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

特許第6335527号
(P6335527)

(45)発行日　平成30年5月30日(2018.5.30)　(24)登録日　平成30年5月11日(2018.5.11)

(51)Int.Cl.　F I
G06F 12/00 (2006.01)　G06F 12/00 ５２０Ｅ
G06F 3/12 (2006.01)　G06F 3/12 ３３４

請求項の数 8　(全 20 頁)

(21)出願番号　特願2014-13570(P2014-13570)
(22)出願日　平成26年1月28日(2014.1.28)
(65)公開番号　特開2015-141523(P2015-141523A)
(43)公開日　平成27年8月3日(2015.8.3)
審査請求日　平成29年1月20日(2017.1.20)

(73)特許権者　000001007
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　100114775
弁理士　高岡　亮一
(72)発明者　住谷　茉莉
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

審査官　田中　啓介

最終頁に続く

(54)【発明の名称】システム、システムの制御方法およびコンピュータプログラム

(57)【特許請求の範囲】
【請求項１】
処理対象のファイルを受信し、データ処理部により当該ファイルのデータ処理を行う処理手段と、
受信するファイルに基づいて前記データ処理部を複数起動する起動手段と、
前記ファイルを一意に識別するための識別情報を前記ファイルから算出する算出手段と、
前記ファイルに対して前記データ処理を行うことでエラーが発生した場合、算出された前記ファイルの識別情報を登録する登録手段と、
新たに受信したファイルから算出された識別情報が、登録されている前記識別情報に含まれているか否かを判断する判断手段と、を備え、
前記処理手段は、
前記データ処理部が少なくとも一つ停止したことを条件として、前記判断手段により新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断して、前記識別情報に含まれていない場合にデータ処理を行い、
少なくとも一つの前記データ処理部が停止後に、他のデータ処理部が新たに受信したファイルのデータ処理を正常に終了したことを条件として、前記判断手段により新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断せずにデータ処理を行う

ことを特徴とするシステム。
【請求項２】
前記処理手段は、前記データ処理部が少なくとも一つ停止したことを条件として、前記判断手段により新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断して、前記識別情報に含まれている場合に当該ファイルを不正ファイルであると見なしてデータ処理を行わない
ことを特徴とする請求項１に記載のシステム。
【請求項３】
前記識別情報はハッシュ値である
ことを特徴とする請求項１または２に記載のシステム。
【請求項４】
前記登録手段は、予め定められた規定時間内に前記データ処理部が前記データ処理を完了できない場合、処理中の前記ファイルの識別情報を登録する
ことを特徴とする請求項１乃至３のいずれか一項に記載のシステム。
【請求項５】
前記処理手段は、複数のフィルタから構成されるフィルタパイプラインを用いて、前記受信したファイルを印刷データに変換処理する
ことを特徴とする請求項１乃至４のいずれか一項に記載のシステム。
【請求項６】
前記登録手段は、予め定められた規定時間内に前記データ処理部が前記データ処理を完了できない場合、処理中の前記ファイルの識別情報を登録し、前記規定時間は、前記フィルタの各々に設定されている
ことを特徴とする請求項５に記載のシステム。
【請求項７】
システムの制御方法であって、
システムが備える処理手段が、処理対象のファイルを受信し、データ処理部により当該ファイルのデータ処理を行う処理工程と、
システムが備える起動手段が、受信するファイルに基づいて前記データ処理部を複数起動する起動工程と、
システムが備える算出手段が、前記ファイルを一意に識別するための識別情報を前記ファイルから算出する算出工程と、
システムが備える登録手段が、前記ファイルに対して前記データ処理を行うことでエラーが発生した場合、算出された前記ファイルの識別情報を登録する登録工程と、
システムが備える判断手段が、新たに受信したファイルから算出された識別情報が、登録されている前記識別情報に含まれているか否かを判断する判断工程と、を有し、
前記処理工程では、
前記データ処理部が少なくとも一つ停止したことを条件として、前記判断工程において新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断して、前記識別情報に含まれていない場合にデータ処理を行い、
少なくとも一つの前記データ処理部が停止後に、他のデータ処理部が新たに受信したファイルのデータ処理を正常に終了したことを条件として、前記判断工程において新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断せずにデータ処理を行う
ことを特徴とするシステムの制御方法。
【請求項８】
請求項７に記載の制御方法をコンピュータにより実行させることを特徴とするコンピュータプログラム。
【発明の詳細な説明】
【技術分野】

【０００１】
本明細書は、投入された不正データを検出するシステム、システムの制御方法およびコンピュータプログラムに関する。
【背景技術】
【０００２】
サービス利用者から文書データや画像データなどの入力データを受け取り、そのデータを印刷データに変換して印刷装置に出力する、サーバ上のデータ変換サービスが提供されている。このデータ変換サービスは、利用者からサービス内でエラーを発生させる不正データを受け取る可能性がある。不正データには、データ変換処理中にシステムの異常終了を引き起こすものと、処理中に無限ループに入るなどしてタイムアウトを発生させるものの２種類がある。異常終了とタイムアウトは予期せぬ不正データによって引き起こされるため回避するのが難しい。しかし、システム内でエラーが生じると、システムはその後復旧するまでの間、次の入力データを受け取ることができない。不正データの連続投入により故意にシステムが攻撃されると、その他のサービス利用者がデータ変換サービスを利用できなくなる。このような事態を回避するために、サーバシステムは安全性を確保することが求められる。
【０００３】
ここで、システムに安全性を持たせる方法として予備のシステムを持つ方法を挙げる。データ変換部が１つしかないシステムでは、処理待ちのデータ変換要求が滞留することでシステム全体の処理能力が落ちる。この課題に対し、特許文献１は、データ変換部を複数並べ、各データ変換部に均等に処理待ちのデータ変換要求を割り振ることでシステム全体の処理能力を向上させるシステムを提案している。上記サーバ上のデータ変換サービスにおいても、データ変換部を多重化し、システムに冗長性を持たせることでユーザからの不正なデータの投入に対応している。
【先行技術文献】
【特許文献】
【０００４】
【特許文献１】特開平５－２８４１７８号公報
【発明の概要】
【発明が解決しようとする課題】
【０００５】
一般的に、ユーザは入力データ投入後、一定時間以内にシステムから返答がないと同じデータを続けてシステムに投入する傾向がある。ユーザが投入したデータが不正データであった場合、ユーザは多重化されたシステムの一部でエラーを発生させたことを知らずに不正データをシステムに連続投入することになる。上記特許文献１のデータ変換サービスでは、不正データへの対応としてシステムの多重化を行っていた。しかしデータ変換サービスは、悪意のないユーザの不正データの連続投入により、サービスを専有されてしまうことがあった。そしてその間、他のサービス利用者はサービスを一時的に利用できないおそれがある。
【０００６】
本発明は、未知の不正データを受信した場合に、次に同じデータを受け取った際にシステムで同じエラーの発生を防ぐことができ、データ処理の停止を防止するシステム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】
本発明の一実施形態のシステムは、処理対象のファイルを受信し、データ処理部により当該ファイルのデータ処理を行う処理手段と、受信するファイルに基づいて前記データ処理部を複数起動する起動手段と、前記ファイルを一意に識別するための識別情報を前記ファイルから算出する算出手段と、前記ファイルに対して前記データ処理を行うことでエラーが発生した場合、算出された前記ファイルの識別情報を登録する登録手段と、新たに受

信したファイルから算出された識別情報が、登録されている前記識別情報に含まれているか否かを判断する判断手段と、を備える。前記処理手段は、前記データ処理部が少なくとも一つ停止したことを条件として、前記判断手段により新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断して、前記識別情報に含まれていない場合にデータ処理を行い、少なくとも一つの前記データ処理部が停止後に、他のデータ処理部が新たに受信したファイルのデータ処理を正常に終了したことを条件として、前記判断手段により新たに受信した前記ファイルから算出された前記識別情報が、登録されている前記識別情報に含まれているか否かを判断せずにデータ処理を行う。

【発明の効果】

【０００８】

本発明のシステムによれば、未知の不正データを受信した場合に、次に同じデータを受け取った際にシステムで同じエラーの発生を防ぐことができ、データ処理の停止を防止することが可能となる。従って、未知の同一不正データの連続投入により、特定のユーザがシステム全体を専有することを防止することができる。

