

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4847397号

(P4847397)

(45) 発行日 平成23年12月28日(2011.12.28)

(24) 登録日 平成23年10月21日(2011.10.21)

(51) Int.Cl.

F I

G O 6 F 13/00 (2006.01)

G O 6 F 13/00 5 0 0 A

G O 6 F 15/00 (2006.01)

G O 6 F 15/00 3 2 0 D

請求項の数 3 (全 12 頁)

(21) 出願番号 特願2007-148816 (P2007-148816)
 (22) 出願日 平成19年6月5日(2007.6.5)
 (65) 公開番号 特開2008-304969 (P2008-304969A)
 (43) 公開日 平成20年12月18日(2008.12.18)
 審査請求日 平成22年1月14日(2010.1.14)

(73) 特許権者 000233055
 株式会社日立ソリューションズ
 東京都品川区東品川四丁目12番7号
 (74) 代理人 100096954
 弁理士 矢島 保夫
 (72) 発明者 田中 輝明
 東京都品川区東品川4丁目12番7号 日
 立ソフトウェアエンジニアリング株式会社
 内

審査官 安田 太

最終頁に続く

(54) 【発明の名称】 逐次保存型Webアプリケーションシステム

(57) 【特許請求の範囲】

【請求項1】

Webアプリケーションが動作するWebサーバ装置と、該Webサーバ装置にネットワーク経由でアクセスするクライアント端末とを備えたWebアプリケーションシステムであって、前記クライアント端末は、

前記Webサーバ装置に対して、ユーザIDを含むエントリ要求のリクエストを送信する手段と、

前記Webサーバ装置から送信される入力画面を表示し、該入力画面が備える入力フィールドへのユーザ入力を受け付ける手段と、

前記入力フィールドへの入力に応じて、ユーザIDと前記入力画面の入力フィールドへの入力データを含む逐次保存要求のリクエストを前記Webサーバ装置に送信する手段と、

前記入力画面でユーザが入力データの確定を指示したとき、ユーザIDと前記入力画面の入力フィールドへの入力データを含む確定要求のリクエストを前記Webサーバ装置に送信する手段と

を備え、

前記Webサーバ装置は、

ユーザIDと入力画面の入力フィールドへの入力データとを対応させて一時保存する一時保存用記憶手段と、

ユーザIDと入力画面の入力フィールドへの確定された入力データとを対応させて保存する確定データ保存用記憶手段と、

10

20

前記クライアント端末から送信されるエントリ要求のリクエストを受信したとき、そのリクエストに含まれるユーザIDが前記一時保存用記憶手段に記憶されているか否かを判定し、(1)記憶されていない場合は、前記一時保存用記憶手段に当該ユーザIDに対応する初期データレコードを格納すると共に、入力フィールドが初期状態である入力画面を前記クライアント端末に送信し、(2)記憶されていた場合は、前記一時保存用記憶手段に記憶されている当該ユーザIDに対応する入力データを含む入力画面を前記クライアント端末に送信するエントリ処理手段と、

前記クライアント端末から送信される逐次保存要求のリクエストを受信したとき、そのリクエストに含まれる入力データを、前記一時保存用記憶手段の当該ユーザIDに対応するレコードに格納する逐次保存処理手段と、

10

前記クライアント端末から送信される確定要求のリクエストを受信したとき、そのリクエストに含まれるユーザIDに対応して前記一時保存用記憶手段に格納されている入力データに、前記確定要求のリクエストに含まれる入力データを上書きする形で加えて、前記確定データ保存用記憶手段に格納すると共に、当該ユーザIDに対応して前記一時保存用記憶手段に格納されていたデータを削除する確定処理手段と

を備えることを特徴とするWebアプリケーションシステム。

【請求項2】

請求項1に記載のWebアプリケーションシステムにおいて、

前記一時保存用記憶手段は、各ユーザIDに対応するレコード内に更新番号のカラムを備えており、

20

前記エントリ処理手段は、(1)前記エントリ要求のリクエストに含まれるユーザIDが前記一時保存用記憶手段に記憶されておらずに前記一時保存用記憶手段に当該ユーザIDに対応する初期データレコードを格納する場合は、該初期データレコードの更新番号は初期値"0"に初期設定し、(2)前記エントリ要求のリクエストに含まれるユーザIDが前記一時保存用記憶手段に記憶されていた場合は、前記一時保存用記憶手段に記憶されている当該ユーザIDに対応するレコードの更新番号は初期値"0"に初期化し、(3)前記(1)および(2)の何れの場合も前記クライアント端末に送信する入力画面にはその更新番号を含めて送信するものであり、

