(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4467256号
(P4467256)

(45)発行日　平成22年5月26日(2010.5.26)　(24)登録日　平成22年3月5日(2010.3.5)

(51)Int.Cl.		F I		
G06F 21/20	(2006.01)	G06F	15/00	330B
G06F 15/00	(2006.01)	G06F	15/00	310D
H04L 9/32	(2006.01)	H04L	9/00	675D

請求項の数 4　(全 38 頁)

(21)出願番号　特願2003-175139(P2003-175139)
(22)出願日　平成15年6月19日(2003.6.19)
(65)公開番号　特開2005-11098(P2005-11098A)
(43)公開日　平成17年1月13日(2005.1.13)
審査請求日　平成18年5月25日(2006.5.25)

(73)特許権者　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100092152
弁理士　服部　毅巖
(72)発明者　三輪　秀成
愛知県名古屋市東区葵一丁目１６番３８号　株式会社富士通プライムソフトテクノロジ内
(72)発明者　松永　克幸
愛知県名古屋市東区葵一丁目１６番３８号　株式会社富士通プライムソフトテクノロジ内

最終頁に続く

(54)【発明の名称】代理認証プログラム、代理認証方法、および代理認証装置

(57)【特許請求の範囲】
【請求項１】
他の装置への認証手続きを代理する代理認証プログラムにおいて、
コンピュータに、
クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスし、
前記サーバ機能からの応答がユーザ認証を要求する応答であるか否かを当該応答の内容から判断する方法を示した情報である複数の認証許否確認論理が予め定義されており、前記応答が前記複数の認証許否確認論理のいずれかに適合するか否かにより、前記サーバ機能との間のユーザ認証の要否を判断し、
前記サーバ機能との間のユーザ認証が必要と判断した場合、前記クライアント機能との間のセッションの確立の有無を確認し、
前記セッションが未確立の場合、前記クライアント機能に対してユーザ認証を要求し、前記クライアント機能との間のユーザ認証が完了すると前記クライアント機能との間に前記セッションを確立し、
前記サーバ機能との間のユーザ認証が必要と判断した場合であって、前記クライアント機能との間で前記セッションが確立済みになると、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う、
処理を実行させることを特徴とする代理認証プログラム。

【請求項２】
前記クライアント機能からの前記要求を記憶し、前記サーバ機能との間の前記ユーザ認証手続き完了後、記憶しておいた前記要求を前記サーバ機能に送信することを特徴とする請求項１記載の代理認証プログラム。
【請求項３】
他の装置への認証手続きを代理するコンピュータの代理認証方法において、
前記コンピュータが、クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスし、
前記サーバ機能からの応答がユーザ認証を要求する応答であるか否かを当該応答の内容から判断する方法を示した情報である複数の認証許否確認論理が予め定義されており、前記コンピュータが、前記応答が前記複数の認証許否確認論理のいずれかに適合するか否かにより、前記サーバ機能との間のユーザ認証の要否を判断し、
前記コンピュータが、前記サーバ機能との間のユーザ認証が必要と判断した場合、前記クライアント機能との間のセッションの確立の有無を確認し、
前記コンピュータが、前記セッションが未確立の場合、前記クライアント機能に対してユーザ認証を要求し、前記クライアント機能との間のユーザ認証が完了すると前記クライアント機能との間に前記セッションを確立し、
前記コンピュータが、前記サーバ機能との間のユーザ認証が必要と判断した場合であって、前記クライアント機能との間で前記セッションが確立済みになると、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う、
ことを特徴とする代理認証方法。
【請求項４】
他の装置への認証手続きを代理する代理認証装置において、
クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスするアクセス手段と、
前記サーバ機能からの応答がユーザ認証を要求する応答であるか否かを当該応答の内容から判断する方法を示した情報である複数の認証許否確認論理が予め定義されており、前記応答が前記複数の認証許否確認論理のいずれかに適合するか否かにより、前記サーバ機能との間のユーザ認証の要否を判断する認証要否判断手段と、
前記サーバ機能との間のユーザ認証が必要と判断された場合、前記クライアント機能との間のセッションの確立の有無を確認し、前記セッションが未確立の場合、前記クライアント機能に対してユーザ認証を要求し、前記クライアント機能との間のユーザ認証が完了すると前記クライアント機能との間に前記セッションを確立する手段と、
前記サーバ機能との間のユーザ認証が必要と判断された場合であって、前記クライアント機能との間で前記セッションが確立済みになると、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う代理認証手段と、
を有することを特徴とする代理認証装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は他の装置への認証手続きを代理する代理認証プログラム、代理認証方法、および代理認証装置に関し、特に異なる認証方式による認証手続きを行う代理認証プログラム、代理認証方法、および代理認証装置に関する。
【０００２】
【従来の技術】
企業内のネットワークでは、ネットワーク内に複数のサーバ（あるいはサービスを提供するアプリケーションプログラム）が存在し、様々なサービスがネットワークを介して提供される。ただし、企業内のネットワークであっても、一部の機能に関しては使用可能なユ

ーザが制限される。たとえば、社員の出退勤を管理するシステムに対しては、所定の役職以上のユーザ、あるいは人事部等の所定の部署のユーザに対してのみアクセスを許可する必要がある。以下、サービスを提供する機能（サーバアプリケーション）をサーバ機能と呼ぶこととする。

【０００３】

一般的には、サービスを利用可能なユーザを制限するため、サーバ機能毎にユーザ認証が行われる。ユーザ認証では、サーバ機能がユーザに対して、ユーザ名とパスワードとの入力を要求する。入力されたユーザ名とパスワードとの組が予めサーバ機能に登録されていれば、サーバ機能は、そのユーザに許可されている範囲の機能を提供する。このようなユーザ認証は、ネットワーク内にサーバ機能が多数存在する場合でも、原則的に個別に行われる。

【０００４】

ところで、最近の業務処理のシステム化に伴い、社内業務の多くが、その業務を支援するサーバ機能を利用して行われる。そのため、社員が業務を遂行する上で、様々なサーバ機能に対してユーザ認証を行う必要が生じる。この場合、個別のサーバ機能へのユーザ認証を行うには、各ユーザがユーザ名とパスワードとの組を個人で管理しなければならず、パスワード等の管理負担が大きい。また、一日に何度もユーザ認証手続きを行うのは、業務の作業効率低下に繋がる。

【０００５】

そこで、従来はシングルサインオンと呼ばれるシステムが利用されている。シングルサインオンでは、ユーザが一度認証を受けるだけで、そのユーザに対したアクセスを許可している機能をユーザ利用できる。

【０００６】

シングルサインオンを実現する技術の１つにＰＫＩ(Public Key Infrastructure)を用いた技術がある。ＰＫＩは、電子署名と暗号技術を兼ね備え、安全な電子通信を確保する技術である。ＰＫＩシステムでは、ユーザ認証したユーザに対して電子証明書を発行する。そして、電子証明書を保持しているユーザは、その電子証明書に応じた各種サービスの提供を受けることができる。

【０００７】

また、シングルサインオンを実現する別の技術として、プロキシサーバで代理認証する方式がある。この技術では、各サーバ機能に対するユーザ認証のためのデータをプロキシサーバに保持する。そして、ユーザからサーバ機能へのアクセス要求が出されると、プロキシサーバが代理でサーバ機能との間でユーザ認証手続きを行う（たとえば、特許文献１参照）。

【０００８】

また、ユーザ端末からＷｅｂサーバへのプロキシサーバを介したアクセスで一度認証された場合、そのデータをプロキシサーバに記憶しておく。そして、同じユーザ端末による次回以降のログイン時には、プロキシサーバに記憶されたデータを使用する。この方式に寄れば、様々な形式のログインプロトコルに対応することができる（たとえば、特許文献２参照）。

【０００９】

【特許文献１】

特開平１０－１７７５５２号公報（第１図）

【特許文献２】

特開２００２－３２３４０号公報（第１図）

【００１０】

【発明が解決しようとする課題】

しかし、ＰＫＩを用いた技術の場合は、Ｗｅｂブラウザ、Ｗｅｂサーバともに電子証明書に対応した機構を必要とする上、電子証明書のためのインフラを必要とする。そのため、既存のシステムをそのまま流用するのは困難である。

【００１１】
すなわち、一般のプロキシサーバで代理認証する方式では固定的な認証プロトコルを用いるため、サーバ機能毎に認証方式が異なる場合には採用できない。しかも、現実のＷｅｂサーバの認証方式は開発者の自由であるため様々なものが存在する。一つの代理認証型のプロキシサーバがそれらに対し代理認証を行うのは困難であり、代理認証プロキシの有用性を減じさせている。
【００１２】
たとえば、特許文献１に記載された方式では、代理サーバで複数の認証情報を管理し、クライアントに代わってサーバに対して認証情報を送信するだけであり、サーバ毎に異なる手順の認証手続きを実行することができない。具体的には、認証情報を送信するだけで認証可能なサーバもあれば、サーバが提供するフォームデータ内の所定のパラメータとして設定した認証情報を送信することで認証可能なサーバもある。このように、単に認証情報を送信するだけでは、代理認証ができないサーバも多数存在する。
【００１３】
また、特許文献２に示された方式であればサーバ機能毎に異なる認証方式が採用されている場合にも適用可能である。しかし、特許文献２に示されたような認証時のデータを全て記録する方式では、多数のユーザが同じサーバを利用した場合は、認証用の通信データの保存域にほとんど同じだが少しだけ異なるデータを多数保持する必要が生じる。そのため、システムの資源（メインメモリやハードディスクデバイス等）の利用効率が悪くなる。これは、実際の認証用通信データで、ユーザ毎に異なる部分は僅かであり、データの多くは共通のヘッダやフッターによって占められるためである。
【００１４】
一般的に、必要なハードウェア資源の量が不明な場合、システムの安定運用のためには、余裕を持ったハードウェア環境を整えておく必要がある。そのため、システムの規模が肥大化し、システムの構築コストが増加していた。
【００１５】
本発明はこのような点に鑑みてなされたものであり、ハードウェア資源を効率よく利用して、異なる認証方式のサーバへの代理認証を行うことができる代理認証プログラム、代理認証方法、および代理認証装置を提供することを目的とする。
【００１６】
【課題を解決するための手段】
本発明では上記課題を解決するために、図１に示すような処理をコンピュータに実行させる代理認証プログラムが提供される。この代理認証プログラムは、他の装置への認証手続きを代理するためのものであり、コンピュータに対して以下の処理を実行させる。
【００１７】
コンピュータは、クライアント機能２ａ，２ｂ，２ｃ，・・・からの要求に応じて、ネットワークを介して接続されるサーバ機能３ａ，３ｂ，３ｃ，・・・に対してアクセスする（ステップＳ１）。コンピュータは、サーバ機能からの応答がユーザ認証を要求する応答であるか否かを当該応答の内容から判断する方法を示した情報である複数の認証許否確認論理が予め定義されており、サーバ機能からの応答が複数の認証許否確認論理のいずれかに適合するか否かにより、サーバ機能との間のユーザ認証の要否を判断する（ステップＳ２）。サーバ機能との間のユーザ認証が必要と判断した場合、クライアント機能との間のセッションの確立の有無を判断し、セッションが未確立の場合、クライアント機能に対してユーザ認証を要求し、クライアント機能との間のユーザ認証が完了するとクライアント機能との間にセッションを確立する。そして、サーバ機能との間のユーザ認証が必要と判断した場合であって、クライアント機能との間でセッションが確立済みになると、コンピュータは、応答が適合した認証許否確認論理に対応付けて予め定義されている認証手順に従って、サーバ機能との間でユーザ認証手続きを行う（ステップＳ３）。
【００１８】
このような代理認証プログラムによれば、コンピュータにより、クライアント機能から

の要求に応じてサーバ機能にアクセスしたとき、ユーザ認証を要求する応答が返されると、クライアント機能との間のセッションの確立の有無が確認される。セッションが未確立の場合、クライアント機能との間のセッションが確立される。そして、サーバ機能からの応答が適合する認証許否確認論理に対応する処理手順でサーバ機能との間で認証手続きが行われる。

【００１９】

また、上記課題を解決するために他の装置への認証手続きを代理するコンピュータの代理認証方法において、前記コンピュータが、クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバに対してアクセスし、前記サーバ機能からの応答がユーザ認証を要求する応答であるか否かを当該応答の内容から判断する方法を示した情報である複数の認証許否確認論理が予め定義されており、前記コンピュータが、前記応答が前記複数の認証許否確認論理のいずれかに適合するか否かにより、前記サーバ機能との間のユーザ認証の要否を判断し、前記コンピュータが、前記サーバ機能との間のユーザ認証が必要と判断した場合、前記クライアント機能との間のセッションの確立の有無を確認し、前記コンピュータが、前記セッションが未確立の場合、前記クライアント機能に対してユーザ認証を要求し、前記クライアント機能との間のユーザ認証が完了すると前記クライアント機能との間に前記セッションを確立し、前記コンピュータが、前記サーバ機能との間のユーザ認証が必要と判断した場合であって、前記クライアント機能との間で前記セッションが確立済みになると、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う、ことを特徴とする代理認証方法が提供される。

【００２０】

このような代理認証方法によれば、コンピュータにより、クライアント機能からの要求に応じてサーバ機能にアクセスしたとき、ユーザ認証を要求する応答が返されると、クライアント機能との間のセッションの確立の有無が確認される。セッションが未確立の場合、クライアント機能との間のセッションが確立される。そして、サーバ機能からの応答が適合する認証許否確認論理に対応する処理手順でサーバ機能との間で認証手続きが行われる。

【００２１】

また、上記課題を解決するために、他の装置への認証手続きを代理する代理認証装置において、クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスするアクセス手段と、前記サーバ機能からの応答がユーザ認証を要求する応答であるか否かを当該応答の内容から判断する方法を示した情報である複数の認証許否確認論理が予め定義されており、前記応答が前記複数の認証許否確認論理のいずれかに適合するか否かにより、前記サーバ機能との間のユーザ認証の要否を判断する認証要否判断手段と、前記サーバ機能との間のユーザ認証が必要と判断された場合、前記クライアント機能との間のセッションの確立の有無を確認し、前記セッションが未確立の場合、前記クライアント機能に対してユーザ認証を要求し、前記クライアント機能との間のユーザ認証が完了すると前記クライアント機能との間に前記セッションを確立する手段と、前記サーバ機能との間のユーザ認証が必要と判断された場合であって、前記クライアント機能との間で前記セッションが確立済みになると、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う代理認証手段と、を有することを特徴とする代理認証装置が提供される。

【００２２】

このような代理認証装置によれば、アクセス手段により、クライアント機能からの要求に応じてサーバ機能へのアクセスが行われる。そして、サーバ機能から応答が返されると、認証要否判断手段により、ユーザ認証の要否を判断される。ユーザ認証が必要な場合、クライアント機能との間のセッションの確立の有無が確認され、未確立の場合は、クライアント機能との間にセッションが確立される。ユーザ認証が必要な場合であってセッションが確立済みになると、代理認証手段により、応答が適合した認証許否確認論理に対応付

けて予め定義されている認証手順に従って、サーバ機能との間でユーザ認証手続きが行われる。

【００２３】

【発明の実施の形態】

以下、本発明の実施の形態を図面を参照して説明する。

まず、実施の形態に適用される発明の概要について説明し、その後、実施の形態の具体的な内容を説明する。

【００２４】

図１は、実施の形態に適用される発明の概念図である。代理認証装置１は、複数のクライアント機能２ａ，２ｂ，２ｃ，・・・と複数のサーバ機能３ａ，３ｂ，３ｃ，・・・との論理的な通信経路上に設けられている。ここで、論理的な通信経路とは、複数のクライアント機能２ａ，２ｂ，２ｃ，・・・のいずれかと複数のサーバ機能３ａ，３ｂ，３ｃ，・・・のいずれかとの間の通信（少なくとも代理認証を必要とする通信）が、必ず代理認証装置１を経由して行われることを意味する。