【図面の簡単な説明】

【０００９】

【図１】本発明のシステムの全体構成例を示す図である。
【図２】サーバＰＣのハードウエア構成例を示す図である。
【図３】データ処理システムのソフトウエア構成例を示す図である。
【図４】データ処理システムの処理を示す処理フローである。
【図５】警戒態勢が導入された際の処理を示す処理フローである。
【図６】データ処理システムが警戒態勢に入る仕組みを説明する図である。
【図７】データ処理システムが警戒態勢を解除する仕組みを説明する図である。
【図８】印刷データ変換システムのソフトウエア構成例を示す図である。
【図９】印刷データ変換部の動作に関連するコンポーネント群を説明する図である。
【図１０】印刷データ変換システムのソフトウエア構成例を示す図である。
【図１１】印刷データ変換システムの処理の概要を示す処理フローである。
【図１２】ジョブ実行処理を示す処理フローである。
【図１３】エラー処理を示す処理フローである。
【図１４】ブラックリスト検索処理を示す処理フローである。
【図１５】フィルタの処理の概要を示す処理フローである。
【図１６】フィルタのデータ変換処理を示す処理フローである。
【図１７】フィルタのタイムアウト処理を示す処理フローである。
【図１８】印刷データ変換システムのブラックリストの内容を示す図である。
【図１９】警戒態勢時のジョブ実行処理を示す処理フローである。
【図２０】警戒態勢時のエラー処理を示す処理フローである。

【発明を実施するための形態】

【００１０】

（実施例１）

図１は、本発明の一実施形態のシステムの構成例を示す。図１に示すサーバＰＣ８０が、本実施形態のデータ処理システムとして機能する。データ処理システムは、依頼者から入力データと処理依頼を受け取ると処理依頼に従って入力データにデータ処理を行い、依頼者に処理後のデータを出力するサービスを提供する。なお、本明細書におけるデータ処理とはデータ形式の変換や画像を編集する処理などを意味する。図１に示す例では、データ処理を依頼する依頼者５０は、処理要求を送出する機器として、パーソナルコンピュータ２０、タブレットＰＣ３０や携帯端末４０などの機器を利用する。依頼者側の機器とサーバＰＣは、インターネットなど任意のネットワーク１０を介して通信可能である。図１では、プリンタ６０、ロードバランサ７０、サーバＰＣ９０が示されているが、本実施例では必須の構成要素ではないため実施例３以降にて後述する。

【００１１】
図２は、サーバＰＣ８０のハードウエア構成例を示す図である。データ処理システムはＯＳ（オペレーティングシステム）上で動作するアプリケーションプログラムであり、ＨＤＤ２０２あるいはＲＯＭ２０３に格納されている。ＣＰＵ２０５がＯＳとアプリケーションプログラムをＨＤＤ２０２またはＲＯＭ２０３から読み出してＲＡＭ２０４にロードし、実行することで様々な処理が実現される。処理結果はファイルとしてＨＤＤ２０２に格納され、あるいはデータとしてＲＡＭ２０４に記憶される。なお、ＣＰＵ、ＨＤＤ、ＲＯＭ、ＲＡＭはそれぞれ、Ｃｅｎｔｒａｌ Ｐｒｏｃｅｓｓｉｎｇ Ｕｎｉｔ、Ｈａｒｄ Ｄｉｓｋ Ｄｒｉｖｅ、Ｒｅａｄ Ｏｎｌｙ Ｍｅｍｏｒｙ、Ｒａｎｄｏｍ Ａｃｃｅｓｓ Ｍｅｍｏｒｙの略称である。

【００１２】
アプリケーションプログラムは、コンピュータに接続されている入力装置２０７から使用者の入力、各種センサの読み取り値を取得する。さらに出力装置２０６に対して情報を出力し、処理結果を表示する。さらに、通信装置２０８を介してネットワークに接続された他のコンピュータや装置と通信を行う。これらのハードウエアはコントロールバス１００１で互いに接続されていてアプリケーションプログラムから操作できるように構成されている。

【００１３】
図３は、サーバＰＣ８０のデータ処理システムに係るソフトウエア構成例を示す図である。データ処理システムは、データ処理部１００、リスト管理部１０４、ブラックリスト１０５を備える。また、データ処理部１００は、入力受付部１０１、制御部１０２、処理部１０３を備える。これらのコンポーネントはバス１０６でつながっている。コンポーネント間のデータの受け渡しはバス１０６を通して行われる。

【００１４】
入力受付部１０１は、依頼者５０からデータ処理の要求などを受け付ける。制御部１０２は、入力受付部１０１から入力データと処理依頼を受信すると、必要な処理を解析する。処理部１０３は、制御部１０２が解析した内容に従ってデータ処理を行う。

【００１５】
ここで、処理部１０３が不正データに対し処理を行うと、処理部１０３でエラーが発生し、処理が強制的に中断される。また、エラー発生後にエラー処理を行うため、その間は次の入力データを受け付けることができない。そこで、データ処理システムは、受信したファイル数などに応じてデータ処理部１００を複数起動し、多重化することでエラーの原因となる不正データの連続投入に対処する。

【００１６】
ブラックリスト１０５は、不正データの識別情報を保持している。リスト管理部１０４は、データ処理部共通のブラックリスト１０５を管理する。リスト管理部１０４は、制御部１０２から問合せを受付けてブラックリスト１０５を参照し、処理部１０３が不正データを受付けない仕組みを提供する。

【００１７】
以下、図４を用いて、実施例１のデータ処理システムが行う処理を説明する。データ処理部１００は、入力受付部１０１が依頼者５０から入力データと処理依頼を受け取ると、入力データを制御部１０２に渡した後、Ｓ００１の処理を開始する。Ｓ００１では、制御部１０２がリスト管理部１０４に対し、ブラックリスト１０５に入力データの識別情報と一致するものが存在するかを問い合わせる。入力データ識別情報は、データそのものでなくともよく、データのＩＤやハッシュ値など、データが一意に識別可能なものであればよい。

【００１８】
リスト管理部１０４は、入力データ識別情報を抽出し、一致する識別情報がブラックリスト１０５に登録されているかを判定し、結果を制御部１０２に返す。判定の結果、入力データ識別情報と一致する情報がブラックリスト１０５に存在する場合、制御部１０２は

入力データが不正であり処理対象外であるとして、Ｓ００６で入力データに関するエラー通知を行い、処理を終了する。エラー通知処理では、制御部１０２から入力受付部１０１を介して依頼者５０にエラーを通知する。エラーの通知はメールによりエラーメッセージを送信するか又はＷｅｂページにエラーメッセージを出力すること等により行われる。エラーメッセージには入力データのデータ処理が正常に行われなかったことを示すメッセージが含まれる。

【００１９】

一方、入力データ識別情報と一致する情報がブラックリスト１０５に存在しない場合、Ｓ００２に処理が進み、制御部１０２は処理部１０３に入力データと処理依頼を転送し、通常処理が行われる。通常処理では、処理部１０３は依頼された処理を実行し、処理後のデータを制御部１０２に返す。Ｓ００３で、制御部１０２は、通常のデータ処理中にエラーが発生しているか否かを判断する。エラーが発生していない場合、制御部１０２は、受け取ったデータを入力受付部１０１に渡し、入力受付部１０１は依頼元にデータを出力して正常終了する。

【００２０】

通常処理中にエラーが発生した場合、Ｓ００４に処理が進む。Ｓ００４では、制御部１０２は、エラーを引き起こした入力データを不正と見なし、ブラックリスト１０５に登録（Ｓ００４）するようにリスト管理部１０４に依頼する。制御部１０２は、不正データの識別情報がブラックリスト１０５に登録されると、データ処理を終了する（Ｓ００５）。データ処理部１００が正常終了できない場合は、エラー発生後、ログ出力や依頼元へのエラー通知等のエラー処理を行う。

【００２１】

データ処理部１００は、未知の不正データを受け取ると、Ｓ００１の処理でブラックリスト１０５を照合しても不正データであることを判断できない。しかし、未知の不正データを最初に受けたデータ処理部１００が、Ｓ００４の処理で未知の不正データをブラックリスト１０５に登録し、リストを更新する。従って、それ以後はＳ００１の処理で過去に不正と判定された入力データをシステムから遮断することができるようになる。

【００２２】

同じ不正データが連続でシステムに投入されたとき、一つ目のデータ処理部１００により不正データの識別情報が新たにブラックリスト１０５に追加されると、それ以降システムに投入される同一不正データはシステムから弾かれるようになる。従って、エラーが発生したファイルを処理中の処理手段以外の起動中の他の処理手段は、エラーが発生した当該ファイルと同一の識別情報を有する別のファイルを受信した場合に当該ファイルを不正ファイルとして処理しない。以上、本発明のデータ処理システムによれば、未知の不正データを受信した場合に、次に同じデータを受け取った際にシステムで同じエラーの発生を防ぐことができ、データ処理の停止を防止することができる。