前記逐次保存処理手段は、逐次保存要求のリクエストに含まれる更新番号と前記一時保存用記憶手段に格納されている当該ユーザIDに対応するレコードの更新番号とを比較し、逐次保存要求のリクエストの更新番号が前記一時保存用記憶手段に格納されている当該ユーザIDに対応するレコードの更新番号より大きい場合のみ、逐次保存要求のリクエストに含まれる入力データの前記一時保存用記憶手段への格納を行うものであり、

30

前記クライアント端末は、前記Webサーバ装置から入力画面と共に送信される更新番号を記憶し、前記逐次保存要求のリクエストを送信する際に該更新番号を1インクリメントし、インクリメントした更新番号を当該逐次保存要求のリクエストに含めて送信するものである

ことを特徴とするWebアプリケーションシステム。

【請求項3】

請求項1または2に記載のWebアプリケーションシステムにおいて、

40

前記Webサーバ装置は、前記クライアント端末からキャンセル要求のリクエストが送信されたとき、該リクエストに含まれるユーザIDに対応する前記一時保存用記憶手段内のレコードを削除する手段を備えることを特徴とするWebアプリケーションシステム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、Webブラウザをユーザインターフェースとするオンラインシステムにおける入力情報の保存の技術に関するものである。

【背景技術】

【0002】

50

従来のWebアプリケーションシステムでは、入力情報はセッションに一時的に保存され、確定操作を行う際にセッションに一時保存された情報をデータベースなどの記録媒体に保存する方式が用いられることが多い。セッションに一時的に保存された状態では、サーバシャットダウンが発生すると、入力情報は失われてしまう。

【0003】

特許文献1には、メンテナンスなどにより予期できるサーバシャットダウンに対応し、シャットダウン後の操作で入力状態を継続する技術が開示されている。

【特許文献1】特開2005-316634

【発明の開示】

【発明が解決しようとする課題】

10

【0004】

一般的に、Webアプリケーションシステムは対象とするユーザ数が多いため、セッションのタイムアウト時間が設定されていることが普通である。確定操作を行う前にユーザが何らかの事情でWebブラウザを閉じたり、長時間操作しないことでセッションタイムアウトが発生した場合、そこまでのユーザの入力情報は失われてしまう。また、障害などによる予期できないサーバシャットダウンでも、確定操作を行う前の入力情報は失われる。

【0005】

さらに、Webアプリケーションの仕組みとして、ユーザからリクエストが発生しない限りサーバに入力情報が送られることはない。入力項目や確認項目が非常に多い画面では、入力中にセッションのタイムアウトとなってしまう、ボタンを押下した際にタイムアウトエラー画面に遷移して、入力情報が失われてしまう懸念がある。

20

【0006】

本発明の目的は、例えば画面上のボタンを押下しなくてもリクエストを発行してサーバにそこまでの入力状態を保存し、セッションタイムアウトや予期しないサーバシャットダウンが発生しても、発生前の入力情報を保持したまま継続することができる逐次保存型Webアプリケーションシステムの技術を提供することにある。

【課題を解決するための手段】

【0007】

上記目標を達成するために、本発明に係る逐次保存型Webアプリケーションシステムは、Webアプリケーションが動作するWebサーバ装置と、該Webサーバ装置にネットワーク経由でアクセスするクライアント端末とを備えたWebアプリケーションシステムであって、クライアント端末は、Webサーバ装置にユーザIDを含むエントリ要求のリクエストを送信し、Webサーバ装置から送信される入力画面の入力フィールドへのユーザ入力を受け付け、その入力フィールドへの入力に応じて、ユーザIDと前記入力画面の入力フィールドへの入力データを含む逐次保存要求のリクエストをWebサーバ装置に送信し、入力画面でユーザが入力データの確定を指示したとき、ユーザIDと前記入力画面の入力フィールドへの入力データを含む確定要求のリクエストをWebサーバ装置に送信する。Webサーバ装置は、ユーザIDと入力画面の入力フィールドへの入力データとを対応させて一時保存する一時保存用記憶手段と、ユーザIDと入力画面の入力フィールドへの確定された入力データとを対応させて保存する確定データ保存用記憶手段とを備えている。エントリ処理手段は、エントリ要求のリクエストを受信したとき、そのリクエストに含まれるユーザIDが前記一時保存用記憶手段に記憶されているか否かを判定し、(1)記憶されていない場合は、前記一時保存用記憶手段に当該ユーザIDに対応する初期データレコードを格納すると共に、入力フィールドが初期状態である入力画面を前記クライアント端末に送信し、(2)記憶されていた場合は、前記一時保存用記憶手段に記憶されている当該ユーザIDに対応する入力データを含む入力画面を前記クライアント端末に送信する。逐次保存処理手段は、逐次保存要求のリクエストを受信したとき、そのリクエストに含まれる入力データを、前記一時保存用記憶手段の当該ユーザIDに対応するレコードに格納する。確定処理手段は、確定要求のリクエストを受信したとき、そのリクエストに含まれるユーザIDに対応して前記一時保存用記憶手段に格納されている入力データに、前記確定要求のリクエストに含まれる入力デ

