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

特許第6771874号
(P6771874)

(45)発行日　令和2年10月21日(2020.10.21)　(24)登録日　令和2年10月2日(2020.10.2)

(51)Int.Cl.			F I		
G06F	9/48	(2006.01)	G06F	9/48	370
G06F	9/44	(2018.01)	G06F	9/44	
G06F	9/455	(2006.01)	G06F	9/455	150

請求項の数 13　(全 14 頁)

(21)出願番号　特願2015-183180(P2015-183180)
(22)出願日　平成27年9月16日(2015.9.16)
(65)公開番号　特開2017-58952(P2017-58952A)
(43)公開日　平成29年3月23日(2017.3.23)
審査請求日　平成30年9月11日(2018.9.11)

(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　100125254
弁理士　別役　重尚
(72)発明者　木暮　岳史
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
(72)発明者　塚田　祥弘
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　漆原　孝治

最終頁に続く

(54)【発明の名称】情報処理装置、その制御方法及びプログラム

(57)【特許請求の範囲】
【請求項１】
オペレーティングシステム（ＯＳ）とは別に設けられ、拡張プログラムに含まれる命令を解釈して実行するプログラムである複数の仮想マシンを実行する実行手段と、
前記複数の仮想マシンにおける第一の仮想マシンと第二の仮想マシンとの間でメッセージの送信を行うメッセージ送信手段と、
前記第一の仮想マシンによって利用される第一のスレッドと、
前記第二の仮想マシンによって利用される第二のスレッドと、を備え、
前記第一のスレッドに対応するアプリケーションと前記第二のスレッドに対応するアプリケーションが異なる場合、前記メッセージ送信手段は前記第一の仮想マシンと前記第二の仮想マシンとの間でメッセージの送信を行わないことを特徴とする情報処理装置。
【請求項２】
前記拡張プログラムの実行に応じて新たなスレッドを生成するスレッド生成手段と、
前記新たなスレッドを利用する新たな仮想マシンを生成する仮想マシン生成手段と、をさらに備え、
前記第二のスレッドは前記スレッド生成手段によって生成され、前記第二の仮想マシンは前記仮想マシン生成手段によって生成されることを特徴とする請求項１記載の情報処理装置。
【請求項３】
前記スレッド生成手段は前記ＯＳを実行するプロセッサであり、

前記第一の仮想マシンが前記第二のスレッドの生成を要求し、前記第二のスレッドの生成の要求に応じて前記ＯＳを実行するプロセッサが前記第二のスレッドを生成することを特徴とする請求項２記載の情報処理装置。

【請求項４】

少なくとも１つのスレッドを管理するためのスレッド管理情報を生成するスレッド管理情報生成手段をさらに備え、

前記スレッド管理情報は、少なくとも前記第一のスレッドに対応するアプリケーションの識別番号と前記第二のスレッドに対応するアプリケーションの識別番号とを有することを特徴とする請求項１乃至３のいずれか１項に記載の情報処理装置。

【請求項５】

前記第一のスレッドに対応するアプリケーションと前記第二のスレッドに対応するアプリケーションが同じ場合、前記メッセージ送信手段は前記第一の仮想マシンと前記第二の仮想マシンとの間でメッセージの送信を行うことを特徴とする請求項１記載の情報処理装置。

【請求項６】

ネイティブプログラム内に前記第一のスレッドを有し、前記第一のスレッドに対応する前記第一の仮想マシンを生成し、前記拡張プログラムは前記第一の仮想マシン上で動作することを特徴とする請求項１記載の情報処理装置。

【請求項７】

オペレーティングシステム（ＯＳ）とは別に設けられ、拡張プログラムに含まれる命令を解釈して実行するプログラムである複数の仮想マシンを実行する実行ステップと、

前記複数の仮想マシンにおける第一の仮想マシンと第二の仮想マシンとの間でメッセージの送信を行うメッセージ送信ステップと、を有し、

前記第一の仮想マシンによって利用される第一のスレッドに対応するアプリケーションと、前記第二の仮想マシンによって利用される第二のスレッドに対応するアプリケーションとが異なる場合、前記メッセージ送信ステップでは前記第一の仮想マシンと前記第二の仮想マシンとの間でメッセージの送信を行わないことを特徴とする情報処理装置の制御方法。

