

(19) 日本国特許庁(JP)

(12) 公開特許公報(A)

(11) 特許出願公開番号
特開2012-185650
(P2012-185650A)

(43) 公開日 平成24年9月27日(2012.9.27)

(51) Int.Cl.	F I	テーマコード (参考)
G 0 6 F 13/00 (2006.01)	G O 6 F 13/00 5 6 O A	5 B O 8 4
H O 4 N 1/00 (2006.01)	H O 4 N 1/00 C	5 C O 6 2
	H O 4 N 1/00 1 O 7 A	

審査請求 未請求 請求項の数 11 O L (全 19 頁)

(21) 出願番号	特願2011-47910 (P2011-47910)	(71) 出願人	000001007
(22) 出願日	平成23年3月4日 (2011.3.4)		キヤノン株式会社
			東京都大田区下丸子3丁目30番2号
		(74) 代理人	100126240
			弁理士 阿部 琢磨
		(74) 代理人	100124442
			弁理士 黒岩 創吾
		(72) 発明者	中島 康輔
			東京都大田区下丸子3丁目30番2号キヤ ノン株式会社内
		Fターム(参考)	5B084 AA06 AA12 AB04 BB01 DB08 DC03 5C062 AA05 AA13 AA35 AB02 AB17 AB20 AB23 AB38 AB40 AC02 AC05 AC34

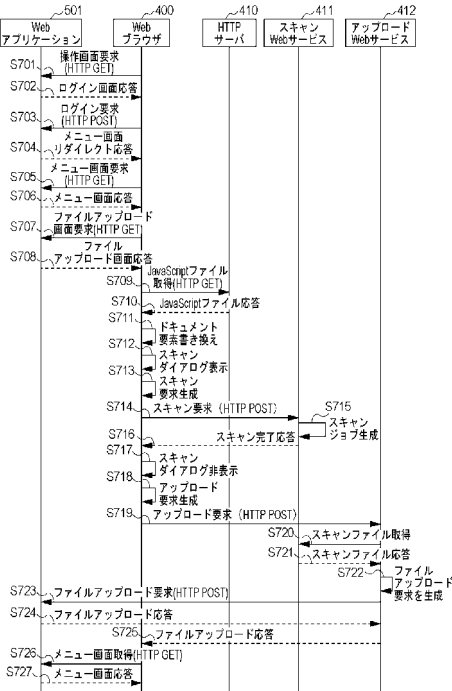
(54) 【発明の名称】 情報処理装置、情報処理システム、情報処理装置の制御方法、及びプログラム

(57) 【要約】

【課題】 ファイルのアップロードを汎用のWebブラウザを用いて簡単に行える仕組みを提供することを目的とする。

【解決手段】 ネットワークに接続する接続手段（218）と、ネットワーク上のWebサーバから画面データを受信し、当該画面データに基づいて画面を表示するWebブラウザ（400）とを備える情報処理装置（101）において、Webブラウザが受信する画面データを書き換えるために使用されるスクリプトを保持し、当該スクリプトを使用して画面データが書き換えられた後（S711）、当該書き換えられた画面データに基づいて表示される画面を介したユーザからの指示に応じて、ファイルの生成および当該生成したファイルのアップロードを実行させる（S714及びS719）。

【選択図】 図7



【特許請求の範囲】**【請求項 1】**

ネットワークに接続する接続手段と、

前記ネットワーク上の Web サーバから画面データを受信し、当該画面データに基づいて画面を表示する Web ブラウザと、

前記 Web ブラウザが受信する画面データを書き換えるために使用されるスクリプトを保持する保持手段と、

前記スクリプトを使用して画面データが書き換えられた後、当該書き換えられた画面データに基づいて表示される画面を介したユーザからの指示に応じて、ファイルの生成および当該生成したファイルのアップロードを実行させる制御手段と、

を備えることを特徴とする情報処理装置。

10

【請求項 2】

原稿上の画像を読み取って、当該画像に基づいて画像ファイルを生成する読取手段を更に備え、

前記制御手段は、前記書き換えられた画面データに基づいて表示される画面を介したユーザからの指示に従って、前記読取手段に前記画像ファイルの生成を実行させることを特徴とする請求項 1 に記載の情報処理装置。

【請求項 3】

前記ファイルの生成を制御する第 1 の Web サービス手段を更に備え、

前記制御手段は、前記第 1 の Web サービス手段に対して前記ファイルの生成を要求することを特徴とする請求項 1 または 2 に記載の情報処理装置。

20

【請求項 4】

前記生成されたファイルを前記 Web サーバにアップロードするアップロード手段を更に備え、

前記制御手段は、前記書き換えられた画面データに基づいて表示される画面を介したユーザからの指示に従って、前記アップロード手段に前記ファイルのアップロードを実行させることを特徴とする請求項 1 から 3 のいずれか 1 項に記載の情報処理装置。

【請求項 5】

前記ファイルのアップロードを制御する第 2 の Web サービス手段を更に備え、

前記制御手段は、前記第 2 の Web サービス手段に対して前記ファイルのアップロードを要求することを特徴とする請求項 1 から 4 のいずれか 1 項に記載の情報処理装置。

30

【請求項 6】

前記保持手段が保持するスクリプトには、第 2 のスクリプトの取得が定義され、

前記第 2 のスクリプトには、前記画面データの書き換えが定義されていることを特徴とする請求項 1 から 5 のいずれか 1 項に記載の情報処理装置。

【請求項 7】

前記保持手段は、前記 Web ブラウザのブックマークとして前記スクリプトを保持することを特徴とする請求項 1 から 6 のいずれか 1 項に記載の情報処理装置。

【請求項 8】

前記保持手段が保持するスクリプトの実行指示をユーザから受け付ける受付手段を更に備え、

前記実行指示を受け付けたことに応じて、前記保持手段が保持するスクリプトが実行されることを特徴とする請求項 1 から 7 のいずれか 1 項に記載の情報処理装置。

40

【請求項 9】

ネットワークに接続する接続手段と、

前記ネットワーク上の Web サーバから画面データを受信し、当該画面データに基づいて画面を表示する Web ブラウザと、

前記 Web ブラウザが受信する画面データを書き換えるために使用されるスクリプトを保持する保持手段と、

前記スクリプトを使用して画面データが書き換えられた後、当該書き換えられた画面デ

50

ータに基づいて表示される画面を介したユーザからの指示に応じて、ファイルの生成および当該生成したファイルのアップロードを実行させる制御手段と、

を備えることを特徴とする情報処理システム。

【請求項 10】

ネットワークに接続する接続手段と、前記ネットワーク上の Web サーバから画面データを受信し、当該画面データに基づいて画面を表示する Web ブラウザと、を備える情報処理装置の制御方法であって、

前記 Web ブラウザが受信する画面データを書き換えるために使用されるスクリプトを保持する保持工程と、

前記スクリプトを使用して画面データが書き換えられた後、当該書き換えられた画面データに基づいて表示される画面を介したユーザからの指示に応じて、ファイルの生成および当該生成したファイルのアップロードを実行させる制御工程と、