30

40

50

ータを上書きする形で加えて、前記確定データ保存用記憶手段に格納すると共に、当該ユーザIDに対応して前記一時保存用記憶手段に格納されていたデータを削除する。

【0008】

前記クライアント端末が逐次保存要求のリクエストを送信するタイミングは任意であるが、例えば、入力画面の1つの入力フィールドに入力データを記入して次の入力フィールドにフォーカスを移したタイミングで送信するとよい。

【0009】

さらに、一時保存用記憶手段は、各ユーザIDに対応するレコード内に更新番号のカラムを備え、エントリ処理手段は、(1)前記エントリ要求のリクエストに含まれるユーザIDが前記一時保存用記憶手段に記憶されておらずに前記一時保存用記憶手段に当該ユーザIDに対応する初期データレコードを格納する場合は、該初期データレコードの更新番号は初期値"0"に初期設定し、(2)前記エントリ要求のリクエストに含まれるユーザIDが前記一時保存用記憶手段に記憶されていた場合は、前記一時保存用記憶手段に記憶されている当該ユーザIDに対応するレコードの更新番号は初期値"0"に初期化し、(3)前記(1)および(2)の何れの場合も前記クライアント端末に送信する入力画面にはその更新番号を含めて送信するものとし、逐次保存処理手段は、逐次保存要求のリクエストに含まれる更新番号と前記一時保存用記憶手段に格納されている当該ユーザIDに対応するレコードの更新番号とを比較し、逐次保存要求のリクエストの更新番号が前記一時保存用記憶手段に格納されている当該ユーザIDに対応するレコードの更新番号より大きい場合のみ、逐次保存要求のリクエストに含まれる入力データの前記一時保存用記憶手段への格納を行うものとし、クライアント端末は、前記Webサーバ装置から入力画面と共に送信される更新番号を記憶し、前記逐次保存要求のリクエストを送信する際に該更新番号を1インクリメントし、インクリメントした更新番号を当該逐次保存要求のリクエストに含めて送信するものとする。とよい。

【0010】

また、キャンセル要求のリクエストが送信されたとき、該リクエストに含まれるユーザIDに対応する前記一時保存用記憶手段内のレコードを削除するようにして、キャンセル機能を実現することもできる。

【発明の効果】

【0011】

本発明の逐次保存型Webアプリケーションシステムによれば、例えば画面上の確定ボタンを押下しなくても、入力フィールドの入力に応じて逐次保存要求のリクエストを発行してサーバに入力状態を一時保存するので、セッションタイムアウトや予期しないサーバシャットダウンが発生しても、発生前の入力情報を再現して入力を継続することができる。また、更新番号により入力の順序を管理するので、リクエストが遅延して先に入力したデータが後に入力したデータより遅れてサーバに到着したとしても、より新しい入力データを採用することができる。さらに、キャンセル機能により、データ確定前の入力のキャンセルも実現できる。

【発明を実施するための最良の形態】

【0012】

以下、本発明を適用した逐次保存型Webアプリケーションシステムの一実施の形態について説明する。

【0013】

図1は、本発明の一実施の形態例を示すシステム構成図である。この実施形態の逐次保存型Webアプリケーションシステムは、Webサーバ101とデータベースサーバ(以下、DBサーバ)102とを備える。クライアントは、Webブラウザ103を利用して、インターネットやイントラネット104を通じてHTTP通信を実施し、サーバにアクセスする。

【0014】

Webサーバ101にはアプリケーションサーバがインストールされており、アプリケーション105がデプロイ(展開)されている。そのアプリケーション105は、画面制御やDBアクセ

スなどの一般的なWebアプリケーションとしての機能のほかに、エントリロジック106、逐次保存ロジック107、キャンセルロジック108、および確定ロジック109の機能を備えている。DBサーバ102にはリレーショナルデータベースがインストールされており、アプリケーション105で使用する情報が保持される。入力値が保存されるテーブルは、一時保存用テーブル110と確定データテーブル111との2テーブルの構成となっている。

【 0 0 1 5 】

ユーザが、クライアントからWebブラウザ103を使用してWebサーバ101にアクセスすると、Webブラウザ103にはエントリ画面112が表示される。該エントリ画面112のエントリボタン114を押下することにより、入力画面113に遷移する。この際、サーバ側ではエントリロジック106が動作する。入力画面113には入力フィールドが複数存在する。入力フィールドに入力（116）されると、リクエストロジック115が起動し、サーバ101にリクエストが発行され、逐次保存ロジック107が動作する。入力画面113でキャンセルボタン117を押下すると、キャンセルロジック108が動作することにより、それまで入力した結果を破棄して、エントリ画面112に戻ることができる。入力画面113で確定ボタン118を押下すると、確定ロジック109が動作することにより、それまで入力した結果を確定させて、完了画面に遷移することができる。