【００２５】

代理認証装置１は、クライアント機能２ａ，２ｂ，２ｃ，・・・がサーバ機能３ａ，３ｂ，３ｃ，・・・に対して実行すべきユーザ認証手続きを、クライアント機能２ａ，２ｂ，２ｃ，・・・の代理で実行することができる。なお、代理認証装置１には、ユーザ認証が未完了であることを示す応答を識別するための複数の認証許否確認論理が予め定義されている。また、各認証許否確認論理に対応付けて、認証手順が定義されている。認証手順は、認証手続きの際にクライアント機能２ａ，２ｂ，２ｃ，・・・側で実行すべき処理内容を、認証方式毎に定義したものである。

【００２６】

クライアント機能２ａ，２ｂ，２ｃ，・・・は、ユーザの操作入力に応答してサーバ機能３ａ，３ｂ，３ｃ，・・・に対して要求を送信する機能を有している。クライアント機能２ａ，２ｂ，２ｃ，・・・は、たとえば、それぞれが個別のクライアントコンピュータに実装されたクライアントアプリケーションである。

【００２７】

サーバ機能３ａ，３ｂ，３ｃ，・・・は、クライアント機能２ａ，２ｂ，２ｃ，・・・から送られた要求に応じた処理を行う。サーバ機能３ａ，３ｂ，３ｃ，・・・は、たとえば、プロキシサーバ等に実装されたサーバアプリケーションである。なお、サーバ機能３ａ，３ｂ，３ｃ，・・・はユーザ認証機能を有しており、ユーザ認証を完了したクライアント機能２ａ，２ｂ，２ｃ，・・・からの要求に対してのみ、処理機能を提供する。また、各サーバ機能３ａ，３ｂ，３ｃ，・・・は、それぞれ異なる認証方式を採用している。

【００２８】

ここで、クライアント機能２ａ，２ｂ，２ｃ，・・・からサーバ機能３ａ，３ｂ，３ｃ，・・・へアクセスする場合、代理認証装置１において、以下の処理が行われる。

【００２９】

代理認証装置１は、クライアント機能２ａ，２ｂ，２ｃ，・・・からの要求に応じて、ネットワークを介して接続されるサーバ機能３ａ，３ｂ，３ｃ，・・・に対してアクセスする（ステップＳ１）。アクセスされたサーバ機能は、ユーザ認証が必要であり、アクセス要求を出力したクライアント機能のユーザ認証が未完了の場合、ユーザ認証を要求する応答を代理認証装置１に返す。

【００３０】

代理認証装置１は、サーバ機能からの応答が複数の認証許否確認論理のいずれかに適合するか否かによりユーザ認証の要否を判断する（ステップＳ２）。そして、ユーザ認証が必要と判断した場合、代理認証装置１は、応答が適合した認証許否確認論理に対応付けて予め定義されている認証手順に従って、サーバ機能との間でユーザ認証手続きを行う（ステップＳ３）。

【００３１】

このような代理認証装置１によれば、クライアント機能からの要求に応じてサーバ機能にアクセスしたとき、ユーザ認証の未完了を示す応答が返されると、その応答が適合する認証許否確認論理に対応する処理手順でサーバ機能との間で認証手続きが行われる。サーバ機能からの応答を解析してサーバ機能が採用する認証方式に応じた認証手順を判断し、その認証手順で代理認証を行うことで、認証手順等を格納しておくべき記憶容量が少なくてすむ。

【００３２】

たとえば、複数のクライアント機能２ａ，２ｂ，２ｃ，・・・が同一のサーバ機能３ａにアクセスする場合を考える。このとき、従来の技術（特許文献２参照）のように全ての認証データを記録し、そのデータから認証手続きを再現することで代理認証を行う場合、クライアント機能２ａ，２ｂ，２ｃ，・・・毎の認証手続きを保存しなければならない。本発明のように、サーバ機能３ａからの応答によって該当する認証手続きを決定し、その認証手続きで代理認証を行うようにすれば、サーバ機能３ａに対応付けて１つの認証手続きを保持しておけばよい。

【００３３】

しかも、サーバ機能からの応答によって必要な認証手続きを決定するため、各サーバ機能３ａ，３ｂ，３ｃ，・・・は、それぞれ異なる認証方式を採用していても、それぞれの認証方式に応じた認証手続きで代理認証を行うことができる。

【００３４】

なお、クライアント機能２ａ，２ｂ，２ｃ，・・・毎のユーザ認証情報（たとえば、ユーザ名とパスワード）を保持する必要はあるが、これらは短い文字列からなる情報である。そのため、ユーザ認証情報は、認証手続きの処理手順を全て記録する場合に比べて、少ない記憶容量で済む。

【００３５】

以下に、本実施の形態を具体的に説明する。以下の実施の形態は、プロキシサーバに代理認証装置の機能を実装し、Ｗｅｂサーバへの代理認証を行う場合の例である。

【００３６】

図２は、本実施の形態を提供するネットワークシステムの構成例を示す図である。図２に示すように、複数のクライアント１１，１２，１３，・・・、複数のサーバ３１，３２，３３，・・・およびプロキシサーバ１００でシステムが構成されている。

【００３７】

クライアント１１，１２，１３，・・・は、ユーザが使用するコンピュータである。クライアント１１，１２，１３，・・・とプロキシサーバ１００との間は、たとえば、ＬＡＮ(Local Area Network)で接続される。ユーザは、クライアント１１，１２，１３，・・・を用いて、プロキシサーバ１００に対してログイン（ベーシック(Basic)認証ログイン）を行うことができる。本実施の形態では、各クライアント１１，１２，１３，・・・には、クライアント機能としてＷｅｂブラウザが実装されており、Ｗｅｂブラウザによってサーバ３１，３２，３３，・・・にアクセスするものとする。

【００３８】

サーバ３１，３２，３３，・・・は、クライアント１１，１２，１３，・・・からの要求に応じて所定のサービスを提供するサーバコンピュータである。サーバ３１，３２，３３，・・・とプロキシサーバ１００との間は、ＬＡＮ、高速専用回線、インターネット等で接続される。サーバ３１，３２，３３，・・・には、サーバ機能等が実装され、サーバ機能毎にユーザ認証機能を有している。本実施の形態では、各サーバ３１，３２，３３，・・・には、サーバ機能としてＷｅｂサーバが実装され、Ｗｅｂサーバによってクライアントのブラウザに対するコンテンツの提供サービス等を行う。

【００３９】

プロキシサーバ１００は、クライアント１１，１２，１３，・・・の代理でサーバ３１，３２，３３，・・・にアクセスするコンピュータである。プロキシサーバ１００には、複数のログインアルゴリズム（ベーシック認証やフレーム認証）が定義されている。サーバ

３１，３２，３３，・・・からユーザ認証が要求された場合、プロキシサーバ１００は要求されたユーザ認証用のログインアルゴリズムを識別し、該当するアルゴリズムでログイン手続きを行うことができる。たとえば、プロキシサーバ１００は、サーバ３１，３２，３３，・・・にアクセスする際、ユーザ認証を要求されると、クライアント１１，１２，１３，・・・に代わってユーザ認証手続きを行い、サーバ３１，３２，３３，・・・にログインする。

【００４０】

このようなシステムにより、ユーザはクライアント（たとえば、クライアント１１）を利用してサーバ３１，３２，３３，・・・にアクセスできる。そして、１つのＷｅｂサーバ（たとえば、サーバ３１）からユーザ認証が要求されると、プロキシサーバ１００からクライアント１１にユーザ認証が要求される。すると、ユーザがクライアント１１を操作し、プロキシサーバ１００に対してベーシック認証ログインを行う。

【００４１】

プロキシサーバ１００は、クライアント１１から送られるユーザ認証情報を、内部に保持するユーザ情報テーブルと照合することでベーシック認証ログインを認め、サーバ３１に対してログインする。その際、プロキシサーバ１００は、サーバ３１の認証方式に合ったアルゴリズムでログイン手続きを行う。

【００４２】

その後、クライアント１１から他のＷｅｂサーバ３２，３３，・・・へのアクセスの際にユーザ認証が要求されると、そのＷｅｂサーバの認証方式に合ったアルゴリズムで、プロキシサーバ１００がログイン手続きを行う。ログイン手続き完了後は、クライアント１１が要求するサービスがＷｅｂサーバにおいて提供され、結果がクライアント１１に渡される。

【００４３】

図３は、プロキシサーバのハードウェア構成例を示す図である。プロキシサーバ１００は、ＣＰＵ(Central Processing Unit)１０１によって装置全体が制御されている。ＣＰＵ１０１には、バス１０７を介してＲＡＭ(Random Access Memory)１０２、ハードディスクドライブ（ＨＤＤ:Hard Disk Drive）１０３、グラフィック処理装置１０４、入力インタフェース１０５、および通信インタフェース１０６が接続されている。

【００４４】

ＲＡＭ１０２には、ＣＰＵ１０１に実行させるＯＳ(Operating System)のプログラムやアプリケーションプログラムの少なくとも一部が一時的に格納される。また、ＲＡＭ１０２には、ＣＰＵ１０１による処理に必要な各種データが格納される。ＨＤＤ１０３には、ＯＳやアプリケーションプログラムが格納される。

【００４５】

グラフィック処理装置１０４には、モニタ１１が接続されている。グラフィック処理装置１０４は、ＣＰＵ１０１からの命令に従って、画像をモニタ１１の画面に表示させる。入力インタフェース１０５には、キーボード１２とマウス１３とが接続されている。入力インタフェース１０５は、キーボード１２やマウス１３から送られてくる信号を、バス１０７を介してＣＰＵ１０１に送信する。

【００４６】

通信インタフェース１０６は、ネットワーク１０に接続されている。通信インタフェース１０６は、ネットワーク１０を介して、他のコンピュータ（クライアント１１，１２，１３，・・・やサーバ３１，３２，３３，・・・）との間でデータの送受信を行う。

【００４７】

以上のようなハードウェア構成によって、本実施の形態の処理機能を実現することができる。なお、図３には、プロキシサーバ１００のハードウェア構成を示したが、クライアント１１，１２，１３，・・・やサーバ３１，３２，３３，・・・も同様のハードウェア構成で実現することができる。

【００４８】

次に、プロキシサーバ１００の機能について具体的に説明する。なお、以下の説明では、クライアント１１からサーバ３１やサーバ３２に対してアクセスする場合を例に採る。

【００４９】

図４は、プロキシサーバの機能を示すブロック図である。なお、図４に示すように、プロキシサーバ１００には、クライアント１１に実装されたＷｅｂブラウザ１１ａとサーバ３１，３２に実装されたＷｅｂサーバ３１ａ，３２ａとの間で通信を行うことができる。

【００５０】

プロキシサーバ１００は、データベース１１０、セッション情報テーブル１２１、タイマ１２２、タイムアウト駆動部１２３、ブラウザ入力受付部１３１、プロキシサーバ認証部１３２、プロキシサーバ認証要求部１３３、通信状態保持部１３４、Ｗｅｂサーバ識別部１３５、Ｗｅｂサーバ要求生成部１３６、Ｗｅｂサーバ通信部１３７、レスポンス分析部１３８、認証処理判定部１３９、ログイン処理命令生成部１４０、コンテンツ変換部１４１、およびブラウザ応答生成部１４２を有している。

【００５１】

データベース１１０には、代理認証に必要なデータが格納されている。データベース１１０の内容は、ユーザ情報テーブル１１１とＷｅｂサーバ情報１１２とに分かれる。

【００５２】

ユーザ情報テーブル１１１は、クライアント１１，１２，１３，・・・を使用するユーザに関する情報が登録されたデータベースである。たとえば、ユーザ情報テーブル１１１には、各ユーザがプロキシサーバ１００にベーシックログインするためのユーザ認証情報や、ユーザがサーバ３１，３２，３３，・・・にログインするためのユーザ認証情報が、ユーザ毎に格納されている。本実施の形態では、ユーザ情報テーブル１１１の内容は、システムの管理者によって予め設定されるものとする。

【００５３】

Ｗｅｂサーバ情報１１２は、代理認証の対象とするＷｅｂサーバに関する情報である。Ｗｅｂサーバ情報１１２は、システムテーブル１１２ａとレスポンス分析・ログイン命令テーブル１１２ｂとに別れている。本実施の形態では、Ｗｅｂサーバ情報１１２の内容は、システムの管理者によって予め設定されるものとする。

【００５４】

システムテーブル１１２ａは、サーバ機能毎にＷｅｂサーバのＵＲＬ(Uniform Resource Locator)や認証方式が管理されている。

レスポンス分析・ログイン命令テーブル１１２ｂには、Ｗｅｂサーバへのログインに必要な情報が格納されている。

【００５５】

セッション情報テーブル１２１には、クライアントとの間で確立したセッションに関する情報が登録されている。セッションに関する情報には、タイムアウト時刻も含まれる。セッション情報テーブル１２１の内容は、システムの運用中に動的に追加や削除される。

【００５６】

ここで、プロキシサーバ１００の他の要素を説明する前に、各情報のデータ構造例について詳細に説明する。

図５は、ユーザ情報テーブルのデータ構造例を示す図である。ユーザ情報テーブル１１１には、プロキシサーバ認証情報に対応付けて、システムＩＤ、ユーザ名およびパスワードの組がＷｅｂサーバ毎に設定されている。各欄の横方向に並べられた情報同士が互いに関連づけられて、各ユーザのユーザ情報を構成している。

【００５７】

プロキシサーバ認証情報の欄は、ユーザ名の欄とパスワードの欄とに分かれている。ユーザ名の欄には、ユーザ名が設定される。パスワードの欄には、ユーザ名に対応するパスワードが設定される。

【００５８】

システムＩＤの欄には、ユーザ認証対象となるＷｅｂサーバの識別情報が設定される。図

５の例では、「Ｓｙｓ」に続くアルファベットがサーバ（ネットワーク上のノード）の識別子であり、アルファベットに続く数字がサーバ内のＷｅｂサーバの識別子である。従って、「ＳｙｓＡ０」と「ＳｙｓＡ１」とは、同じサーバ内の別々のＷｅｂサーバを示している。

【００５９】

ユーザ名の欄には、Ｗｅｂサーバに設定されているユーザ認証情報のユーザ名が設定される。パスワードの欄には、Ｗｅｂサーバに設定されているユーザ認証情報のパスワードが設定される。

【００６０】

図６は、システムテーブルのデータ構造例を示す図である。システムテーブル１１２ａには、システムＩＤ、仮想ＵＲＬ、実ＵＲＬ、および認証タイプの欄が設けられている。各欄の横方向に並べられた情報同士が互いに関連づけられて、各Ｗｅｂサーバに関する情報を構成している。

【００６１】

システムＩＤの欄には、Ｗｅｂサーバの識別情報が設定される。

仮想ＵＲＬの欄には、クライアントからＷｅｂサーバにアクセスするための仮想的なＵＲＬが設定される。Ｗｅｂブラウザからは、仮想ＵＲＬを指定したアクセス要求が出される。

【００６２】

実ＵＲＬの欄には、Ｗｅｂサーバの実際の所在を示すＵＲＬが設定される。なお、仮想ＵＲＬと実ＵＲＬとは同一の場合もある。

認証タイプの欄には、Ｗｅｂサーバにおけるユーザ認証方式の名称が設定される。図６の例では、認証タイプとして「フォーム」、「ベーシック」、「ｎｕｌｌ」が設定されている。