を備えることを特徴とする情報処理装置の制御方法。

【請求項 11】

請求項 10 に記載の情報処理装置の制御方法をコンピュータに実行させるためのプログラム。

【発明の詳細な説明】

【技術分野】

【0001】

本発明は、Web サーバから画面データを受信し、当該画面データに基づいて画面を表示する Web ブラウザを備える情報処理装置、情報処理システム、情報処理装置の制御方法、及びプログラムに関する。

【背景技術】

【0002】

PC などの情報処理装置がネットワーク上の Web サーバに接続され、Web サーバにより提供される操作画面を、情報処理装置が備える Web ブラウザ上に表示することが知られている。最近ではスキャナやプリンタを備えた MFP (Multi Function Peripheral) の中でも、上述したような Web ブラウザを備えているものがある。MFP は、Web サーバにより提供される操作画面を MFP の Web ブラウザに表示し、ユーザからの各種指示を受け付ける。

【0003】

そして、MFP の Web ブラウザで表示される操作画面を介してユーザが各種指示を行うことにより、MFP のスキャナやプリンタを制御する技術が知られている。特許文献 1 では、Web アプリケーションによって提供された操作画面にファイルアップロード (RFC 1867 に従う) が定義されている場合、ユーザの送信指示を受けて、MFP がスキャナを制御してファイルを生成し、Web サーバへアップロードする。

【先行技術文献】

【特許文献】

【0004】

【特許文献 1】特開 2005-149320 号公報

【発明の概要】

【発明が解決しようとする課題】

【0005】

特許文献 1 の場合、MFP の Web ブラウザがスキャナを制御することを可能とするために、汎用の Web ブラウザを MFP 用にカスタマイズして MFP に搭載しなければならない。そのため、MFP 毎に Web ブラウザの種類が異なる場合、MFP 毎に Web ブラウザの拡張実装をしなければならないという課題がある。特に最近では、MFP の操作部にタブレット PC を使用するという構成も存在し、より多様な Web ブラウザに対応していかなければならなくなっている。

【0006】

10

20

30

40

50

本発明は、上記の問題点に鑑みなされたものであり、ファイルのアップロードを汎用のWebブラウザを用いて簡単に行える仕組みを提供することを目的とする。

【課題を解決するための手段】

【0007】

上記の目的を達成するために本発明の情報処理装置は、ネットワークに接続する接続手段と、前記ネットワーク上のWebサーバから画面データを受信し、当該画面データに基づいて画面を表示するWebブラウザと、前記Webブラウザが受信する画面データを書き換えるために使用されるスクリプトを保持する保持手段と、前記スクリプトを使用して画面データが書き換えられた後、当該書き換えられた画面データに基づいて表示される画面を介したユーザからの指示に応じて、ファイルの生成および当該生成したファイルのアップロードを実行させる制御手段と、を備えることを特徴とする。

10

【発明の効果】

【0008】

本発明によれば、ファイルのアップロードを汎用のWebブラウザを用いて簡単に行える仕組みを提供することが可能となる。

【図面の簡単な説明】

【0009】

【図1】本発明の実施形態における情報処理システムの全体図である。

【図2】本発明の実施形態におけるMF P 1 0 1の構成を示すブロック図である。

【図3】本発明の実施形態におけるサーバ102の構成を示すブロック図である。

20

【図4】本発明の実施形態におけるMF P 1 0 1のソフトウェア構成を示す図である。

【図5】本発明の実施形態におけるサーバ102のソフトウェア構成を示す図である。

【図6】本発明の実施形態におけるMF P 1 0 1の画面遷移を示す図である。

【図7】本発明の実施形態における情報処理システムの処理シーケンスを示す図である。

【図8】本発明の実施形態におけるファイルアップロード画面の一例を示す図である。

【図9】本発明の実施形態におけるブックマークレットの一例を示す図である。

【図10】本発明の実施形態におけるスキャン要求の一例を示す図である。

【図11】本発明の実施形態におけるアップロード要求の一例を示す図である。

【図12】本発明の実施形態におけるファイルアップロード要求の一例を示す図である。

【図13】本発明の実施形態におけるファイルアップロード完了応答の一例を示す図である。

30

【図14】本発明の実施形態におけるWebブラウザ400がJavaScriptを処理してドキュメントを書き換える処理の流れを表すフローチャートである。

【図15】本発明の実施形態におけるWebブラウザ400がJavaScriptを処理してスキャンファイルを生成する処理の流れを表すフローチャートである。

【図16】本発明の実施形態におけるWebブラウザ400がJavaScriptを処理してアップロード要求を生成する処理の流れを表すフローチャートである。

【発明を実施するための形態】

【0010】

以下、図面を参照して本発明の実施の形態を詳しく説明する。なお、以下の実施の形態は特許請求の範囲に係る発明を限定するものでなく、また実施の形態で説明されている特徴の組み合わせの全てが発明の解決手段に必須のものとは限らない。

40

【0011】

(第1の実施形態)

本発明に係る第1の実施形態について説明する。第1の実施形態では、ブックマークレットを用いて画面データ(HTMLデータ)を書き換えることにより、ファイル参照ボタンの押下に応じて原稿をスキャンしてファイルを生成し、送信ボタンの押下に応じてファイルをアップロードすることを可能とする。

【0012】

図1は、第1の実施形態における情報処理システムの全体図である。LAN110上(

50

ネットワーク上)には、MFP101、サーバ102(Webサーバ)が互いに通信可能に接続されている。

図2は、MFP101の構成を示すブロック図である。CPU211を含む制御部210は、MFP101全体の動作を制御する。CPU211は、ROM212に記憶された制御プログラムを読み出して読取制御や送信制御などの各種制御を行う。RAM213は、CPU211の主メモリ、ワークエリア等の一時記憶領域として用いられる。

【0013】

HDD214は、画像データや各種プログラムを記憶する。操作部I/F215は、操作部219と制御部210とを接続する。操作部219には、タッチパネル機能を有する液晶表示部やキーボードなどが備えられている。また、MFP101には後述するWebブラウザ機能が備えられており、MFP101のWebブラウザはサーバ102から受信したHTMLデータを解析し、受信したHTMLデータの記述に基づく操作画面を操作部219の液晶表示部に表示する。

【0014】

プリンタI/F216は、プリンタ220と制御部210とを接続する。プリンタ220で印刷すべき画像データはプリンタI/F216を介して制御部210から転送され、プリンタ220において記録媒体上に印刷される。

スキャナI/F217は、スキャナ221と制御部210とを接続する。スキャナ221は、原稿上の画像を読み取って画像データ(画像ファイル)を生成し、スキャナI/F217を介して制御部210に入力する。

ネットワークI/F218は、制御部210(MFP101)をLAN110に接続する。ネットワークI/F218は、LAN110上の外部装置(例えば、サーバ102)に画像データや情報を送信したり、LAN110上の外部装置から各種情報を受信したりする。

【0015】

図3は、サーバ102の構成を示すブロック図である。CPU311を含む制御部310は、サーバ102全体の動作を制御する。CPU311は、ROM312に記憶された制御プログラムを読み出して各種制御処理を実行する。RAM313は、CPU311の主メモリ、ワークエリア等の一時記憶領域として用いられる。HDD314は、画像データや各種プログラムを記憶する。