【請求項８】

前記拡張プログラムの実行に応じて新たなスレッドを生成するスレッド生成ステップと、

前記新たなスレッドを利用する新たな仮想マシンを生成する仮想マシン生成ステップと、を有し、

前記第二のスレッドは前記スレッド生成ステップにおいて生成され、前記第二の仮想マシンは前記仮想マシン生成ステップにおいて生成されることを特徴とする請求項７記載の情報処理装置の制御方法。

【請求項９】

前記スレッド生成ステップでは、前記第一の仮想マシンが前記第二のスレッドの生成を要求し、前記第二のスレッドの生成の要求に応じて前記ＯＳを実行するプロセッサが前記第二のスレッドを生成することを特徴とする請求項８記載の情報処理装置の制御方法。

【請求項１０】

少なくとも１つのスレッドを管理するためのスレッド管理情報を生成するスレッド管理情報生成ステップをさらに有し、

前記スレッド管理情報は、少なくとも前記第一のスレッドに対応するアプリケーションの識別番号と前記第二のスレッドに対応するアプリケーションの識別番号とを有することを特徴とする請求項８又は９記載の情報処理装置の制御方法。

【請求項１１】

前記第一のスレッドに対応するアプリケーションと前記第二のスレッドに対応するアプリケーションが同じ場合、前記メッセージ送信ステップでは前記第一の仮想マシンと前記第二の仮想マシンとの間でメッセージの送信を行うことを特徴とする請求項７記載の情報

処理装置の制御方法。
【請求項１２】
ネイティブプログラム内に前記第一のスレッドを有し、前記第一のスレッドに対応する前記第一の仮想マシンを生成し、前記拡張プログラムは前記第一の仮想マシン上で動作することを特徴とする請求項７記載の情報処理装置の制御方法。
【請求項１３】
情報処理装置の制御方法をコンピュータに実行させるプログラムであって、
前記情報処理装置の制御方法は、
オペレーティングシステム（ＯＳ）とは別に設けられ、拡張プログラムに含まれる命令を解釈して実行するプログラムである複数の仮想マシンを実行する実行ステップと、
前記複数の仮想マシンにおける第一の仮想マシンと第二の仮想マシンとの間でメッセージの送信を行うメッセージ送信ステップと、を有し、
前記第一の仮想マシンによって利用される第一のスレッドに対応するアプリケーションと、前記第二の仮想マシンによって利用される第二のスレッドに対応するアプリケーションが異なる場合、前記メッセージ送信ステップでは前記第一の仮想マシンと前記第二の仮想マシンとの間でメッセージの送信を行わないことを特徴とするプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、情報処理装置、その制御方法及びプログラムに関し、特に、非同期処理を実行する情報処理装置、その制御方法及びプログラムに関する。
【背景技術】
【０００２】
機能を拡張するために拡張プログラムをアドインする情報処理装置としての画像形成装置が知られている。このような画像形成装置は基本的な機能を実現するための制御プログラムの実行環境の他に、拡張プログラムを実行するための実行環境を有する。
【０００３】
具体的には、画像形成装置では、オペレーティングシステム（ＯＳ）上に、画像形成装置のプリンタ部、ＦＡＸ部やスキャナ部を制御するための制御プログラムと、拡張プログラムの実行環境である拡張プログラム実行プラットフォームとが動作する。拡張プログラム実行プラットフォーム上には、拡張プログラムシステムサービスと拡張プログラムが動作する（例えば、特許文献１参照。）。
【０００４】
例えば、拡張プログラムを実行することによって外部装置に格納される画像データを取得して印刷する印刷処理を行う場合、外部装置との通信処理と、Ｕｓｅｒ Ｉｎｔｅｒｆａｃｅ（ＵＩ）処理とを同期して実行することがある。このとき、外部装置通信処理の実行中にＵＩ処理は中断されるため、外部装置通信処理の実行に何らかの要因で時間を要すると、ＵＩ処理が実行されず、ＵＩを介した入力等を随時受け付けることができない。そこで、通常、印刷処理では、通信処理とＵＩ処理を非同期で実行する。このとき、外部装置通信処理の実行中であってもＵＩ処理は中断されないため、通信処理でトラブルが生じても、ＵＩ処理が実行され、その結果、ＵＩを介した入力等を随時受け付けることができる。
【０００５】
通信処理とＵＩ処理を非同期で実行、すなわち、非同期処理を実行するには、拡張プログラムにおいて複数のＯＳのスレッドを利用する必要がある。ここで、余裕のある実行環境、例えば、Ｊａｖａ（登録商標）を利用可能な実行環境を搭載可能な画像形成装置では、拡張プログラムを実行する仮想マシン（Ｖｉｒｔｕａｌ Ｍａｃｈｉｎｅ：ＶＭ）が複数のＯＳのスレッドを利用することができる。その結果、余裕のある実行環境を搭載可能な画像形成装置では、非同期処理を実行することができる。
【先行技術文献】