【 0 0 1 6 】

図2は、DBサーバ102のデータベースにおいて入力画面の値を蓄積するテーブルの構成図である。一時保存用テーブル201（図1の110）および確定データテーブル202（図1の111）の2テーブル構成となっている。これは、入力情報のキャンセル機能を実現するためである。一般的なWebアプリケーションと異なり、本実施形態のシステムでは、クライアントでの入力値は逐一DBサーバ102に送信されてテーブルが更新される。従って、もしテーブル構成が1つとすると、入力途中の情報のキャンセル（なかったことにする）機能を実現することが難しくなってしまう。このため、本実施形態のシステムでは、確定ボタンを押下するまでの入力途中状態の入力値は一時保存テーブル201に格納し、確定ボタン押下時の入力値を確定データテーブル202に格納するようにしている。これにより、キャンセル機能が容易に実現できる。

【 0 0 1 7 】

本実施形態では、ユーザから氏名および住所などの情報を入力画面で入力してもらうものとしている。2つのテーブルは、何れも、ユーザID203、氏名205、および住所206のカラムを持つ。入力値を保持するカラムには空（またはNULL）が格納される可能性があるため、NOTNULL制約などの制約はつけない。一時保存用テーブル201のみ、更新番号204カラムを持つ。更新番号204カラムの使用方法は、後のフローチャートの説明で述べる。

【 0 0 1 8 】

一時保存用テーブル201には入力中のデータが格納され、確定データテーブル202には確定したデータが格納される。例えば、ユーザIDが「user001」のレコード207は、一度「田中輝明」「東京都品川区東品川4-12-7」で確定したが、住所を「愛知県名古屋市中区栄4-14-5」に変更しようとしている最中のデータである。ユーザIDが「user002」のレコード208は、新規に「鈴木太郎」「神奈川県横浜市戸塚区品野町549-6」で登録しようとしている最中のデータである。

【 0 0 1 9 】

図3は、本実施形態における画面遷移の例を示す図である。ユーザはWebブラウザ103においてエントリ画面301（図1の112）を表示し、ユーザID302およびパスワード303を入力した後、ログインボタン304（図1のエントリボタン114）を押下すると、入力画面305（図1の113）に遷移する。入力画面305で氏名306および住所307などの情報を入力フィールドに入力し、確定ボタン308（図1の118）を押下すると、完了画面310に遷移する。入力画面305でキャンセルボタン309（図1の117）を押下すると、入力途中のデータを破棄してエントリ画面301に戻る。完了画面310では、確定した入力値の氏名311および住所312などが表示される。

【 0 0 2 0 】

図4は、エントリロジック106の処理概要を示すフローチャートである。エントリロジック106は、エントリ画面112のエントリボタン114により起動されるサーバ側処理である(ステップ401)。初めに、クライアントから送られてきたリクエストからユーザIDを取得する(ステップ402)。一時保存用テーブル201を参照して、取得したユーザIDのデータが存在するかチェックする(ステップ403)。データが存在しない場合、一時保存用テーブル201に初期データを作成する(ステップ404)。初期データは主キーのユーザID203と更新番号204のカラムのみが格納されたデータであり、更新番号204の初期値は「0」(固定)とする。セッションタイムアウトなどにより入力が入断された場合は、一時保存用テーブル201にデータが存在している。この場合は、ステップ403から405に進み、一時保存用テーブル201の当該レコードの更新番号204を「0」にリセットして(ステップ405)、入力途中の情報を取得する(ステップ406)。ステップ404またはステップ406の情報を元に、入力画面の表示処理を行い(ステップ407)、エントリロジック106の処理を終了する(ステップ408)。

【0021】

図5は、リクエストロジック115の処理概要を示すフローチャートである。リクエストロジック115は、入力画面113の入力フィールド(例えば、図3の306, 307など)にユーザが入力すること(116)により起動するクライアント側ロジック(Javascript(登録商標))である(ステップ501)。初めに、画面からユーザIDおよび入力値を取得する(ステップ502, 503)。更新番号はJavascriptで管理しておき(初期値「1」)、リクエストロジック115が実行される毎に1ずつインクリメントして採番する(ステップ504)。取得したユーザID、入力値、および採番した更新番号をHTTPリクエストにてサーバ101に送信し(ステップ505)、リクエストロジック115を終了する(ステップ507)。