【００６３】

フォーム認証は、認証されていない要求をＷｅｂサーバ３１ａ側でＨＴＭＬフォームにリダイレクトするシステムである。クライアント（プロキシサーバ１００がＷｅｂブラウザ１１ａの代わりにクライアントとして機能する）には、認証用のフォームが渡される。そこで、フォームに対して認証情報を入力し、認証情報を含むフォームをＷｅｂサーバ３１ａに送信する。

【００６４】

ベーシック認証では、認証されていない要求を受け取ったＷｅｂサーバ３１ａから認証を要求する応答が返されると、クライアント（プロキシサーバ１００がＷｅｂブラウザ１１ａの代わりにクライアントとして機能する）に認証情報（ユーザ名とパスワード）入力用のダイアログが表示される。そのダイアログに対して認証情報を入力すると、その認証情報がＷｅｂサーバ３１ａに送信される。

【００６５】

ｎｕｌｌは、認証が不要であることを示している。

なお、仮想ＵＲＬは、ＨＴＴＰ(HyperText Transfer Protocol)のクッキー送信範囲の制限を回避し、外部（Ｗｅｂサーバ３１，３２，３３，・・・側）のネットワーク上のサーバに対し、内部（クライアント１１，１２，１３，・・・側）にあるサーバと同じようにシングルサインオンを提供するために利用される。すなわち、本実施の形態では、プロキシサーバ１００でのユーザ認証が完了しているか否かを示すセッション情報を、クライアントにおいてＨＴＴＰクッキーで管理している。このＨＴＴＰクッキーはその仕様上、ドメイン名が後方一致している場合に限り、異なるドメインに対して設定することができる。

【００６６】

たとえば、プロキシサーバ１００のドメイン名がaaa.bbb.comであり、ユーザがアクセスしたいサーバ機能のドメイン名がddd.bbb.comである場合は、bbb.comの部分が同じ（後方一致）である。したがって、プロキシサーバ１００が仮想ＵＲＬを用いなくてもクッキー

機構が機能する。しかし、対象となるＷｅｂサーバのドメイン名がxxx.zzz.comだった場合には後方一致せず、プロキシサーバ１００がその通信を操作し、認証情報をクッキーに付加したとしても、ブラウザ側は仕様によりaaa.bbb.comプロキシサーバ１００のクッキーとは認められない。そのため、本実施の形態の認証ロジックが成立しない。

【００６７】

そこで本実施の形態では、xxx.zzz.comの仮想ＵＲＬをxxx.bbb.comと設定する。ユーザはＷｅｂブラウザからＵＲＬとしてxxx.bbb.comを指定して、プロキシサーバ１００経由でアクセスを行う。プロキシサーバ１００は内部で仮想ＵＲＬ変換を行い、実際の通信をxxx.zzz.comに対して行う。プロキシサーバ１００はxxx.zzz.comからの応答データとクッキーを組み合わせてユーザのＷｅｂブラウザに返送する。このときＷｅｂブラウザはＵＲＬがxxx.zzz.comのＷｅｂサーバからではなく、ＵＲＬがxxx.bbb.comのＷｅｂサーバからの通信と判断する。そのため、クッキーは無事受け入れられる。これにより、本発明の効果がインターネット全体に適用可能で、実用性が高まる。

【００６８】

図７は、レスポンス分析・ログイン命令テーブルのデータ構造例を示す図である。レスポンス分析・ログイン命令テーブル１１２ｂには、システムＩＤ、ログイン処理情報、および認証識別情報の欄が設けられている。各欄の横方向に並べられた情報同士が互いに関連づけられ、各Ｗｅｂサーバへのユーザ認証手続きに必要な情報を構成している。

【００６９】

システムＩＤの欄には、Ｗｅｂサーバの識別情報が設定される。

ログイン処理情報の欄には、Ｗｅｂサーバへのユーザ認証（ログイン）手続きに必要な情報が設定される。ログイン処理情報の欄には、認証ＵＲＬ、メソッド、フォーム名の欄が設けられている。

【００７０】

認証ＵＲＬの欄には、該当Ｗｅｂサーバへログインする際にアクセスすべきＵＲＬが設定される。

メソッドの欄には、ＨＴＴＰにおけるリクエストの種類（メソッド）が設定さえる。メソッドには、「ＧＥＴ」や「ＰＯＳＴ」がある。ＧＥＴは、Ｗｅｂサーバから情報を引き出すためのメソッドである。ＰＯＳＴは、Ｗｅｂサーバに対して情報を送信するためのメソッドである。

【００７１】

フォーム名の欄には、認証情報を設定すべきフォームの名称が設定される。フォーム名の欄には、ユーザ、パスワード、付加データの欄が設けられている。ユーザの欄には、ユーザ名を設定すべきフォームの名称が設定される。パスワードの欄には、パスワードを設定すべきフォームの名称が設定される。付加データの欄には、ユーザ認証に際して付加すべき情報を設定するためのフォームの名称が設定される。この付加データの欄には、１つのシステムＩＤに対応付けて１個あるいは複数（ｍ個：ｍは１以上の自然数）の情報を設定することができる。

【００７２】

認証識別情報の欄には、Ｗｅｂサーバからの応答が認証を要求しているか否かを識別するための情報が設定される。認証識別情報の欄には、レスポンスコード、認証ヘッダ、および識別内容データの欄が設けられている。

【００７３】

レスポンスコードの欄には、Ｗｅｂサーバからの応答の種別を示すコード（レスポンスコード）が設定される。識別ヘッダの欄には、認証の要求を示す応答のヘッダに含まれるべき情報が設定される。この識別ヘッダの欄には、１つのシステムＩＤに対応付けて１個あるいは複数（ｎ個：ｎは１以上の自然数）の情報を設定することができる。識別内容データの欄には、認証の要求を示す応答の内容に含まれるべき情報が設定される。

【００７４】

図８は、セッション情報テーブルのデータ構造例を示す図である。セッション情報テーブ

ル１２１には、クライアントとの間で確立したセッションに関する情報が設定される。セッション情報テーブル１２１には、セッションＩＤ、ユーザ名、ログイン時刻、およびタイムアウト予告時刻の欄が設けられている。各欄の横方向に並べられた情報同士が互いに関連づけられ、セッション毎の情報を構成している。

【００７５】

セッションＩＤの欄には、確立されたセッションの識別情報が設定される。ユーザ名の欄には、セッションの確立を要求したユーザの識別情報が設定される。ログイン時刻の欄には、ユーザがプロキシサーバ１００に対してログイン（ユーザ認証が成功したこと）した時刻が設定される。タイムアウト予告時刻には、セッションを切断する時刻が設定される。

【００７６】

以上が、プロキシサーバ１００で管理される各データベースのデータ構造である。以下、図４の説明に戻り、プロキシサーバ１００の他の要素について説明する。

【００７７】

タイマ１２２は、時刻を計時する機能を有している。ＯＳは、タイマ１２２を用いて定期的にタイマーイベントを発生させ、そのタイマーイベントを契機にタイムアウト駆動部１２３が起動される。

【００７８】

タイムアウト駆動部１２３は、起動されるとセッション情報テーブル１２１を走査し、タイムアウト予定時刻を過ぎたセッション情報をセッション情報テーブル１２１から削除する。

【００７９】

ブラウザ入力受付部１３１は、Ｗｅｂブラウザ１１ａからの入力を受け付ける。ブラウザ入力受付部１３１は、受け付けた内容をプロキシサーバ認証部１３２、通信状態保持部１３４、およびＷｅｂサーバ識別部１３５に渡す。

【００８０】

プロキシサーバ認証部１３２は、Ｗｅｂブラウザ１１ａからの入力がプロキシサーバへの認証情報を含むか否かを確認する。入力内容に、既に認証済みを示すプロキシサーバセッションＩＤ情報が含まれていた場合は、プロキシサーバ認証部１３２は、セッション情報テーブル１２１を確認し、有効な接続か否かを確認する。有効な接続だった場合、プロキシサーバ認証部１３２は、タイムアウト時間を延長するために、"タイムアウト予定時刻"を更新することもできる。

【００８１】

また、入力情報にユーザ名とパスワード情報（AUTHORIZEDヘッダ内に含まれる）が含まれていた場合は、プロキシサーバ認証部１３２は、ユーザ情報テーブル１１１からプロキシサーバ認証用のユーザ名、パスワードと比較する。一致する場合は、プロキシサーバ認証部１３２は、セッション情報テーブル１２１にセッション情報を登録する。一致しない場合は、プロキシサーバ認証部１３２は、プロキシサーバ認証要求部１３３に対してプロキシサーバ認証を依頼する。

【００８２】

プロキシサーバ認証部１３２は、Ｗｅｂブラウザ１１ａからの入力内容にプロキシサーバへの認証情報（セッションＩＤ、ユーザ名、パスワード）を含まない場合は、何もしない（認証情報を含まないことは許可される）。

【００８３】

プロキシサーバ認証要求部１３３は、プロキシサーバ認証部１３２からの依頼に応じて、ＨＴＴＰのUNAUTHORIZEDレスポンスを生成する。そして、プロキシサーバ認証要求部１３３は、生成したUNAUTHORIZEDレスポンスをユーザのＷｅｂブラウザ１１ａに返す。ここで、UNAUTHORIZEDレスポンスとは、Ｗｅｂブラウザ１１ａに対して認証情報の送信を要求する通信パケットである。

【００８４】

通信状態保持部１３４は、後で使用するために、ブラウザから受けたＨＴＴＰ要求（ＭＥＴＨＯＤ，ＵＲＬ，ヘッダ、ＰＯＳＴデータ）を保存する。
Ｗｅｂサーバ識別部１３５は、Ｗｅｂブラウザからの入力内容がプロキシ通信要求の場合、そのプロキシ通信要求に基づいて通信先のＵＲＬを識別する。なお、Ｗｅｂブラウザ１１ａから指定されるＵＲＬは仮想ＵＲＬである。Ｗｅｂサーバ識別部１３５は、要求されたＵＲＬが含まれる情報をシステムテーブル１１２ａから検索する。検索時にはシステムテーブル１１２ａの仮想ＵＲＬの項目を対象にする。要求されたＵＲＬが仮想ＵＲＬに一致するか、あるいは仮想ＵＲＬの配下に含まれる場合、そのシステムエントリを用いる。Ｗｅｂサーバ識別部１３５は、要求されたＵＲＬを含む仮想ＵＲＬに対応する実ＵＲＬをＷｅｂサーバ要求生成部１３６に渡す。

【００８５】
以後の処理でプロキシサーバ１００から実際のＷｅｂサーバ３１ａ，３２ａへ通信がなされる場合は、仮想ＵＲＬが実ＵＲＬに変換される。なお、仮想ＵＲＬと実ＵＲＬは同じでもかまわない。

【００８６】
なお、仮想ＵＲＬ、実ＵＲＬともに物理的には同じ計算機であってもよい。本実施の形態では、サーバ機能がＵＲＬ毎に管理されるため、一つの計算機を複数のシステムとして扱うことが許される。

【００８７】
Ｗｅｂサーバ要求生成部１３６は、ＨＴＴＰ要求を生成する。その際、通信状態保持部１３４のデータを利用する。また、セッション情報テーブル１２１を確認し、プロキシサーバ１００で認証済みの場合は、プロキシサーバ１００のユーザ名等の認証情報を、ＨＴＴＰのクッキーとして、ＷｅｂサーバへのＨＴＴＰ要求に付加する。そして、Ｗｅｂサーバ要求生成部１３６は、クッキーが付加されたＨＴＴＰ要求をＷｅｂサーバ通信部１３７に渡す。

【００８８】
また、Ｗｅｂサーバ要求生成部１３６は、レスポンス分析部１３８による応答内容の分析の結果、Ｗｅｂサーバ３１ａ，３２ａに対するフォーム形式のログインが完了していたとき、応答内容を受け取る。すると、Ｗｅｂサーバ要求生成部１３６は、認証応答で得られたクッキー情報やセッション情報テーブル１２１から生成したプロキシサーバのユーザ名等の情報をＨＴＴＰクッキーに含める。そして、Ｗｅｂサーバ要求生成部１３６は、通信状態保持部１３４に保存されているユーザのＷｅｂブラウザからの要求を組み合わせ、Ｗｅｂサーバ通信部１３７からＷｅｂサーバ３１ａ，３２ａに送信する。

【００８９】
Ｗｅｂサーバ通信部１３７は、生成されたＨＴＴＰ要求をシステムテーブル１１２ａの実ＵＲＬから生成した通信先に送る。また、Ｗｅｂサーバ通信部１３７は、Ｗｅｂサーバからの通信応答をレスポンス分析部１３８に渡す。

【００９０】
レスポンス分析部１３８は、システムテーブル１１２ａの認証タイプを元に処理を切り替える。また、レスポンス分析部１３８は、認証タイプが「ｎｕｌｌ」の場合は、そのシステムにはアクセス制限は存在しないので、応答内容をコンテンツ変換部１４１に渡す。

【００９１】
レスポンス分析部１３８の認証の結果、ログイン処理が成功しており、かつシステムテーブル１１２ａの認証タイプがフォームだった場合、レスポンス分析部１３８は応答内容をＷｅｂサーバ要求生成部１３６に渡す。認証タイプがフォーム以外の場合、レスポンス分析部１３８はコンテンツ変換部１４１に応答内容を渡す。

【００９２】
さらに、レスポンス分析部１３８は、認証タイプが「ベーシック」の場合はサーバ応答のＨＴＴＰヘッダを調べ、UNAUTHORIZEDか否かを確認する。レスポンス分析部１３８は、UNAUTHORIZEDの場合は認証処理判定部１３９へ応答内容を渡す。それ以外の場合は、レスポ

ンス分析部１３８はコンテンツ変換部１４１に応答内容を渡す。
【００９３】
認証タイプが「フォーム」の場合、レスポンス分析部１３８はレスポンス分析・ログイン命令テーブル１１２ｂを参照する。認証識別情報の項を用い、応答ヘッダ中のレスポンスコード、特定の識別ヘッダ、固有のコンテンツ識別内容データを比較する。これらが一致した場合は、レスポンス分析部１３８は該当するＷｅｂサーバから認証上の問題でアクセス拒否されたとみなし、認証処理判定部１３９に応答内容を渡す。それ以外は、レスポンス分析部１３８はコンテンツ変換部１４１に応答内容を渡す。
【００９４】
認証処理判定部１３９はセッション情報テーブル１２１を調べ、現在の通信要求者がプロキシサーバへの認証を経ているか確認する。セッション情報テーブル１２１に当該ユーザ名が存在しない場合はプロキシサーバ認証要求部１３３に処理を移す。存在する場合はログイン処理命令生成部１４０に移る。
【００９５】
ログイン処理命令生成部１４０は、システムテーブル１１２ａの現在のシステムＩＤ（要求されたＵＲＬに対応するシステムＩＤ）と一致するユーザ名（通信要求を出したユーザのユーザ名）／パスワードの組をユーザ情報テーブル１１１から得る。また、ログイン処理命令生成部１４０は、システムテーブル１１２ａを調べ、認証タイプによって処理を切り替える。
【００９６】
認証タイプが「ベーシック」だった場合、ログイン処理命令生成部１４０は、先にユーザ情報テーブル１１１から得た、特定のＷｅｂサーバ向けのユーザ名/パスワードをＨＴＴＰのベーシック認証に対応した送信用データを生成する。
【００９７】
認証タイプが"フォーム"だった場合、ログイン処理命令生成部１４０はレスポンス分析・ログイン命令テーブル１１２ｂのログイン処理情報の項の情報とユーザ情報テーブル１１１から得た、特定のＷｅｂサーバ向けのユーザ名／パスワードを組み合わせ、ＨＴＴＰのフォーム通信のデータを生成する。
【００９８】
ログイン処理命令生成部１４０は、ログイン処理用に生成した送信データをＷｅｂサーバ通信部１３７より該当するＵＲＬに送信する。Ｗｅｂサーバからの応答は上記と同様レスポンス分析部１３８で分析される。
【００９９】
コンテンツ変換部１４１は、応答内容をレスポンス分析部１３８から受け取ったとき、その応答がコンテンツ書き換えの条件を満たす場合、Ｗｅｂサーバ３１ａ，３２ａからの応答の内容（ＨＴＭＬ文書）を書き換える。書き換えの際には、現在の利用者の情報をセッション情報テーブル１２１とユーザ情報テーブル１１１を使って、ユーザ毎に異なるＨＴＭＬ変換を実施する。
【０１００】
ブラウザ応答生成部１４２は、コンテンツ変換部１４１より得たＷｅｂサーバからの応答に、セッション情報テーブル１２１から得た情報を元に生成したプロキシサーバ認証情報を含んだＨＴＴＰクッキーを組み合わせ、リモートＷｅｂブラウザに送信する。
【０１０１】
以上のような構成のシステムにおける処理をフローチャートを参照して説明する。
図９は、代理認証処理の手順を示す第１のフローチャートである。これは、Ｗｅｂブラウザ１１ａからＷｅｂサーバ３１ａにアクセスする場合の例である。以下、図９に示す処理をステップ番号に沿って説明する。
【０１０２】
［ステップＳ１１］Ｗｅｂブラウザ１１ａは、ユーザからの操作入力によるＷｅｂサーバ３１ａへのアクセス指示を受け付ける。たとえば、表示画面からＷｅｂサーバ３１ａのＵ