【特許文献】
【０００６】
【特許文献１】特開２０１４－７５０８８号公報
【発明の概要】
【発明が解決しようとする課題】
【０００７】
しかしながら、メモリ等が十分に備えられていない余裕の無い実行環境、例えば、Ｌｕａの利用を前提とする実行環境を搭載する画像形成装置では、ＶＭが複数のＯＳのスレッドを利用することができず、非同期処理を実行することができないという問題がある。
【０００８】
本発明の目的は、余裕の無い実行環境であっても、非同期処理を実行することができる情報処理装置、その制御方法及びプログラムを提供することにある。
【課題を解決するための手段】
【０００９】
上記目的を達成するために、本発明の情報処理装置は、オペレーティングシステム（ＯＳ）とは別に設けられ、拡張プログラムに含まれる命令を解釈して実行するプログラムである複数の仮想マシンを実行する実行手段と、前記複数の仮想マシンにおける第一の仮想マシンと第二の仮想マシンとの間でメッセージの送信を行うメッセージ送信手段と、前記第一の仮想マシンによって利用される第一のスレッドと、前記第二の仮想マシンによって利用される第二のスレッドと、を備え、前記第一のスレッドに対応するアプリケーションと前記第二のスレッドに対応するアプリケーションが異なる場合、前記メッセージ送信手段は前記第一の仮想マシンと前記第二の仮想マシンとの間でメッセージの送信を行わないことを特徴とする。
【発明の効果】
【００１０】
本発明によれば、余裕の無い実行環境であっても非同期処理を実行することができる。
【図面の簡単な説明】
【００１１】
【図１】本発明の実施の形態に係る情報処理装置としてのＭＦＰの主要部の構成を概略的に示すブロック図である。
【図２】図１のＭＦＰにおける拡張アプリケーションの実行環境の一例を説明するためのブロック図である。
【図３】図２におけるＯＳ上において拡張アプリケーションを起動する場合の処理の流れを説明するための図である。
【図４】図３におけるステップＳ３０４で生成されるスレッド管理情報の構成の一例を説明するための図である。
【図５】図２におけるＯＳ上において拡張アプリケーションから新たなＶＭスレッドを生成する場合の処理の流れを説明するための図である。
【図６】図５のステップＳ５０１において送出されるスレッド情報の構成の一例を説明するための図である。
【図７】図１のＭＦＰを含む印刷システムの機器構成の一例を説明するための図である。
【図８】図７における外部装置に格納されている画像データを取得して印刷するための拡張アプリケーションを実行するときの処理の流れを説明するための図である。
【図９】図８のステップＳ８０３において表示される操作画面の一例を示す図である。
【図１０】図８のステップＳ８０２やステップＳ８１３において２つのＶＭの間で実行されるメッセージ送信処理を説明するための図である。
【図１１】図１０のステップＳ１００２の各ＶＭ及び拡張ＶＭシステムサービスの間における処理の流れを説明するための図である。
【図１２】外部装置としてのホストＰＣに格納された拡張アプリケーションをＭＦＰにインストールするときの処理の流れを説明するための図である。