【0022】

リクエストロジック115の起動はJavascriptのイベント実装になるため、起動タイミングは、入力フィールドの内容が変わる、入力フィールドのフォーカスを移動したとき、など、Javascriptのイベント実装であればどのタイミングでも可能である。しかし、イベントが発生するたびにHTTPリクエストが発生し、その先で後述のサーバ側ロジック107が起動してデータベース更新処理が行われるため、業務量およびデータベース性能を考慮してイベントを選択することが望ましい(入力フィールドの内容が変わるタイミングの場合、文字を1文字打ち込むタイミングでデータベース更新処理が実行されることになり、高負荷がかかる)。なお、リクエストロジック115は、例えばAjax(Asynchronous Javascript + XML)のような非同期通信手法を用いることによって実現されるが、他の方法を用いてもよい。

【0023】

図6は、逐次保存ロジック107の処理概要を示すフローチャートである。逐次保存ロジック107は、図5のリクエストロジック115から送られたHTTPリクエストを受信することにより起動するサーバ側ロジックである(ステップ601)。初めに、受信したリクエストからユーザID、更新番号、および入力値を取得する(ステップ602)。次に、入力値のサイズチェックを行う(ステップ603)。この時点ではまだユーザの入力が完了していないため、このチェックは入力値が適正かどうかをチェックするものではなく、対応するデータベースのカラムサイズに収まっているか否かをチェックするものである。サイズオーバーである場合は、逐次保存ロジックを終了する(ステップ604)。入力値がカラムサイズ内である場合、ユーザIDをキーに一時保存用テーブル201に更新用ロック(排他制御のため)を取得する(ステップ605)。ロックの粒度はレコード単位でのロックとすることが望ましい。ロック取得後、一時保存用テーブル201に当該ユーザIDのデータが存在するかチェックする(ステップ606)。存在する場合、リクエストから取得した更新番号と一時保存用テーブル201の更新番号とを比較する(ステップ608)。リクエストから取得した更新番号の方が大きければ、一時保存用テーブル201を更新して(ステップ610)、逐次保存ロジックを終了する(ステップ611)。

【0024】

逐次保存ロジック107はリクエストロジック115から起動される。前述の通り、リクエストロジック115はJavaScriptにより非同期に起動され、かつ、HTTPリクエストを発行するため、逐次保存ロジック107が起動される順序は保証されない。このため、更新番号をJavaScriptにより管理しその番号をサーバに送信することで、HTTPリクエストがより最新の入力値情報であるかを確認できる。HTTPリクエストが遅延してサーバ101に到達して逐次保存ロジック107が起動された場合、ステップ608のチェックによりデータベースを更新せずに逐次保存ロジックを終了する（ステップ609）。

【0025】

また、HTTPリクエストが遅延して、キャンセルロジック108および確定ロジック109の起動より後にサーバ101に到達するケースもあり得る。この場合、既に確定しているデータに対して入力中の更新をしないようにする必要がある。後述するが、キャンセルロジック108および確定ロジック109では一時保存用テーブル201からデータを削除するため、既に確定しているデータについては、ステップ606のデータの存在チェックによりデータベースを更新せずに逐次保存ロジックを終了することになる（ステップ607）。

【0026】

図7は、HTTPリクエストが遅延する様子を示す図である。画面の入力状態が701 702 703 704と変化したとする。702、703、704に変化するタイミングでリクエストロジック115が起動してHTTPリクエストが発行されたとする。サーバ101側では、まず、702に更新されたときのHTTPリクエストが到達し、705のようにデータベースが更新される。次に、703に更新されたときのHTTPリクエストが遅延し、704に更新されたときのHTTPリクエストが先に到達したとすると、706のようにデータベースが更新される。その後、遅延した703に更新したときのHTTPリクエストが到達すると、リクエストの更新番号<データベースの更新番号となっているため、ステップ608のチェックによりデータベースが更新されずに済む。

【0027】

図8は、キャンセルを実施するサーバ側ロジック108のフローチャートである。キャンセルロジック108は、入力画面112のキャンセルボタン117により起動されるサーバ側処理である（ステップ801）。初めに、キャンセルを指示するリクエストからユーザIDを取得する（ステップ802）。ユーザIDをキーに一時保存用テーブル201に更新用ロックを取得する（ステップ803）。ロック取得後、一時保存用テーブル201に当該ユーザIDのデータが存在するかチェックする（ステップ804）。存在する場合、一時保存用テーブル201から当該ユーザIDをキーにデータを削除する（ステップ805）。存在しないケースは、ボタン連打などによるキャンセルロジックおよび確定ロジックの多重起動や、ブラウザの戻るボタンによるものなどが考えられる。いずれも正常な画面遷移ではないので、データベースの更新処理は実施しない。最後に、エントリ画面112を表示する処理を実施し（ステップ806）、キャンセルロジックを終了する（ステップ807）。