【００２３】

（実施例２）

実施例１のデータ処理システムでは、データ処理部１００は入力データを受け取るたびに入力データ識別情報を算出し、それ以前に不正と判定されたデータの識別情報が登録されたリストに一致する情報がないか検索を行う。しかし、例えば入力データからデータの識別情報としてハッシュ値を使用する場合、ハッシュ値算出の処理は時間がかかる。そのため、入力データのサイズが大きいほどシステムの処理速度が低下する。処理速度の低下はサービス利用者による入力データの再投入を増加させ、入力データの連続投入を引き起こす原因にもなる。そのため、不正データの連続投入に備えつつ、処理速度の低下を抑える手段を構築する必要がある。

【００２４】

図５を用いて、システムの状態によって処理を変更するデータ処理システムの処理について説明する。データ処理システムの構成は実施例１と同じである。また、図４で説明したステップと同じ処理については、既に説明したステップの番号と同一の番号を付与し、

特に断りが無い限り説明は省略する。

【００２５】

Ｓ８００で処理が開始し、Ｓ８０１で制御部１０２は、システムの状態の判定処理を行う。制御部１０２は、判定結果によってシステムの処理内容を一部変更する。システムの状態は、警戒態勢（警戒モード）と通常状態の２種類がある。本実施例では、不正データの投入によりデータ処理部が１つ以上停止するとシステムの状態は警戒態勢となり、警戒態勢時にデータ処理部での処理が規定回数連続で正常に終了すると警戒態勢は解除されるものとする。また、データ処理システムの状態の管理はリスト管理部１０４が担う。

【００２６】

制御部１０２は、Ｓ８０１で警戒態勢であると判断したときのみ実施例１と同様にＳ００１の判定処理を行う。警戒態勢でない場合はＳ００１の判定処理を行わずにＳ００２の通常処理に移る。つまり、データ処理システムが正常な入力データを受け取っている間は受け取った入力データを審査することなく通常処理を行う。

【００２７】

Ｓ００３で、制御部１０２は、エラーが発生したか否かを判断する。不正データが投入されエラーが発生したときは、Ｓ８０３に処理が進み、警報を発令する。つまり、データ処理システムの状態は警戒モードに移行するようにモード切替を行う。これにより、データ処理システムは、不正データに対する警戒を強める。警戒態勢時には、制御部１０２は入力データの審査を行い、審査に通過したデータのみを通常処理の対象とする。

【００２８】

Ｓ００３で、通常処理中にエラーが発生した場合、Ｓ８０３に処理は進み、制御部１０２は警報を発令する。警報発令の例を図６に示す。図６に示すように、制御部１０２は、リスト管理部１０４にアラートメッセージ３０１を送る。リスト管理部１０４は、アラートメッセージ３０１を受け取ると警戒フラグ３０２を立てる。つまり、連続して受信するファイルを分散処理する冗長化されたデータ処理システムにおいて、１つ以上のデータ処理部でエラーが発生すると、データ処理システムは警戒態勢にシフトする。

【００２９】

警戒フラグ３０２が立っている間にデータを投入されたデータ処理部１００では、Ｓ８０１の処理で警戒態勢であると判断され、Ｓ００１の入力データの審査を入力があるたびに行う。Ｓ００２で、制御部１０２が、通常処理は正常に終了したと判断した場合、Ｓ８０２に処理は進み、リスト管理部１０４に対して正常終了報告をする。

【００３０】

図７は、リスト管理部１０４が正常終了報告を受け、警戒態勢が解除される例を示す。正常終了報告の処理では、制御部１０２がリスト管理部１０４にコンプリーションメッセージ３０３を送信する。コンプリーションメッセージ３０３は、警戒フラグ３０２が立っている間にデータを投入され正常終了した場合のみ送信することにしてもよい。警戒フラグ３０２が立っている間に、コンプリーションメッセージ３０３が規定回数連続でリスト管理部１０４に届いたとき、リスト管理部１０４は警戒フラグ３０２を下ろし、警戒態勢を解除する。つまり不正データによるエラー発生後、データ処理部１００に不正なデータが規定回数続けて投入されなかったとき、不正データ投入の可能性が下がったとして、データ処理システムは警戒態勢を解除する。

【００３１】

以上のように、実施例２のデータ処理システムは、不正データの投入により１つ以上のプロセスでエラーが発生したときにシステムの状態を警戒態勢に移行し、警戒態勢時にプロセスが規定回数連続で正常に終了したときに警戒態勢を解除する。本システムによれば、安全性を確保する必要がある場合にのみ入力データを解析する構成とし、処理速度の低下を抑えつつ、不正データの連続投入による特定ユーザの意図的なサービス専有を回避することができる。

【００３２】

（実施例３）

実施例１、２の構成を持ったデータ処理システムの一例として、印刷データの変換を行う、サーバ上の印刷データ変換サービスについて図１を再び参照して説明する。実施例３のシステムは、印刷変換サービスを要求する依頼者側の装置２０乃至４０、プリンタ６０、サービス提供側の装置７０乃至９０から構成される。図１に示すサーバＰＣ８０が、本実施形態の印刷データ変換システムとして機能する。印刷データの変換を依頼する依頼者５０は、印刷データの変換要求を送信する機器として、パーソナルコンピュータ２０、タブレットＰＣ３０、携帯端末４０などの機器を利用する。依頼者５０は、これらの機器の内部で動作しているアプリケーション（不図示）を用いて、ネットワーク１０上に存在する印刷データ変換部３３０（図８を参照して後述）に対して印刷データの変換を依頼する。

【００３３】

印刷データ変換部３３０は、サーバＰＣ８０上で周知のＵＲＬを公開している、いわゆるＷｅｂサービスとして動作しているプログラムである。サーバＰＣ８０上ではＵＲＬに対して到着した依頼を印刷データ変換部３３０に転送するＷｅｂ通信部３１０も動作している。ＵＲＬへ到着する依頼は通常多数になるため、単一のサーバＰＣ８０では処理させず、複数のサーバＰＣ８０に処理させる構成とする。従って、ロードバランサ７０が複数のサーバＰＣ８０の前段に設置され、各サーバＰＣ８０に転送される依頼が均一に配分される。

【００３４】

印刷データ変換部３３０が提供するサービスは、依頼者５０が指定するデータをプリンタ６０で印刷できる形式に変換することである。依頼者が指定するデータにはドキュメント、画像ファイルなど各種のアプリケーションプログラムで作成されたフォーマットが存在する。一方、プリンタ６０が印刷データとして受け付けるデータは前述のフォーマットとは異なり、ＰＤＬで記述されたデータや、印刷部数等の印刷を制御するための命令とパラメータを備えたＰＪＬで記述されたデータとなっている。ＰＤＬは、Ｐａｇｅ Ｄｅｓｃｒｉｐｔｉｏｎ Ｌａｎｇｕａｇｅの略称である。また、ＰＪＬは、Ｐｒｉｎｔ Ｊｏｂ Ｌａｎｇｕａｇｅの略称である。

【００３５】

このように、依頼者５０とプリンタ６０の間には扱うデータ形式の差異が存在する為に、その差異を埋るプログラムが必要とされる。通常この変換処理を行うのがプリンタドライバと呼ばれるプログラムである。プリンタドライバは、依頼者５０が操作する前述のアプリケーションプログラムが動作しているＰＣ上で動作させることが通常である。しかし、プリンタドライバの処理は比較的ＣＰＵ、メモリなどのリソースを必要とする重い処理であるため、依頼者５０の操作するＰＣの保持するリソースが少ない場合には印刷処理が実用的でなくなる程時間のかかる処理になってしまう。依頼者５０が操作する端末（パーソナルコンピュータ２０、タブレットＰＣ３０や携帯端末４０）などリソースの比較的少ない端末からの印刷の需要が高まるにつれプリンタドライバの処理をインターネット上のサービスで行わせることの重要性が増している。

【００３６】

依頼者５０は、各種端末から前述のＵＲＬに対して印刷データをプリンタ６０で印刷できる形式に変換するように依頼を行う。依頼に応じてサーバＰＣ８０上で動作している印刷データ変換部３３０が生成した変換後のデータは、依頼元の端末に対して返送される。依頼者５０の指示によっては変換後のデータは、印刷データ変換部３３０がアクセス可能なファイルサーバＰＣ９０で保管される場合もある。印刷データ変換部３３０をはじめとするソフトウエア群が動作するサーバＰＣ８０のハードウエア構成は、図７の構成と同一であるため説明を省略する。