【発明を実施するための形態】
【００１２】
以下、本発明の実施の形態について図面を参照しながら詳細に説明する。本実施の形態では情報処理装置としての画像形成装置、例えば、ＭＦＰ（Multifunction Peripheral）に本発明を適用した場合について説明する。しかしながら、本発明は画像形成装置であるプリンタに適用してもよく、さらには、画像形成機能を備えない情報処理装置、例えば、サーバ、ＰＣやスマート家電機器に適用してもよい。具体的には、拡張プログラムがアドインされて非同期処理を実行する情報処理装置であれば、本発明を適用することができる。また、本実施の形態に記載されている構成要素はあくまで例示に過ぎず、本発明の範囲は本実施の形態に記載されている構成要素によって限定されることはない。
【００１３】
まず、本発明の第１の実施の形態について説明する。
【００１４】
図１は、本実施の形態に係る情報処理装置としてのＭＦＰの主要部の構成を概略的に示すブロック図である。本実施の形態では、ＭＦＰ１００が余裕の無い実行環境、例えば、Ｌｕａを利用する実行環境しか搭載せず、ＶＭが１つのスレッドのみしか利用できないことを前提とする。
【００１５】
図１において、ＭＦＰ１００はコントローラユニット１０１を備え、コントローラユニット１０１には、画像入力デバイスであるスキャナ１０２や画像出力デバイスであるプリンタ１０３だけでなく、操作部１０４も接続される。コントローラユニット１０１は、スキャナ１０２で読み取られた画像データをプリンタ１０３によって印刷出力するコピー機能を実現する制御を行う。また、コントローラユニット１０１はＣＰＵ１０５、ＲＯＭ１０６、ＲＡＭ１０７及びストレージ１０８を有し、ＣＰＵ１０５は、ＲＯＭ１０６に格納されているブートプログラムによってＯＳを起動する。ＣＰＵ１０５は起動したＯＳ上でストレージ１０８に格納されているプログラムを実行して各種処理を実行する。ＲＡＭ１０７は、ＣＰＵ１０５の作業領域を提供し、さらに、画像データを一時記憶するための画像メモリ領域を提供する。ストレージ１０８はプログラムや画像データを格納する。
【００１６】
さらに、コントローラユニット１０１は、操作部Ｉ／Ｆ（操作部インターフェース）１０９、ネットワークＩ／Ｆ（ネットワークインターフェース）１１０、ＵＳＢホストＩ／Ｆ１１１、画像バスＩ／Ｆ（画像バスインターフェース）１１２を有する。ＣＰＵ１０５にはＲＯＭ１０６、ＲＡＭ１０７及びストレージ１０８だけでなく操作部Ｉ／Ｆ１０９、ネットワークＩ／Ｆ１１０、ＵＳＢホストＩ／Ｆ１１１及び画像バスＩ／Ｆ１１２がシステムバス１１４を介して接続される。操作部Ｉ／Ｆ１０９は、タッチパネルを有する操作部１０４とのインターフェースであり、操作部１０４の画面に表示する画像データを操作部１０４へ出力する。また、操作部Ｉ／Ｆ１０９は、操作部１０４においてユーザによって入力された情報をＣＰＵ１０５に送出する。ネットワークＩ／Ｆ１１０は、ＭＦＰ１００をＬＡＮに接続するためのインターフェースである。ＵＳＢホストＩ／Ｆ１１１は、データを格納する外部記憶装置であるＵＳＢストレージ１１３と通信するためのインターフェースであり、ストレージ１０８に格納されているデータをＵＳＢストレージ１１３へ記憶させるために送出する。また、ＵＳＢホストＩ／Ｆ１１１は、ＵＳＢストレージ１１３に格納されているデータを受け取り、受け取ったデータをＣＰＵ１０５に伝送する。ＵＳＢストレージ１１３は、ＵＳＢホストＩ／Ｆ１１１に対して着脱可能である。なお、ＵＳＢホストＩ／Ｆ１１１には、ＵＳＢストレージ１１３を含む複数のＵＳＢデバイスが接続可能である。画像バスＩ／Ｆ１１２は、システムバス１１４と、画像データを高速で転送する画像バス１１５とを接続し、且つデータ形式を変換するバスブリッジである。画像バス１１５は、ＰＣＩバスまたはＩＥＥＥ１３９４等によって構成される。画像バス１１５にはデバイスＩ／Ｆ１１６、スキャナ画像処理部１１７及びプリンタ画像処理部１１８が接続される。デバイスＩ／Ｆ１１６には、スキャナ１０２及びプリンタ１０３が接続され