ＲＬが関連付けられたアンカー表示の選択入力（マウスによるクリック等）を受け付ける。

【０１０３】

［ステップＳ１２］Ｗｅｂブラウザ１１ａは、プロキシサーバ１００に対して、Ｗｅｂサーバ３１ａへのアクセス要求（ブラウザ要求）を送信する。

［ステップＳ１３］プロキシサーバ１００では、ブラウザ入力受付部１３１がブラウザ要求を受信する。

【０１０４】

［ステップＳ１４］プロキシサーバ１００において、指定されたＵＲＬに基づき、アクセス対象のＷｅｂサーバ３１ａに対応した認証許否確認論理とログイン処理論理（認証手続き）とを取得する。認証許否確認論理は、システムテーブル１１２ａの認証タイプと、レスポンス分析・ログイン命令テーブル１１２ｂにおける認証識別情報とによって特定される。ログイン処理論理は、システムテーブル１１２ａの認証タイプと、レスポンス分析・ログイン命令テーブル１１２ｂのログイン処理情報とによって特定される。

【０１０５】

［ステップＳ１５］プロキシサーバ１００は、アクセス要求内のセッションを特定するクッキーの有無を確認する。そして、プロキシサーバ１００は、クッキーが有る場合、該当するセッションがセッション情報テーブル１２１に登録されているか否かを判断する。該当するセッションが有る場合、処理がステップＳ１６に進められる。該当するセッションが無い場合、処理がステップＳ１７に進められる。

【０１０６】

［ステップＳ１６］プロキシサーバ１００は、セッション情報テーブル１２１内の該当するセッションの情報からユーザ名を取得する。その後、処理がステップＳ３１（図１０参照）に進められる。

【０１０７】

［ステップＳ１７］プロキシサーバ１００は、受信したアクセス要求にAuthorizedヘッダが含まれるか否かを判断する。Authorizedヘッダは、Ｗｅｂブラウザ１１ａがユーザ認証情報を送信するために付加する情報である。Ｗｅｂブラウザ１１ａは、プロキシ認証レスポンスを受け取ったときに、プロキシサーバ１００へのユーザ認証情報（ユーザ名とパスワード）をAuthorizedヘッダに含めて、アクセス要求をプロキシサーバ１００に送信するように設計されている。Authorizedヘッダが含まれる場合には、処理がステップＳ１８に進められる。Authorizedヘッダが含まれない場合には、処理がステップＳ３１（図１０参照）に進められる。

【０１０８】

［ステップＳ１８］プロキシサーバ１００は、アクセス要求からAuthorizedヘッダからユーザ名/パスワード取得を取得する。

［ステップＳ１９］プロキシサーバ１００は、ユーザ情報テーブル１１１を参照し、ユーザ情報を取得する。具体的には、プロキシサーバ１００は、Authorizedヘッダに含まれていたユーザ名を、ユーザ情報テーブル１１１のプロキシサーバ認証情報のユーザ名の欄から検索する。そして、プロキシサーバ１００は、該当するユーザ名に対応付けられたパスワードを取得する。

【０１０９】

［ステップＳ２０］プロキシサーバ１００は、認証が成功したか否かを判断する。具体的には、プロキシサーバ１００は、Authorizedヘッダに含まれていたユーザ認証情報と一致するプロキシサーバ認証情報がユーザ情報テーブル１１１に登録されていれば、認証成功と判断する。該当するユーザ名がユーザ情報テーブル１１１に登録されていないか、該当するユーザ名のパスワードが異なっている場合には、認証失敗と判断する。認証が成功した場合、処理がステップＳ２２に進められる。認証失敗の場合、処理がステップＳ２１に進められる。

【０１１０】

［ステップＳ２１］認証に失敗した場合、プロキシサーバ１００は、プロキシ認証レスポンスを生成し、Ｗｅｂブラウザ１１ａに返信する。プロキシ認証レスポンスとは、プロキシサーバへのユーザ認証手続きを要求するパケットである。その後、処理が終了し、Ｗｅｂブラウザ１１ａからの要求を待つ。

【０１１１】

［ステップＳ２２］認証に成功した場合、プロキシサーバ１００は、セッションを生成する。生成したセッションには、ユニークなセッションＩＤが付与される。

【０１１２】

［ステップＳ２３］プロキシサーバ１００は、生成したセッションにユーザ情報を登録する。具体的には、プロキシサーバ１００は、生成したセッションに付与されたセッションＩＤに対し、ユーザ名、ログイン時刻、およびタイムアウト予定時刻を関連付けて、セッション情報テーブル１２１に登録する。ログイン時刻は、セッションの生成時刻である。タイムアウト予定時刻は、たとえば、セッション生成から所定の時間後の時刻を設定する。

【０１１３】

図１０は、代理認証処理の手順を示す第２のフローチャートである。以下、図１０に示す処理をステップ番号に沿って説明する。

［ステップＳ３１］プロキシサーバ１００は、ブラウザからのアクセス要求を受け取り、内部に保存する。

【０１１４】

［ステップＳ３２］プロキシサーバ１００は、Ｗｅｂサーバ３１ａのＵＲＬ（実ＵＲＬ）を取得し、Ｗｅｂサーバへのアクセス要求を生成する。生成されたアクセス要求は、プロキシサーバ１００によってＷｅｂサーバ３１ａに送信される。

【０１１５】

［ステップＳ３３］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答を受信する。

［ステップＳ３４］プロキシサーバ１００は、応答内容を解析し、Ｗｅｂサーバ３１ａによる処理結果を判定する。具体的には、プロキシサーバ１００は、システムテーブル１１２ａを参照し、Ｗｅｂサーバ３１ａの認証タイプを判断する。たとえば、プロキシサーバ１００は、認証タイプがベーシックであり、Ｗｅｂサーバ３１ａからの応答のＨＴＴＰヘッダがUNAUTHORIZEDであれば、ベーシック認証の手続きが必要であると判断する。また、プロキシサーバ１００は、認証タイプがフォームの場合、レスポンス分析・ログイン命令テーブル１１２ｂを参照し、応答内容が該当するシステムＩＤに対応する認証識別情報に合致する場合には、フォーム認証の手続きが必要であるものと判断する。

【０１１６】

［ステップＳ３５］プロキシサーバ１００は、アクセスが許可されたか否かを判断する。すなわち、ベーシックやフォームの認証手続きを必要と判断するための条件のいずれにも該当しなければ、アクセスが許可されたものと判断される。アクセスが許可された場合、処理がステップＳ６１（図１２参照）に進められる。アクセスが許否された場合、処理がステップＳ４１（図１１参照）に進められる。

【０１１７】

図１１は、代理認証処理の手順を示す第３のフローチャートである。以下、図１１に示す処理をステップ番号に沿って説明する。

［ステップＳ４１］プロキシサーバ１００は、Ｗｅｂサーバ３１ａへのアクセスが拒否された場合、セッション情報テーブル１２１を参照し、Ｗｅｂブラウザ１１ａを介してアクセス要求を出したユーザのユーザ名に対応するセッションの有無を判断する。セッションがない場合、処理がステップＳ４２に進められる。セッションがある場合、処理がステップＳ４４に進められる。

【０１１８】

［ステップＳ４２］セッションがない場合、プロキシサーバ認証要求部１３３が、Ｗｅｂ

サーバ３１ａから受信したクッキーをＷｅｂブラウザ１１ａのプロキシ認証レスポンス（応答内容）に設定する。

【０１１９】

［ステップＳ４３］プロキシサーバ１００は、プロキシ認証レスポンスを生成し、Ｗｅｂブラウザ１１ａに返信する。その後、処理が終了する。

［ステップＳ４４］セッションがある場合、プロキシサーバ１００へのユーザ認証が完了しているため、プロキシサーバ１００は、セッション情報テーブル１２１からユーザ情報（ユーザ名等）を取得する。

【０１２０】

［ステップＳ４５］プロキシサーバ１００は、ユーザ情報テーブル１１１からＷｅｂサーバ３１ａにログインするためのユーザ認証情報を取得する。具体的には、プロキシサーバ１００は、ステップＳ４４で取得したユーザ名に対応付けて設定されているＷｅｂブラウザ３１ａ（システムＩＤで判別する）のユーザ名とパスワードとを取得する。

【０１２１】

［ステップＳ４６］プロキシサーバ１００は、ログインリクエストを生成する。

［ステップＳ４７］プロキシサーバ１００は、システムテーブル１１２ａを参照し、ベーシック認証か否かを判断する。ベーシック認証の場合、処理がステップＳ４８に進められる。ベーシック認証ではない場合、処理がステップＳ４９に進められる。

【０１２２】

［ステップＳ４８］プロキシサーバ１００は、保存してあるブラウザ要求を取り出し、ログインリクエストに組み合わせる。

［ステップＳ４９］プロキシサーバ１００は、Ｗｅｂサーバ３１ａへ要求（ログインリクエスト）を送信する。

【０１２３】

［ステップＳ５０］プロキシサーバ１００は、Ｗｅｂサーバ３１ａから応答された処理結果を受信する。

［ステップＳ５１］プロキシサーバ１００は、処理結果の内容を解析し、ログインに成功したか否かを判断する。ログインに成功した場合には、処理がステップＳ５４に進められる。ログインに失敗した場合には、処理がステップＳ５２に進められる。

【０１２４】

［ステップＳ５２］ログインに失敗した場合、プロキシサーバ１００は、エラーメッセージを生成する。

［ステップＳ５３］プロキシサーバ１００は、エラーメッセージを含む例外を発生させ、処理を終了する。

【０１２５】

［ステップＳ５４］プロキシサーバ１００は、成功したログイン処理がベーシック認証か否かを判断する。ベーシック認証の場合、処理がステップＳ３１（図１０参照）に進められる。ベーシック認証では無い場合、処理がステップＳ５５に進められる。

【０１２６】

［ステップＳ５５］プロキシサーバ１００は、通信状態保持部１３４から保存済のブラウザからの要求を取り出す。その後、処理がステップＳ３１（図１０参照）に進められる。

【０１２７】

次に、ステップＳ３５においてアクセスが許可されたと判断されたときの処理について説明する。

図１２は、代理認証処理の手順を示す第４のフローチャートである。以下、図１２に示す処理をステップ番号に沿って説明する。

【０１２８】

［ステップＳ６１］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答が書き換え対象か否かを、Ｗｅｂサーバ３１ａのＵＲＬから判断する。書き換え対象の場合、処理がステップＳ６２に進められる。書き換え対処ではない場合、処理がステップＳ６３に進め

られる。
【０１２９】
［ステップＳ６２］プロキシサーバ１００は、コンテンツの書き換えを行う。書き換え方法は、たとえば、ＵＲＬに対応付けて予め定義されている。
［ステップＳ６３］プロキシサーバ１００は、書き換えたコンテンツをブラウザレスポンスとして設定する。
【０１３０】
［ステップＳ６４］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからのレスポンスヘッダをブラウザレスポンスに設定する。
［ステップＳ６５］プロキシサーバ１００は、送信相手のＷｅｂブラウザに対応するセッションがあるか否かを判断する。セッションがあれば、処理がステップＳ６６に進められる。セッションがなければ、処理がステップＳ６７に進められる。
【０１３１】
［ステップＳ６６］プロキシサーバ１００は、プロキシのセッションＩＤを応答クッキーに埋め込む。
［ステップＳ６７］プロキシサーバ１００は、Ｗｅｂブラウザ１１ａに対して応答を送信する。
【０１３２】
［ステップＳ６８］Ｗｅｂブラウザ１１ａは、アクセスの結果を表示する。
以上のようにして、プロキシサーバ１００において代理認証が行われる。
次に、Ｗｅｂブラウザ１１ａによりプロキシサーバ１００にログインし、Ｗｅｂサーバ３１ａ等にアクセスする際の流れを具体的に説明する。以下の例では、Ｗｅｂサーバでベーシック認証を行う場合と、フォーム認証を行う場合とについて説明する。また、Ｗｅｂサーバ３１ａの名称を「ＷｅｂサーバＡ」、Ｗｅｂサーバ３２ａの名称を「ＷｅｂサーバＢ」とする。
【０１３３】
図１３は、Ｗｅｂサーバでベーシック認証を行う場合の処理手順を示す第１のシーケンス図である。図１３は、プロキシサーバ１００にログインしていないＷｅｂブラウザ１１ａからの要求に対して、プロキシ認証レスポンスを返すまでの処理である。以下、図１３に示す処理をステップ番号に沿って説明する。
【０１３４】
［ステップＳ７１］Ｗｅｂブラウザ１１ａは、プロキシサーバ１００に対する最初の要求を送信する。この例では、ＧＥＴ要求「GET(URL__X)」を送信している。
【０１３５】
［ステップＳ７２］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。最初の要求では、Authorizedヘッダは含まれていない。Authorizedヘッダが無い場合、認証不要のリクエストであると仮定して処理が進められる。
【０１３６】
［ステップＳ７３］プロキシサーバ１００は、Ｗｅｂサーバ３１ａに要求を送信する。この例では、ＧＥＴ要求「GET(URL__X)」を送信している。
［ステップＳ７４］Ｗｅｂサーバ３１ａは、プロキシサーバに対して応答「401(Unauthorized,WWW-Authenticate)」を返す。この例では、レスポンスコードが４０１の応答が返されている。この応答は、ベーシック認証が未完了であるためにアクセスを拒否したことを示している。
【０１３７】
［ステップＳ７５］プロキシサーバ１００は、レスポンスを解析する。
［ステップＳ７６］プロキシサーバ１００は、セッションの有無を確認する（ここで、プロキシサーバ１００とＷｅｂブラウザ１１ａとの間のセッションをＰＰＳセッションとする）。この段階では、Ｗｅｂブラウザ１１ａからプロキシサーバ１００にログインされていないため、セッションも存在していない。

