERWINSKI S E ET AL: "An architecture for a secure service discovery service" MOBICOM '99. PROCEEDINGS OF THE 5TH ANNUAL ACM/IEEE INTERNATIONAL CONFERENCE ON MOBILE COMPUTING AND NETWORKING, NEW YORK, NY: ACM, US, vol. CONF. 5, 15 August 1999

(57) Sammendrag

Ved en terminal hvor kommunikasjon kan finne sted via et nett, blir det mottatt ADF i hvilken egenskapsinformasjon om dataene er lagret, og basert på kommunikasjonssystemene (vanlig/SSL) som blir brukt til å motta denne ADF og det kommunikasjonssystem som ble brukt til å motta JAR, hvor entiteten av de ovennevnte data er lagret, blir det bestemt om et kommunikasjonsmønster er tillatt. JAR blir mottatt hvis kommunikasjonsmønsteret er tillatt og JAR blir ikke mottatt hvis ikke kommunikasjonsmønsteret ikke er tillatt av sikkerhetsgrunner. Data blir med andre ord bare innhentet når kommunikasjonsmønsteret er tillatt.


INNHENTINGSPROSESS
FREMSKAFF NELASTINGSSIDE
S1
VENT PÅ KOMMANDO
S2
S3
KOMMANDO OM Å INNHENTE?
NO
JA
S4
RETUR
IDENTITETSFORSKYVNINGSMØNSTER OG FORB.MODUS (A→B)
ADF-INNSAMLINGSPROSESS
S5
S6
AUTOMATISK INNHENTING?
NEI
S7
JA
FORESPØRSEL
S8
FORTSETT?
NEI
JA
IDENTITETSFORSKYVNINGSMØNSTER OG FORB.MODUS (B→C)
S9
S10
SSL → VANLIG?
JA
NEI
S12
JAR-INNHENTINGSPROSESS
S11
AVBRUDDSPROSESS

Teknisk område

Foreliggende oppfinnelse vedrører en fremgangsmåte for å fremskaffe data for å tilveiebringe data via et nett, og terminaler for å fremskaffe data ved hjelp av denne fremgangsmåten.

Teknisk bakgrunn

På grunn av den siste utvikling av kommunikasjonsnett er det meget utbredt å fremskaffe data via nett slik som internett, (nedlasting). De data som er fremskaffet via nett slik som internett, blir vanligvis lagret i et fast lager slik som en hardplate eller hardisk. De data som er lagret i det faste lager selv etter at CDU (Random Access Memory, Sentralenhet) og RAM (Random Access Memory, direktelager) er initialisert ved omstarting av terminalen ved å slå av strømmen til terminalen og så slå den på igjen, eller ved å resette terminalen, kan aksesseres på nytt ved å lese ut dataene fra det faste lager hvor data er lagret.

Når data blir fremskaffet via nett slik som internett, kan også de fremskaffede data lagres i et midlertidig lager slik som RAM. Et eksempel på data av denne typen er en Java-applet. En Java-applet er et program som er tilveiebrakt ved å benytte Java (Java-applikasjon). En Java-applet blir tilveiebrakt via nett slik som internett, og blir lagret i et midlertidig lager i terminalen. En Java-applet som er tilveiebrakt i terminalen blir brukt via nettleseren til å se på nettsider skrevet i HTML (Hyper Text Markup Languague) og den virtuelle Java-maskin. Som beskrevet ovenfor blir det midleritidige RAM-lager initialisert når terminalen startes på nytt, og de data som er lagret i det midlertidige lager blir eliminert. Når data som er fremskaffet via nett slik som internett blir lagret i et midlertidig lager, kan de ikke brukes igjen med mindre dataene blir fremskaffet på nytt via nettet, slik som internett, etter at terminalen er omstartet.

Det finnes et antall Java-applikasjoner som i motsetning til en Java-applet, blir lagret i et fast lager etter å ha blitt fremskaffet fra nett, slik som internett, og som ikke må fremskaffes på nytt fra nett slik som internett selv etter at terminalen er startet på nytt. Det finnes også Java-applikasjoner som er lagret i et fast lager i terminalen og som ikke behøver å bli fremskaffet på nytt fra et nett. I forbindelse med beskrivelsen av foreliggende oppfinnelse vil imidlertid uttrykket "Java-applikasjon" heretter referere til

en Java-applikasjon som er fremskaffet fra et nett, siden det å fremskaffe data fra et nett er en forutsetning.

Det skal også bemerkes at selv om de data som er fremskaffet fra nettene, slik som internett, blir lagret i et fast lager eller i et midlertidig lager, blir de vanligvis mottatt fra nettet, slik som internett, som en fil. Når en Java-applikasjon som for eksempel består av en enkelt fil, blir fremskaffet fra en nettserver ved hjelp av http (Hyper Text Transfer Protocol), finner dette sted i en sekvens, med andre ord tilkobling til nettserveren, anmodning om informasjon, mottagelse av responsen og frakobling fra nettserveren. Ettersom brukeren anmoder om en Java-applikasjon ved å betjene terminalen, starter nedlasting i dette tilfellet umiddelbart, og forbindelsen mellom nettjeneren og terminalen blir opprettholdt inntil nedlastingen er ferdig. En melding blir fremvist på terminalen som indikerer at en fil er blitt lastet ned.

I denne fremgangsmåten for å fremskaffe data kan terminalbrukeren ikke kjenne til spesiell informasjon slik som filstørrelsen til Java-applikasjonen før nedlastingen av denne begynner; dermed kan terminalbrukeren ikke forutsi den mengde med tid som er nødvendig for nedlasting av Java-applikasjonen. Det er derfor et problem at når en Java-applikasjon består av et antall filer, kan bruk av terminalen begrenses på grunn av en lenger nedlastingstid enn forventet. Dette er uhyre alvorlig, spesielt for terminaler slik som mobiltelefoner hvor en nettleser er installert, som har et begrenset kommunikasjonsområde eller prosesskapasitet. Nødvendig egenskapsinformasjon slik som filnavnet på Java-applikasjonen og filstørrelsen kan selvsagt fremvises på nettsiden i brukerens terminal, men det er bekymringsfullt at en del ukorrekt egenskapsinformasjon kan gis til brukeren på grunn av feilaktige beskrivelser eller i bedragerisk hensikt.

For å unngå de ovennevnte problemer blir det antydet å dele en Java-applikasjon i to filer, en ADF (Applikasjonsdeskriptorfil) som inneholder egenskapsinformasjon og JAR (Java-arkiv) som inneholder dataentiteten, og å motta disse i rekkefølge ved terminalen. JAR er en filetype hvor en eller et antall filer er organisert i en fil. JAR kan laste ned et antall filer i en operasjon, for derved å spare den tid som er nødvendig for å laste ned

hver fil i en separat operasjon. Når en fil er inndelt i to filer, ADF og JAR, er imidlertid problemer med å trygge sikkerheten ikke tatt hensyn til i det hele tatt.

Publikasjonen "Adaptive Multimedia Messaging based on MPEG-7- The M3-Box", Proceedings 2nd int'l symposium on mobile multimedia systems & applications, 10 November 2000 (2000-11-10), av Jörg Heuer et al. viser en framgangsmåte for innhenting av data bestående av et første mottagelsestrinn i en terminal som kan kommunisere via et nett for å motta, via nettet, en første dataenhet hvor egenskapsinformasjon om data er lagret; et bestemmelsestrinn i terminalen for å bestemme om dataene kan framskaffes basert på en kommunikasjonsmodus som ble brukt til å motta en annen dataenhet hvor en entitet av dataene er lagret; og et annet mottagelsestrinn hvor den andre dataenheten blir mottatt fra nettsiden når et bestemt utfall i bestemmelsestrinnet er "ja".

Publikasjonen "An architecture for a secure discovery service", Mobicom'99, proceedings of the 5th annual ACM/IEEE international conference on mobile computing and networking, New York, NY: ACM, US, vol. CONF. 5, 15 August 1999, av Czerwinski S. et al beskriver et sikkert Service Discovery Service (SDS) som anvender en todelt mekanisme der klientforespørsler og tjenestebeskriverleser sendes mellom tilbyder og klient ved å bruke XML i det første steget. All informasjon som er sendt mellom kilent og server er kryptert for å unngå uredelig bruk. I tillegg er streng autentisering av endepunktene brukt. Følgelig kan en innholdspresentasjon av den undersøkte kontoen bli generert og sendt til mottakeren som kan bruke output-anordningen til å se igjennom presentasjonen.

Beskrivelse av oppfinnelsen

Formålet med foreliggende oppfinnelse er å tilveiebringe en fremgangsmåte for å fremskaffe data og en terminal som kan gi tilstrekkelig høy sikkerhet når oppdelte data blir fremskaffet.