【００３７】

図８は、印刷データ変換部３３０のソフトウエア構成例を示す図である。サーバＰＣ８０は、Ｗｅｂ通信部３１０、アプリケーション部３２０、印刷データ変換部３３０、ロケータ３４０、ロガー３５０を備える。Ｗｅｂ通信部３１０は、依頼者５０が操作したパー

ソナルコンピュータ２０上のアプリケーションからＨＴＴＰプロトコルで送信されたリクエストを受け取る。ロードバランサ７０は、Ｗｅｂ通信部３１０の前段に位置し、複数のサーバＰＣ８０上で動作しているＷｅｂ通信部３１０に振り分ける。この振り分ける方式は種々知られているが、その詳細は本発明には関係がないため説明を省略する。

【００３８】

ここで、サーバＰＣ８０は必ずしも物理的な一台のサーバＰＣを意味する必要はない。近年は、単一のＰＣ上に複数の仮想的なＰＣを出現させる仮想化技術が一般的となっており、ソフトウエアからみれば自身が動作しているオペレーティングシステムが仮想的に作り出されたものか否かは判別がつかないか、あるいは判別する必要がない。従って本発明におけるサーバＰＣ８０は「場合によっては仮想化されたＰＣ」を意味する場合もある。これ以降、「ＰＣ」という表現は物理的なＰＣと仮想化されたＰＣの両方を意味するものとして扱う。

【００３９】

アプリケーション部３２０は、要求の送付先として指定されたＵＲＬを解析し、関連付けられている印刷データ変換部３３０を決定するソフトウエアである。通常、Ｗｅｂ通信部３１０とアプリケーション部３２０は複数種のサービスを扱うことが可能であり、ＵＲＬから対応するサービスを特定して要求を振り分けることができる。本実施例ではその複数種のサービスの一つが印刷データ変換部３３０であることを想定している。このサービスの他にもサービスが稼働しているが、発明の説明においては印刷データ変換部３３０以外のサービスは省略する。

【００４０】

印刷データ変換部３３０は、実施例１、２のデータ処理部１００に対応し、入力受付部１０１、制御部１０２、処理部１０３のそれぞれに相当するコンポーネントで構成される。データ処理部１００と同様に、受信したファイル数などに応じて複数の印刷データ変換部３３０が起動される。変換データの受け渡しは、本実施例３で新たに導入する、フィルタパイプラインを用いる。また、リスト管理部１０４に相当するコンポーネントの他、印刷データ変換サービスを実現するために必要な、各印刷データ変換サービスに共通するコンポーネントを２つ追加する。

【００４１】

印刷データ変換部３３０は単一のプロセスではなく、複数プロセスが共同して動作することで印刷データ変換サービスを提供する形態をとっている。以下では「印刷データ変換サービス」という場合にはこれら複数のプロセスを総称していることに留意されたい。ロケータ３４０とロガー３５０は、印刷データ変換部３３０とは独立して存在するプロセスである。アプリケーション部３２０は、複数の要求を並列実行させるために印刷データ変換部３３０を複数起動し、各々に要求を振り分ける。この場合でも、ロケータ３４０とロガー３５０は各々一個のプロセスとして存在している。アプリケーション部３２０はさらに、印刷データ変換部３３０が動作する際に依頼者５０からの一連の要求を特定する為のセッションを管理する。具体的には、同一の依頼者５０からの要求は同一の印刷データ変換部３３０に転送するセッション・アフィニティーを実現している。

【００４２】

（印刷データ変換サービス）

図９は、印刷データ変換部３３０の動作に関連するソフトウエアコンポーネント群の関連を説明する図である。印刷データ変換部３３０は先に説明したように複数のプロセスの共同体として動作している。その構成要素は印刷サービスゲートウェイ２１１、プロキシモジュール２２０、フィルタホスト２４０と各種フィルタ群（２７０，２８０，２９０）である。ここでは説明の為に各種フィルタは３種類示してあるが、実際には多くのフィルタが存在しており、さらにその組み合わせは固定化されておらず、生成される印刷データ変換部３３０毎に異なるフィルタが使用される可能性がある。

【００４３】

（印刷サービスゲートウェイ）

印刷データ変換部３３０に対するサービス依頼を受け付けるのは、プロセスとして存在する印刷サービスゲートウェイ２１１である。印刷サービスゲートウェイ２１１は実施例１の入力受付部１０１に相当する。印刷サービスゲートウェイ２１１の起動はアプリケーション部３２０が行う。印刷サービスゲートウェイ２１１はプロキシモジュール２２０をロードし、Ａｐｐｌｉｃａｔｉｏｎ Ｐｒｏｇｒａｍｍｉｎｇ Ｉｎｔｅｒｆａｃｅ（ＡＰＩ）で提供している関数群を呼び出すことで変換処理の実行を依頼し、さらに結果を受領する。

【００４４】

（ロケータ）

印刷データ変換部３３０を構成するコンポーネント群が利用するロケータ３４０は、特別な役割を担っている。ロケータ３４０は、アプリケーション部３２０が起動した時点で既に起動している特殊なプロセスである。ロケータ３４０は、コンポーネントを生成する機能を有する。さらに、ロケータ３４０は、他のコンポーネントからの求めに応じて別のコンポーネントのアクセスポイントを返すという機能を提供する。アクセスポイントとはＴＣＰ／ＩＰのリスンポートを指す。あるコンポーネントは別のコンポーネントが公開しているアクセスポイントを取得し、そのアクセスポイントにアクセスすることでそのコンポーネントが提供する機能を利用する。

【００４５】

（ジョブコントローラ）

実施例１の制御部１０２に対応するのはジョブコントローラ２３０である。プロキシモジュール２２０を介してジョブ実行を依頼されたジョブコントローラ２３０は、ジョブ、およびジョブに含まれている印刷データを解析して変換に必要なフィルタを選定する。次に、ジョブコントローラ２３０は、ジョブに含まれている変換先の印刷データ形式を取得し、変換元の印刷データ形式と変換先の印刷データ形式の両者が確定した時点で、この変換に必要なフィルタ群を選定する。必要なフィルタの組み合わせとその変換順番は固定的な知識としてジョブコントローラ２３０に実装されている。

【００４６】

必要な複数のフィルタを選定したあとは、これらフィルタをロードしているフィルタホスト２４０が取得される。この例では変換に必要なフィルタとして、フィルタＡ２７０、フィルタＢ２８０、フィルタＣ２９０が必要であり、それらはそれぞれフィルタホスト２４０にロードされている。

【００４７】

ジョブコントローラ２３０は、フィルタＡ２７０、フィルタＢ２８０、フィルタＣ２９０との間で、パイプライン３００と呼ばれるデータ転送チャネルを作成する。パイプライン３００は基本的に片方向にのみデータを転送する。ジョブコントローラ２３０→フィルタＡ２７０→フィルタＢ２８０→フィルタＣ２９０→ジョブコントローラ２３０と一周するようにパイプライン３００が構成される。また、ジョブコントローラ２３０はプロキシモジュール２２０との間にもパイプライン３００を作成する。このようにパイプライン３００を作成することで、プロキシモジュール２２０から印刷データが複数のコンポーネントを還流し、最後に変換が終了した形で再びプロキシモジュール２２０に戻ってくる。

【００４８】

（フィルタ、フィルタホスト）

フィルタは、データ変換処理のみを実装したライブラリモジュールの形式で用意されており、実施例１の処理部１０３に相当する。フィルタホスト２４０はプロセスとして存在し、実行時に指定されたフィルタをロードする。フィルタホスト２４０は、コントロールバス５０２への接続とメッセージ送受信、ジョブコントローラ２３０との通信、フィルタのログ出力のロガー３５０への転送などの処理を担当する。

【００４９】

（ロガー）

ロガー３５０は、プロセスとして同一コントロールバス５０２上に一つ存在する。コン

トロールバス５０２上のコンポーネントは、ジョブの実行中にログ出力を行う。各コンポーネントは、ログデータをロガー３５０に対してネットワークを介して送り、ロガー３５０が一定量のログデータが蓄積した時点でログファイルに出力することになっている。