、デバイスＩ／Ｆ１１６は画像データの同期系／非同期系の変換を行う。スキャナ画像処理部１１７は入力画像データを補正し、加工し、又は編集する。プリンタ画像処理部１１８はプリント出力画像データに対してプリンタ１０３に応じた補正、解像度変換等を行う。

【００１７】

図２は、図１のＭＦＰにおける拡張アプリケーション（拡張プログラム）の実行環境の一例を説明するためのブロック図である。本実施の形態では、ストレージ１０８に格納されているプログラムを、ＣＰＵ１０５がＲＡＭ１０７にロードして実行することにより、ＯＳ上に以下説明する図２の各モジュールが実現される。

【００１８】

図２において、ＣＰＵ１０５によって起動されたＯＳ２０１上には、プリンタ機能、ＦＡＸ機能やスキャナ機能を実現するためのネイティブプログラム２０２と、ＶＭ２０３とが動作している。ＶＭ２０３は拡張アプリケーションを制御するプログラムを理解して実行するモジュールであり、拡張アプリケーションは必ずＶＭ２０３上で動作する。ネイティブプログラム２０２内には、プリンタ１０３やスキャナ１０２等の画像処理ユニットを制御するためのネイティブスレッド２０４と、ＶＭ２０３を動作させるためのＶＭスレッド２０５とが存在する。本実施の形態では、ＶＭ２０３として３つのＶＭ２０３ａ、ＶＭ２０３ｂ及びＶＭ２０３ｃが生成されている。また、ＶＭスレッド２０５として、ＶＭ２０３ａ、ＶＭ２０３ｂ及びＶＭ２０３ｃの合計数に対応する数である３つのＶＭスレッド２０５ａ、ＶＭスレッド２０５ｂ及びＶＭスレッド２０５ｃが生成されている。ＶＭシステムサービス２０６は各拡張アプリケーション２０７ａ，２０７ｂによって共通に利用されるユーティリティライブラリであり、複数の機能を提供する。各拡張アプリケーション２０７ａ，２０７ｂは、自身を実行するために必要な機能をＶＭシステムサービス２０６から選択する。ＭＦＰ１００では、各拡張アプリケーション２０７ａ，２０７ｂからＶＭシステムサービス２０６が提供する各機能を呼び出すことにより、拡張アプリケーションを開発する手間を省き、さらに、ＭＦＰ１００の各モジュールへアクセスすることができる。ＶＭシステムサービス２０６はモジュールとして標準ＶＭシステムサービス２０８と拡張ＶＭシステムサービス２０９を有する。標準ＶＭシステムサービス２０８は、例えば、ファイルシステムの「ｏｐｅｎ」、「ｃｌｏｓｅ」、「ｒｅａｄ」や「ｗｒｉｔｅ」という基本的なサービスを提供し、ＶＭ２０３がＶＭとして機能するための最低限の機能を実現させる。拡張ＶＭシステムサービス２０９は、ＭＦＰ１００の各モジュールへのアクセス機能や、ＯＳの各機能を実現する。