【０１３８】
この例では、レスポンスコードが４０１であり（ログインを要求するレスポンス）、かつＰＰＳセッションが存在しない場合、プロキシサーバ１００へログイン処理が開始される。
【０１３９】
［ステップＳ７７］プロキシサーバ１００は、プロキシ認証レスポンス「Unauthorized,WWW-Authenticate」をＷｅｂブラウザ１１ａに送信する。
図１４は、Ｗｅｂサーバでベーシック認証を行う場合の処理手順を示す第２のシーケンス図である。図１４は、Ｗｅｂブラウザ１１ａからプロキシサーバ１００にログインし、Ｗｅｂページを取得するまでの処理である。以下、図１４に示す処理をステップ番号に沿って説明する。
【０１４０】
［ステップＳ８１］Ｗｅｂブラウザ１１ａは、ユーザからのユーザ名とパスワードとの入力を受け付ける。
［ステップＳ８２］Ｗｅｂブラウザ１１ａは、Authorizedヘッダにユーザ認証情報を含めた要求「GET(URL__X, Authorization/PPS)」を、プロキシサーバ１００に送信する。
【０１４１】
［ステップＳ８３］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。このとき、Authorizedヘッダありと判断される。
［ステップＳ８４］プロキシサーバ１００は、セッション（ＰＰＳセッション）の有無を確認する。この時点では、セッションは存在しない。Authorizedヘッダがあってもセッションが存在しない場合、未認証であると判断される。
【０１４２】
なお、ステップＳ８３とステップＳ８４との処理は、順番が逆であってもよい。図９に示すフローチャートでは、セッションの確認（ステップＳ１５）の後、Authorizedヘッダを確認（ステップＳ１６）している。
【０１４３】
［ステップＳ８５］プロキシサーバ１００は、ユーザ名とパスワードとの組を照合（マッチング）する。照合が成功した場合、以下の処理が行われる。
［ステップＳ８６］プロキシサーバ１００は、セッションを生成する。
【０１４４】
［ステップＳ８７］プロキシサーバ１００は、Ｗｅｂサーバ３１ａ(SubSystem)用のユーザ認証情報（認証キー）を取得する。
［ステップＳ８８］プロキシサーバ１００は、Ｗｅｂサーバ３１ａ用のAuthorizedヘッダを生成する。
【０１４５】
［ステップＳ８９］プロキシサーバ１００は、Ｗｅｂブラウザ１１ａから送られた要求からクッキー（PPSCookie)やプロキシサーバ１００へ認証するためのユーザ認証情報を取り除き、Ｗｅｂサーバ３１ａ用のユーザ認証情報を追加した要求「GET(URL__X,Authorization/Sub)」を、Ｗｅｂサーバ３１ａに送信する。
【０１４６】
［ステップＳ９０］Ｗｅｂサーバ３１ａは、ユーザ認証情報に基づいてユーザ認証を行う。Ｗｅｂサーバ３１ａは、認証に成功すると、指定されたＷｅｂページ（ＵＲＬ＿Ｘのページ）をプロキシサーバ１００に送信する。なお、この送信のパケットのレスポンスコードは「２００」である。
【０１４７】
［ステップＳ９１］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答（レスポンス）を解析する。
［ステップＳ９２］プロキシサーバ１００は、Ｗｅｂサーバ３１ａから送られたＷｅｂページにプロキシサーバ１００用のクッキー(PPSSCookie)を追加してＷｅｂブラウザ１１ａ

に送信する。このクッキーでは、プロキシサーバ１００のドメイン名と後方一致するＵＲＬへのアクセスの際に有効となるように設定されている。

【０１４８】

［ステップＳ９３］Ｗｅｂブラウザ１１ａは、受け取ったＷｅｂページを表示すると共に、プロキシサーバ１００から送られたクッキーを保存する。

以上のようにして、Ｗｅｂブラウザ１１ａを使用するユーザは、プロキシサーバ１００へログインするだけで、Ｗｅｂサーバ３１ａへのアクセスが可能となる。

【０１４９】

次に、引き続き同じＷｅｂサーバ３１ａにアクセスする際の処理手順を説明する。

図１５は、ベーシック認証を完了したＷｅｂサーバへのアクセス手順を示すシーケンス図である。以下、図１５に示す処理をステップ番号に沿って説明する。

【０１５０】

［ステップＳ１０１］Ｗｅｂブラウザ１１ａは、ユーザの操作入力に応答してＷｅｂページ(URL__Y)の取得要求「GET(URL__Y,PPSCookie,Authorization/PPS)」をプロキシサーバ１００に送信する。この要求には、ステップＳ９２（図１４参照）で保存したクッキーやプロキシサーバ１００へログインするためのユーザ認証情報が含まれる。

【０１５１】

［ステップＳ１０２］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。このとき、Authorizedヘッダありと判断される。

［ステップＳ１０３］プロキシサーバ１００は、セッション（ＰＰＳセッション）の有無を確認する。この例ではすでにセッションが存在するため、認証済と判断され、以下の処理が行われる。

【０１５２】

［ステップＳ１０４］プロキシサーバ１００は、Ｗｅｂサーバ３１ａ(SubSystem)用のユーザ認証情報（認証キー）を取得する。

［ステップＳ１０５］プロキシサーバ１００は、Ｗｅｂサーバ３１ａ用のAuthorizedヘッダを生成する。

【０１５３】

［ステップＳ１０６］プロキシサーバ１００は、Ｗｅｂブラウザ１１ａから送られた要求からクッキー（PPSCookie)やプロキシサーバ１００へ認証するためのユーザ認証情報を取り除き、Ｗｅｂサーバ３１ａ用のユーザ認証情報を追加した要求「GET(URL__Y,Authorization/a:Sub)」を生成する。そして、生成した要求をＷｅｂサーバ３１ａに送信する。

【０１５４】

［ステップＳ１０７］Ｗｅｂサーバ３１ａは、指定されたＷｅｂページ（ＵＲＬ＿Ｙのページ）をプロキシサーバ１００に送信する。

［ステップＳ１０８］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答（レスポンス）を解析する。アクセスが拒否されていないので、以下の処理に進める。

【０１５５】

［ステップＳ１０９］プロキシサーバ１００は、Ｗｅｂサーバ３１ａから送られたＷｅｂページにプロキシサーバ１００用のクッキー(PPSSCookie)を追加してＷｅｂブラウザ１１ａに送信する。

【０１５６】

次に、プロキシサーバ１００へのログイン完了後に、新たなＷｅｂサーバにアクセスする場合の例を説明する

図１６は、ベーシック認証を必要とする新たなＷｅｂサーバへのアクセス手順を示すシーケンス図である。以下、図１６に示す処理をステップ番号に沿って説明する。

【０１５７】

［ステップＳ１１１］Ｗｅｂブラウザ１１ａは、ユーザの操作入力に応答してＷｅｂページ(URL__x)の取得要求「GET(URL__x,PPSCookie)」をプロキシサーバ１００に送信する。この要求には、ステップＳ９２（図１４参照）で保存したクッキーが含まれる。なお、以

前と異なるＷｅｂサーバ３２ａへの要求なので、Authorizedヘッダは送られない。また、送信アドレスが仮想アドレスであるため、プロキシサーバ１００用のクッキーの適用範囲内であると判断され、そのクッキーが要求に含められている。

【０１５８】

［ステップＳ１１２］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。このとき、Authorizedヘッダなしと判断され、認証不要な要求と仮定して以降の処理が進められる。

【０１５９】

［ステップＳ１１３］プロキシサーバ１００は、Ｗｅｂサーバ３２ａに対して、要求を転送する。この要求では、プロキシサーバ１００用のクッキーは取り除かれている。

【０１６０】

［ステップＳ１１４］Ｗｅｂサーバ３２ａは、ユーザ認証を要求する応答をプロキシサーバ１００に返す。この応答のレスポンスコードは「４０１」である。

［ステップＳ１１５］プロキシサーバ１００は、応答を解析し、ユーザ認証が要求されていることを認識する。具体的には、レスポンスコードが「４０１」であることを検出し、ベーシック認証が要求されていることを認識する。

【０１６１】

［ステップＳ１１６］プロキシサーバ１００は、セッションの有無を確認する。この例では、既にセッションが存在するため、Ｗｅｂブラウザ１１ａへのユーザ認証用のダイアログ等の表示を行わず、以下のような自動認証処理を行う。

【０１６２】

［ステップＳ１１７］プロキシサーバ１００は、Ｗｅｂサーバ３２ａ用のユーザ認証情報を取得する。

［ステップＳ１１８］プロキシサーバ１００は、Ｗｅｂサーバ３２ａ用のAuthorizedヘッダを生成する。

【０１６３】

［ステップＳ１１９］プロキシサーバ１００は、Ｗｅｂブラウザ１１ａから送られた要求からクッキー（PPSCookie）やプロキシサーバ１００へ認証するためのユーザ認証情報を取り除き、Ｗｅｂサーバ３２ａ用のユーザ認証情報を追加した要求「GET(URL__x,Authorization/b:SUB)」を、Ｗｅｂサーバ３２ａに送信する。

【０１６４】

［ステップＳ１２０］Ｗｅｂサーバ３２ａは、ユーザ認証情報に基づいてユーザ認証を行う。Ｗｅｂサーバは、認証に成功すると、指定されたＷｅｂページ（URL__xのページ）をプロキシサーバ１００に送信する。

【０１６５】

［ステップＳ１２１］プロキシサーバ１００は、Ｗｅｂサーバからの応答（レスポンス）を解析する。アクセスが拒否されていないので、以下の処理に進める。

【０１６６】

［ステップＳ１２２］プロキシサーバ１００は、Ｗｅｂサーバ３２ａから送られたＷｅｂページをＷｅｂブラウザ１１ａに送信する。

このように、既にプロキシサーバ１００にログインしていれば、新たなＷｅｂサーバ３２ａにアクセスする場合であってもユーザ認証情報の入力は不要となる。

【０１６７】

次に、フォーム認証を行う際の手順を説明する。

図１７は、Ｗｅｂサーバでフォーム認証を行う場合の処理手順を示す第１のシーケンス図である。図１７は、プロキシサーバ１００にログインしていないＷｅｂブラウザ１１ａからの要求に対して、プロキシ認証レスポンスを返すまでの処理である。以下、図１７に示す処理をステップ番号に沿って説明する。

【０１６８】

［ステップＳ１３１］Ｗｅｂブラウザ１１ａは、プロキシサーバ１００に対する最初の要

求を送信する。この例では、ＧＥＴ要求「GET(URL__X)」を送信している。
【０１６９】
［ステップＳ１３２］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。最初の要求では、Authorizedヘッダは含まれていない。Authorizedヘッダが無い場合、認証不要のリクエストであると仮定して処理が進められる。
【０１７０】
［ステップＳ１３３］プロキシサーバ１００は、Ｗｅｂサーバ３１ａにする要求を送信する。この例では、ＧＥＴ要求「GET(URL__X)」を送信している。
［ステップＳ１３４］Ｗｅｂサーバ３１ａは、プロキシサーバに対して応答を返す。この例では、要求拒否の応答(NGレスポンス(SSSCookie))が返されている。要求拒否の応答には、クッキー(SSSCookie)が含まれる。この応答は、フォーム認証が未完了であるためにアクセスを拒否したことを示している。
【０１７１】
［ステップＳ１３５］プロキシサーバ１００は、レスポンスを解析する。
［ステップＳ１３６］プロキシサーバ１００は、セッションの有無を確認する（ここで、プロキシサーバ１００とＷｅｂブラウザ１１ａとの間のセッションをＰＰＳセッションとする）。この段階では、Ｗｅｂブラウザ１１ａからプロキシサーバ１００にログインされていないため、セッションも存在していない。
【０１７２】
この例では、要求拒否の応答が返され、かつＰＰＳセッションが存在しない場合、プロキシサーバ１００へログイン処理が開始される。
［ステップＳ１３７］プロキシサーバ１００は、プロキシ認証レスポンス「Unauthorized,WWW-Authenticate」をＷｅｂブラウザ１１ａに送信する。
【０１７３】
図１８は、Ｗｅｂサーバでフォーム認証を行う場合の処理手順を示す第２のシーケンス図である。図１８は、Ｗｅｂブラウザ１１ａからプロキシサーバ１００にログインし、Ｗｅｂページを取得するまでの処理である。以下、図１８に示す処理をステップ番号に沿って説明する。
【０１７４】
［ステップＳ１４１］Ｗｅｂブラウザ１１ａは、ユーザからのユーザ名とパスワードとの入力を受け付ける。
［ステップＳ１４２］Ｗｅｂブラウザ１１ａは、Authorizedヘッダにユーザ認証情報を含めた要求「GET(URL__X, Authorization/PPS)」を、プロキシサーバ１００に送信する。
【０１７５】
［ステップＳ１４３］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。このとき、Authorizedヘッダありと判断される。
［ステップＳ１４４］プロキシサーバ１００は、セッション（ＰＰＳセッション）の有無を確認する。この時点では、セッションは存在しない。Authorizedヘッダがあってもセッションが存在しない場合、未認証であると判断される。
【０１７６】
なお、ステップＳ１４３とステップＳ１４４との処理は、順番が逆であってもよい。図９に示すフローチャートでは、セッションの確認（ステップＳ１５）の後、Authorizedヘッダを確認（ステップＳ１６）している。
【０１７７】
［ステップＳ１４５］プロキシサーバ１００は、ユーザ名とパスワードとの組を照合（マッチング）する。照合が成功した場合、以下の処理が行われる。
［ステップＳ１４６］プロキシサーバ１００は、セッションを生成する。
【０１７８】
［ステップＳ１４７］プロキシサーバ１００は、Ｗｅｂサーバ３１ａ(SubSystem)用のユーザ認証情報（認証キー）を取得する。

［ステップＳ１４８］プロキシサーバ１００は、Ｗｅｂサーバ３１ａに対してログイン要求「POST(URL__Login, User,Pass, SSSCookie)」を送信する。このログイン要求には、ユーザ認証情報(User,Pass)やクッキー(SSSCookie)が含まれる。

【０１７９】

［ステップＳ１４９］Ｗｅｂサーバ３１ａは、ユーザ認証情報に基づいてユーザ認証を行う。Ｗｅｂサーバ３１ａは、認証に成功すると、認証後に送信すべきものとして予め設定されているＷｅｂページをプロキシサーバ１００に送信する。なお、この送信のパケットのレスポンスコードは「２００」である。

【０１８０】

［ステップＳ１５０］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答（レスポンス）を解析する。これにより、Ｗｅｂサーバ３１ａに対するログインが成功したことが認識される。

【０１８１】

「ステップＳ１５１］プロキシサーバ１００は、最初にＷｅｂブラウザ１１ａから送られた要求にＷｅｂサーバ３１ａ用のクッキー(SSSCookie)を追加して新たな要求「GET(URL__X, SSSCookie)」を生成し、Ｗｅｂサーバ３１ａに対して送信する。

【０１８２】

［ステップＳ１５２］Ｗｅｂサーバ３１ａは、クッキー(SSSCookie)により、既に認証しているユーザからのアクセスであると認識し、要求されたＷｅｂページをプロキシサーバ１００に送信する。

【０１８３】

［ステップＳ１５３］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答（レスポンス）を解析する。

［ステップＳ１５４］プロキシサーバ１００は、Ｗｅｂサーバ３１ａから送られたＷｅｂページにＷｅｂサーバ３１ａ用のクッキー(SSSCookie)とプロキシサーバ１００用のクッキー(PPSSCookie)とを追加してＷｅｂブラウザ１１ａに送信する。プロキシサーバ１００用のクッキーでは、プロキシサーバ１００のドメイン名と後方一致するＵＲＬへのアクセスの際に有効となるように設定されている。