【００５０】

（変換ジョブ）

ここで、パイプライン３００を還流するデータについて説明する。このデータは「変換ジョブ」と呼ばれる。変換処理は、プロキシモジュール２２０のＡＰＩを通じて依頼される際に、変換ジョブ（以降、単にジョブと称す）の形で引き渡される。ジョブは、概念的には、変換されるべき対象データと変換処理用の設定値などをまとまりのある形式に束ねた「チケット」と呼ばれるデータ構造から構成される。ジョブには、識別できるようにユニークな識別情報が与えられる（ジョブＩＤ）。生成されたジョブは、ファイルとしてＨＤＤ２０２などの記憶装置に格納される。また記憶装置から取り出す際には、ジョブＩＤを指定してそのファイルを取得して内容を参照することが可能である。また、ジョブを破棄する行為はファイルの削除として実現される。ジョブは、この印刷データの内容を直接ファイル内に含むか、またはファイル化した印刷データのファイルパスなどの参照情報を含んでいる。さらにジョブは、印刷を行わせるプリンタ６０が処理できる印刷ファイル形式についての情報も含んでいる。また、印刷部数や印刷の面付指示などの印刷の処理方法を指定する印刷設定も含む。

【００５１】

（不正ファイル投入時のエラー）

印刷データ変換部３３０は、不正データを含むジョブが投入されると、ジョブを正常に行うことができない。不正データには２種類存在する。ひとつはフィルタでの変換処理中にアクセス違反やオーバーフローなどのエラーを発生させ、フィルタを異常終了させるデータである。印刷データ変換部３３０は、変換処理中にフィルタが異常終了すると、フィルタをロードしたフィルタホスト２４０のプロセスもともに終了する。従って、フィルタ群とジョブコントローラ２３０をつないでいたパイプライン３００が切れる。

【００５２】

もうひとつは、フィルタでの変換処理中に無限ループに入るなどしてフィルタを長時間専有するデータである。各フィルタは、変換処理開始時にオペレーティングシステムのシステムタイマーを起動させ、処理にかかる時間を計る。変換処理が規定時間内に終了しなかった場合、システムタイマーからフィルタにタイムアウトメッセージが送信される。また、これらのデータ依存のエラーの他、印刷データ変換部３３０は不正な変換対象データを含まないジョブに対しても、データ投入のタイミングによりエラーを発生させることがある。

【００５３】

以下、本明細書を上記の印刷データ変換部３３０に適用した実施形態について説明する。図１０は、本実施例における印刷データ変換部３３０の構成例を示す。印刷データ変換部３３０は、複数の印刷データ変換部３３０、ロケータ３４０、ロガー３５０、ゲートキーパー２０１を備える。各コンポーネントは、同一コントロールバス５０２で接続されている。

【００５４】

印刷データ変換部３３０は、未知の不正データを１つ投入されると、まず以前に不正と判断されたデータの識別情報を保持するブラックリスト２００から一致する情報の検索（以下、ブラックリスト検索と呼ぶ）をする。一致する情報が検出されない場合、ブラックリスト検索が終了すると印刷データ変換部３３０は通常の変換処理を開始する。不正データにより、フィルタホスト２４０にロードされ変換処理を実行するフィルタでは変換処理中にエラーが発生する。

【００５５】

そこで、印刷データ変換部３３０は、エラー処理としてまずこの不正データの識別情報をゲートキーパー２０１が管理するブラックリスト２００に追加する。その後、エラー通

知やログ出力などエラー処理として必要な処理を行う。印刷データ変換部３３０は、エラー処理が完了するまで同プロセスで次のジョブを受け付けることはできない。しかし、前述のエラー処理中に同じ不正データが同一サーバＰＣ８０上の別の印刷データ変換部３３０のプロセスに投入されたとき、ブラックリスト検索により、投入されたデータの識別情報が検出されるようになる。前述のエラー処理中の印刷データ変換部３３０のプロセスが復旧するのを待つことなく、同一サーバＰＣ８０に並列する印刷データ変換部３３０のプロセスから、連続投入された同一不正データを遮断することが可能となる。

【００５６】

ゲートキーパー２０１は、同一サーバＰＣ８０上で起動され同一コントロールバス５０２でつながれた複数の印刷データ変換部３３０に対し、１つだけ存在する。ゲートキーパー２０１は、実施例１のリスト管理部１０４と同様、データの識別情報の算出、ブラックリスト検索とジョブコントローラ２３０への結果通知、ブラックリスト２００へのデータ識別情報の追加の処理を担当する。いずれもジョブコントローラ２３０からの依頼により行う。

【００５７】

また、本実施例３では、変換対象データの識別情報として変換対象データのハッシュ値を用いる。ハッシュ値を求める方法には例えばＳＨＡ－２等のハッシュ関数が知られているが、ハッシュ関数はこれに限定されない。変換対象データが一意に識別できるハッシュ値を求める手法であれば、どのハッシュ関数が適用されてもよい。ハッシュ値の算出対象は変換対象データ全体に限らず、変換対象データの一部、変換対象データとチケットを合わせたジョブ全体、あるいはジョブの一部であってもよい。不正データのハッシュ値を保持するブラックリスト２００の例を図１８に示す。図１８に示すブラックリストには、ハッシュ値とその登録時刻などの情報が登録されている。

【００５８】

ロケータ３４０により、印刷データ変換部３３０の変換処理実行に必要なコンポーネントが生成され、ジョブコントローラ２３０がプロキシモジュール２２０からジョブ実行を依頼された後の処理のフローについて述べる。

【００５９】

図１１は、ジョブコントローラ２３０が生成された時点から開始するイベントループ処理である。ジョブコントローラ２３０は、自身のプロセスが終了するまで２つのイベントを待ち受けながらループし続ける。イベントは、「ジョブ取得」と「エラー発生」である。まず、Ｓ１０１で、ジョブコントローラ２３０は、変換対象データを取得したか否かを判断する。変換対象データを取得した場合、処理はＳ２００に進み、ジョブコントローラ２３０は図１２に示すジョブ実行処理を実行する。続いて、Ｓ１０２で、ジョブコントローラ２３０は、データ変換処理中にエラーが発生したか否かを判断する。エラーが発生した場合、処理はＳ３００に進み、ジョブコントローラ２３０は図１３に示すエラー処理を実行する。

【００６０】

図１２は、ジョブコントローラ２３０によるジョブ実行処理（Ｓ２００）の処理フローを示す。Ｓ２０１で、ジョブコントローラ２３０は、ゲートキーパー２０１にブラックリスト検索処理を依頼するため、ゲートキーパー２０１に変換対象データを渡す。ブラックリスト検索処理（Ｓ４００）は図１４を用いて後述する。Ｓ２０２でジョブコントローラ２３０は、ゲートキーパー２０１からブラックリスト検索結果を受け取ると、Ｓ２０３で変換対象データがブラックリスト２００に登録されているか否か判断する。変換対象データがブラックリスト２００に登録されていると判断した場合、Ｓ２０７で、ジョブコントローラ２３０はプロキシモジュール２２０にエラーを通知し、ジョブ実行処理を終了する。

【００６１】

一方、変換対象データがブラックリスト２００に登録されていない場合、Ｓ２０４に処理は進み、ジョブコントローラ２３０はフィルタ群にデータ変換処理を依頼する。データ

変換処理（Ｓ５００）は図１６を用いて後述する。フィルタ群がデータ変換処理を終了すると、Ｓ２０５で、ジョブコントローラ２３０は出力データを取得する。ジョブコントローラ２３０は出力データを受け取ると、Ｓ２０６で、プロキシモジュール２２０に出力データを渡し、ジョブ実行処理を終了する。

【００６２】

図１４は、ゲートキーパー２０１によるブラックリスト検索処理（Ｓ４００）の処理フローを示す。ゲートキーパー２０１は、ジョブコントローラ２３０からブラックリスト検索処理の依頼と変換対象データを受け取ると、Ｓ４０１で、変換対象データのハッシュ値を算出する。Ｓ４０２で、ゲートキーパー２０１は、算出したハッシュ値と一致する情報をブラックリスト２００内で検索し、Ｓ４０３で検索結果をジョブコントローラ２３０に返す。

【００６３】

図１６は、各フィルタによるデータ変換処理（Ｓ５００）の処理フローを示す。データ変換は、ジョブに応じて複数フィルタが行う。図１０に示す例では、フィルタ群はｎ個のフィルタ（フィルタ１（２４１）、フィルタ２（２４２）、…、フィルタｎ（２４３））から構成されている。各フィルタは、Ｓ５０１で各々が担当するデータ変換を行った後に、Ｓ５０２で変換後のデータをパイプライン３００に出力し、次のフィルタにデータを渡す。パイプライン３００でつながれたすべてのフィルタが処理を終えると、出力データはパイプライン３００に乗ってジョブコントローラ２３０に送られる。