【００１９】

ＶＭ２０３は各拡張アプリケーション２０７ａ，２０７ｂを解釈して実行する。ＶＭ２０３は、拡張アプリケーションのスレッド毎に生成される。図２の実行環境では、非同期処理を実行する拡張アプリケーション２０７ａのために２つのＶＭスレッド２０５ａ，２０５ｂが生成され、各ＶＭスレッド２０５ａ，２０５ｂに対応して２つのＶＭ２０３ａ，ＶＭ２０３ｂが生成される。また、同期処理を実行する拡張アプリケーション２０７ｂのために１つのＶＭスレッド２０５ｃが生成され、ＶＭスレッド２０５ｃに対応して１つのＶＭ２０３ｃが生成される。

【００２０】

図１に戻り、ＭＦＰ１００の操作部１０４の画面には、各拡張アプリケーション２０７ａ，２０７ｂを示すアイコンが表示される。ユーザによるいずれかのアイコンの選択が操作部１０４を介して操作部Ｉ／Ｆ１０９によって検知されると、操作部Ｉ／Ｆ１０９はその旨をＣＰＵ１０５に送信する。その旨を受け取ったＣＰＵ１０５はユーザによって選択された拡張アプリケーション２０７ａ又は拡張アプリケーション２０７ｂを起動する。

【００２１】

図３は、図２におけるＯＳ上において拡張アプリケーションを起動する場合の処理の流れを説明するための図である。

【００２２】

まず、ネイティブスレッド２０４に拡張アプリケーション起動要求が通知され（ステップＳ３０１）、ネイティブスレッド２０４は続くステップＳ３０２においてＶＭスレッド生成処理を行う。このとき、ネイティブスレッド２０４はＯＳ２０１に対してＶＭスレッド生成要求を送出する。スレッド生成要求を受けたＯＳ２０１（スレッド生成手段）はＶＭスレッド２０５を生成する（ステップＳ３０３）。次いで、生成されたＶＭスレッド２０５（ＶＭ生成手段、スレッド管理情報生成手段）は後述するスレッド管理情報４００（図４参照。）を生成し（ステップＳ３０４）、さらに、ＶＭスレッド２０５上で動作する（を利用する）ＶＭ２０３を生成させる（ステップＳ３０５）。生成されたＶＭ２０３は拡張アプリケーションの読み込みを実行して標準ＶＭシステムサービス２０８に読み込み要求を送出する（ステップＳ３０６）。読み込み要求を受けた標準ＶＭシステムサービス２０８は拡張アプリケーションを実行するためのデータ（以下、「拡張アプリケーションデータ」という。）をＶＭ２０３に読み込ませる（ステップＳ３０７）。次いで、ＶＭ２０３は読み込んだ拡張アプリケーションデータに基づいて拡張アプリケーションを実行する（ステップＳ３０８）。すなわち、図３の処理では、拡張アプリケーションを起動する場合にネイティブスレッド２０４とは別の新たなＶＭスレッド２０５が生成され、さらに、ＶＭスレッド２０５を利用するＶＭ２０３が生成される。

【００２３】

図４は、図３におけるステップＳ３０４で生成されるスレッド管理情報の構成の一例を説明するための図である。

【００２４】

スレッド管理情報４００は拡張アプリケーションを起動する際に生成されたＶＭスレッド２０５を管理するために用いられる。スレッド管理情報４００は、アプリケーションＩＤ４０１、スレッドＩＤ４０２及びキューＩＤ４０３を有する。アプリケーションＩＤ４０１は生成されたＶＭスレッド２０５に対応するアプリケーションを識別するための識別番号である。スレッドＩＤ４０２は各ＶＭスレッド２０５を識別するための識別番号である。キューＩＤ４０３は各ＶＭスレッド２０５に対応するメッセージを格納するキューを識別するための識別番号である。図４のスレッド管理情報の構成の一例では、ＯＳ２０１上において、アプリケーションＡに対応し、キューＩＤがキュー１であるＶＭスレッド１と、アプリケーションＡに対応し、キューＩＤがキュー２であるＶＭスレッド２とが生成されている。さらに、アプリケーションＢに対応し、キューＩＤがキュー３であるＶＭスレッド３とが生成されている。