【0028】

図9は、確定処理を実施するサーバ側ロジック109のフローチャートである。確定ロジック109は、入力画面112の確定ボタン118により起動されるサーバ側処理である（ステップ901）。初めに、確定を指示するリクエストからユーザID、更新番号、および入力値を取得する（ステップ902）。次に、入力値の単項目チェックを行う（ステップ903）。確定処理なので、このチェックでは入力値が適正かどうかをチェックする。チェックエラーである場合は、入力画面にエラーメッセージを表示する処理を行い（ステップ904）、確定ロジックを終了する（ステップ905）。

【0029】

ステップ903のチェックを通過した場合、ユーザIDをキーに一時保存用テーブル201に更新用ロックを取得する（ステップ906）。ロック取得後、一時保存用テーブル201に当該ユーザIDのデータが存在するかチェックする（ステップ907）。存在する場合、リクエストから取得した入力値で一時保存用テーブル201を更新する（ステップ910）。処理の優先度は、逐次保存ロジック107へのリクエストより確定ロジック109へのリクエストの方が高い

10

20

30

40

50

ため、ここでは更新番号のチェックを実施する必要はない。存在しないケースはキャンセルロジック108で述べたケースと同様となる。すでにキャンセルまたは確定がなされた後なので、確定済を通知する画面を表示する処理を実施し（ステップ908）、確定ロジックを終了する（ステップ909）。

【0030】

ステップ910の一時保存用テーブル201への更新の後、必要があれば関連項目チェックを行う（ステップ911）。チェックエラーである場合は、入力画面にエラーメッセージを表示する処理を行い（ステップ912）、確定ロジックを終了する（ステップ913）。チェックを通過した場合、一時保存用テーブル201から確定データテーブル202にデータをコピーし（ステップ914）、一時保存用テーブル201から当該データを削除する（ステップ915）。その後、完了画面表示処理を行い（ステップ916）、確定ロジックを終了する（ステップ917）。

10

【0031】

図10は、画面遷移の具体例の説明図である。ユーザはエントリ画面1001（図1の112）を表示し、ユーザID等（不図示）を入力してエントリボタン1002（図1の114）を押下する。入力画面1003（図1の113）に遷移し、名前欄1004を入力し、住所欄1005を入力する前にタイムアウトが発生したとする。ユーザは再度エントリ画面1006を表示し、ユーザID等を入力してエントリボタン1007を押下する。入力画面1008に遷移すると、タイムアウト前に入力されていた名前欄1009の情報が復元されている。