【００６４】

図１６に示すデータ変換処理は、処理が正常に終了した場合のフローである。Ｓ５０１のデータ変換では、不正データによるエラーが発生する可能性があり、データ変換処理を最後まで正常に行えるとは限らない。そのため、各フィルタはジョブコントローラ２３０と同様にロードされた時点からイベントループ処理を開始し、エラーの発生を監視している。

【００６５】

図１５は、フィルタによるイベントループ処理（Ｓ６００）を示す。フィルタは、自身のプロセスが終了するまで２つのイベントを待ち受けながらループし続ける。イベントは、データ取得とタイムアウトである。Ｓ６０１のイベントで、ジョブコントローラ２３０または別のフィルタから変換対象データを取得すると、フィルタは図１６のデータ変換処理（Ｓ５００）を実行する。Ｓ６０２でフィルタは、データ変換処理中にタイムアウトメッセージを受信したか否かを判断する。自身が担う変換処理が規定時間内に終了せず、システムタイマーからタイムアウトメッセージを受信した場合、処理はＳ７００に進み、タイムアウト処理を実行する。

【００６６】

図１７は、タイムアウト処理（Ｓ７００）の処理フローを示す。タイムアウトメッセージを受けたフィルタは、Ｓ７０１で実行中のデータ変換処理を中断し、Ｓ７０２でジョブコントローラ２３０にタイムアウト通知を送信する。この通知を受け取ったジョブコントローラ２３０が、図１１に示すイベントＳ１０２を発生させる。

【００６７】

イベントＳ１０２は、フィルタからのタイムアウト通知だけでなく、不正な変換対象データを原因とするフィルタの異常終了によっても発生する。エラーによりフィルタが異常終了すると、フィルタはジョブコントローラ２３０にメッセージを送信することができない。しかしフィルタ群のいずれかのフィルタが異常終了すると、パイプライン３００は断絶される仕組みになっている。従って、ジョブコントローラ２３０はパイプライン３００の状態を確認することにより異常終了を確認し、イベントＳ１０２を発生させることができる。

【００６８】

図１３は、イベントＳ１０２発生時に実行されるジョブコントローラ２３０によるエラー処理（Ｓ３００）の処理フローを示す。Ｓ３０１で、ジョブコントローラ２３０は、ゲ

ートキーパー２０１にエラーを引き起こした変換対象データのハッシュ値をブラックリスト２００に登録するよう依頼する。ゲートキーパー２０１はジョブコントローラ２３０からの依頼を受けると、先にＳ４０１で算出したハッシュ値をブラックリストに登録する。ジョブコントローラ２３０はプロキシモジュール２２０にエラー通知（Ｓ２０７）をし、エラー処理を終了する。

【００６９】

本実施例では、データ変換処理（Ｓ５００）においてエラーを発生させた変換対象データはすべてＳ３０１でブラックリストに登録するが、前述したようにデータ依存ではなくタイミング依存のエラーが発生する可能性も考慮してよい。例えば、エラー発生（Ｓ１０２）後、データ変換処理（Ｓ５００）のリトライをし、再びエラーが発生したら変換対象データをブラックリスト登録するようにしてもよい。

【００７０】

（実施例４）

実施例２のシステムを実施例３の印刷データ変換部３３０に適用した実施形態について説明する。本実施形態４におけるジョブコントローラ２３０のイベントループ処理は図１１と同じであり、ジョブコントローラ２３０が受け取るイベントは変わらないが、イベント発生時に行う処理が一部異なる。以下、実施例３と同じ処理については説明を省略する。

【００７１】

ジョブコントローラ２３０が変換対象データを取得したときのイベントＳ１０１におけるジョブ実行処理（Ｓ２１０）を図１９に示す。図１９は、実施例２の「警戒態勢」を印刷データ変換部３３０に導入したときのジョブ実行処理となる。印刷データ変換部３３０の「現在の状態」（警戒態勢であるか否か）は、ゲートキーパー２０１が保持し管理する。Ｓ２１０のジョブ実行処理では、始めにジョブコントローラ２３０はゲートキーパー２０１に対し「現在の状態」を問い合わせる。Ｓ２１２で、ジョブコントローラは、ゲートキーパー２０１が保持している「現在の状態」が警戒態勢であるか否かの返答を受取る。

【００７２】

Ｓ２０２で、警戒態勢でない場合、処理はＳ２０４に進み、データ変換処理（Ｓ５００）を実行するようフィルタ群に変換対象データを渡す。Ｓ２０５で、ジョブコントローラ２３０がフィルタ群から出力データを受け取ると、実施例３と同様に、Ｓ２０６でプロキシモジュール２２０に変換後のデータを出力する。最後に、ジョブコントローラ２３０は、ゲートキーパー２０１にコンプリーションメッセージ３０３を送ることにより処理終了通知（Ｓ２１３）をする。

【００７３】

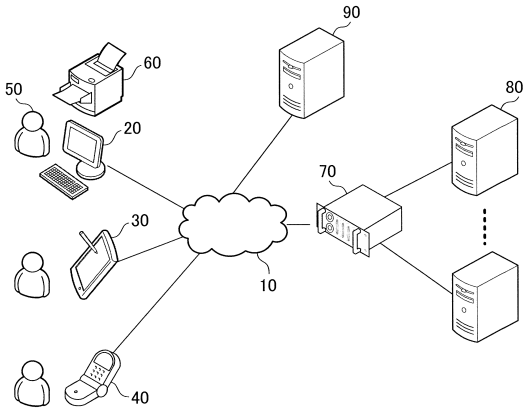
Ｓ２０２において、警戒態勢である場合、ジョブコントローラ２３０はゲートキーパーにブラックリスト検索を依頼する。ゲートキーパー２０１は、ブラックリスト検索処理（Ｓ４００）を行う。Ｓ２１３で、ジョブコントローラ２３０は、ブラックリスト検索結果をゲートキーパー２０１から受け取る。Ｓ２０３以降の処理は図１２で説明した処理と同様である。

【００７４】

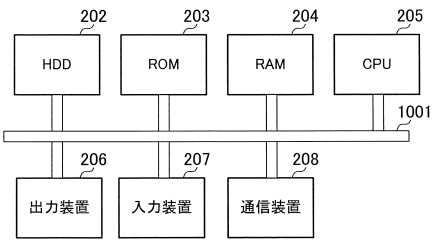
ジョブコントローラ２３０でのエラー発生時のイベントＳ１０２における、エラー処理（Ｓ３１０）を図２０に示す。Ｓ３１０でエラー処理が開始し、Ｓ３１１で、ジョブコントローラ２３０は、ゲートキーパー２０１にアラートメッセージ３０１を送って警報発令をする。その後、実施例３と同様にブラックリスト登録依頼（Ｓ３０１）をゲートキーパー２０１にし、プロキシモジュール２２０にエラー通知をする（Ｓ２０７）。ゲートキーパー２０１は、複数稼働している各印刷データ変換部３３０のジョブコントローラ２３０から順次送られてくるアラートメッセージ３０１とコンプリーションメッセージ３０３から現在の状態を判断する。ゲートキーパー２０１がアラートメッセージ３０１を受理すると現在の状態は「警戒態勢」になり、「警戒態勢」である間にコンプリーションメッセージ３０３を規定回数連続で受理したときに「警戒態勢」を解除する。

【００７５】
以上、本発明の印刷データ変換システムによれば、未知の不正データを受信した場合に、次に同じデータを受け取った際にシステムで同じエラーが発生せず、データ処理の停止を防止することができる。
【００７６】
（その他の実施形態）
また、本発明は、以下の処理を実行することによっても実現される。即ち、上述した実施形態の機能を実現するソフトウエア（コンピュータプログラム）を、ネットワーク又は各種記憶媒体を介してシステム或いは装置に供給する。そして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）がプログラムを読み出して実行する処理である。この場合、そのプログラム、及び該プログラムを記憶した記憶媒体は本発明を構成することになる。