【０１８４】

［ステップＳ１５５］Ｗｅｂブラウザ１１ａは、受け取ったＷｅｂページを表示すると共に、プロキシサーバ１００から送られたクッキーを保存する。

以上のようにして、Ｗｅｂブラウザ１１ａを使用するユーザは、プロキシサーバ１００へログインするだけで、Ｗｅｂサーバ３１ａへのアクセスが可能となる。

【０１８５】

次に、引き続き同じＷｅｂサーバ３１ａにアクセスする際の処理手順を説明する。

図１９は、フォーム認証を完了したＷｅｂサーバへのアクセス手順を示すシーケンス図である。以下、図１９に示す処理をステップ番号に沿って説明する。

【０１８６】

［ステップＳ１６１］Ｗｅｂブラウザ１１ａは、ユーザの操作入力に応答してＷｅｂページ(URL__Y)の取得要求「GET(URL__Y,PPSCookie,Authorization/PPS)」をプロキシサーバ１００に送信する。この要求には、ステップＳ１５５（図１８参照）で保存したクッキーやプロキシサーバ１００へログインするためのユーザ認証情報が含まれる。

【０１８７】

［ステップＳ１６２］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。このとき、Authorizedヘッダありと判断される。

［ステップＳ１６３］プロキシサーバ１００は、セッション（ＰＰＳセッション）の有無を確認する。この例ではすでにセッションが存在するため、認証済と判断され、以下の処理が行われる。

【０１８８】

［ステップＳ１６４］プロキシサーバ１００は、Ｗｅｂブラウザ１１ａから送られた要求

にＷｅｂサーバ３１ａ用のユーザ認証情報を追加した要求「GET(URL__Y,SSSCookie)」を生成し、Ｗｅｂサーバ３１ａに送信する。

【０１８９】

［ステップＳ１６５］Ｗｅｂサーバ３１ａは、指定されたＷｅｂページ（ＵＲＬ＿Ｙのページ）をプロキシサーバ１００に送信する。

［ステップＳ１６６］プロキシサーバ１００は、Ｗｅｂサーバ３１ａからの応答（レスポンス）を解析する。アクセスが拒否されていないので、以下の処理に進める。

【０１９０】

［ステップＳ１６７］プロキシサーバ１００は、Ｗｅｂサーバ３１ａから送られたＷｅｂページをＷｅｂブラウザ１１ａに送信する。

次に、プロキシサーバ１００へのログイン完了後に、新たなＷｅｂサーバにアクセスする場合の例を説明する。

【０１９１】

図２０は、フォーム認証を必要とする新たなＷｅｂサーバへのアクセス手順を示すシーケンス図である。以下、図２０に示す処理をステップ番号に沿って説明する。

【０１９２】

［ステップＳ１７１］Ｗｅｂブラウザ１１ａは、ユーザの操作入力に応答してＷｅｂページ(URL__x)の取得要求「GET(URL__x,PPSSCookie)」をプロキシサーバ１００に送信する。この要求には、ステップＳ１５５（図１８参照）で保存したクッキーが含まれる。なお、以前と異なるＷｅｂサーバ３２ａへの要求なので、Authorizedヘッダは送られない。また、送信アドレスが仮想アドレスであるため、プロキシサーバ１００用のクッキーの適用範囲内であると判断され、そのクッキーが要求に含められている。

【０１９３】

［ステップＳ１７２］プロキシサーバ１００では、Authorization（Authorizedヘッダ）の確認が行われる。このとき、Authorizedヘッダなしと判断され、認証不要な要求と仮定して以降の処理が進められる。

【０１９４】

［ステップＳ１７３］プロキシサーバ１００は、Ｗｅｂサーバ３２ａに対して、要求を転送する。この要求では、プロキシサーバ１００用のクッキーは取り除かれている。

【０１９５】

［ステップＳ１７４］Ｗｅｂサーバ３２ａは、ユーザ認証を要求する応答をプロキシサーバ１００に返す。この例では、要求拒否の応答(NGレスポンス(SSSCookie))が返されている。要求拒否の応答には、クッキー(SSSCookie)が含まれる。この応答は、フォーム認証が未完了であるためにアクセスを拒否したことを示している。

【０１９６】

［ステップＳ１７５］プロキシサーバ１００は、応答を解析し、ユーザ認証が要求されていることを認識する。

［ステップＳ１７６］プロキシサーバ１００は、セッションの有無を確認する。この例では、既にセッションが存在するため、Ｗｅｂブラウザ１１ａへのユーザ認証用のダイアログ等の表示を行わず、以下のような自動認証処理を行う。

【０１９７】

［ステップＳ１７７］プロキシサーバ１００は、Ｗｅｂサーバ３２ａ(SubSystem)用のユーザ認証情報（認証キー）を取得する。

［ステップＳ１７８］プロキシサーバ１００は、Ｗｅｂサーバ３２ａに対してログイン要求「POST(URL__Login,SSSCookie)」を送信する。このログイン要求には、クッキー(SSSCookie)が含まれる。

【０１９８】

［ステップＳ１７９］Ｗｅｂサーバ３２ａは、ユーザ認証情報に基づいてユーザ認証を行う。Ｗｅｂサーバ３２ａは、認証に成功すると、認証後に送信すべきものとして予め設定されているＷｅｂページをプロキシサーバ１００に送信する。なお、この送信のパケット

のレスポンスコードは「２００」である。
【０１９９】
［ステップＳ１８０］プロキシサーバ１００は、Ｗｅｂサーバ３２ａからの応答（レスポンス）を解析する。これにより、Ｗｅｂサーバ３２ａに対するログインが成功したことが認識される。
【０２００】
［ステップＳ１８１］プロキシサーバ１００は、最初にＷｅｂブラウザ１１ａから送られた要求にＷｅｂサーバ３２ａ用のクッキー(SSSCookie)を追加して新たな要求「GET(URL_x,SSSCookie)」を生成し、Ｗｅｂサーバ３２ａに対して送信する。
【０２０１】
［ステップＳ１８２］Ｗｅｂサーバ３２ａは、クッキー(SSSCookie)により、既に認証しているユーザからのアクセスであると認識し、要求されたＷｅｂページをプロキシサーバ１００に送信する。
【０２０２】
［ステップＳ１８３］プロキシサーバ１００は、Ｗｅｂサーバ３２ａからの応答（レスポンス）を解析する。
［ステップＳ１８４］プロキシサーバ１００は、Ｗｅｂサーバ３２ａから送られたＷｅｂページにＷｅｂサーバ３２ａ用のクッキー(SSSCookie)を追加してＷｅｂブラウザ１１ａに送信する。
【０２０３】
［ステップＳ１８５］Ｗｅｂブラウザ１１ａは、受け取ったＷｅｂページを表示すると共に、プロキシサーバ１００から送られたクッキーを保存する。
以上のように、ベーシック認証、フォーム認証等の認証方式に合わせて、代理認証を行うことができる。しかも、認証手順は予め定義されており、各Ｗｅｂサーバに対するユーザ毎のユーザ認証情報を保持しておくだけでよい。
【０２０４】
なお、ベーシック認証の認証手順とフォーム認証の認証手順とは、上記の手順で表された相違点以外に、送信データの生成方法も異なる。そこで、以下に、ログイン処理命令生成部１４０による認証方式毎の送信データ生成方法について説明する。
【０２０５】
図２１は、ベーシック認証時の送信データ生成手順を示すフローチャートである。以下、図２１に示す処理をステップ番号に沿って説明する。
［ステップＳ２０１］ログイン処理命令生成部１４０は、ユーザ名とパスワードとの文字列の間に‘：’をいれて、それぞれを連結する。
【０２０６】
［ステップＳ２０２］ログイン処理命令生成部１４０は、連結した文字列を所定のアルゴリズムでエンコードする。たとえば、Base64と呼ばれるアルゴリズムでエンコードする。
【０２０７】
［ステップＳ２０３］ログイン処理命令生成部１４０は、エンコードされた認証文字列の前に“Basic”という文字列を挿入する。
［ステップＳ２０４］ログイン処理命令生成部１４０は、生成した文字列をヘッダ名「AUTHORIZATION」と連結し、認証ヘッダを生成する。
【０２０８】
［ステップＳ２０５］ログイン処理命令生成部１４０は、通信状態保持部１３４からＵＲＬ、ヘッダ、ボディ情報を引き出す。
［ステップＳ２０６］ログイン処理命令生成部１４０は、ヘッダ部に認証ヘッダを挿入する。
【０２０９】
［ステップＳ２０７］ログイン処理命令生成部１４０は、Ｗｅｂサーバにヘッダを送信する。

［ステップＳ２０８］ログイン処理命令生成部１４０は、Ｗｅｂサーバにボディ情報を送信する。

【０２１０】

図２２は、フォーム認証時の送信データ生成手順を示すフローチャートである。以下、図２２に示す処理をステップ番号に沿って説明する。

［ステップＳ２１１］ログイン処理命令生成部１４０は、ユーザフォーム名とユーザ名とをフォームデータ生成情報保持テーブル１４０ａに登録する。

【０２１１】

ここで、フォームデータ生成情報保持テーブル１４０ａは、ログイン処理命令生成部１４０が保持するデータテーブルであり、フォーム名（ユーザフォーム名やパスワードフォーム名）と登録情報（ユーザ名やパスワード文字列）との組を登録することができる。

【０２１２】

［ステップＳ２１２］ログイン処理命令生成部１４０は、パスワードフォーム名とパスワード文字列との組を、フォームデータ生成情報保持テーブル１４０ａに登録する。

【０２１３】

［ステップＳ２１３］ログイン処理命令生成部１４０は、ステップＳ２１１とステップＳ２１２とで登録した各々の文字列を、ＵＲＬエンコードルールに従って変換する。

【０２１４】

［ステップＳ２１４］ログイン処理命令生成部１４０は、変換した文字列の間に‘＆’を挿入して連結する。

［ステップＳ２１５］ログイン処理命令生成部１４０は、通信状態保持部１３４からクッキー情報を取得する。

【０２１５】

［ステップＳ２１６］ログイン処理命令生成部１４０は、Ｗｅｂサーバに“Content-Type:application/x-www-form-urlencoded”ヘッダを送信する。

［ステップＳ２１７］ログイン処理命令生成部１４０は、Ｗｅｂサーバにクッキーヘッダを送信する。

【０２１６】

［ステップＳ２１８］ログイン処理命令生成部１４０は、Ｗｅｂサーバに認証フォームデータを送信する。

以上説明したように、プロキシサーバ１００が代理認証を行うことで、ユーザは、Ｗｅｂサーバ毎にログイン手続きを行う必要が無くなり、利便性が向上する。

【０２１７】

なお、上記の例では、プロキシサーバが代理認証を行っているが、この機能をクライアントに実装することもできる。

図２３は、代理認証機能を実装したクライアントの概念図である。図２３に示すように、クライアント２００にはブラウザプログラム２１０とプロキシプログラム２２０とが実装される。ブラウザプログラム２１０は、Ｗｅｂブラウザとしてクライアント２００を機能させるためのプログラムである。また、プロキシプログラム２２０は、プロキシサーバ１００の機能をクライアント２００で実現するためのプログラムである。プロキシプログラム２２０は、ユーザ情報テーブル２２１を有している。ユーザ情報テーブル２２１、各サーバ３１，３２，３３に実装されたＷｅｂサーバにログインするためのユーザ認証情報（ユーザ名とパスワード）が登録されている。

【０２１８】

なお、プロキシプログラム２２０は、クライアント２００のユーザの代理認証を行えばよい。そのため、クライアント２００のＯＳへ正しくログインしたユーザに対しては、プロキシプログラム２２０においてユーザ認証を行わなくてもよい。

【０２１９】

以上説明したように、本発明の実施の形態によれば、複数の認証を要するＷｅｂシステムを利用する際に、ユーザがログインを行う数が１回ですむ。

しかも、従来のシングルサインオンシステムとは異なり、認証を要しないＷｅｂページを使用する場合には、ユーザのログイン操作は０回になる。すなわち、従来のシングルサインオン技術では、予め代理認証システムにログインすることが前提となり、かならず１回のユーザ認証処理が発生する。ところが、本実施の形態によれば、Ｗｅｂサーバでのユーザ認証の要否をプロキシサーバが判断し、ユーザ認証を要求しないＷｅｂサーバへのアクセスで有ればプロキシサーバへのログインを要求しない。その結果、ユーザの操作性が向上する。

【０２２０】

また、本実施の形態では、ＷｅｂサーバのＵＲＬ毎にユーザ認証情報を管理しているため、同一サーバコンピュータ内に複数のＷｅｂサーバが実装されていたときでも、Ｗｅｂサーバ毎の代理認証が可能である。

【０２２１】

また、従来の代理認証技術には、認証通信を完全に記録しそのまま再現するタイプのものがある。この技術では、冗長な記憶領域を必要とした。一方、本実施の形態の方式では認証用の通信データをプログラムの計算処理によって生成するため、ユーザが増加した場合も記憶容量の増加はごく僅かである。

【０２２２】

また、本実施の形態ではＷｅｂサーバ側でセッションのタイムアウト発生によりそのセッションが停止しても、次回の該当するＷｅｂサーバへのアクセスの際に自動的にログインされセッションが再度確立することができる。そのため、Ｗｅｂサーバでのセッションのタイムアウトをユーザが意識せずにすむ。

【０２２３】

また、従来のシングルサインオンでは、シングルサインオンを実現するためのサーバが必要であり個人での運用・利用は困難だった。本実施の形態では、プロキシサーバの機能を用意にクライアントに実装することができ（図２３参照）、クライアント上でのシングルサインオンが実現可能となる。

【０２２４】

なお、既存のＷｅｂブラウザにはユーザの入力したユーザ名やパスワードを記憶し、パスワード入力の手間を削減する機構が存在する。しかし、この機能は個々のブラウザ毎に異なり、複数のブラウザを利用するユーザはユーザ名やパスワードのデータを共用できない。図２３に示すような単一ユーザ形態であれば、複数のブラウザで統一されたシングルサインオンを実現できる。

【０２２５】

ところで、上記の例では、Ｗｅｂサーバにアクセスする場合の例を用いて説明しているが、ネットワーク経由のユーザ認証を要求する他のアプリケーションに対しても同様の処理で代理認証を行うことができる。

【０２２６】

また、上記の例では、Ｗｅｂサーバがベーシック認証またはフォーム認証を要求する場合の例を示したが、他の認証方式に適用することもできる。その場合、適用すべき認証方式の判別基準をレスポンス分析・ログイン命令テーブル１１２ｂに設定するとともに、該当する認証方式への認証手順を定義しておく。ログイン処理命令生成部１４０は、定義された手順で代理認証の処理を実行する。

【０２２７】

なお、上記のプロキシサーバ１００や代理認証装置１の処理機能は、サーバコンピュータに所定の代理認証プログラムを実行させることにより実現することができる。その場合、プロキシサーバ１００等が有すべき機能の処理内容を記述した代理認証プログラムが提供される。サーバコンピュータは、クライアントコンピュータからの要求に応答して、代理認証プログラムを実行する。これにより、上記処理機能がサーバコンピュータ上で実現され、処理結果がクライアントコンピュータに提供される。

【０２２８】

処理内容を記述した代理認証プログラムは、サーバコンピュータで読み取り可能な記録媒体に記録しておくことができる。サーバコンピュータで読み取り可能な記録媒体としては、磁気記録装置、光ディスク、光磁気記録媒体、半導体メモリなどがある。磁気記録装置には、ハードディスク装置（ＨＤＤ）、フレキシブルディスク（ＦＤ）、磁気テープなどがある。光ディスクには、ＤＶＤ(Digital Versatile Disc)、ＤＶＤ－ＲＡＭ(Random Access Memory)、ＣＤ－ＲＯＭ(Compact Disc Read Only Memory)、ＣＤ－Ｒ(Recordable)／ＲＷ(ReWritable)などがある。光磁気記録媒体には、ＭＯ(Magneto-Optical disc)などがある。