【図面の簡単な説明】

20

【0032】

【図1】本発明の一実施の形態例を示すシステム構成図である。

【図2】データベースにおいて入力画面の値を蓄積するテーブルの構成図である。

【図3】本実施の形態における画面遷移図である。

【図4】入力画面へ遷移する際のサーバ側ロジックのフローチャートである。

【図5】フィールド入力中にリクエストを実施するクライアント側ロジックのフローチャートである。

【図6】逐次保存を実施するサーバ側ロジックのフローチャートである。

【図7】HTTPリクエストが遅延する様子を示す図である。

【図8】キャンセルを実施するサーバ側ロジックのフローチャートである。

30

【図9】確定処理を実施するサーバ側ロジックのフローチャートである。

【図10】画面遷移の具体例の説明図である。

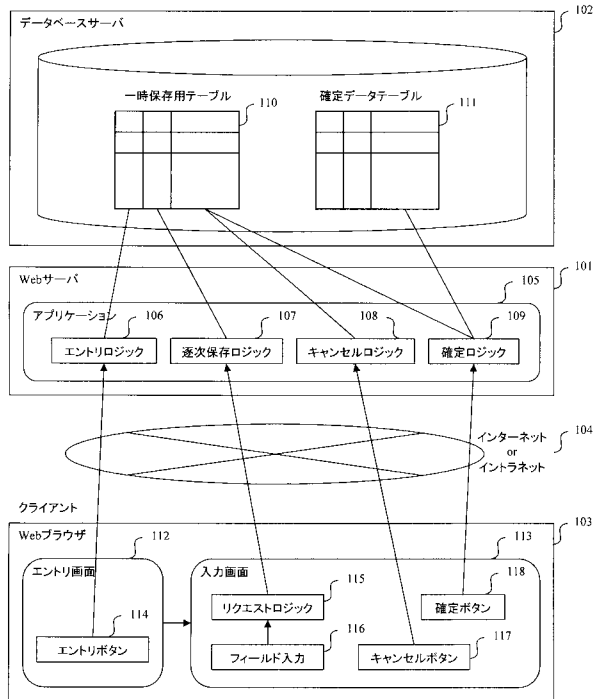
【符号の説明】

【0033】

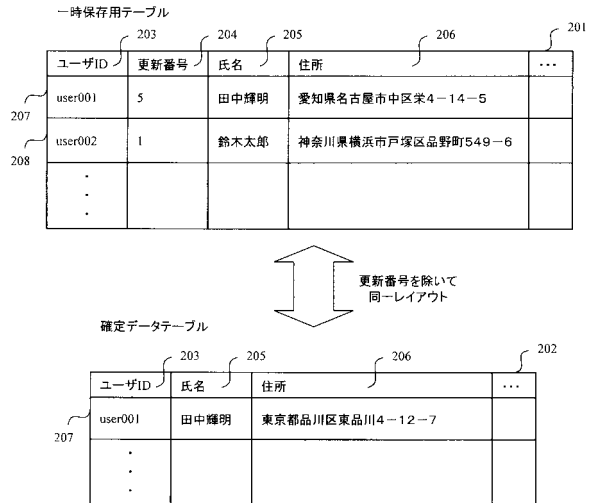
101...Webサーバ、102...データベースサーバ、103...Webブラウザ、104...インターネットorイントラネット、105...Webアプリケーション、106...エントリロジック、107...逐次保存ロジック、108...キャンセルロジック、109...確定ロジック、110...一時保存用テーブル、111...確定データテーブル、112...エントリ画面、113...入力画面、114...エントリボタン、115...リクエストロジック、116...フィールド入力イベント、117...キャンセルボタン、118...確定ボタン。