For å oppnå det ovennevnte formål, tilveiebringer foreliggende oppfinnelse en fremgangsmåte for å tilveiebringe data som omfatter et første mottagelsestrinn i en

terminal som kan kommunisere via et nett, for å motta fra nettsiden en første dataenhet der egenskapsinformasjon om data er lagret; et bestemmelsestrinn i terminalen for å bestemme om dataene kan fremskaffes basert på en kommunikasjonsmåte som ble brukt i det første mottagelsestrinn, og en kommunikasjonsmåte som blir brukt til å motta en annen dataenhet i hvilken en entitet av dataene er lagret; og et annet mottagelsestrinn hvor den annen dataenhet blir mottatt fra nettsiden når et forutbestemt resultat i bestemmelsestrinnet er "ja" og den annen dataenhet ikke blir mottatt fra nettsiden når et bestemt resultat i bestemmelsestrinnet er "nei".

Ved hjelp av denne fremgangsmåten blir det bestemt, ved å sammenligne den kommunikasjonsmåte som ble brukt da den første dataenhet ble mottatt med den kommunikasjonsmåte som er brukt for å motta den annen dataenhet, om den annen dataenhet kan mottas. Det vil si at innsamling av data kan forbys når kommunikasjonsmåten er uriktig koblet fra det tidspunkt da den første dataenhet blir mottatt, til det tidspunkt da den annen dataenhet blir mottatt. Tilstrekkelig høy sikkerhet kan med andre ord sikres når det fremskaffes delte data.

Foreliggende oppfinnelse tilveiebringer også en fremgangsmåte for fremskaffelse av data hvor innsamling av dataene ikke blir tillatt når sikkerheten til en kommunikasjonsmåte som blir brukt til å motta den annen dataenhet er lavere enn sikkerheten til den kommunikasjonsmåte som ble brukt i det første mottagelsestrinn i bestemmelsestrinnet.

Foreliggende oppfinnelse tilveiebringer også en fremgangsmåte for å fremskaffe data hvor innsamling av dataene ikke er tillatt når sikkerheten til en kommunikasjonsmåte som blir brukt til å motta den annen dataenhet er mindre enn sikkerheten til den kommunikasjonsmåte som ble brukt i det første mottagelsestrinn i bestemmelsestrinnet.

Foreliggende oppfinnelse tilveiebringer også en fremgangsmåte for å fremskaffe data hvor en kommunikasjonsmåte for å motta den første dataenhet er en måte som benytter krypteringskommunikasjon, og hvor innsamling av dataene ikke blir tillatt når en kommunikasjonsmåte for å motta den annen dataenhet er en måte som ikke bruker kryptering i bestemmelsestrinnet.

Foreliggende oppfinnelse tilveiebringer også en fremgangsmåte for å fremskaffe data hvor innsamling av dataene ikke blir tillatt når en kommunikasjonsmåte som blir brukt i det første mottagelsestrinn, og en kommunikasjonsmåte som blir brukt til å motta den annen dataenhet, ikke stemmer over ens i bestemmelsestrinnet.

Foreliggende oppfinnelse tilveiebringer også en fremgangsmåte for å fremskaffe data hvor dataene er et dataprogram som kan utføres i terminalen.

Foreliggende oppfinnelse tilveiebringer, i en hvilken som helst av de fremgangsmåter for å fremskaffe data som er beskrevet ovenfor, en fremgangsmåte for å fremskaffe data hvor dataprogrammet er et dataprogram som utfører kommunikasjon.

Foreliggende oppfinnelse i henhold til noen av de fremgangsmåter for å fremskaffe data som er beskrevet ovenfor, tilveiebringer også en fremgangsmåte for å fremskaffe data hvor terminalen er en mobiltelefon.

Foreliggende oppfinnelse tilveiebringer også, i en terminal, en første mottagelsesanordning for å motta en første dataenhet hvor egenskapsinformasjonen om dataene blir lagret fra en nettside; en bestemmelsesanordning for å bestemme om innsamling av dataene er rimelig basert på en kommunikasjonsmåte som ble brukt til å motta den første dataenhet ved hjelp av den første mottagelsesanordning, og en kommunikasjonsmåte som ble brukt til å motta en annen dataenhet, hvor en dataentitet blir lagret fra nettsiden; og en annen mottagelsesanordning som mottar den annen dataenhet fra nettsiden når et bestemt resultat av bestemmelsesanordningen er "ja" og som ikke mottar den annen dataenhet fra nettsiden når et bestemt resultat fra bestemmelsesanordningen er "nei".

Kort beskrivelse av tegningene

Figur 1 er et skjema som viser den grunnleggende prosess for å fremskaffe en Java-applikasjon i terminalen i henhold til utførelsesformen av foreliggende oppfinnelse.

Figur 2 er et skjema som viser kommunikasjonsmønstrene til denne terminalen.

Figur 3 er et diagram som illustrerer de fremviste meldinger i denne terminalen.

Figur 4 er et blokkskjema som viser utformingen av det dataleveringssystemet som benytter denne terminalen.

Figur 5 er et blokkskjema som viser utformingen av nøkkelenhetene til denne terminalen.

Figur 6 er et begrepsmessig skjema som viser utformingen av prosesstabeller PT1 og PT2 i denne terminalen.

Figur 7 er et flytskjema som viser den prosess som utføres av denne terminalen under innhentingen av en Java-applikasjon.

Beste måte for utførelse av oppfinnelsen

I det etterfølgende vil foretrukne utførelsesformer av foreliggende oppfinnelse bli forklart under henvisning til figurene. Den foreliggende oppfinnelse er ikke begrenset til de følgende utførelsesformer og forskjellige endringer er mulige uten å avvike fra oppfinnelsens omfang og rekkevidde.

[Grunnleggende idé]

Først vil den grunnliggende idé for fremgangsmåte til å fremskaffe data i henhold til oppfinnelsen bli forklart.

Figur 1 viser den grunnleggende prosess hvor terminalen i henhold til foreliggende oppfinnelse fremskaffer en Java-applikasjon. Som vist på figur 1, under innsamlingen og utførelsen av Java-applikasjonen, blir prosessen i terminalen utført i rekkefølgen A, B, C, D. Terminalen sender med andre ord først ut en innsamlingsanmodning til nettet

etter den side som skal fremskaffe en Java-applikasjon, og aksesserer den relevante side (Trinn A). I denne tilstand mater så terminalbrukeren inn en ordre for å fremskaffe den Java-applikasjon som er beskrevet på denne siden ved å betjene terminalen, slik at terminalen sender ut innhentingsanmodningen som reaksjon på denne ordren, for å fremskaffe ADF for den relevante Java-applikasjon og lagre den i et fast lager (Trinn B). Terminalen sender så ut til nettet innhentingsanmodningen for JAR som svarer til denne ADF, fremskaffer den relevante JAR og lagrer den i et fast lager (Trinn C). Når så terminalbrukeren beordrer utførelse av den fremskaffede Java-applikasjonen (det program som befinner seg i JAR) ved å betjene terminalen, blir den relevante Java-applikasjonen utført ved terminalen (Trinn D).

Den grunnleggende prosess er akkurat som beskrevet ovenfor, bortsett fra tilstandsendringen i henhold til den kommunikasjonsmodus som benyttes til å oppnå hvert trinn. En operasjon der den samme kommunikasjonsmåte blir brukt i trinnene A-C, er for eksempel forskjellig fra en operasjon hvor en vanlig kommunikasjonsmodus der det ikke er noen kryptering blir brukt i Trinn A, men deretter blir kommunikasjonen utført ved hjelp av protokollen SSL (Secure Sockets Layer) som er den protokoll som benyttes til å sende og motta informasjon ved kryptering til Trinn C. Virkemåten til terminalen som svarer til slike tilstander blir beskrevet i det følgende.

Figur 2 er et diagram som viser kommunikasjonsmønstrene til terminalen i henhold til foreliggende oppfinnelse, og som vist i dette diagrammet, er kommunikasjonsmønstrene P1 til P8 mulige kommunikasjonsmønstre for terminalen. Kommunikasjonsmønstrene P1 til P8 er alle de mulige varianter av rekkefølgen til hver kommunikasjonsmåte eller kommunikasjonsmodus (vanlig/SSL) for Trinnene A-C.