【０２２９】

代理認証プログラムを流通させる場合には、たとえば、その代理認証プログラムが記録されたＤＶＤ、ＣＤ－ＲＯＭなどの可搬型記録媒体が販売される。

代理認証プログラムを実行するサーバコンピュータは、たとえば、可搬型記録媒体に記録された代理認証プログラムを、自己の記憶装置に格納する。そして、サーバコンピュータは、自己の記憶装置から代理認証プログラムを読み取り、代理認証プログラムに従った処理を実行する。なお、サーバコンピュータは、可搬型記録媒体から直接代理認証プログラムを読み取り、その代理認証プログラムに従った処理を実行することもできる。

【０２３０】

（付記１）　他の装置への認証手続きを代理する代理認証プログラムにおいて、

コンピュータに、

クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスし、

ユーザ認証が未完了であることを示す応答を識別するための複数の認証許否確認論理が予め定義されており、前記サーバ機能からの応答が前記複数の認証許否確認論理のいずれかに適合するか否かによりユーザ認証の要否を判断し、

ユーザ認証が必要な場合、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う、

処理を実行させることを特徴とする代理認証プログラム。

【０２３１】

（付記２）　前記ユーザ認証手続きでは、予め前記クライアント機能に対応付けて格納されている前記サーバ機能へのユーザ認証情報を、前記クライアント機能からの前記要求に付加して前記サーバ機能へ送信することを特徴とする付記１記載の代理認証プログラム。

【０２３２】

（付記３）　前記クライアント機能からの前記要求を記憶し、前記サーバ機能との間の前記ユーザ認証手続き完了後、記憶しておいた前記要求を前記サーバ機能に送信することを特徴とする付記１記載の代理認証プログラム。

【０２３３】

（付記４）　ユーザ認証が未完了であることを示す前記応答を受け取った際に、前記クライアント機能との間のセッションの確立の有無を確認し、

前記セッションが未確立の場合、前記クライアント機能に対してユーザ認証を要求し、

ユーザ認証が完了すると前記クライアント機能との間にセッションを確立し、前記サーバ機能との間の前記ユーザ認証手続きを行う、

ことを特徴とする付記１記載の代理認証プログラム。

【０２３４】

（付記５）　前記クライアント機能からアクセス可能な前記サーバ機能が複数有る場合、前記サーバ機能毎の認証許否確認論理が設定されていることを特徴とする付記１記載の代理認証プログラム。

【０２３５】

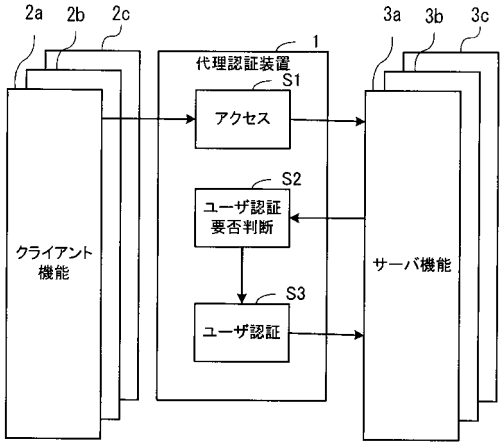
（付記６）　前記認証許否確認論理では、前記サーバ機能が実装されている装置の識別情報と前記装置内での前記サーバ機能の所在を示す情報との組み合わせにより前記サーバ機能が特定されていることを特徴とする付記１記載の代理認証プログラム。

【０２３６】
（付記７）　他の装置への認証手続きを代理するための代理認証方法において、
クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスし、
ユーザ認証が未完了であることを示す応答を識別するための複数の認証許否確認論理が予め定義されており、前記サーバ機能からの応答が前記複数の認証許否確認論理のいずれかに適合するか否かによりユーザ認証の要否を判断し、
ユーザ認証が必要な場合、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う、
ことを特徴とする代理認証方法。
【０２３７】
（付記８）　他の装置への認証手続きを代理する代理認証装置において、
クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスするアクセス手段と、
ユーザ認証が未完了であることを示す応答を識別するための複数の認証許否確認論理が予め定義されており、前記サーバ機能からの応答が前記複数の認証許否確認論理のいずれかに適合するか否かによりユーザ認証の要否を判断する認証要否判断手段と、
ユーザ認証が必要な場合、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う代理認証手段と、
を有することを特徴とする代理認証装置。
【０２３８】
（付記９）　他の装置への認証手続きを代理する代理認証プログラムを記録したコンピュータ読み取り可能な記録媒体において、
前記コンピュータに、
クライアント機能からの要求に応じて、ネットワークを介して接続されるサーバ機能に対してアクセスし、
ユーザ認証が未完了であることを示す応答を識別するための複数の認証許否確認論理が予め定義されており、前記サーバ機能からの応答が前記複数の認証許否確認論理のいずれかに適合するか否かによりユーザ認証の要否を判断し、
ユーザ認証が必要な場合、前記応答が適合した前記認証許否確認論理に対応付けて予め定義されている認証手順に従って、前記サーバ機能との間でユーザ認証手続きを行う、
処理を実行させることを特徴とする代理認証プログラムを記録したコンピュータ読み取り可能な記録媒体。
【０２３９】
【発明の効果】
以上説明したように本発明では、複数の認証許否確認論理を予め定義しておき、サーバ機能からの応答に応じた認証許否確認論理を判断し、その認証許否確認論理に対応付けられた認証手順でサーバ機能に対するユーザ認証手続きを行うようにした。このように認証許否確認論理に応じた認証手順に基づいてユーザ認証が行われるため、複数のクライアントの代理認証を行う場合であっても、クライアント毎の認証手順を記憶する必要が無い。その結果、少ないハードウェア資源の利用で代理認証機能が実現可能となる。
【図面の簡単な説明】
【図１】実施の形態に適用される発明の概念図である。
【図２】本実施の形態を提供するネットワークシステムの構成例を示す図である。
【図３】プロキシサーバのハードウェア構成例を示す図である。
【図４】プロキシサーバの機能を示すブロック図である。
【図５】ユーザ情報テーブルのデータ構造例を示す図である。
【図６】システムテーブルのデータ構造例を示す図である。
【図７】レスポンス分析・ログイン命令テーブルのデータ構造例を示す図である。

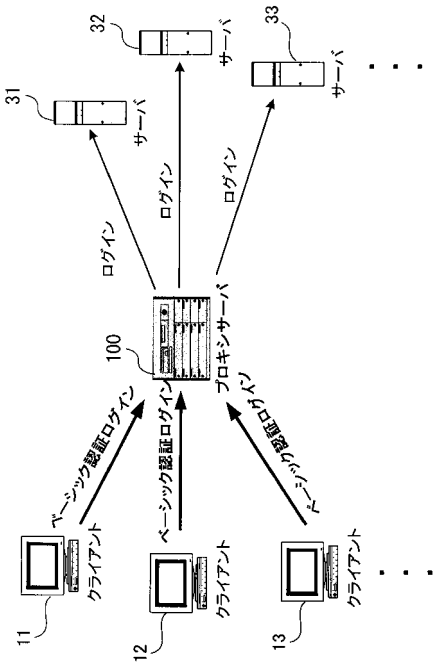
【図８】セッション情報テーブルのデータ構造例を示す図である。
【図９】代理認証処理の手順を示す第１のフローチャートである。
【図１０】代理認証処理の手順を示す第２のフローチャートである。
【図１１】代理認証処理の手順を示す第３のフローチャートである。
【図１２】代理認証処理の手順を示す第４のフローチャートである。
【図１３】Ｗｅｂサーバでベーシック認証を行う場合の処理手順を示す第１のシーケンス図である。
【図１４】Ｗｅｂサーバでベーシック認証を行う場合の処理手順を示す第２のシーケンス図である。
【図１５】ベーシック認証を完了したＷｅｂサーバへのアクセス手順を示すシーケンス図である。
【図１６】ベーシック認証を必要とする新たなＷｅｂサーバへのアクセス手順を示すシーケンス図である。
【図１７】Ｗｅｂサーバでフォーム認証を行う場合の処理手順を示す第１のシーケンス図である。
【図１８】Ｗｅｂサーバでフォーム認証を行う場合の処理手順を示す第２のシーケンス図である。
【図１９】フォーム認証を完了したＷｅｂサーバへのアクセス手順を示すシーケンス図である。
【図２０】フォーム認証を必要とする新たなＷｅｂサーバへのアクセス手順を示すシーケンス図である。
【図２１】ベーシック認証時の送信データ生成手順を示すフローチャートである。
【図２２】フォーム認証時の送信データ生成手順を示すフローチャートである。
【図２３】代理認証機能を実装したクライアントの概念図である。
【符号の説明】
１　代理認証装置
２　クライアント
３　サーバ
１１，１２，１３，・・・　クライアント
３１，３２，３３，・・・　サーバ
１００　プロキシサーバ
１１０　データベース
１１１　ユーザ情報テーブル
１１２　Ｗｅｂサーバ情報
１１２ａ　システムテーブル
１１２ｂ　レスポンス分析・ログイン命令テーブル
１２１　セッション情報テーブル
１２２　タイマ
１２３　タイムアウト駆動部
１３１　ブラウザ入力受付部
１３２　プロキシサーバ認証部
１３３　プロキシサーバ認証要求部
１３４　通信状態保持部
１３５　Ｗｅｂサーバ識別部
１３６　Ｗｅｂサーバ要求生成部
１３７　Ｗｅｂサーバ通信部
１３８　レスポンス分析部
１３９　認証処理判定部
１４０　ログイン処理命令生成部
１４１　コンテンツ変換部

１４２　ブラウザ応答生成部

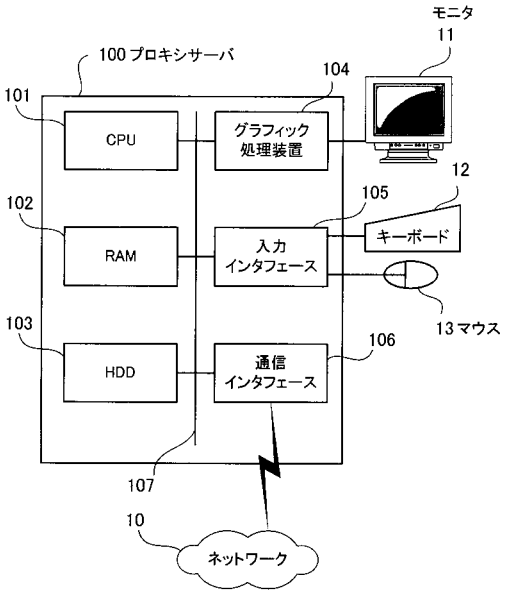
【図１】



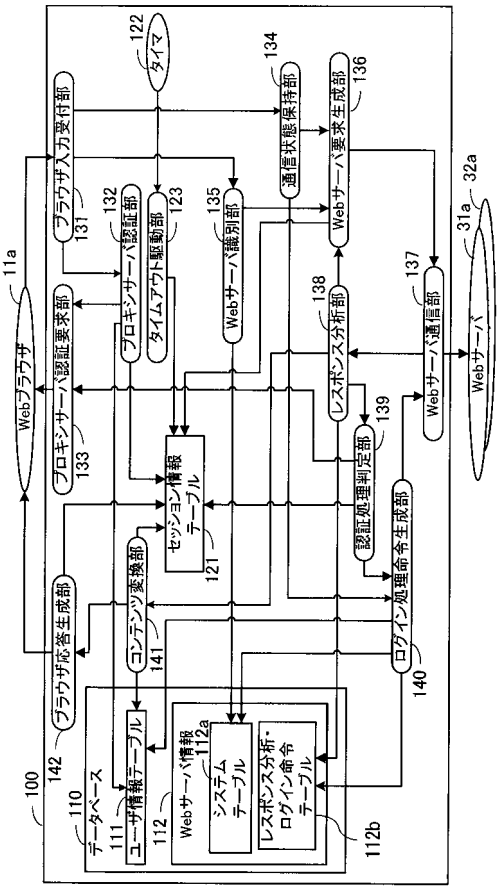
【図２】



【図３】



【図４】



【図５】

111 ユーザ情報テーブル

プロキシサーバ認証情報		システムID	ユーザ名	パスワード	システムID	ユーザ名	パスワード	システムID	ユーザ名	パスワード
ユーザ名	パスワード									
User1	Pass1	SysA0	uu1a0	pp1a0	SysA1	uu1a1	pp1a1	SysB0	uu1b0	pp1b0
User2	Pass2	-	-	-	SysA1	uu2a1	pp2a1	SysB0	uu2b0	pp2b0
User3	Pass3	SysA0	uu3a0	pp3a0	SysA1	uu3a1	pp3a1	SysB0	uu3b0	pp3b0
User4	Pass4	SysA0	uu4a0	pp4a0	SysA1	uu4a1	pp4a1	SysB0	uu4b0	pp4b0

システムID	仮想URL	実URL	認証タイプ	システムID	ユーザ名	パスワード
SysC0	http://CCC/c0/	http://ccc/c3/	ベーシック	SysC0	uu3c0	pp3c0
SysD0	http://dddd/d0/	http://dddd/d0/	null	-	-	-

【図６】

112a システムテーブル

システムID	仮想URL	実URL	認証タイプ
SysA0	http://aaa/a0/	http://aaa/a0/	フォーム
SysB0	http://bbb/b0/	http://bbb/b0/	ベーシック
SysA1	http://aaa/a1/	http://aaa/a1/	フォーム
SysC0	http://CCC/c0/	http://ccc/c3/	ベーシック
SysD0	http://dddd/d0/	http://dddd/d0/	null

【図７】

112b レスポンス分析・ログイン命令テーブル

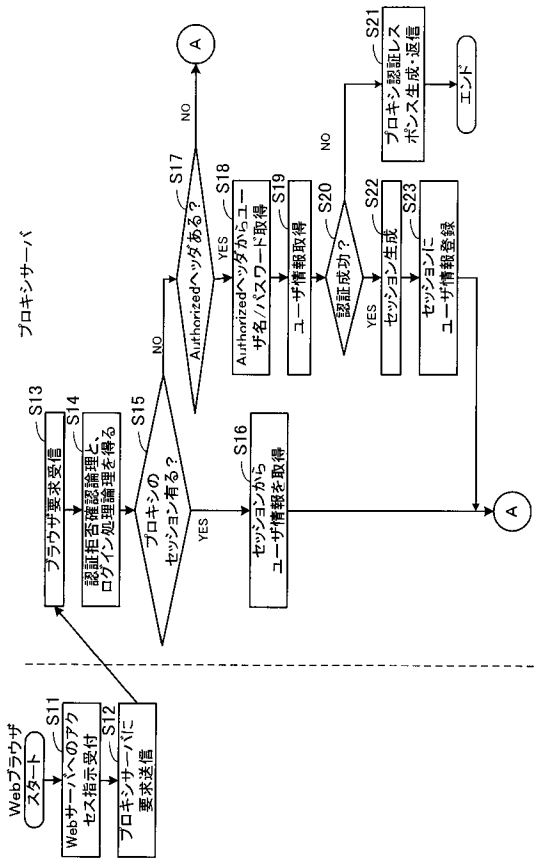
システムID	ログイン処理情報						認証識別情報		
	認証URL	メソッド	フォーム名				レスポンスコード	識別ヘッダ(n個)	識別内容データ
			ユーザ	パスワード	付加データ(m個)				
SysA0	http://aaa/a0/login	GET	user	password			305	Location,xxxxxx	-
SysA1	http://aaa/a1/login	POST	USR	PASS	id=x,xxx=yyyy,...		200	-	"Unauthorized..."