ネットワークI/F315は、制御部310(サーバ102)をLAN110に接続する。ネットワークI/F315は、LAN110上の他の装置との間で各種情報を送受信する。

【0016】

図4は、MFP101のソフトウェア構成を説明するための図である。図4に示す各機能部は、MFP101が有しているCPU211が制御プログラムを実行することにより実現される。

MFP101は、Webブラウザ400、HTTPサーバ410、及びジョブ制御部420を備えている。

Webブラウザ400は、HTTPプロトコルにしたがってリクエストを送信してレスポンスを受信する。そして、受信したレスポンスを解析して、操作画面を表示する。

Webブラウザ400は、JavaScriptエンジン401およびブックマークレット402を有する。JavaScriptエンジン401は、受信したレスポンスに含まれるJavaScriptを処理する。Webブラウザ400は、JavaScriptを処理することにより、HTTP通信することなくHTMLデータを書き換えたり、各種WebサービスへHTTPリクエストを送信したりすることができる。ブックマークレット402は、Webブラウザ400のブックマークに登録されたJavaScriptアプリケーションである。ブックマークよりブックマークレット402が選択されると、JavaScriptエンジン401はブックマークレット402を処理する。

【0017】

HTTPサーバ410は、Webブラウザ400からのリクエストを受信して、HTMLデータやJavaScriptファイルなどを応答したり、WebサービスへWebブラウザ400からのリクエストを送信したりする。

HTTPサーバ410が公開するWebサービスには、スキャンWebサービス411、およびアップロードWebサービス412がある。スキャンWebサービス411、およびアップロードWebサービス412は、WebアプリケーションからHTTPリクエストを受信し、WebアプリケーションへHTTPレスポンスを応答する。Webアプリケーション以外にも、JavaScriptエンジン401がブックマークレットを処理することによって、Webブラウザ400からリクエストが送信され、Webブラウザ400へレスポンスを応答する場合もある。

10

スキャンWebサービス411（第1のWebサービス）は、受信したHTTPリクエストを解析して、MFP101のスキナ211を制御して原稿上の画像を読み取って画像ファイルを生成するためのジョブ制御命令を生成する。そして、ジョブ制御部420にジョブ制御を指示する。また、ジョブ制御部420のジョブ完了通知を受けて、HTTPレスポンスを応答する。

【0018】

アップロードWebサービス412（第2のWebサービス）は、受信したHTTPリクエストを解析して、multi part / form - dataでエンコードしたデータを生成する。そして、生成したデータをリクエストとしてWebアプリケーションへ送信して、Webアプリケーションより送信完了画面を受信する。本実施の形態では、スキャンWebサービスが送信するスキャン画像を受信して、Webアプリケーションへアップロードする。アップロードWebサービス412は、送信完了画面をHTTPレスポンスとして応答する。

20

ジョブ制御部420は、指示されるジョブ制御命令を処理することによって、プリンタ230による印刷処理、スキナ221による読取処理、或いはネットワークI / F 218を介した送信処理などを実行する。また、ジョブ処理が完了すると、指示元に対してジョブ処理完了を通知する。

【0019】

図5は、サーバ102のソフトウェア構成を説明するための図である。図5に示す各機能部は、サーバ102が有しているCPU311が制御プログラムを実行することにより実現される。

30

サーバ102は、HTTPサーバ500、およびデータベース510を備えている。HTTPサーバ500は、Webブラウザ400からのリクエストを受信して、HTMLデータやJavaScriptファイルなどを応答したり、Webアプリケーション501へWebブラウザ400からのリクエストを送信したりする。

Webアプリケーション501は、Webブラウザ420からのリクエストに応じて、Webブラウザ420で表示すべきHTMLデータを生成する。また、フォーム認証要求を処理してセッション情報を表すクッキーを発行する。また、ファイルアップロード要求を処理してデータベース510にデータを登録する。生成されたHTMLデータやクッキーはHTTPレスポンスとしてWebブラウザ420へ送信する。

40

データベース510は、Webアプリケーション501の認証情報を管理したり、Webブラウザ400よりWebアプリケーション501へアップロードされるデータを管理したりする。

【0020】

ここまでで本実施形態におけるシステム構成の概要を説明した。引き続き、本実施形態におけるシステムの処理の概要を説明する。

【0021】

図6は、本実施形態における情報処理システムにおけるWebブラウザ400の画面遷移の一例を示す図である。まず、Webブラウザ400の画面構成について説明する。図6中の操作画面に示すように、Webブラウザ400は、タイトルバー、URLバー、ブ

50

ックマークボタン、およびコンテンツ表示領域を含む。タイトルバーは、HTMLデータのタイトルを表示する領域である。URLバーは、表示するコンテンツのURLを、入力または表示する領域である。ブックマークボタンは、予めWebブラウザ400に登録され、保持されているブックマークを呼び出すためのボタンである（ユーザからの実行指示の受付）。

図6の場合、“スキャン”という名称でJavaScriptがブックマークに登録されている。ブックマークに登録されるJavaScriptはブックマークレットと呼ばれる。ブックマークレットに記述されるJavaScriptの具体的な内容については図9を用いて後述する。コンテンツ表示領域は、指定されたURL先のコンテンツを表示する領域である。

10

【0022】

次に、各操作画面の内容と画面遷移について説明する。図6中の操作画面のうち、601から604はWebアプリケーション501が応答するコンテンツ（画面データ）をWebブラウザ400が処理して表示する操作画面である。一方、613から615は、Webブラウザ400がブックマークレットを処理することによって、Webアプリケーション501が応答するコンテンツ（画面データ）を書き換えて表示する操作画面である。

ログイン画面601は、ユーザIDとパスワードを入力してWebアプリケーション501へ認証するための画面である。ログイン画面601は、ユーザIDフィールド、パスワードフィールド、およびログインボタンを含む。ユーザIDフィールドにはWebアプリケーション501での認証に必要なユーザIDが入力される。パスワードフィールドにはWebアプリケーション501での認証に必要なパスワードが入力される。ログインボタンは、入力されたユーザIDおよびパスワードに基づいてWebアプリケーション501へ認証を要求するためのボタンである。Webブラウザ400は、ログインボタンが押下されると、Webアプリケーション501が認証に成功した場合に限りメニュー画面へ遷移する。

20

【0023】

メニュー画面602は、Webアプリケーション501が提供するドキュメント管理機能を利用するためのメニュー画面である。Webアプリケーション501が提供するドキュメント管理機能には、フォルダ作成、ファイルアップロード、ファイル削除の機能がある。メニュー画面602は各種機能ボタンおよびログアウトボタンを含む。また、Webアプリケーションが管理するドキュメントフォルダの一覧、およびドキュメントファイルの一覧を表示する。Webブラウザ400は、機能ボタンが押下されると、選択された機能に対応した操作画面に遷移する。また、ログアウトボタンが押下されると、ログイン画面601へ遷移する。