Figur 3 er et skjema som viser de fremviste meldinger i terminalen i henhold til foreliggende oppfinnelse. I dette skjema blir den fremviste melding på det tidspunkt da overgang fra Trinn A til Trinn B begynner og den fremviste melding fra det tidspunkt hvor overgangen til Trinn B til Trinn C begynner, som svarer til kommunikasjonsmønsteret P1 til P8 indikeres, og virkemåten til terminalen kan identifiseres fra disse fremviste meldinger. I dette skjema svarer også de koblingsmodi som brukes ved overgang fra Trinn A til Trinn B og de forbindelsesmodi som brukes

ved overgang fra Trinn B til Trinn C til hvert kommunikasjonsmønster og de fremviste meldinger svarer til kommunikasjonsmønstrene og til hver forbindelsesmodus. I noen tilfeller blir imidlertid de fremviste meldinger bestemt uten å ta hensyn til forbindelsesmodiene, og "-" blir indikert i kolonnen for en slik forbindelsesmodus. Forbindelsesmodus-typene er "hold i live"-modus der et antall data kan videresendes ved å opprettholde forbindelsen, og "ikke holde i live"-modusen hvor forbindelsen blir kansellert etter en datautveksling. "Ikke hold i live"-modusen er den vanlige datautvekslingsmodus i HTTP og "hold i live"-modusen er den vanlige datautvekslingsmodus i HTTPS (Hyper Text Transfer Protocol Security) som svarer til SSL.

I den foreliggende utførelsesform blir de operasjoner som er vist på figur 3 utført. Disse operasjonene vil bli beskrevet mer detaljert i det etterfølgende. Et eksempel på en operasjon vil bli forklart nå for å forklare skjema. I henhold til kommunikasjonsmønster P6 (det kommunikasjonsmønster der alle kommunikasjonsmåter fra Trinn A til C er SSL), selv ved overgang fra Trinn A til Trinn B når kommunikasjonsmåten er "ikke hold i live", er for eksempel den fremviste melding ved begynnelsen av overgangen fra Trinn A til Trinn B "SSL-kommunikasjon begynner". Ved overgang fra Trinn B til Trinn C når forbindelsesmåten er "hold i live", er det heller ingen fremvisning ved begynnelsen.

Det mest distinktive punkt i dette skjema er et i kommunikasjonsmønstrene P3 og P4, er overgangen fra Trinn B til Trinn C ikke tillatt. Som vist på Figur 2 er i kommunikasjonsmønstrene P3 og P4 kommunikasjonsmåten i Trinn B SSL-protokollen, og kommunikasjonsmåten i Trinn C er vanlig. I den foreliggende utførelsesform er med andre ord innsamlingsmønsteret for å fremskaffe ADF ved kommunikasjon under bruk av krypteringskommunikasjonsmodusen for deretter å fremskaffe JAR ved hjelp av kommunikasjon som benytter den vanlige kommunikasjonsmodus ikke tillatt. Grunnen til dette vil bli forklart i det etterfølgende.

Når en Java-applikasjon fremskaffet fra nettet blir utført, og den utførte Java-applikasjonen utfører kommunikasjonen via et nett, er den modus som brukes når denne

Java-applikasjonen utfører kommunikasjonen via et nett vanligvis den kommunikasjonsmodus som ble brukt da terminalen fremskaffet Java-applikasjonen. Når for eksempel en Java-applikasjon fremskaffet ved hjelp av SSL-kommunikasjon blir utført, og denne Java-applikasjonen utfører kommunikasjonen via et nett, vil vedkommende kommunikasjonsmodus bli begrenset til SSL. På terminalsiden, så lenge kommunikasjonsmodusen var SSL når Java-applikasjonen ble fremskaffet, kan det derfor bestemmes at personlig informasjon om terminalbrukeren ikke vil bli overført i klartekst selv når denne Java-applikasjonen overfører personlig informasjon om terminalbrukeren ved etterpå å benytte nettet.

Det skal bemerkes at i den foreliggende utførelsesform, når en Java-applikasjon blir fremskaffet fra et nett ved å bruke ADF og JAR, utfører denne Java-applikasjon kommunikasjon i samme kommunikasjonsmodus som da JAR ble fremskaffet. Som man kan se fra kommunikasjonsmønstrene P3 og P4, når kommunikasjonsmodusen under innhentingen av ADF er SSL, og kommunikasjonsmodusen under innsamlingen av JAR er vanlig, så vil med andre ord den Java-applikasjonen som befinner seg i denne JAR utføre kommunikasjon på den vanlige kommunikasjonsmodus når den utfører kommunikasjon ved bruk av et nett.

En bruker har imidlertid en tendens til å anta at kommunikasjonsmåten eller kommunikasjonsmodusen for å fremskaffe JAR, er SSL hvis kommunikasjonsmodusen for å fremskaffe ADF er SSL. Hvis det er tillatt med en forskjell mellom kommunikasjonsmodusen for å fremskaffe ADF og kommunikasjonsmodusen for å fremskaffe JAR, så er risikoen at personlig informasjon om terminalbrukeren blir overført i klartekst mot brukerens vilje. Når personlig informasjon blir overført i klartekst mot terminalbrukeren vilje, er det også en risiko for at en tredje person med bedragerisk hensikt også videre, under hånden kan skaffe seg tilgang til personlig informasjon om terminalbrukeren. For å unngå slike problemer er en overgang fra Trinn B til Trinn C ikke tillatt i kommunikasjonsmønstrene P3 og P4, som vist på Figur 3. På Figur 2 er kommunikasjonsmåten i Trinn B og kommunikasjonsmåten i Trinn C forskjellige selv i kommunikasjonsmønstrene P1 og P2, men disse kommunikasjonsmønstrene er tillatt i den foreliggende utførelsesform siden dataene ble

overført ved hjelp av en Java-applikasjon som er utført ved terminalen, ikke er særlig sikrere enn hva en bruker sannsynligvis vil kreve.

[Utførelsesform]

I det følgende vil dataleveringssystemet ved å benytte terminal T i henhold til den foreliggende utførelsesform bli beskrevet.

Figur 4 er et blokkdiagram som viser utformingen av dataleveringssystemet ved å benytte terminal T, og som vist i dette skjema, er dette dataleveringssystemet det system som gjør det mulig for terminal T å benytte WWW (World Wide Web, verdensveven).

I dette skjemaet er terminal T en terminal, i dette tilfellet en mobiltelefon, for å motta pakkekommunikasjonstjensten til mobilpakke-kommunikasjonsnettet MPN, som skal fjernkobles til mobilpakke-kommunikasjonsnettet MPN og mobiltelefonnettet (ikke vist). Mobiltelefonnettet er nettet for å fremskaffe en anropstjeneste for vanlige mobiltelefoner, og terminal T kan motta denne anropstjenesten.

De detaljerte funksjoner og utforminger av terminal T vil bli beskrevet senere. Det mobile pakkekommunikasjonsnettet MPN består av et antall abonnent-, pakkebehandlingsinretninger PS, en portserver GWS og kommunikasjonslinjer for å sammenkoble disse.

Basisstasjoner BS er for eksempel plassert med jevne mellomrom, og hver basisstasjon dekker et område med en radius på 500 m og utfører radiokommunikasjon med terminal T innenfor sin radiosone.

Abonnent-pakkebehandlingsinnretningen PS er det datasystemet som er installert i abonnent-pakkeswitchestasjonen for å betjene et antall basisstasjoner BS, og det mottar pakkeutvekslingsanmodninger fra terminal T og videresender de mottatte pakker til den adresserte terminal T via andre abonnent-pakkebehandlingsinnretninger PS og basisstasjoner BS.

Portserveren GWS er det datasystem som er installert i pakkeport-videresendingssentralen for innbyrdes sammenkobling av forskjellige nett, slik som det mobile pakkekommunikasjonsnett MPN og internett INET, og som utfører utvekslinger av kommunikasjonsprotokoller som er forskjellige mellom nettene. Utvekslingen av kommunikasjonsprotokoller er i dette tilfellet spesielt innbyrdes utvekslinger av overføringsprotokoller for det mobile pakkekommunikasjonsnett som det mobile pakkekommunikasjonsnett MPN er tilpasset og hvis overføringsprotokoller internett INET er tilpasset.

Den overføringsprotokoll som internett INET er tilpasset inneholder TCP/IP (Transmisjonsstyreprotokoll/Internettprotokoll) for nettlaget og transportlaget som er en OSI-referansemodell og protokollen for HTTP, HTTPS osv. som er fremskaffet på denne TCP/IP. Den protokoll som det mobile pakkekommunikasjonsnett MPN er tilpasset, som inneholder den protokoll hvor TCP/IP er forenklet (heretter kalt TL) og den protokoll som er ekvivalent med HTTP og HTTPS (heretter kalt AL). Terminalen T benytter med andre ord WWW på AL.