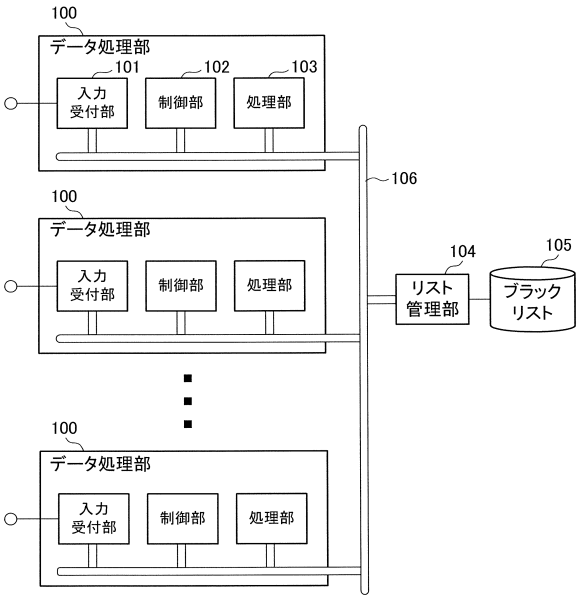
【図１】



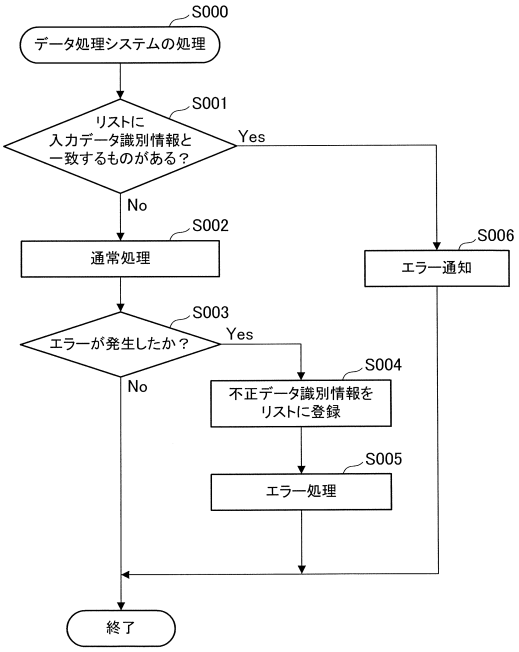
【図２】



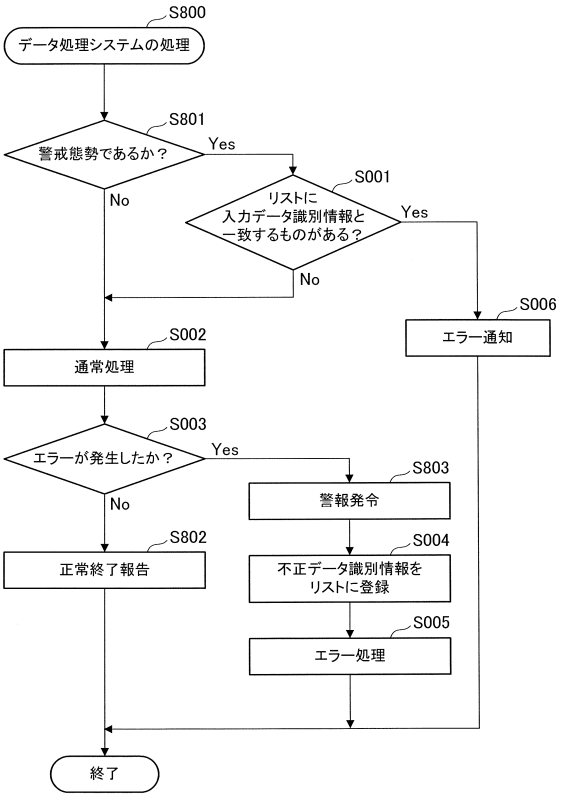
【図３】



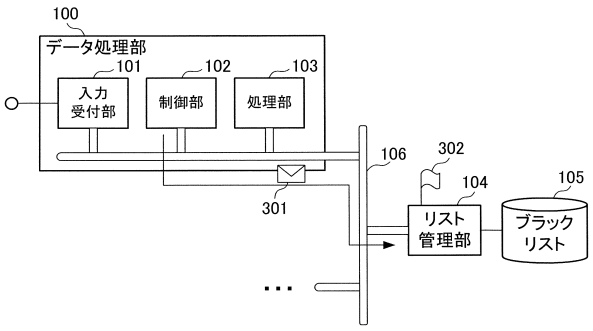
【図４】


S000
データ処理システムの処理
S001
リストに
入力データ識別情報と
一致するものがある？
Yes
No
S002
通常処理
S003
エラーが発生したか？
Yes
No
S004
不正データ識別情報を
リストに登録
S005
エラー処理
S006
エラー通知
終了

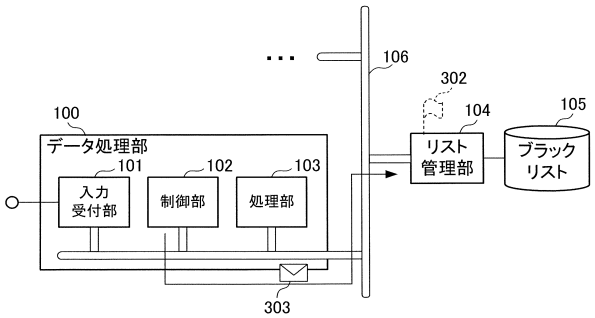
【図５】


S800
データ処理システムの処理
S801
警戒態勢であるか？
Yes
No
S001
リストに
入力データ識別情報と
一致するものがある？
Yes
No
S002
通常処理
S003
エラーが発生したか？
Yes
No
S803
警報発令
S004
不正データ識別情報を
リストに登録
S005
エラー処理
S802
正常終了報告
S006
エラー通知
終了

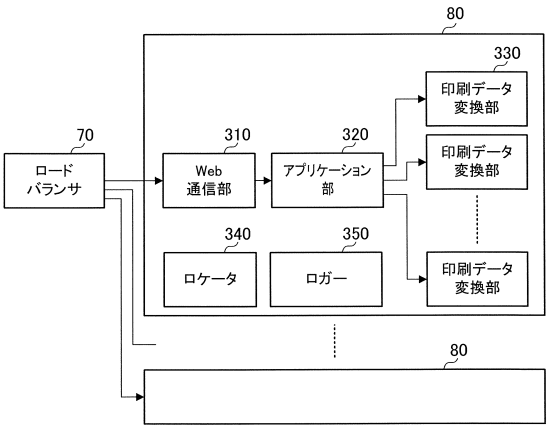
【図６】


100
データ処理部
101
入力
受付部
102
制御部
103
処理部
106
302
104
105
301
リスト
管理部
ブラック
リスト

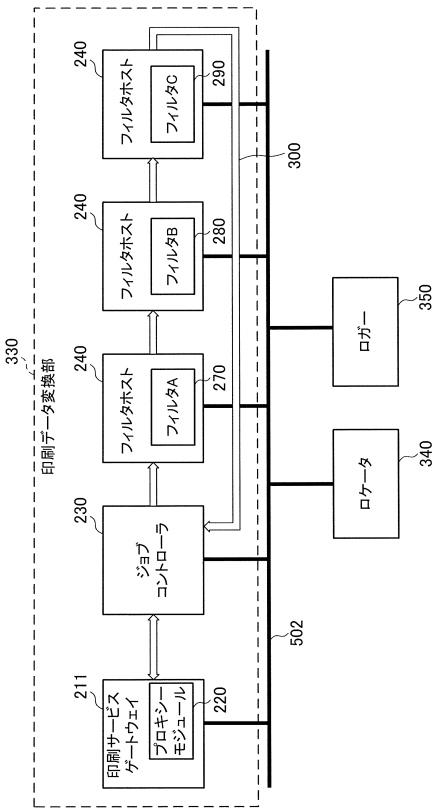
【図７】


106
302
104
105
100
データ処理部
101
入力
受付部
102
制御部
103
処理部
リスト
管理部
ブラック
リスト
303

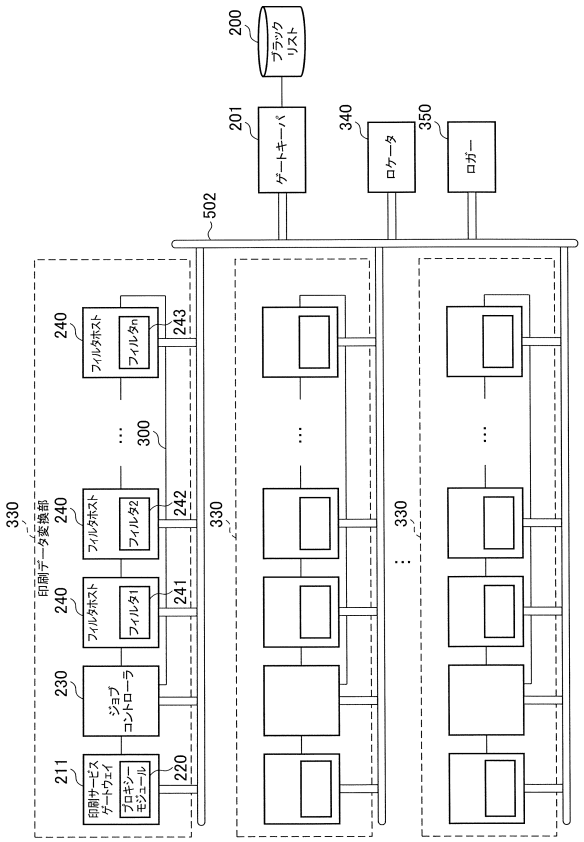
【図８】


80
330
印刷データ
変換部
70
310
320
ロード
バランサ
Web
通信部
アプリケーション
部
印刷データ
変換部
340
350
ロケータ
ロガー
印刷データ
変換部
80