30

【0024】

ファイルアップロード画面603は、ファイルをアップロードするための操作画面である。操作画面603は、ファイル名入力フィールド、ファイル参照ボタン、アップロードボタン、および戻るボタンを含む。Webブラウザ400は、ファイル参照ボタンが押下されると、ファイル選択画面604を表示する。また、アップロードボタンが押下されると、参照ボタンより選択された画像データをmultipart/form-dataでエンコードしてWebアプリケーション501へアップロードする。Webブラウザ400はアップロード処理が完了するとメニュー画面602へ遷移する。また、戻るボタンが押下された場合、メニュー画面602へ遷移する。また、ブックマークボタンが押下された場合、Webブラウザ400は、ブックマークレットを処理してスキャンアップロード画面613へ遷移する。

40

【0025】

ファイル選択画面604は、MFP101のHDD214に格納された画像データを選択するためのファイル選択ダイアログを表示する操作画面である。ファイル選択ダイアログは、ファイルリスト、OKボタン、および戻るボタンを含む。Webブラウザ400は、ファイルリストよりファイル名が選択されてOKボタンが押下されると、操作画面60

50

3へ遷移する。そしてWebブラウザ400は、操作画面603中のファイル名入力フィールドにファイル選択ダイアログで選択されたファイル名を表示する。また、Webブラウザ400は、戻るボタンが押下されると、ファイル選択を中止して操作画面603へ遷移する。

【0026】

スキャンアップロード画面613は、スキャンして生成されたファイルをアップロードするための操作画面である。図6に示すように、ファイルアップロード画面603に“スキャンできます”メッセージが追加される。メッセージは、Webブラウザ400がブックマークレットを処理した結果表示される。また、Webブラウザ400は、ブックマークレットを処理した結果、ファイル参照ボタン、およびアップロードボタン押下した場合の処理内容も変更する。その結果、Webブラウザ400は、参照ボタンが押下されると、ファイル選択画面604を表示するのではなく、スキャン画面614を表示する。また、アップロードボタンが押下されると、Webアプリケーション501へファイルアップロードするのではなく、アップロードWebサービス412へアップロード要求を送信する。アップロードWebサービス412がアップロード完了を応答すると、メニュー画面602へ遷移する。また、戻るボタンが押下された場合は、メニュー画面602へ遷移する。

10

【0027】

スキャン画面614は、MFP101のスキナ211を制御して原稿を読み取りファイル生成するためのスキャンダイアログを表示する操作画面である。スキャンダイアログは、ファイル名入力フィールド、スキャン開始ボタン、および戻るボタンを含む。ファイル名入力フィールドには、原稿を読み取って生成するファイル名が入力される。ファイル名が入力されてスキャン開始ボタンが押下されると、Webブラウザ400は、スキャンWebサービス411へスキャン要求を送信する。スキャンWebサービス411がファイル生成完了を応答すると、スキャン完了画面615へ遷移する。また、戻るボタンが押下された場合は、スキャンアップロード画面613へ遷移する。

20

スキャン完了画面615は、原稿を読み取りファイル生成が完了したことを示す操作画面である。戻るボタンを押下された場合、スキャンアップロード画面613へ遷移する。

【0028】

図7を用いて、本実施形態における情報処理システムの、スキャンしてファイルをアップロードするまでの処理シーケンスを説明する。ステップS701からS708はWebアプリケーション501へログインしてファイルアップロード画面を表示する処理である。ステップS709からS711はブックマークレットを処理してWebアプリケーション501のコンテンツ（画面データ）を書き換える処理である。ステップS712からS716は、スキャンファイル生成する処理である。ステップS717からS726は、Webアプリケーション501へファイルをアップロードする処理である。

30

ステップS701では、Webブラウザ400が操作画面の取得を要求するHTTPリクエストをWebアプリケーション501へ送信する。

ステップS702では、Webアプリケーション501がログイン画面601を応答するHTTPレスポンスをWebブラウザ400へ送信する。

40

【0029】

続いて、ログイン画面601のログインボタンが押下されると、ステップS703の処理を開始する。ステップS703では、Webブラウザ400がログインを要求するHTTPリクエストをWebアプリケーション501へ送信する。

ステップS704では、Webアプリケーション501は、Webブラウザ400から送信された認証情報に基づいて認証処理を行う。認証処理に成功した場合、セッション情報を表すクッキーを生成する。そして、Webアプリケーション501はWebブラウザ400に対して、生成したクッキーの設定、およびメニュー画面602へのリダイレクトを指示するHTTPレスポンスを送信する。

【0030】

50

ステップS705では、Webブラウザ400がメニュー画面602を要求するHTTPリクエストをWebアプリケーション501へ送信する。このときWebブラウザ400は、ステップS704で応答されたクッキー設定指示に従ってクッキーをHTTPリクエストに設定する。

ステップS706では、Webアプリケーション501がメニュー画面602を応答するHTTPレスポンスをWebブラウザ420へ送信する。Webアプリケーション501は、Webブラウザ400からのHTTPリクエストに有効なクッキーが設定されていることを判断した上で、メニュー画面602を応答する。仮に有効なクッキーが設定されていない場合は、ステップS702と同じくログイン画面601を応答する。以降の処理においても、Webブラウザ400はクッキーを設定したHTTPリクエストをWebアプリケーション501へ送信する。

10

【0031】

続いて、メニュー画面602のファイルアップロード機能ボタンが押下されると、ステップS707の処理を開始する。ステップS707では、Webブラウザ400がファイルアップロード画面603を要求するHTTPリクエストをWebアプリケーション501へ送信する。

ステップS708では、Webアプリケーション501がファイルアップロード画面603を応答するHTTPレスポンスを送信する。図8に、ステップS708で送信されるHTTPレスポンスのメッセージボディの一例を示す。メッセージボディにはファイルアップロード画面がHTML形式で記述されている。図8のHTMLデータでは、送信ボタン“アップロード”が押下されると、フォーム構成要素をmultipart/form-dataでエンコードして、Webアプリケーション501の“UploadFile”が示す送信先へ送信することが表されている。また、送信されるフォーム構成要素は、フォルダ名“交通費精算／2011年2月”、およびファイル参照ボタンで指定されたファイルであることが表されている。

20

【0032】

以上が、Webアプリケーション501へログインしてファイルアップロード画面603を表示するまでの処理である。続いて、ブックマークよりブックマークレット402が選択されると、Webブラウザ400がJavaScriptエンジン402によってブックマークレットを処理して、ステップS709からステップS711の処理を実行する。図9に、ブックマークレットの一例を示す。ブックマークレットはJavaScriptで記述されている。図9に示すブックマークレットは、“https://192.168.0.1:8443/js/scan.js”の示すJavaScriptファイルをソースとするスクリプト要素（第2のスクリプト）をドキュメントに追加することを表している。なお、“192.168.0.1”はローカルホストを表す。また、ここでのドキュメントとは、Webブラウザ400が受信する画面データのことを意味する。