Portserveren GWS mottar også HTTP- (eller HTTPS-)meldingen som benytter GET-metoden fra terminal T, den kontrollerer den URL (Uniform Resource Locator) som befinner seg i HTTP (eller HTTPS) som benytter denne GET-metoden. Hvis denne URL er den generelle URL på internett INET, blir HTTP- (eller HTTPS-)meldingen som benytter denne GET-metoden videresendt til internet INET og den respons som sendes fra internett INET som svarer til HTTP- (eller HTTPS-)meldingen som benytter denne GET-metoden, blir sendt tilbake til denne terminalen T. Hvis URL som befinner seg i HTTP- (eller HTTPS) som benytter GET-metoden, er en som indikerer ressursposisjonen til seg selv, portserveren GWS sender denne ressursen korrelert med en HTTP- (eller HTTPS-)melding som benytter denne GET-metoden tilbake til terminal T.

IP-serveren W er den server som er koblet til internett INET og som leverer forskjellige tjenester til klienter som benytter WWW. Når IP-serveren W mottar en anmodning om en HTTP- (eller HTTPS-)melding som benytter GET-metoden via internett INET,

sender den spesielt tilbake den ressurs (arbeidsfilen i den foreliggende utførelsesform) som er identifisert av den URL som befinner seg i HTTP- (eller HTTPS-)meldingen som benytter denne GET-metoden. I den foreliggende utførelsesform er formålet med IP-serveren W å overføre en Java-applikasjon, og enhver kommunikasjon mellom IP-serveren W og terminalen T blir utført ved hjelp av HTTP- (og HTTPS) og AL. Både IP-serveren W og terminalen T understøtter også datautvekslingsmodusen for å "holde i live"/"ikke hold i live" for HTTP (og HTTPS) og SSL i HTTPS.

I det følgende blir utformingen av terminal T forklart. I dette avsnittet vil imidlertid utformingen av nøkkelenheter som direkte angår foreliggende oppfinnelse bli forklart.

Figur 5 er et blokkskjema som viser utformingen av nøkkelenheten i terminal T, og som vist i dette skjemaet har terminal T en innebygd sender/mottager-enhet 11 (for eksempel utstyrt med en antenne, en radioenhet, en sender, en mottager osv.) for å utføre radiokommunikasjon med basisstasjonen BS, en lydfrembringende enhet 12 (for eksempel bestående av en lydkilde, en høyttaler osv.) for å frembringe lyd, en innmatingsenhet 13 som er utstyrt med taster ved hjelp av hvilke innmatingsoperasjoner slik som tallinnmating og bokstavinnmating blir utført, en flytende krystallskjerm 14 som inneholder fremvisningsdomene for et spesielt sted, og en styreenhet 15 som styrer hver av disse enhetene.

Styreenheten 15 har en innebygd CPU 151 som utfører hver styreoperasjon, nettleseren som blir utført ved hjelp av CPU 151, programvaren for å utføre den virtuelle Java-maskin, slik programvare som styrer programmet for terminal T, den informasjon som er nødvendig for tilkobling til portserver GWS, prosesstabellen PTI som vil bli beskrevet senere, ROM (leselager) 152 hvor PT2 osv. er lagret, et flash-lager 153 i hvilket mottatte data, oppsettingsinnholdet til brukeren (for eksempel evnen til automatisk innehenting av JAR) osv. er lagret for å tillate dem å bli brukt igjen selv etter at terminalen er omstartet, og RAM 154 som er lagret for å hindre mottatte data fra å bli brukt igjen etter at terminalen er startet på nytt, og som skal brukes som arbeidsminne for CPU 151.

Når effektbryteren (ikke vist) blir slått på, leser CPU 151 av og utfører det styreprogram som er lagret i ROM 152, styrer så ROM 152, flash-lageret 153, RAM 154, hver av enhetene 11-13 og den flytende krystallskjermen 14 i overensstemmelse med kommandoer som brukeren har matet inn fra styreprogrammet og innmatingsenheten 13. CPU 151 aktiverer også i samsvar med den kommando som er matet inn fra innmatingsenheten 13, nettleseren og er i stand til å utføre kommunikasjon på denne nettleseren i samsvar med kommandoen fra innmatingsenheten 13. Videre er CPU 151 i stand til å styre kommunikasjonsprosessen basert på prosesstabeller PT1 og PT2 (se Figur 6) som er lagret i ROM 152. Spesifikke operasjoner ved hjelp av denne funksjonen vil bli beskrevet senere.

CPU 151 aktiverer også den virtuelle Java-maskin når den utfører en Java-applikasjon som er lagret i flash-lageret 153 og utfører denne Java-applikasjonen på denne virtuelle Java-maskinen. Videre aktiverer CPU 151 den virtuelle Java-maskin og nettleseren når den aksesserer en Java-applikasjon (en Java-applet) som er lagret i RAM 154, og utfører denne Java-applikasjonen på den virtuelle Java-maskinen og nettleseren.

Innsamling av data ved hjelp av terminal T blir utført først, ved å fremskaffe HTML-data fra hjemme-URL (ressursposisjonen på port GWS som den skal aksessere først) som er lagret i ROM 152 ved hjelp av CPU 151 som aksesserer nettleseren som er lagret i ROM 152, og utfører denne nettleseren. Basert på disse data oppfordrer så den flytende krystallskjermen 14 om fremvisning av dialogskjermen, og data blir fremskaffet når brukeren betjener innmatingsenheten 13 etter fremvisningen av denne dialogskjermen.

[Operasjon for fremskaffelse av Java-applikasjon]

Figur 7 er et flytskjema som viser flyten til prosessen som terminal T utfører når en Java-applikasjon blir fremskaffet, og heretter ved hovedsakelig å referere til Figur 2, Figur 3, Figur 6 og Figur 7, vil operasjonene til terminal T for å fremskaffe en Java-applikasjon fra IP-server W bli forklart i forbindelse med hvert kommunikasjonsmønster. I de etterfølgende forklaringer blir imidlertid de operasjoner som er overlappende, utelatt så meget som mulig. Ved terminal T blir det antatt at

nettleseren allerede er blitt aktivert. Når det gjelder utførelsesformen av innhentingen av Java-applikasjonen som er gjenstand for innsamlingen, kan den lagres i enten et fastlager eller midlertidig lager, men bare det tilfellet der en Java-applikasjon blir lagret i et fast lager etter innhentingen, vil bli forklart for å unngå komplisering av forklaringen.

(1) Tilfellet med kommunikasjonsmønster P1

Som vist på Figur 2 er kommunikasjonsmodiene for kommunikasjonsmønster P1 i vanlig modus i trinn A og trinn B, og i SSL i trinn C.

Først fremskaffer terminal T siden (heretter kalt nedlastingsside) for å tilveiebringe den Java-applikasjon som er gjenstand for innhentingen (Trinn S1). Spesielt styrer CPU 151 i terminal T (se Figur 5) sender/mottager-enheten 11 basert på den kommando som bruker den matet inn fra innmatingsenheten 13 og sender IP-serveren W en HTTP-melding ved hjelp av GET-metoden, hvor denne kommandoen er i overensstemmelse med sender/mottager-enheten 11 (se Figur 4). Som reaksjon blir HTML-data fra nedlastingssiden som er formålet sendt tilbake fra IP-serveren W. Disse HTML-dataene blir mottatt av sender/mottager-enheten 11 og blir overført til CPU 151 fra sender/mottager-enheten 11. CPU 151 lagrer disse HPML-dataene i RAM 154 og forsyner brukergrensesnittet med ytterligere tolkning og utførelse av disse dataene. Fremvisningen av dette brukergrensesnittet vises følgelig på den flytende krystallskjermen 14.

Terminal T venter så på en innmating av en kommando av brukeren (Trinn S2), og innhentingsprosessen avsluttes hvis den kommando som mates inn ikke er kommandoen for å fremskaffe den tilsiktede Java-applikasjonen (Trinn S3). Spesielt identifiserer CPU 151 i terminal T innholdet av kommandoen fra brukeren basert på den kommando som er innmatet fra innmatingsenheten 13 og det aktuelle brukergrensesnitt og innhentingsprosessen avsluttes hvis kommandoer som er forskjellige fra innholdet for å fremskaffe Java-applikasjonen, slik som å fremskaffe en annen side blir matet inn, eller effektbryteren (ikke vist) blir slått av.