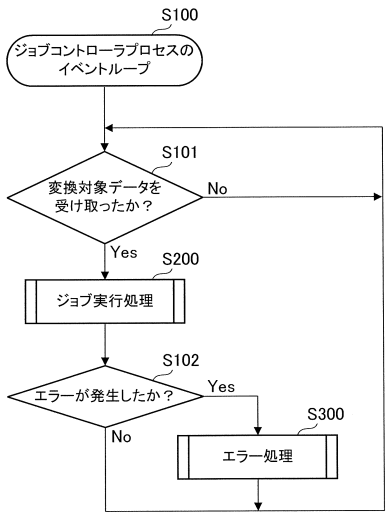
【図９】



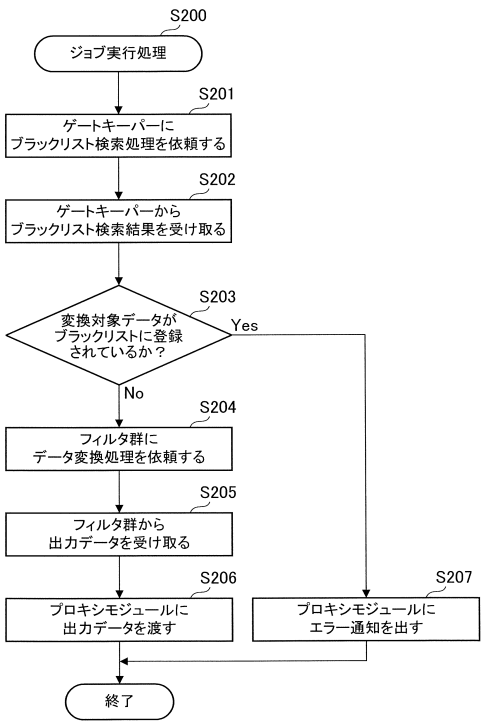
【図１０】



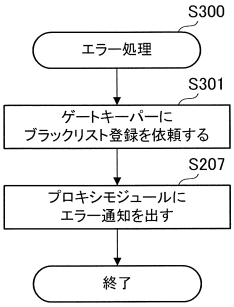
【図１１】



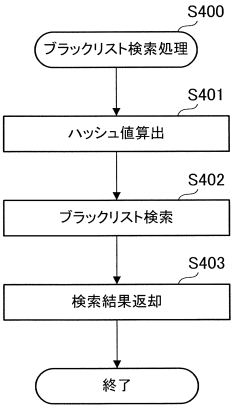
【図１２】



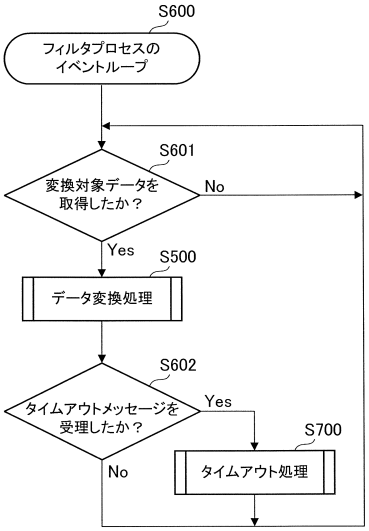
【図１３】



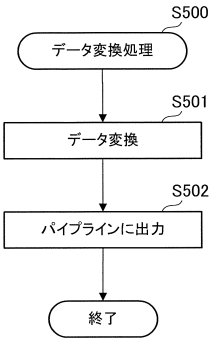
【図１４】



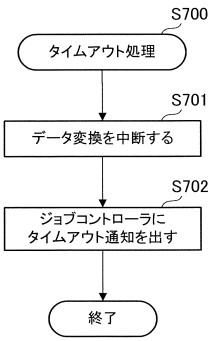
【図１５】



【図１６】



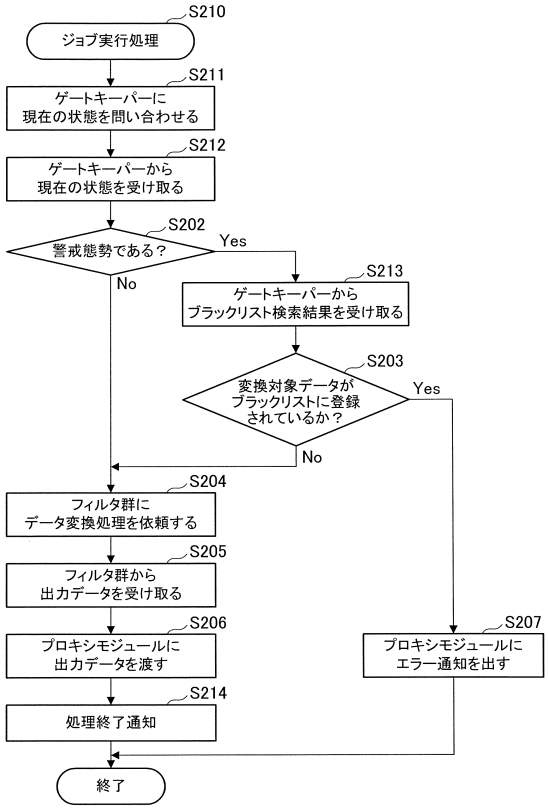
【図１７】



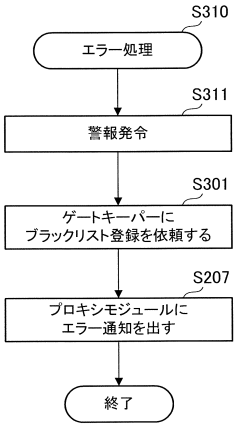
【図１８】

No.	ハッシュ値	登録時刻
1	2f12cc6cdf008899cde08a3aed3aaf66 8175afefc762269842fc1f6d7f753a96	2013/04/30 8:48
2	a8604ea8969ec78b2d59a7e4831469e9 b5661830f1519e14f697dadb9dcd49bc	2012/11/01 11:50
3	a1da8f0e55222542d954f4ef1c03bfeb 836daedf757db6ab20c34c85134609ed	2012/07/24 9:59
・ ・ ・	・ ・ ・	・ ・ ・

【図１９】


S210
ジョブ実行処理
S211
ゲートキーパーに
現在の状態を問い合わせる
S212
ゲートキーパーから
現在の状態を受け取る
S202
警戒態勢である？
Yes
No
S213
ゲートキーパーから
ブラックリスト検索結果を受け取る
S203
変換対象データが
ブラックリストに登録
されているか？
Yes
No
S204
フィルタ群に
データ変換処理を依頼する
S205
フィルタ群から
出力データを受け取る
S206
プロキシモジュールに
出力データを渡す
S207
プロキシモジュールに
エラー通知を出す
S214
処理終了通知
終了

【図２０】


S310
エラー処理
S311
警報発令
S301
ゲートキーパーに
ブラックリスト登録を依頼する
S207
プロキシモジュールに
エラー通知を出す
終了

フロントページの続き

(56)参考文献 特開２００８－２８７４１５（ＪＰ，Ａ）
特開平０９－２２３０３５（ＪＰ，Ａ）
特開２００９－０７５６３４（ＪＰ，Ａ）
特開２０１２－２４２８９５（ＪＰ，Ａ）
特開２００２－１６２８９９（ＪＰ，Ａ）
国際公開第２０１０／１３４１８２（WO，Ａ１）
特開２０１０－１０２５４３（ＪＰ，Ａ）
米国特許出願公開第２０１２／００６９３８４（ＵＳ，Ａ１）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｂ４１Ｊ５／００－５／５２
Ｂ４１Ｊ２１／００－２１／１８
Ｂ４１Ｊ２９／００－２９／７０
Ｇ０６Ｆ３／０９－３／１２
Ｇ０６Ｆ９／４６－５４
Ｇ０６Ｆ１２／００、１３／００
Ｈ０４Ｎ１／００