30

【0033】

ステップS709では、Webブラウザ400がHTTPサーバ410へJavaScriptファイルの取得を要求するHTTPリクエストを送信する。

ステップS710では、HTTPサーバ410がJavaScriptファイルを応答するHTTPレスポンスを送信する。

40

ステップS711では、Webブラウザ400がJavaScriptファイルをスクリプト要素としてドキュメントに追加する。HTTPサーバ410から取得するJavaScriptファイルには、JavaScriptファイルのロードが完了するとドキュメントを構成する各種要素を書き換える処理が定義されている。Webブラウザ400は、JavaScriptファイルを読み込むと、現在表示しているファイルアップロード画面603のドキュメント要素を書き換えてスキャンアップロード画面613を表示する。ドキュメント要素書き換え処理の詳細については図14を用いて後述する。Webブラウザ400は、ドキュメント要素書き換え処理の結果、ファイルアップロード画面のファイル参照ボタンが押下されると原稿をスキャンしてファイルを生成し、送信ボタンが押下

50

されるとファイルをアップロードすることを実現する。

スキャンアップロード画面613のファイル参照ボタンが押下されると、ステップS712の処理を開始する。ステップS712では、Webブラウザ400はスキャン画面614を表示する。

【0034】

続いて、スキャン画面614のスキャン開始ボタンが押下されると、ステップS713の処理を開始する。ステップS713では、Webブラウザ400がスキャン要求を生成する。図10にスキャン要求の一例を示す。スキャン要求はスキャンWebサービスへ送信されるHTTPリクエストである。図10の場合、ホスト“192.168.0.1:8443”および相対URI“/scan/post”が示すスキャンWebサービス411に対する要求が定義されている。要求内容は、スキャンサイズ自動、フルカラー、および片面読みの設定で原稿をスキャンし、“2月6日.pdf”というファイル名でファイル生成することとなっている。本発明の実施形態では、スキャンサイズ、カラー、片面両面読み、およびファイル名のみを指定するスキャン要求を示したが、本発明はこれに限定されない。その他のスキャン設定や、ファイルフォーマット設定、保存先設定を指定するスキャン要求であってもよい。また、本発明の実施形態では、スキャン画面614に示したようにファイル名のみユーザ指定可能としたが、本発明はこれに限定されない。各種スキャン設定、ファイルフォーマット設定、保存先設定を、スキャン画面614により入力可能としてもよい。

【0035】

ステップS714では、Webブラウザ400がスキャンWebサービス411へスキャン要求を送信する。

ステップS715では、スキャンWebサービス411がジョブ制御部420にジョブ制御にジョブ生成を指示する。

続いて、ジョブ制御部420よりジョブ完了が通知されると、ステップS716の処理を開始する。ステップS716では、スキャンWebサービス411がWebブラウザ400へスキャン完了を応答するHTTPリクエストを送信する。

【0036】

続いて、スキャン完了画面615の戻るボタンが押下されると、ステップS717の処理を開始する。ステップS717では、Webブラウザ411がファイルアップロード画面613を表示する。このとき、Webブラウザ400は、スキャン画面614に入力されたファイル名をスキャンアップロード画面613のファイル名入力フィールドに自動的に設定する。

【0037】

続いて、スキャンアップロード画面613のアップロードボタンが押下されると、ステップS718の処理を開始する。ステップS718では、Webブラウザ400がアップロード要求を生成する。図11にアップロード要求の一例を示す。アップロード要求はアップロードWebサービス412へ送信されるHTTPリクエストである。図11の場合、ホスト“192.168.0.1:8443”および相対URI“/upload/multipart”が示すアップロードWebサービスに対する要求が定義されている。要求内容は、ファイル名“2月6日.pdf”が示すファイル、およびフォルダ名“交通費精算/2011年2月”をmultipart/form-dataでエンコードして、URI“https://docs.xxx.yy/upload/UploadFile”の示すWebアプリケーション501へアップロードすることとなっている。また、Webアプリケーション501へアップロードするために必要なクッキーおよびファイルアップロード画面の文字セットを送信する。

【0038】

ステップS719では、Webブラウザ400がアップロード要求をアップロードWebサービス412へ送信する。Webブラウザ400がアップロード要求を生成して送信するまでの処理の詳細については図16を用いて後述する。

ステップS720では、アップロードWebサービス412がスキャンWebサービスに対してファイル名を指定してファイル取得を要求する。

ステップS721では、スキャンWebサービス411がスキャンWebサービスに対して指定されたファイル名のファイルを応答する。このとき応答するファイルは、ステップS715で生成したファイルである。

【0039】

本発明の実施形態では、アップロードWebサービスがスキャンWebサービスに対してスキャンして生成したファイルを要求するとしたが、本発明はこれに限定されない。スキャンWebサービス411がHTTPサーバ410へファイルを送信し、アップロードWebサービス412がHTTPサーバ410へファイルの取得を要求してもよい。またスキャンWebサービス411がスキャンして生成したファイルをアップロードWebサービス412へ送信してもよい。

【0040】

ステップS722では、アップロードWebサービス412がファイルアップロード要求を生成する。図12にファイルアップロード要求の一例を示す。ファイルアップロード要求はWebアプリケーション501へ送信されるmultipart/form-dataでエンコードされたHTTPリクエストである。図12に示す通り、HTTPリクエストは、Webブラウザ400より要求されたファイル名及びフォルダ名（ステップS718）、及びスキャンWebサービス411より応答されたファイル（ステップS721）のバイナリデータでデータパートを形成している。また、メッセージボディは、Webブラウザ400より要求された文字セット（ステップS721）でエンコードされている。また、リクエストヘッダには、Webブラウザ400より要求されたクッキー（ステップS721）が設定されている。

【0041】

ステップS723では、アップロードWebサービス412がステップS722で生成したファイルアップロード要求をWebアプリケーション501へ送信する。

ステップS724では、Webアプリケーション501はファイルアップロード完了を応答するHTTPレスポンスをアップロードWebサービス412へ送信する。図13にHTTPレスポンスの一例を示す。図13に示すように、HTTPレスポンスは“https://docs.xxy.yy/menu”が示すWebアプリケーション501のメニュー画面602へのリダイレクトを指示している。

ステップS725では、アップロードWebサービス412が、ステップS724でWebアプリケーション501より応答されたHTTPレスポンスをWebブラウザ400へ転送する。このHTTPレスポンスはステップS719で送信したHTTPリクエストの応答である。

ステップS726では、Webブラウザ400が、ステップS725で応答されたHTTPレスポンスの内容を解釈した結果、Webアプリケーション501に対してメニュー画面602取得を要求するHTTPリクエストを送信する。

ステップS727では、Webアプリケーション501はメニュー画面602をアップロードWebサービス412へ応答する。この結果、Webブラウザ400はメニュー画面602を表示することが可能となる。

【0042】

続いて、図14から16に示すフローチャートを用いて、Webブラウザ400のJavaScript処理の流れを説明する。図14から16に示す処理は全て、Webブラウザ400がJavaScriptエンジン401を制御してJavaScriptを処理することにより実現される。

【0043】

図14は、Webブラウザ400がJavaScriptを処理してドキュメント（画面データ）を書き換える処理の流れを表すフローチャートである。Webブラウザ400は、ブックマークレット402を処理した結果JavaScriptをドキュメント要素