Hvis derimot den kommando som er innmatet i Trinn S2 er anmodningen for å fremskaffe den tilsiktede Java-applikasjonen (Trinn S3) ved hjelp av HTML-data som er tilveiebrakt i Trinn S1, så forskyves mønsteret fra Trinn A til Trinn B og forbindelsesmodusen blir identifisert (Trinn S4). Den prosess som svarer til det identifiserte resultat, blir så utført, og ADF som er objektet blir fremskaffet (Trinn S5). I dette tilfellet er kommunikasjonsmodiene i både Trinn A og B vanlige; i overensstemmelse med prosesstabellen PT1 som er vist på Figur 6, blir derfor prosessen for å fremskaffe ADF ved hjelp av vanlig kommunikasjon utført. Under denne prosessen blir "blir fremskaffet" fremvist på den flytende krystallskjermen 14. Som vist i prosesstabell PT1 på Figur 6, blir behandlingsinnholdet bestemt uansett hva forbindelsesmodusen er i dette tilfellet. Kommunikasjonsmodusen i Trinn B blir heller ikke bestemt av brukeren, men i Trinn S4 blir den avdekket ved å referere til den URL som befinner seg i HTML-data som fremskaffes i Trinn S1 for å tilveiebringe ADF. Spesielt fremviser begynnelsen av URL kommunikasjonsmodusen for å aksessere WWW-serveren, og når URL begynner med "http", vil kommunikasjonsmodusen være "vanlig" siden HTTP blir fremvist. Hvis URL begynner med "https", vil kommunikasjonsmodusen være "SSL" siden HTTPS blir fremvist.

Når innhentingen av ADF er fullstendig, og hvis den automatiske innsamling av JAR ikke er tillatt (Trinn S6), fortsetter terminal T med å spørre brukeren om JAR skal innhentes (Trinn S7). Når kommandoen for å fortsette innhentingen av JAR blir innmatet som reaksjon på denne forespørselen (Trinn S8), fortsetter den til prosessen i Trinn S9. Når imidlertid kommandoen for å avbryte innhentingen av JAR blir matet inn, blir avbruddsprosessen utført (Trinn S12), og prosessen går tilbake til Trinn S1. Hvis den automatiske innhenting av JAR blir tillatt (Trinn S6), fortsetter prosessen også umiddelbart til Trinn S9.

Terminal T er utstyrt med funksjonen for å innstille tillatt/forby den automatiske innhenting av JAR. CPU 151 setter/resetter den utpekte bit i flash-lageret 153 som er biten for å innstille tillatt/forby for den automatiske innsamling av JAR som reaksjon på den kommando som er matet inn fra innmatingsenheten 13. Ved å referere til denne biten kan derfor CPU 151 identifisere tillatt/forby-innstillingen for den automatiske

innhenting av JAR. I avbruddsprosessen i Trinn S12 avbryter også CPU 151 prosessen for å hente inn Java-applikasjonen, forkaster ADF som er objektet og blir lagret i flash-lageret 153, og trekker maksimal fordel av lagervolumet i flash-lageret 153.

Deretter blir forskyningsmønsteret fra Trinn B til Trinn C og forbindelsesmodusen identifisert, og innhentingsprosessen av JAR blir utført basert på det identifiserte resultat (Trinn S10, S11). I dette tilfellet er kommunikasjonsmodusen i Trinn B ordinær, og kommunikasjonsmodusen i Trinn C er SSL; som vist i prosesstabell PT2, vil derfor prosessen være den hvor JAR blir fremskaffet ved på nytt å starte SSL-kommunikasjon. Under denne prosessen blir "SSL-kommuniasjon begynner (blir autentisert)" fremvist på den flytende krystallskjermen 14 som skjermmelding. Den prosess som stemmer over ens med forskyvningsmønster P1 på Figur 3 blir således riktig utført. Som vist i prosesstabellen PT2 på Figur 6, blir prosessinnholdet bestemt uansett hva forbindelsesmodusen er i dette tilfellet. Kommunikasjonsmodusen i Trinn C vil også bli avdekket i Trinn S9 ved å referere til den URL som befinner seg i den ADF-filen som er fremskaffet i Trinn S5 for å tilveiebringe JAR. Når ADF er fremskaffet, hvis URL begynner med "http", vil kommunikasjonsmodusen være "vanlig". Hvis URL begynner med "https", vil kommunikasjonsmodusen være "SSL".

(2) Tilfellet med kommunikasjonsmønster P2

Som vist på Figur 2 er kommunikasjonsmodiene for kommunikasjonsmønster P2 vanlige i Trinn B og er SSL i Trinn A og C.

I Trinnene 1-5 blir først den samme prosess som i kommunikasjonsmønster P1 utført. Siden kommunikasjonsmodusen i Trinn A imidlertid er SSL, blir HTTPS i GET-metoden sendt til IP-server W i Trinn S1. Kommunikasjonsmodusen er også SSL i Trinn A, og kommunikasjonsmodusen er vanlig i Trinn B under denne prosessen; som vist i prosesstabellen PT1 som er vist på Figur 6 vil derfor kommunikasjonen bli stanset, og prosessen for å fremskaffe ADF ved hjelp av vanlig kommunikasjon vil bli utført. Under denne prosessen blir "SSL side avsluttes" fremvist på en flytende krystallskjerm 14 som skjermmelding.

Etter Trinn S9 er også kommunikasjonsmodusen i Trinn B vanlig, og kommunikasjonsmodusen i Trinn C er SSL; derfor vil den samme prosess som kommunikasjonsmodus P1 bli utført. Den prosess som svarer til kommunikasjonsmønster P2 på Figur 3, blir derfor riktig utført.

(3) Tilfellet med kommunikasjonsmønster P3

Som vist på Figur 2 er kommunikasjonsmodiene i kommunikasjonsmønster P3 vanlige i Trinn A og C, og er SSL i Trinn B.

I trinnene S1 til S5 blir først den samme prosess som kommunikasjonsmønster P1 utført. Under denne prosessen er imidlertid kommunikasjonsmodusen i Trinn A vanlig, og kommunikasjonsmodusen i Trinn B er SSL; derfor vil, som vist i prosesstabell PT1 som er vist i Figur 6, prosessen være den for å fremskaffe ADF ved på nytt starte SSL-kommunikasjon. I løpet av denne prosessen blir "SSL-kommunikasjon begynner" (blir autentisert) fremvist på den flytende krystallskjerm 14 som skjermmelding. Selv om forbindelsesmodusen er "hold i live", starter SSL-kommunikasjon på nytt i denne prosessen til tross for forbindelsesmodusen siden omkobling til SSL-kommunikasjonsmodus fra den vanlige kommunikasjonsmodus uten å avslutte prosessen er umulig.

Deretter forskyves mønsteret fra Trinn B til Trinn C og forbindelsesmodusen blir identifisert (Trinn S9). Under denne prosessen er kommunikasjonsmodusen i Trinn B SSL, og kommunikasjonsmodusen i Trinn C er vanlig; det bestemte resultat i Trinn S10 vil derfor være "ja" og behandlingen går tilbake til Trinn S1 via frakoblingsprosessen i Trinn S12. Java-applikasjonsobjektet blir med andre ord ikke fremskaffet, og den prosess som svarer til kommunikasjonsmønster P3 på figur 3 blir riktig utført.

(4) Tilfellet med kommunikasjonsmønster P4

Som vist på Figur 2 er kommunikasjonsmodiene for kommunikasjonsmønster P4 vanlig i Trinn C, og SSL i Trinnene A og B.

I trinnene S1 til S5 blir først den samme prosess som kommunikasjonsmønster P1 utført. Siden kommunikasjonsmodusen i Trinn A imidlertid er SSL, blir en HTTPS-melding ifølge GET-metoden sendt til IP-serveren W i Trinn S1. I denne prosessen er også kommunikasjonsmodusen i Trinnene A og B SSL; derfor blir prosessen som er i overensstemmelse med forbindelsesmodusen fra Trinnene A til B utført. Hvis forbindelsesmodusen med andre ord er "ikke hold i live", blir prosessen for å innhente ADF ved å starte SSL-kommunikasjon utført. Hvis forbindelsesmodusen er "hold i live", blir prosessen for å innhente ADF ved å fortsette SSL-kommunikasjon utført. I det førstnevnte tilfellet blir "SSL-kommunikasjon begynner" fremvist på den flytende krystallskjermen 14, og i det sistnevnte tilfellet blir ingen melding fremvist på den flytende krystallskjermen 14. En melding blir ikke fremvist når forbindelsesmodusen er "hold i live" fordi en ny prosess med SSL-kommunikasjon ikke starter.

Kommunikasjonsmodusen i Trinn B er også SSL og kommunikasjonsmodusen i Trinn C er vanlig, derfor blir den samme prosess som kommunikasjonsmønster P3 utført. Den prosess som svarer til kommunikasjonsmønster P4 på Figur 3 blir således riktig utført.

(5) Tilfellet med kommunikasjonsmønster P5

Som vist på Figur 2 er kommunikasjonsmodiene for kommunikasjonsmønster P5 vanlig i Trinn A og er SSL i Trinn B og C.