40

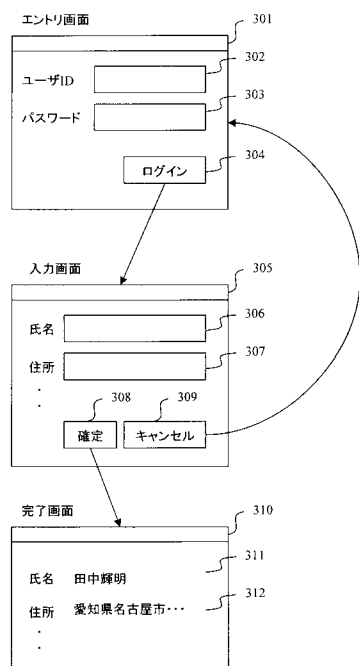
【図 1】



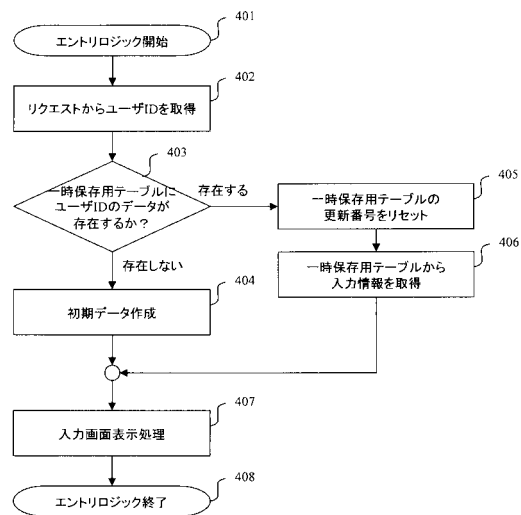
【図 2】



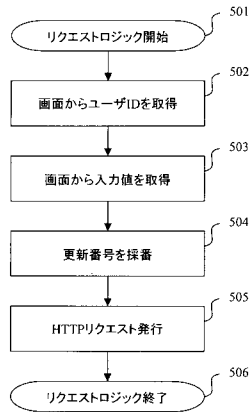
【図 3】



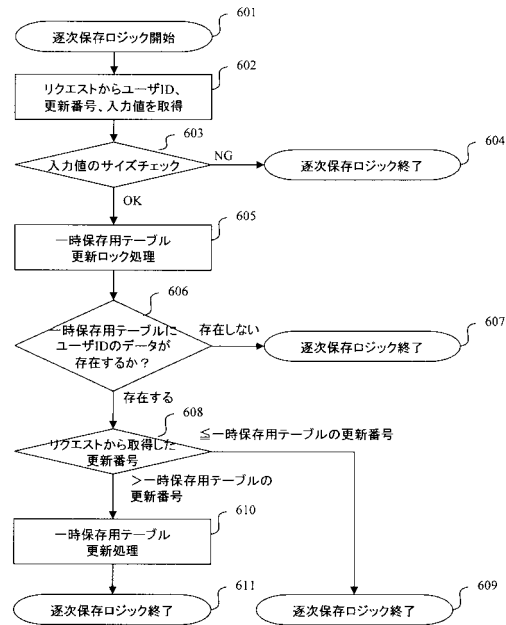
【図 4】



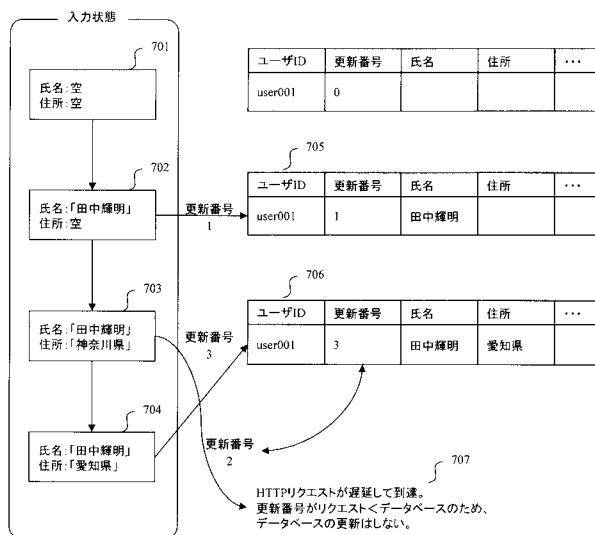
【図 5】



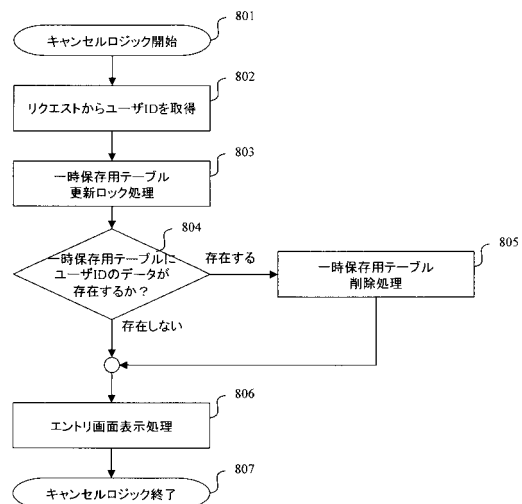
【図 6】



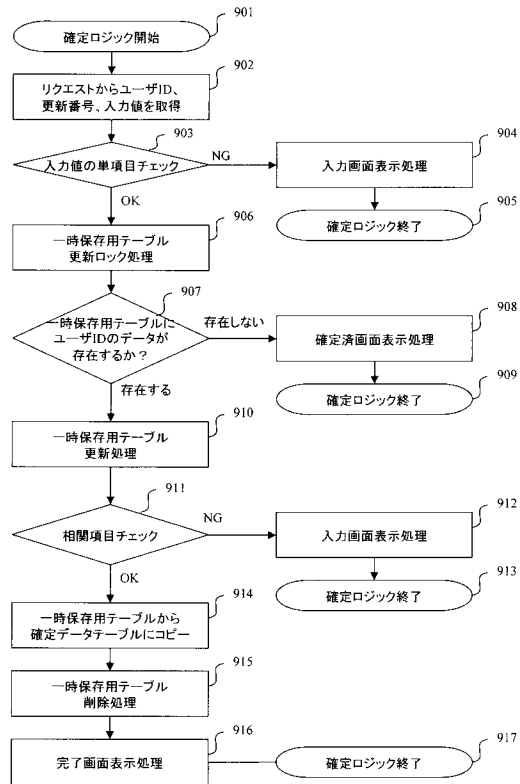
【図 7】



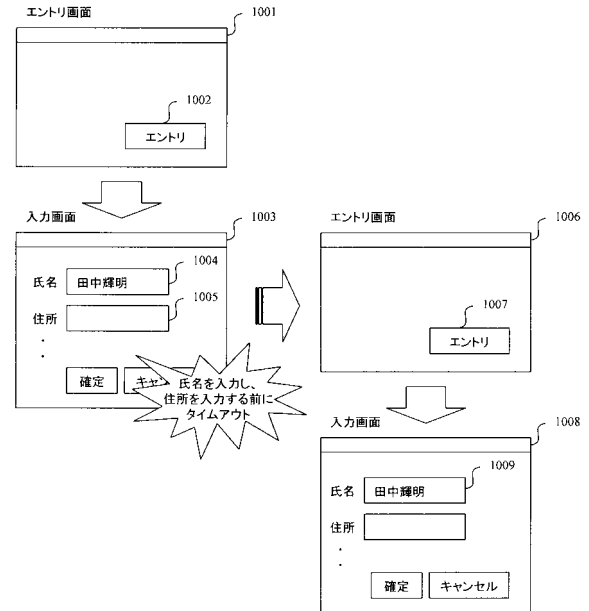
【図 8】



【図 9】



【図 10】



フロントページの続き

(56)参考文献 特開2001-306512(JP,A)
特開2003-058501(JP,A)
特開2003-288650(JP,A)
特開2000-259565(JP,A)

(58)調査した分野(Int.Cl., DB名)
G06F 13/00
G06F 15/00