【図８】

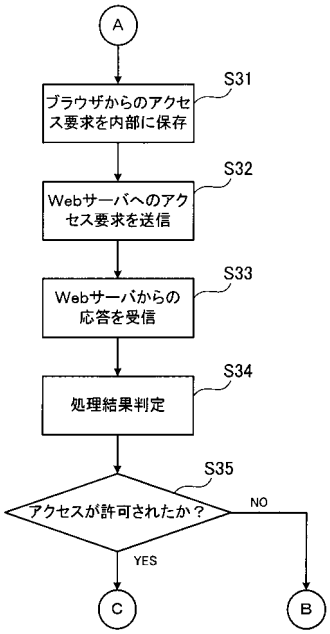
121 セッション情報テーブル

セションID	ユーザ名	ログイン時刻	タイムアウト予告時刻
2313403	User1	2002/10/1 20:22	2002/10/1 22:22
0352334	User3	2002/10/1 21:00	2002/10/1 21:30

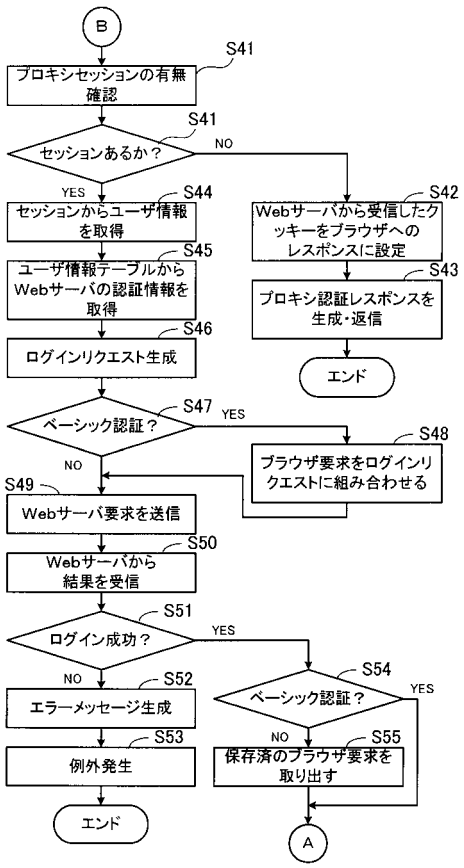
【図９】



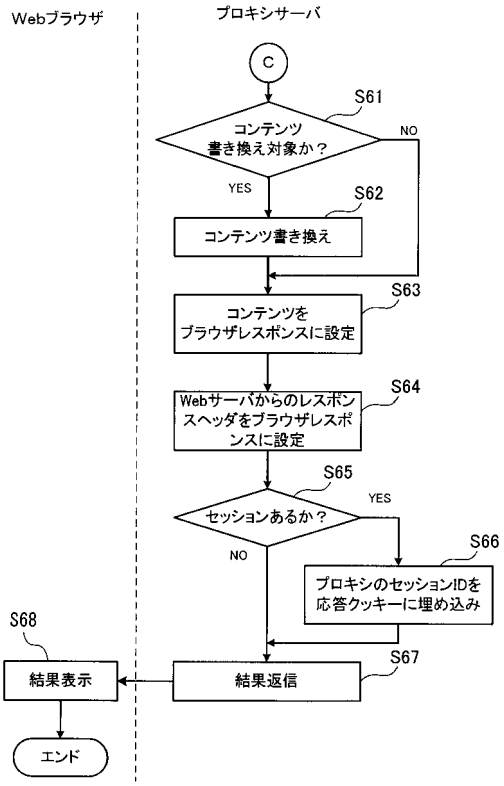
【図１０】



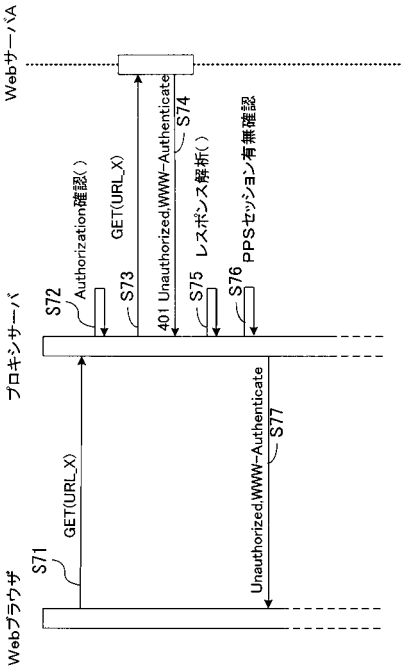
【図１１】



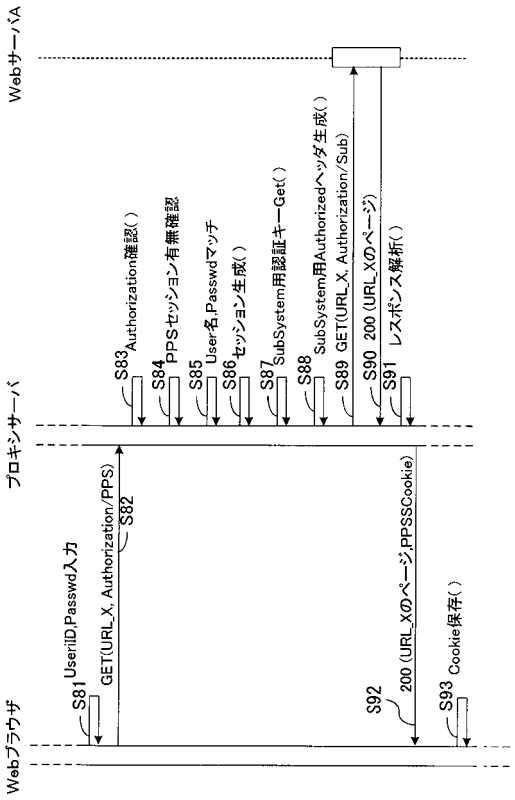
【図１２】



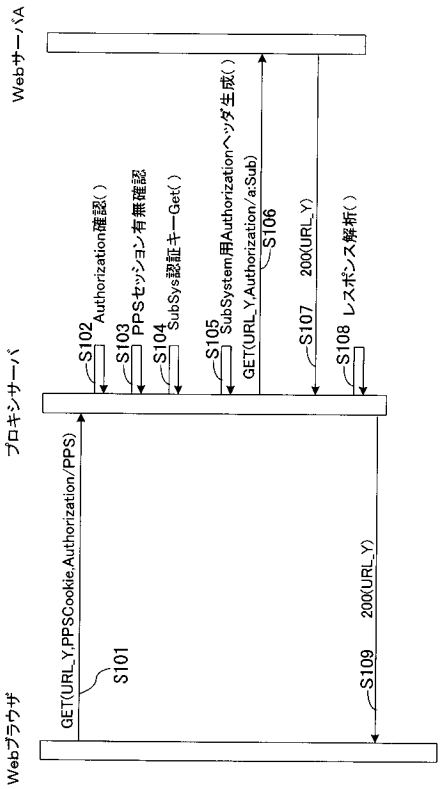
【図１３】



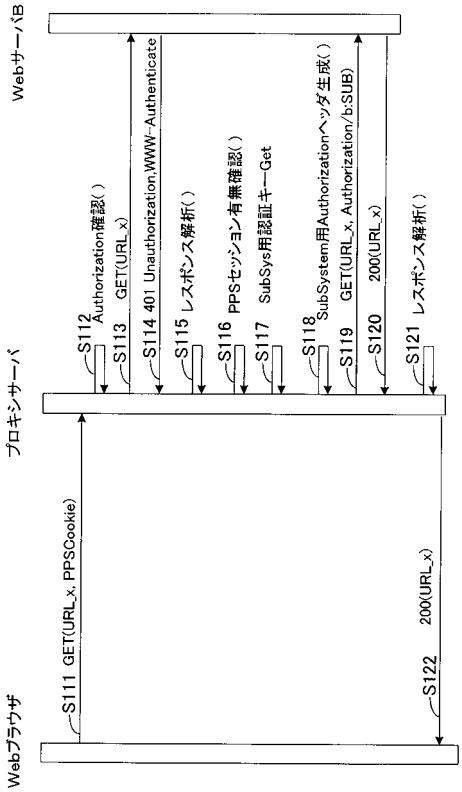
【図１４】



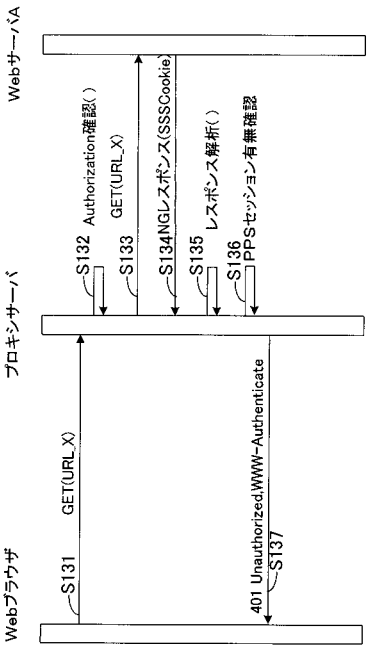
【図１５】



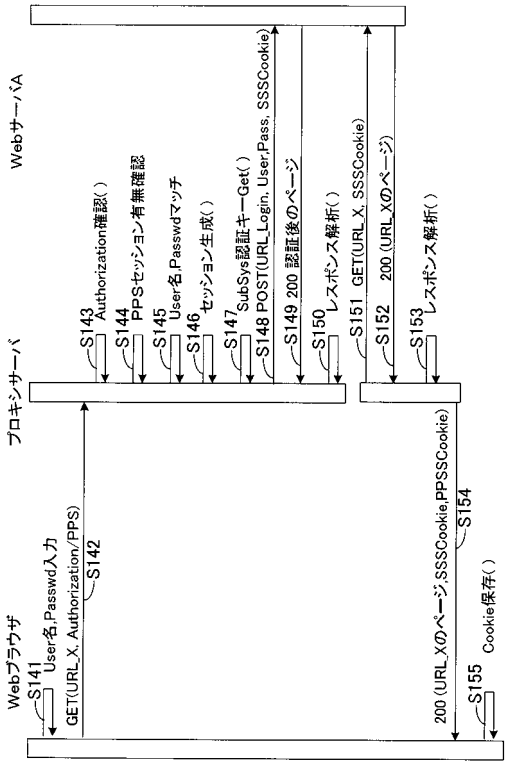
【図１６】



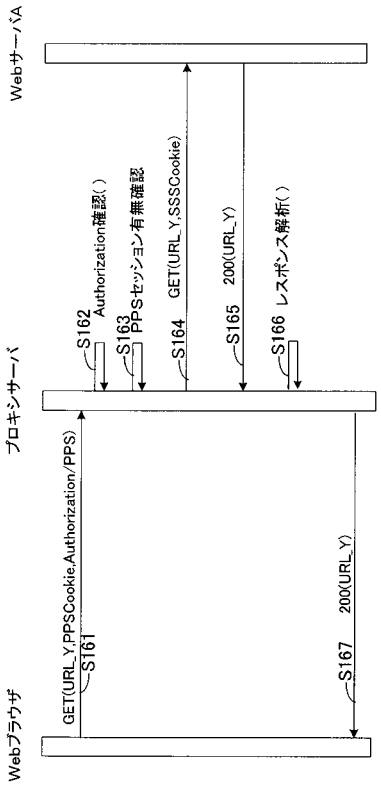
【図１７】



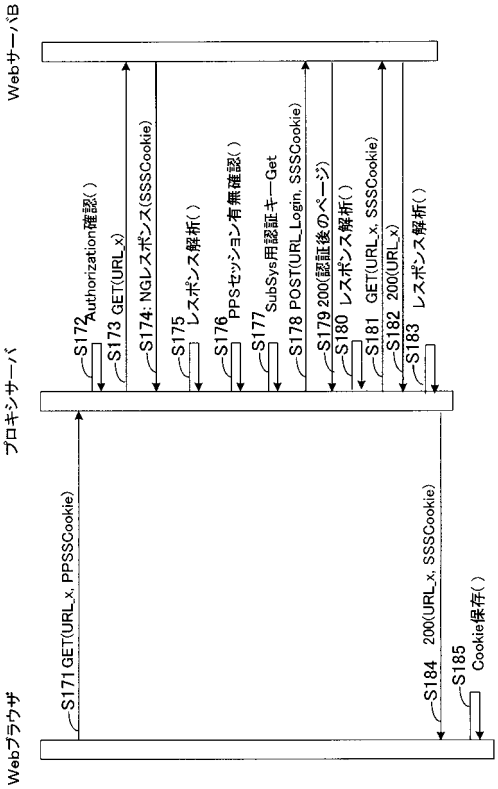
【図１８】



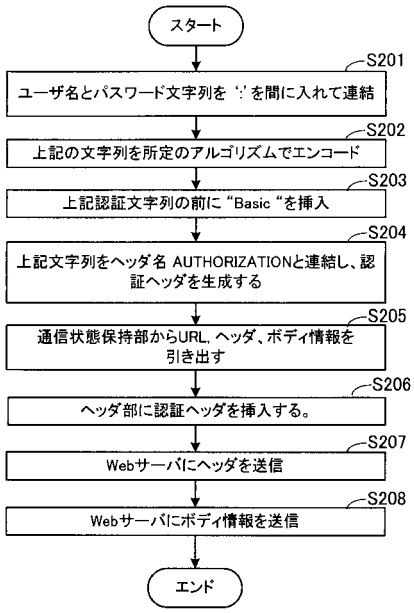
【図１９】



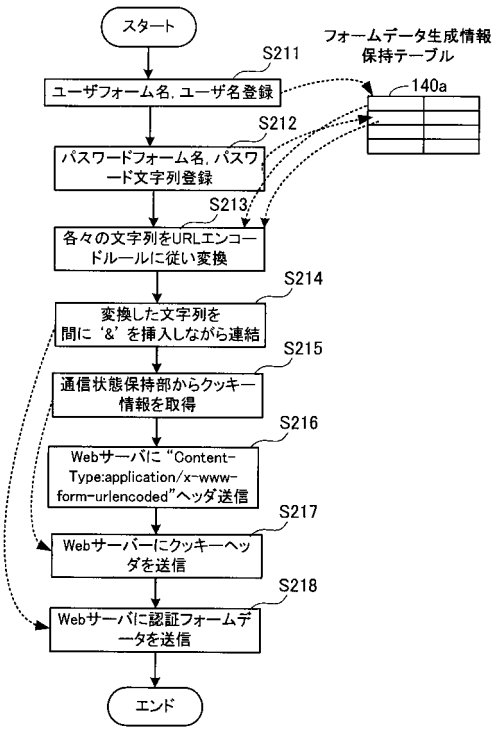
【図２０】



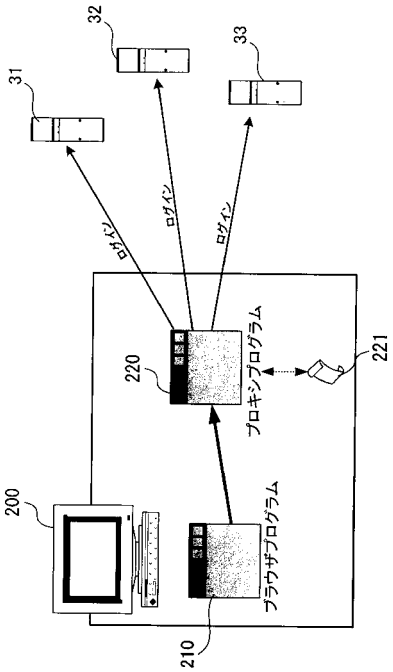
【図２１】



【図２２】



【図２３】



フロントページの続き

審査官　平井　誠

(56)参考文献　特開２００２－２０２９５５（ＪＰ，Ａ）
特開２００２－０３２２１６（ＪＰ，Ａ）
特開２００２－３３４０５６（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 21/20
G06F 15/00
H04L 9/32