I Trinnene S1 til S5 blir først den samme prosess som kommunikasjonsmønster P3 utført. I prosessen etter trinn S9, siden kommunikasjonsmodusen er SSL i Trinn B og C, blir så den prosess som er i overensstemmelse med forbindelsesmodusen fra Trinn B til Trinn C utført basert på prosessen i Tabell PT2 som er vist på Figur 6. Prosessen for å innhente JAR blir med andre ord utført ved å starte SSL-kommunikasjon når forbindelsesmodusen er "ikke hold i live", og prosessen med å innhente JAR ved å fortsette SSL-kommunikasjon blir utført når forbindelsesmodusen er "hold i live". I det førstnevnte tilfellet blir "SSL-kommunikasjon begynner" fremvist på den flytende krystallskjermen 14 og i det sistnevnte tilfellet blir ingen melding fremvist på den flytende krystallskjermen 14. Den prosess som svarer til kommunikasjonsmønster P6 i Figur 3 blir således riktig utført.

(6) Tilfellet med kommunikasjonsmønster P6

Som vist på Figur 2 er kommunikasjonsmodusen for kommunikasjonsmønster P6 SSL i Trinnene A, B og C.

I Trinnene S1 til S5 blir først den samme prosess som kommunikasjonsmønster P4 utført. I prosessen etter trinn S9 blir så den samme prosess som kommunikasjonsmønster P5 utført. Den prosess som svarer til kommunikasjonsmønster P6 på Figur 3 blir så riktig utført.

(7) Tilfellet med kommunikasjonsmønster P7

Som vist på Figur 2 er kommunikasjonsmodusen for kommunikasjonsmønster P7 vanlig i Trinn A, B og C.

I Trinnene S1 til S5 blir først den samme prosess som kommunikasjonsmønster P1 utført. I prosessen i Trinn S9 og videre blir så prosessen med å innhente JAR ved å fortsette vanlig kommunikasjon i samsvar med prosesstabell PT2 som er vist på Figur 6 utført siden kommunikasjonsmodusen i Trinn B og Trinn C er vanlig. I løpet av denne prosessen blir "innhentes" fremvist på den flytende krystallskjermen 14 som den fremviste melding. Prosessen som svarer til kommunikasjonsmønster P7 på Figur 3 er således riktig utført.

(8) Tilfellet med kommunikasjonsmønster P5

Som vist på Figur 2 er kommunikasjonsmodiene til kommunikasjonsmodus P8 SSL i Trinn A og vanlige i Trinn B og C.

I Trinnene S1 til S5 blir først det kommunikasjonsmønster som P2 utført. I prosessen i trinn S9 og deretter, blir det samme kommunikasjonsmønster som P7 utført. Den prosess som svarer til kommunikasjonsmønster P8 på Figur 3 blir således riktig utført.

[Supplement]

Som forklart ovenfor, i henhold til foreliggende utførelsesform, blir den prosess som svarer til hvert kommunikasjonsmønster som er vist på Figur 2 og Figur 3 riktig utført, og ADF blir innhentet ved kommunikasjon i den krypterte kommunikasjonsmodus. Mønsteret for å hente inn en Java-applikasjon ved å fremskaffe JAR ved hjelp av kommunikasjon i et vanlig kommunikasjonsmønster kan derfor elimineres.

Når derfor terminalbrukeren utfører den fremskaffede Java-applikasjon og den fremskaffede Java-applikasjon utfører kommunikasjon ved å bruke nettet, kan overføring av personlig informasjon i klartekst ved hjelp av Java-applikasjonen mot terminalbrukeren vilje hindres. Som i den ovennevnte utførelsesform kan også en mobiltelefon benyttes som en terminal. Databehandlingskapasiteten til en mobiltelefon er liten sammenlignet med en terminal slik som en lommedatamaskin og båndbredden til kommunikasjonsveien er smal sammenlignet med kabelkommunikasjon; derfor er muligheten for at operasjonen av terminalen blir begrenset over en lengre tidsperiode enn brukeren venter, høyere i den kjente teknikk. I den foreliggende utførelsesform blir likevel en slik begrensning unngått ved å gjøre det mulig å innhente en Java-applikasjon ved å dele den inn i ADF og JAR, og ved å tillate brukeren som refererte til egenskapsinformasjon på ADF å bestemme om JAR skal hentes inn.

I den ovenfor beskrevne utførelsesform ble det vist et eksempel på å tillate en Java-applikasjon å bli brukt uten å innhente Java-applikasjonen igjen selv etter at terminalen var startet på nytt, ved å lagre den innhentede Java-applikasjonen i et fast lager; innhenting av Java-applikasjonen på nytt etter at terminalen er slått på ved å lagre den innhentede Java-applikasjonen i et midlertidig lager er imidlertid også mulig.

I den ovenfor beskrevne utførelsesform er også en Java-applikasjon bare et eksempel på data som kan innhentes, men andre programmer eller data kan også fremskaffes. Foreliggende oppfinnelse kan med andre ord anvendes når en type data som er inndelt i to enheter blir fremskaffet fra internett.

I den ovenfor beskrevne utførelsesform er også kommunikasjonsmønstrene P1 og P2 på Figur 2 tillatt, men disse kan forbys. Å forby et innsamlingsmønster hvor

kommunikasjonssystemet ifølge Trinn B og kommunikasjonssystemet i Trinn C ikke stemmer over ens, er med andre ord mulig.

I den ovenfor beskrevne utførelsesform er også en mobiltelefon et eksempel på en terminal, men i henhold til foreliggende oppfinnelse kan andre terminaler som kan utføre kommunikasjon via et nett over kabel eller eter benyttes.

I den ovenfor beskrevne utførelsesform er også kommunikasjonsmønstrene P3 og P4 på Figur 2, uten unntak, ikke tillatt, men slike begrensninger gjelder ikke foreliggende oppfinnelse. Ved for eksempel å spørre brukeren om innhenting er mulig når kommunikasjonsmønsteret er P3 og P4 på Figur 2, kan JAR mottas og lagres når brukeren godkjenner det. Ved å tillate lagring av en type Java-applikasjon i ADF, kan ellers JAR mottas og lagres selv om kommunikasjonsmønsteret er P3 og P4 på Figur 2 hvis Java-applikasjonen ikke er av den type som ikke utfører kommunikasjon.

I den ovenfor beskrevne utførelsesform er det også vist eksempler på videresending av data ved å bruke HTTP (og HTTPS) og AL, men i henhold til foreliggende oppfinnelse kan enhver protokoll som kan oppnå kryptert kommunikasjon anvendes. Det kan selvsagt være en hvilken som helst kommunikasjonsprotokoll som svarer til "hold i live" og "ikke hold i live". Det kan også være en kommunikasjonsprotokoll som ikke svarer til disse.

I den ovennevnte utførelsesform er det også vist et eksempel på konvertering av en kommunikasjonsprotokoll ved en portserver GWS, men dette er bare et eksempel. Ved for eksempel å tillate HTTP (og HTTPS) og SSL å bli behandlet ved terminalen, kan kommunikasjon utføres direkte mellom terminalen og IP-serveren, eller den kan utføres etter å ha gått gjennom et antall omforminger.

P a t e n t k r a v

1.

Fremgangsmåte for innhenting av en applikasjon i en terminal,

k a r a k t e r i s e r t v e d:

et første mottagelsestrinn (S5) i en terminal (T) som kan kommunisere via et nett (MPN) for å motta, på en vanlig eller sikker kommunikasjonsmodus, via nettet, en første dataenhet (ADF) hvor egenskapsinformasjon om en applikasjon er lagret;

et trinn (S9) for å identifisere kommunikasjonsmodusen for å motta en andre dataenhet (JAR), i hvilken en entitet av nevnte applikasjon er lagret, som vanlig eller sikker;

et bestemmelsestrinn (S10) for å bestemme om sikkerheten til den identifiserte kommunikasjonsmodusen brukt for å motta nevnte andre dataenhet er lavere enn sikkerheten til kommunikasjonsmodusen brukt i nevnte første mottagelsestrinn (S5);

og et ytterligere trinn, i hvilket nevnte andre dataenhet (JAR) er mottatt fra nevnte nettverksside (S11) når et bestemt utfall i nevnte bestemmelsestrinn er "nei", og nevnte andre dataenhet ikke er mottatt fra nevnte nettside når det er "ja" (S12).

2.

Fremgangsmåte ifølge krav 1, hvori en sikker kommunikasjonsmodus for å hente nevnte første dataenhet (ADF) er en modus som bruker krypteringskommunikasjon, og innhenting av nevnte data blir bestemt som "nei" når en kommunikasjonsmodus for å motta nevnte andre dataenhet er en modus som ikke bruker kryptering i nevnte bestemmelsestrinn (S10).

3.

Fremgangsmåte ifølge krav 1, hvori innhenting av nevnte data er bestemt som "nei" når en kommunikasjonsmodus som var brukt i nevnte første mottagelsestrinn og en

kommunikasjonsmodus for å motta nevnte andre dataenhet er forskjellige i nevnte bestemmelsestrinn (S10).