10

20

30

40

50

にロード完了すると処理の流れを開始する。

ステップS1401では、ドキュメントを構成するform要素を全て取得する。続いて、ステップS1401で取得した全てのform要素について、ステップS1402からS1414までの処理を繰り返す。

ステップS1402では、form要素のエンコードタイプがmultipart/form-dataかどうかを判断する。エンコードタイプがmultipart/form-dataである場合はステップS1403の処理へ進む。エンコードタイプがmultipart/form-dataでない場合は、次のフォーム要素についてステップS1402からS1414までの処理を繰り返す。

ステップS1403ではform要素を構成する要素を取得する。具体的には、input要素、select要素、およびtextarea要素などが挙げられる。 10

【0044】

続いて、ステップS1403で取得した全てのform構成要素について、ステップS1404からS1411までの処理を繰り返す。ステップS1404では、form構成要素がinput要素のタイプfileである場合、ステップS1405の処理へ進む。そうでない場合、次のinput要素についてステップS1404からS1411までの処理を繰り返す。

ステップS1405ではform構成要素（タイプfile）のクローンノードを生成する。ステップS1406ではクローンノードのタイプをhiddenに設定する。

ステップS1407ではspan要素を生成する。ステップS1408では、ステップS1407で生成したspan要素の子要素として、input要素のタイプtextおよびタイプbuttonを追加する。このbutton要素には、ボタンが押下されると前述のスキャンダイアログを表示するようイベント処理が追加してある。そしてステップS1410でspan要素をform構成要素（input要素タイプfile）と置換する。ステップS1407からS1410までの処理の結果、ファイルアップロード画面603のファイル参照ボタンは、スキャンダイアログを表示するためのスキャンボタンへと変更される。また、ファイルアップロード画面603のファイル名入力フィールドは、JavaScriptよりファイル名を入力可能なテキストフィールドへと変更される。以下、このspan要素をダミーノードと呼ぶ。 20

ステップS1411では前述のクローンノードをform要素の子要素として追加する。図8に示すcheck関数のようにform構成要素の入力チェックなどをJavaScriptで処理している場合に備えて、元のform構成要素はファイルアップロード画面から削除せずに追加しておく必要がある。 30

【0045】

以上、ステップS1404からS1411までの処理を繰り返すことによって、ファイルアップロード画面603を構成するファイル名入力フィールドおよびファイル参照ボタンの書き換えが実現される。

【0046】

ステップS1412では、form構成要素の書き換え処理を行ったか否かを判断する。具体的には、ファイル名入力フィールドおよびファイル参照ボタンの書き換えがこれに該当する。ファイル名入力フィールドおよびファイル参照ボタンの書き換え処理を行っている場合、ステップS1413へ進む。そうでない場合（具体的にはform構成要素がinput要素タイプfileではなかった場合である）、次のform構成要素についてステップS1404からS1414の処理を繰り返す。 40

ステップS1413では、form構成要素のaction属性の値をコピーして別属性として設定する。ステップS1414では、action属性の値を、ファイルアップロード要求を生成する関数に変更する。この処理の結果、ファイルアップロード画面603のアップロードボタンを押下した場合に、JavaScriptを処理して生成されたアップロード要求をアップロードWebサービスに送信するよう動作することになる。なお、処理を行う前は、Webブラウザ400がmultipart/form-data 50

エンコードでWebアプリケーション501へファイルアップロードするように動作するものである。ステップS1413で元のaction属性の値を別属性として退避させたのは、ファイルアップロード要求のリクエストヘッダ“X-UPLOAD-ACTION”にWebアプリケーション501の送信先を設定するために必要とするからである。

【0047】

以上、ステップS1402からS1414までの処理を繰り返すことによって、ファイルアップロード画面603のアップロード動作の書き換えが実現される。

ステップS1416では、スキャンアップロード画面613中にメッセージを表示するためのspan要素を生成する。そして、ステップS1417で、span要素を所定の位置を指定してドキュメントに追加する。この結果、“スキャンできます”メッセージがスキャンアップロード画面613へ表示される。

10

【0048】

図15は、Webブラウザ400がJavaScriptを処理してスキャンファイルを生成する処理の流れを表すフローチャートである。Webブラウザ400は、スキャン画面614よりスキャン開始ボタンが押下されると処理の流れを開始する。

ステップS1501では、スキャン画面614のファイル名入力フィールドにファイル名が入力されているか否かを判断する。スキャン画面614にファイル名が入力されている場合、ステップS1502の処理へ進む。そうでない場合、ステップS1512の処理へ進む。ステップS1512では、ファイル名が入力されていない警告メッセージを表示して処理を終了する。

20

【0049】

ステップS1502では、図10に示したスキャン要求を生成する。そして、ステップS1503で、スキャンWebサービス411にスキャン要求を送信する。

ステップS1504では、スキャンWebサービスよりスキャン完了応答を受信したか否かを判断する。スキャン完了応答を受信した場合、ステップS1505の処理へ進む。スキャン完了応答を受信していない場合、ステップS1504の処理を繰り返す。

ステップS1505では、スキャン画面614に入力されたファイル名をダミーノードのファイル名に設定する。ステップS1505では、スキャン画面614に入力されたファイル名をクローンノードに設定して処理を終了する。この結果、スキャンアップロード画面613にはファイル名が表示される。また、アップロードボタン押下時には、図8に示したcheck関数によるファイル名入力チェックも正常に終了し、action属性に設定したアップロード要求生成関数が呼び出される。

30

【0050】

図16は、Webブラウザ400がJavaScriptを処理してアップロード要求を生成する処理の流れを表すフローチャートである。Webブラウザ400は、スキャンアップロード画面613のアップロードボタンが押下された結果、対応するform要素のアップロード要求生成関数が呼び出されると処理の流れを開始する。

ステップS1601では、アップロード要求のHTTPヘッダ(X-UPLOAD-ACTION、X-UPLOAD-CHARSET、およびX-UPLOAD-COOKIE)に、Webアプリケーション501の送信先、文字セット、クッキーを設定する。

40

ステップS1602では、現在のform要素を構成する要素を取得する。具体的には、input要素、select要素、およびtextarea要素などが挙げられる。

【0051】

続いて、ステップS1602で取得した全てのform構成要素について、ステップS1603からS1606までの処理を繰り返す。ステップS1603では、form構成要素がダミーノードの子要素か否かを判断する。ダミーノードの子要素ではないと判断した場合、ステップS1604の処理へ進む。ダミーノードの子要素であると判断した場合、次のform構成要素についてステップS1603からS1606の処理を繰り返す。

【0052】

ステップS1604では、form構成要素にname属性が設定されているか否かを

50

判断する。name属性が設定されている場合、ステップS1605の処理へ進む。設定されていない場合、次のform構成要素についてステップS1603からS1606の処理を繰り返す。

ステップS1605では、form構成要素がクローンノードであるか否かを判断する。クローンノードである場合、ステップS1606の処理へ進む。クローンノードでない場合、ステップS1616へ進む。