【００２５】

図５は、図２におけるＯＳ上において拡張アプリケーションから新たなＶＭスレッドを生成する場合の処理の流れを説明するための図である。図５の処理を実行する際、図４の処理が実行されてＶＭスレッド２０５ａ及びＶＭ２０３ａが生成されていることを前提とする。

【００２６】

まず、ＶＭ２０３ａは拡張アプリケーション２０７ａから要求されてスレッド生成処理を行い（ステップＳ５０１）、拡張ＶＭシステムサービス２０９に対して新たなＶＭスレッド２０５（２０５ｂ）の生成を要求するスレッド生成要求を送出する。このとき、スレッド生成要求として図６に示すスレッド情報６００を送出する。スレッド情報６００は、新たなＶＭスレッド２０５ｂの生成後に実行される拡張アプリケーション２０７ｂの実行ファイルの情報６０１と、拡張アプリケーション２０７ｂにおいて実行される実行関数の情報６０２とを指定する。さらに、スレッド情報６００は、ＶＭスレッド２０５ｂの優先度の情報６０３と、ＶＭスレッド２０５ｂのスタックサイズの情報６０４とを指定する。

【００２７】

図５に戻り、スレッド情報６００を受け取った拡張ＶＭシステムサービス２０９はＶＭスレッド生成処理を行い、ＯＳ２０１へスレッド生成要求を送出する（ステップＳ５０２）。ステップＳ５０２で送出されるスレッド生成要求は、スレッド情報６００における優先度の情報６０３とスタックサイズの情報６０４に見合うリソースのＶＭスレッド２０５

ｂの生成を要求する。次いで、ＯＳ２０１は、スレッド生成要求で要求された優先度やスタックサイズに見合うＶＭスレッド２０５ｂを生成する（ステップＳ５０３）。生成されたＶＭスレッド２０５ｂはスレッド管理情報４００を生成し（ステップＳ５０４）、さらに、ＶＭスレッド２０５ｂ上で動作する（を利用する）新たなＶＭ２０３ｂを生成させる（ステップＳ５０５）。次いで、生成されたＶＭ２０３ｂは拡張アプリケーションの読み込みを実行して標準ＶＭシステムサービス２０８に読み込み要求を送出する（ステップＳ５０６）。読み込み要求を受けた標準ＶＭシステムサービス２０８は拡張アプリケーションデータをＶＭ２０３ｂに読み込ませる（ステップＳ５０７）。具体的には、スレッド情報６００の実行ファイルの情報６０１が指定するファイルをＶＭ２０３ｂに読み込ませる。次いで、ＶＭ２０３ｂは読み込んだ拡張アプリケーションデータに基づいて拡張アプリケーションを実行する（ステップＳ３０８）。具体的には、スレッド情報６００の実行関数の情報６０２が指定する関数を実行する。

【００２８】

図５の処理によれば、ＶＭスレッド２０５ａとは別に新たなＶＭスレッド２０５ｂが拡張アプリケーション２０７ｂの実行に応じて生成され、さらに、新たなＶＭスレッド２０５ｂを利用する新たなＶＭ２０３ｂが生成される。したがって、実質的に複数のスレッド（ＶＭスレッド２０５ａ及びＶＭスレッド２０５ｂ）を利用することができる。その結果、ＶＭが１つのスレッドのみしか利用できない余裕の無い実行環境であっても、非同期処理を実行することができる。