4.

Fremgangsmåte ifølge krav 1, hvor dataene er et dataprogram som kan utføres ved terminalen (T).

5.

Fremgangsmåte ifølge krav 4, hvor dataprogrammet er et dataprogram som utfører kommunikasjon.

6.

Fremgangsmåte ifølge krav 1, hvor terminalen (T) er en mobiltelefon.

7.

En terminal (T), k a r a k t e r i s e r t v e d:

en første mottagelsesanordning for å motta, i en vanlig eller sikker kommunikasjonsmodus, en første dataenhet (ADF) i hvilken egenskapsinformasjon om en applikasjon er lagret fra en nettside;

middel for å identifisere kommunikasjonsmodusen for å motta en andre dataenhet (JAR), i hvilken en entitet av nevnte applikasjon er lagret, som vanlig eller sikker;

en bestemmelsesanordning for å bestemme om innhenting av dataene er rimelig basert på om sikkerheten til den identifiserte kommunikasjonsmodusen for å motta nevnte andre dataenhet (JAR) er lavere enn sikkerheten til kommunikasjonsmodusen brukt for å motta nevnte første dataenhet (ADF) fra nevnte nettside; og

en ytterligere mottagelsesanordning som mottar den andre dataenhet fra nettsiden når et bestemt utfall av bestemmelsesanordningen er "nei", og som ikke mottar den andre dataenhet fra nettsiden når det er "ja".

FIG. 1

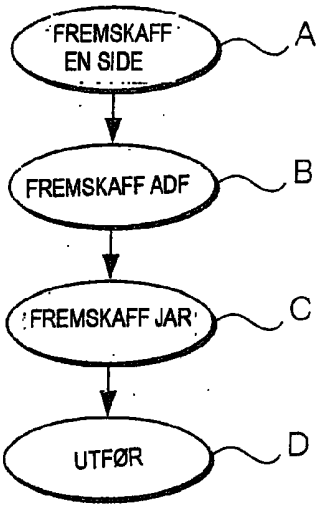


FIG. 2

KOMMUNIKASJONS-MØNSTER	TRINN		
	A (FREMSKAFF EN SIDE)	B (FREMSKAFF ADF)	C (FREMSKAFF JAR)
P1	VANLIG	VANLIG	SSL
P2	SSL	VANLIG	SSL
P3	VANLIG	SSL	VANLIG
P4	SSL	SSL	VANLIG
P5	VANLIG	SSL	SSL
P6	SSL	SSL	SSL
P7	VANLIG	VANLIG	VANLIG
P8	SSL	VANLIG	VANLIG

FIG. 3

KOMMUNIKASJONS MØNSTER	FORBINDELSES-MODUS (A→B)	FREMVIST MELDING NÅR PROSESSEN ER I GANG (A→B)	FORBINDELSES-MODUS (B→C)	FREMVIST MELDING NÅR PROSESSEN ER I GANG (B→C)
P1	-	"INNHENTES"	-	"SSL-KOMMUNIKASJON BEGYNNER (AUTENTISERES)"
P2	-	"SSL-SIDE AVSLUTTES"	-	SSL-KOMMUNIKASJON BEGYNNER (AUTENTISERES)
P3	-	"SSL-KOMMUNIKASJON BEGYNNER (AUTENTISERES)"	-	KAN IKKE BEHANDLES
P4	"IKKE HOLD I LIVE"	"SSL-KOMMUNIKASJON BEGYNNER"	-	KAN IKKE BEHANDLES
	"HOLD I LIVE"	INGEN VISNING	-	KAN IKKE BEHANDLES
P5	-	"SSL-KOMMUNIKASJON BEGYNNER (AUTENTISERES)"	"IKKE HOLD I LIVE"	"SSL-KOMMUNIKASJON BEGYNNER"
			"HOLD I LIVE"	INGEN VISNING
P6	"IKKE HOLD I LIVE"	"SSL-KOMMUNIKASJON BEGYNNER"	"IKKE HOLD I LIVE"	"SSL-KOMMUNIKASJON BEGYNNER"
			"HOLD I LIVE"	INGEN VISNING
	"HOLD I LIVE"	INGEN FREMVISNING	"IKKE HOLD I LIVE"	"SSL-KOMMUNIKASJON BEGYNNER"
			"HOLD I LIVE"	INGEN VISNING
P7	-	"INNHENTES"	-	"INNHENTES"
P8	-	"SSL-SIDE AVSLUTTES"	-	"INNHENTES"