ステップS1606では、form構成要素のname属性の値、value属性の値、およびvalue属性の値から判断されるcontent-typeを対にして、メッセージボディに追加する。

【0053】

一方、ステップS1616では、form構成要素がinput要素のタイプsubmitまたはbuttonであるかを判断する。input要素のタイプsubmitまたはbuttonである場合、次のform構成要素についてステップS1603からS1606の処理を繰り返す。その他のタイプである場合、ステップS1617の処理へ進む。ステップS1617では、form構成要素のname属性の値、およびvalue属性の値を対にして、メッセージボディに追加する。

【0054】

ステップS1607では、押下されているsubmitボタンにname属性が設定されているか否かを判断する。name属性が設定されていない場合、ステップS1609に進み、name属性が設定されている場合、ステップS1608の処理へ進む。ステップS1608では、submitボタンのname属性の値、およびvalue属性の値を対にして、メッセージボディに追加する。

【0055】

以上の処理によって、アップロードWebサービス412が、Webアプリケーション501へファイルアップロードするために必要な情報が、アップロード要求に設定される。最後に、ステップS1609で、生成したアップロード要求をアップロードWebサービス412へ送信して処理を終了する。

【0056】

このように本実施形態では、Webブラウザ400が、ブックマークレットによりWebアプリケーション501を拡張する（Webアプリケーション501から受信する画面データを書き換える）。これにより、ファイル参照ボタンが押下されると原稿をスキャンしてファイルを生成し、送信ボタンが押下されるとファイルをアップロードすることを可能とする。

【0057】

なお、第1の実施形態では、アップロードWebサービス412をMF P101が提供する構成としたが、本発明はこれに限定されない。HTTPサーバを搭載するその他のサーバでアップロードWebサービス412を提供する構成であってもよい。この構成の場合、スキャンWebサービス411は、スキャンして生成したファイルをアップロードWebサービスへ送信する、もしくは、HTTPサーバを搭載するその他のサーバへ送信する構成が好ましい。

【0058】

また、第1の実施形態では、ブックマークレットが取得するJavaScriptファイルをMF P101が提供する構成としたが、本発明はこれに限定されない。HTTPサーバを搭載するその他のサーバでJavaScriptファイルを提供する構成であってもよい。

【0059】

また、第1の実施形態では、Webブラウザ400をMF P101に搭載する構成としたが、本発明はこれに限定されない。タブレットPCやスマートフォンなどの情報装置に搭載されるWebブラウザを使用したシステム構成であってもよい。

【0060】

また、第１の実施形態では、アップロードWebサービス４１２をMFP１０１が提供する構成としたが、Webブラウザ４００の送信機能をJavaScriptより制御することによってファイルアップロードを行う構成であってもよい。この場合、アップロードWebサービスはシステム構成に含まれない。この構成の場合、スキャンWebサービス４１１は、スキャンして生成したファイルを、HTTPサーバを搭載するその他のサーバへ送信する構成が好ましい。

【００６１】

（他の実施形態）

本発明の目的は、以下の処理を実行することによっても達成される。即ち、上述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システム或いは装置に供給し、そのシステム或いは装置のコンピュータ（またはCPUやMPU等）が記憶媒体に格納されたプログラムコードを読み出す処理である。

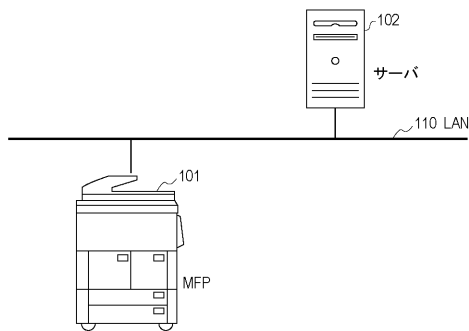
この場合、記憶媒体から読み出されたプログラムコード自体が前述した実施の形態の機能を実現することになり、そのプログラムコード及び該プログラムコードを記憶した記憶媒体は本発明を構成することになる。

【符号の説明】

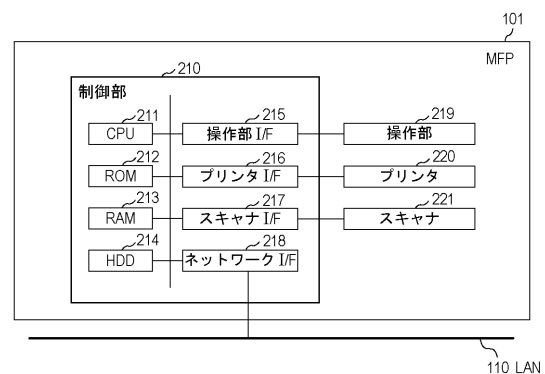
【００６２】

- ４００ Webブラウザ
- ４１０ HTTPサーバ
- ４１１ スキャンWebサービス
- ４１２ アップロードWebサービス
- ５０１ Webアプリケーション