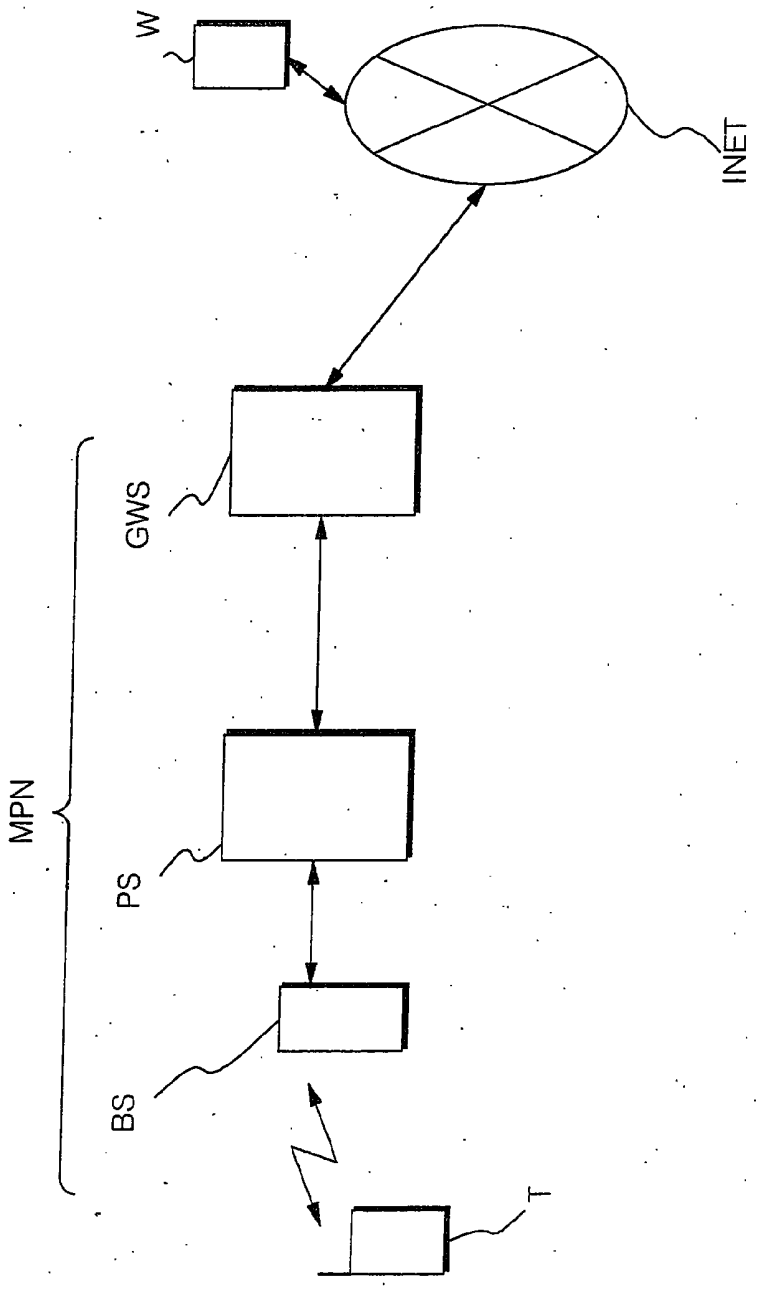


FIG. 4

FIG. 5

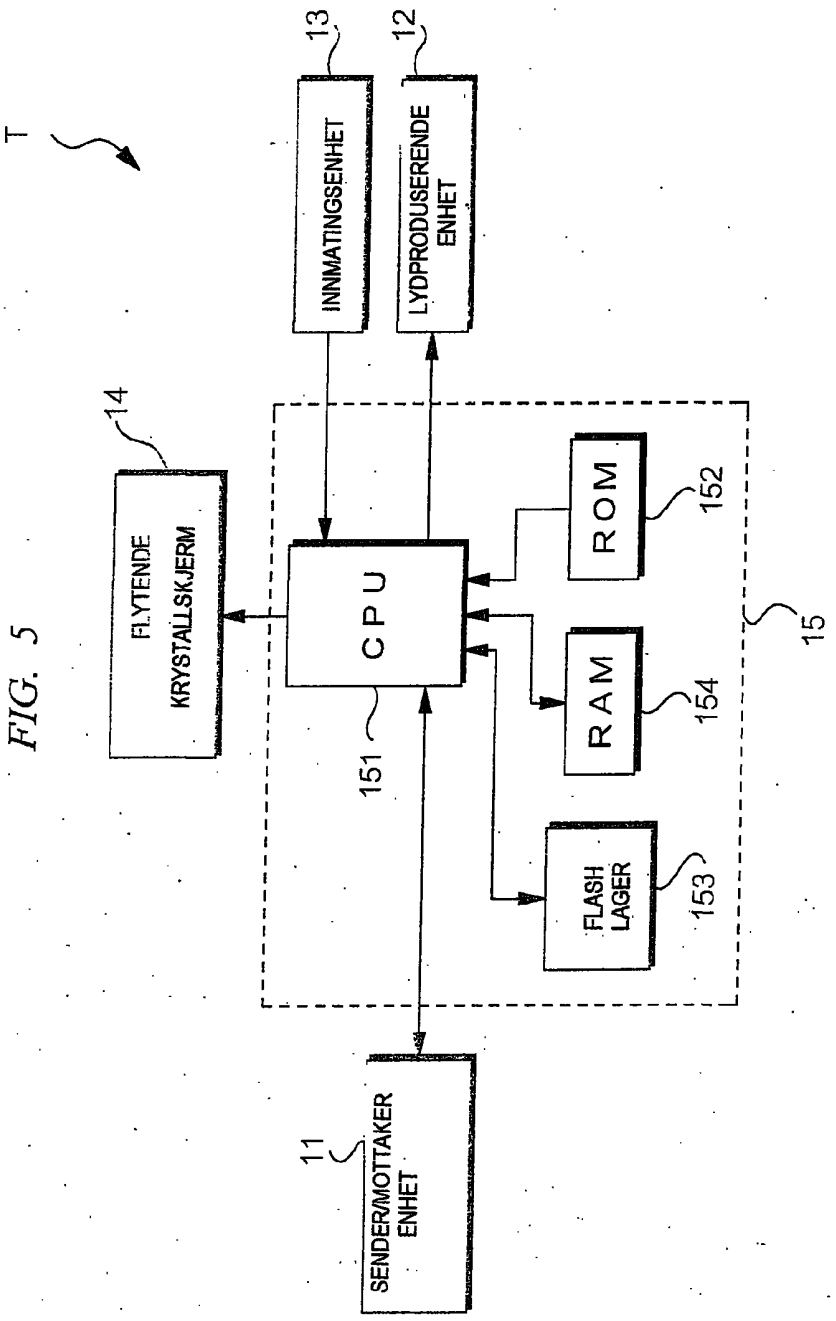

T
13
INNMATINGSENHET
12
LYDPRODUSERENDE ENHET
14
FLYTENDE KRYSTALLSKJERM
CPU
151
ROM
152
RAM
154
FLASH LAGER
153
15
11
SENDER/MOTTAKER ENHET

FIG. 6

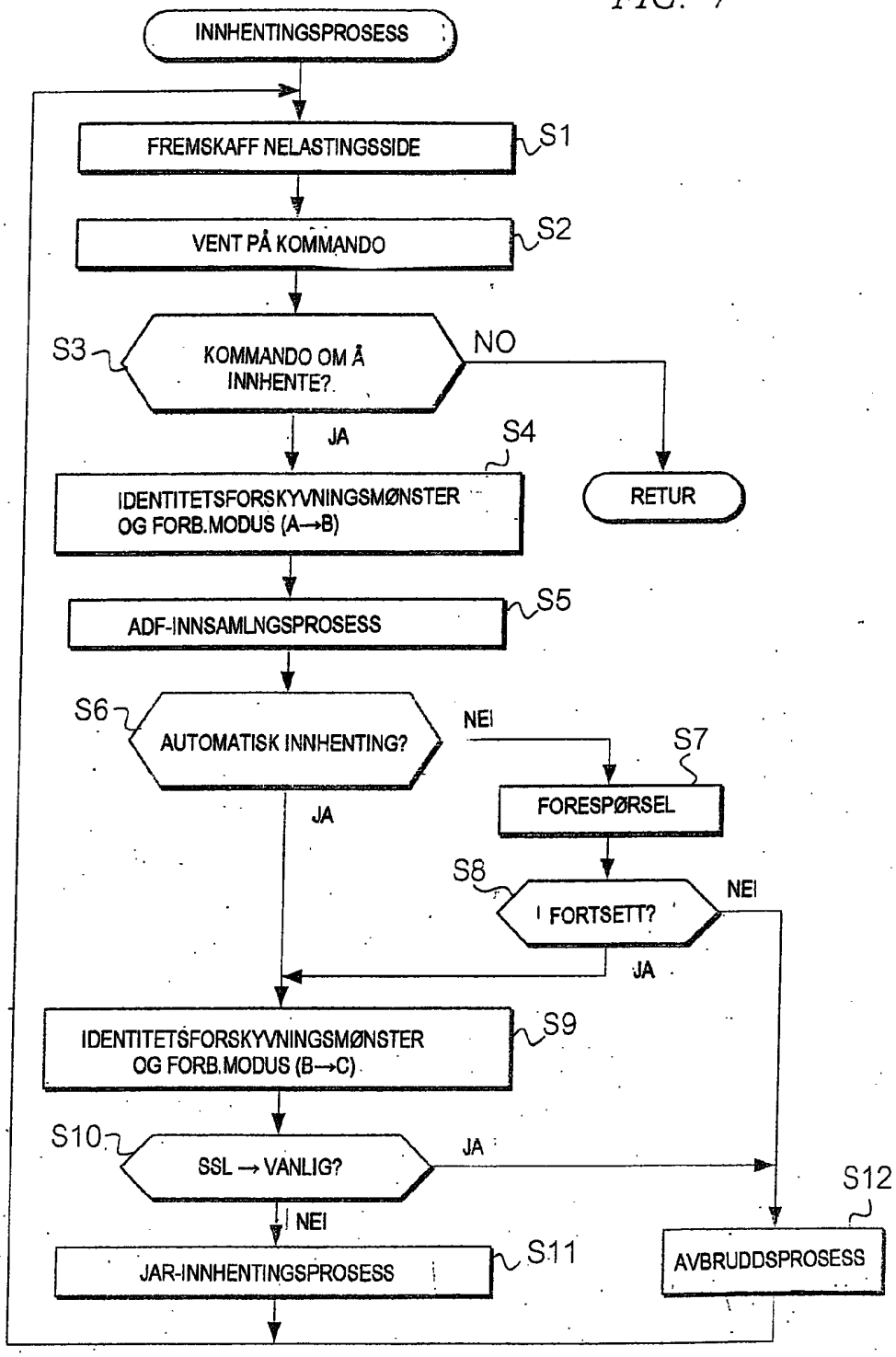
FORSKYVNINGS-MØNSTER		FORBINDELSESMODUS (A→B)	PROSESS (A→B)
TRINN A	TRINN B		
VANLIG	VANLIG	-	VANLIG KOMMUNIKASJON FORT.
VANLIG	SSL	-	NY SSL-KOMMUNIKASJON BEGYNNER
SSL	VANLIG	-	SSL-KOMMUNIKASJON SLUTTER
SSL	SSL	"IKKE HOLD I LIVE"	SSL-KOMMUNIKASJON BEGYNNER
		"HOLD I LIVE"	SSL-KOMMUNIKASJON FORTSETTER

PT1

FORSKYVNINGS-MØNSTER		FORBINDELSESMODUS (B→C)	PROSESS (B→C)
TRINN B	TRINN C		
VANLIG	VANLIG	-	VANLIG KOMMUNIKASJON FORT.
VANLIG	SSL	-	NY SSL-KOMMUNIKASJON BEGYNNER
SSL	VANLIG	-	KAN IKKE BEHANDLES
SSL	SSL	"IKKE HOLD I LIVE"	SSL-KOMMUNIKASJON BEGYNNER
		"HOLD I LIVE"	SSL-KOMMUNIKASJON FORTSETTER

PT2

FIG. 7


INNHENTINGSPROSESS
FREMSKAFF NELASTINGSSIDE
S1
VENT PÅ KOMMANDO
S2
S3
KOMMANDO OM Å INNHENTE?
NO
JA
S4
RETUR
IDENTITETSFORSKYVNINGSMØNSTER OG FORB.MODUS (A→B)
ADF-INNSAMLINGSPROSESS
S5
S6
AUTOMATISK INNHENTING?
NEI
S7
JA
FORESPØRSEL
S8
FORTSETT?
NEI
JA
IDENTITETSFORSKYVNINGSMØNSTER OG FORB.MODUS (B→C)
S9
S10
SSL → VANLIG?
JA
NEI
S12
JAR-INNHENTINGSPROSESS
S11
AVBRUDDSPROSESS