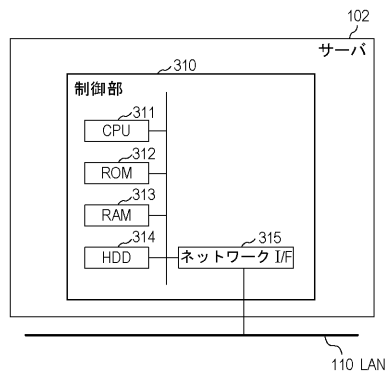
【図１】



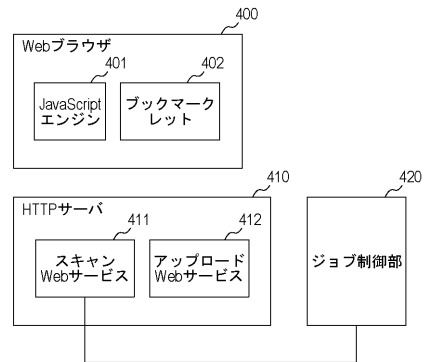
【図２】



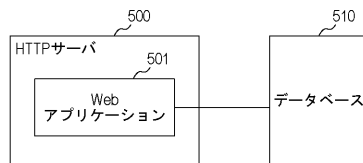
【図 3】



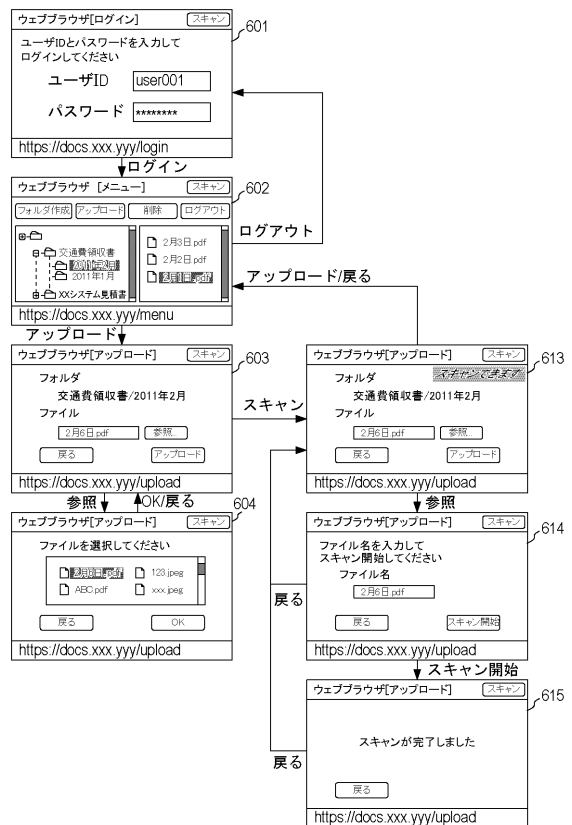
【図 4】



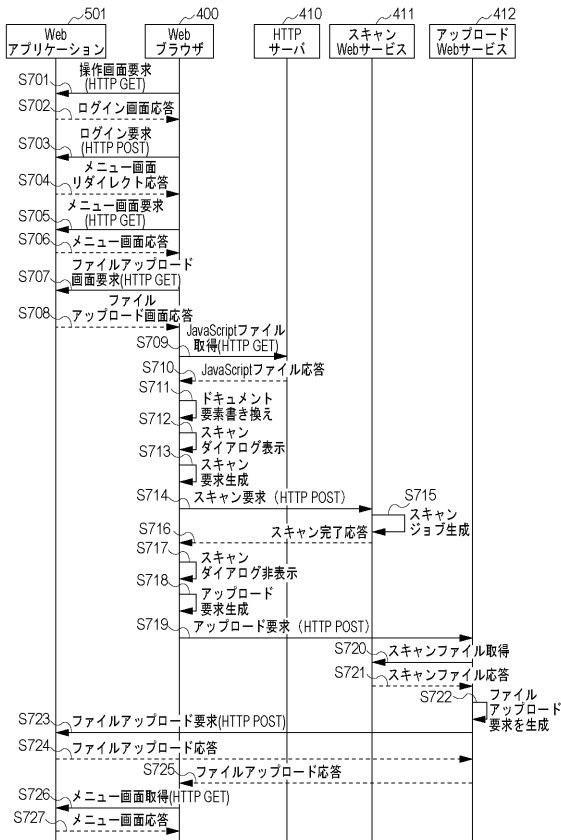
【図 5】



【図 6】



【図 7】



【図 8】

```

<html>
<head>
<title>ファイルアップロード</title>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function check(){
if(document.form.file.value==""){
alert("ファイル名が入力されていません。");
return false;
}else{
return true;
}
}
</script>
</head>
<body>
<form method="POST" name="form" enctype="multipart/form-data" onsubmit="return check()" action="UploadFile">
<p>フォルダ</p>
<div>交通費精算2011年2月</div>
<p>ファイル</p>
<div>
<input type="file" name="file" size="32">
</div>
<input type="hidden" name="folder_name" value="交通費精算2011年2月">
<p>
<input type="button" onclick="history.back()" value="戻る">
<input type="submit" value="アップロード">
</p>
</div>
</body>
</html>
  
```

【図 9】

```

javascript:if(document.getElementById("scan_js")==undefined)
(function){
var s=document.createElement("script");
s.src="https://192.168.0.1:8443/js/scan.js";
s.id="scan_js";
document.body.appendChild(s)
}
})();
  
```

【図 10】

```

POST /scan/post HTTP/1.1
Host: 192.168.0.1:8443
Content-Length: 97
Content-Type: text/plain; charset=UTF-8

{"size":"auto", "color":"auto", "plex":"simplex", "file":"2月6日.pdf"}
  
```

【図 1 1】

```

POST /upload/multipart HTTP/1.1
Host: 132.168.0.1:8443
X-UPLOAD-ACTION: https://docs.xxx.yyy/upload/UploadFile
X-UPLOAD-CHARSET: UTF-8
X-UPLOAD-COOKIE: sessionId=75dbfc6bd0b4e6e35278d901122693c5
{"file": {"filename": "2月6日.pdf", "contentType": "application/pdf"}, "folder_name": "交通費精算(2011年2月)"}

```

【図 1 2】

```

POST /UploadFile HTTP/1.1
Host: docs.xxx.yyy
User-Agent: Mozilla/5.0 (MFP; IR-S/1.3; like Gecko)
Cookie: sessionId=75dbfc6bd0b4e6e35278d901122693c5

Content-Type: multipart/form-data; boundary=-----309001771318958
Content-Length: 10490

-----309001771318958
Content-Disposition: form-data; name="file"; filename="2月6日.pdf"
Content-Type: application/pdf

%PDF-1.4
...
%%EOF
-----309001771318958
Content-Disposition: form-data; name="folder"

交通費精算(2011年2月)
-----309001771318958--

```

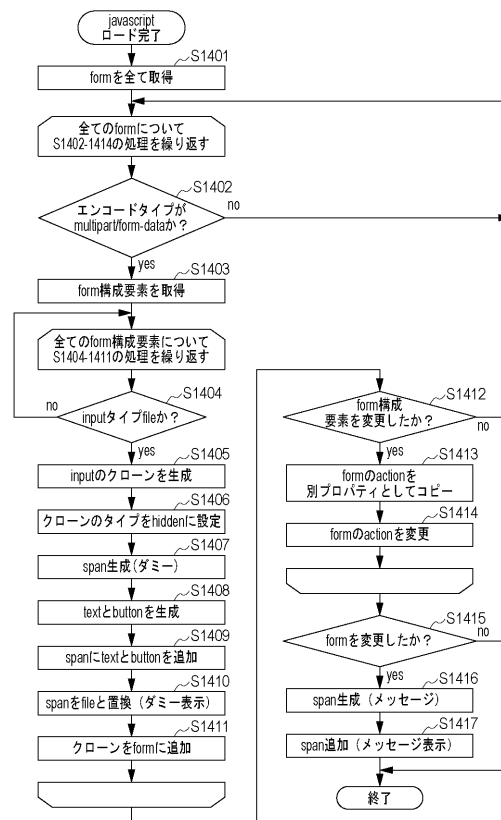
【図 1 3】

```

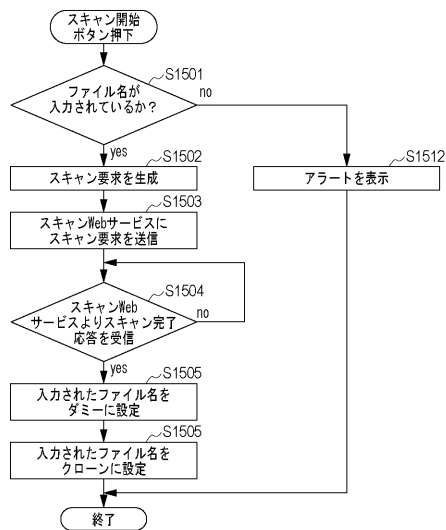
HTTP/1.1 303 See Other
Server: Apache-Coyote/1.1
Date: Sun, 6 Feb 2011 08:27:28 GMT
Location: https://docs.xxx.yyy/menu
Transfer-Encoding: chunked

```

【図 1 4】



【図 15】



【図 16】