【００２９】

図７は、図１のＭＦＰを含む印刷システムの機器構成の一例を説明するための図である。

【００３０】

図７において、印刷システム７００は、ＭＦＰ１００、ネットワークルータ７０１及び外部装置７０２を備える。ＭＦＰ１００には外部装置７０２に格納されている画像データを取得して印刷するための拡張アプリケーションがインストールされている。ネットワークルータ７０１はＭＦＰ１００と外部装置７０２の通信を仲介する。外部装置７０２は画像データを格納し、ＭＦＰ１００からの画像取得要求に応じてＭＦＰ１００へ画像データを送出する。なお、外部装置７０２はイントラネット上に存在しても、インターネット上に存在してもよい。

【００３１】

図８は、図７における外部装置に格納されている画像データを取得して印刷するための拡張アプリケーションを実行するときの処理の流れを説明するための図である。図８の処理を実行する際、図４の処理が実行されてＶＭスレッド２０５ａ及びＶＭ２０３ａが生成されていることを前提とする。

【００３２】

まず、ＶＭ２０３ａは、図５の処理を実行して拡張アプリケーションの実行に応じたＶＭスレッド２０５ｂを生成し、さらに、ＶＭ２０３ｂを生成する（ステップＳ８０１）。生成されたＶＭスレッド２０５ｂは画像データを取得するための処理のスレッドとしてＶＭ２０３ｂによって利用される。次いで、ＶＭ２０３ａはＶＭ２０３ｂに対して画像データの取得を要求する印刷画像取得処理を行い（ステップＳ８０２）、図９に示す操作画面を表示する操作画面表示を行う（ステップＳ８０３）。さらに、ＶＭ２０３ａは他のスレッドからのイベントを待つイベント待ち処理を行う（ステップＳ８０４）。次いで、ＶＭ２０３ａは何らかのイベントを受け取ると、当該受け取ったイベントの種別を判別する（ステップＳ８０５）。ステップＳ８０５の判別の結果、受け取ったイベントが画像データの取得の完了を示していれば、ステップＳ８０６に進み、取得した画像データを印刷する印刷処理を行う。受け取ったイベントがキャンセルの要求を示していれば、ＶＭ２０３ａはＶＭ２０３ｂに対して終了要求を送出する終了処理を行い（ステップＳ８０７）、その後、ＶＭ２０３ａは処理を終了する。

【００３３】

ステップＳ８０１の実行の結果、生成されたＶＭ２０３ｂは他のスレッドからのイベントを待つイベント待ち処理を行い（ステップＳ８１１）、何らかのイベントを受け取ると、当該受け取ったイベントの種別を判別する（ステップＳ８１２）。ステップＳ８１２の判別の結果、受け取ったイベントが終了の要求を示していれば、ＶＭ２０３ｂは処理を終了する。受け取ったイベントが画像データの取得の要求を示していれば、ステップＳ８１３に進み、外部装置７０２から画像データを取得し、さらに、画像データの取得完了後に画像取得完了通知をＶＭ２０３ａに送出する画像データ取得処理を行う。その後、ＶＭ２０３ｂは処理をステップＳ８１１に戻す。

【００３４】

図１０は、図８のステップＳ８０２やステップＳ８１３において２つのＶＭの間で実行されるメッセージ送信処理を説明するための図である。

【００３５】

まず、ＶＭ２０３ａ及びＶＭ２０３ｂ（メッセージ送信手段）はスレッド管理情報４００を参照する。その後、ＶＭ２０３ａ及びＶＭ２０３ｂは、送信元のＶＭスレッドに対応するアプリケーションの識別番号と送信先のＶＭスレッドに対応するアプリケーションの識別番号とを比較する（ステップＳ１００１）。例えば、ステップＳ８０２では、送信元であるＶＭ２０３ａが利用するＶＭスレッド２０５ａに対応するアプリケーションの識別番号と、送信先であるＶＭ２０３ｂが利用するＶＭスレッド２０５ｂに対応するアプリケーションの識別番号とが比較される。ステップＳ１００１の判別の結果、比較される２つのアプリケーションの識別番号が同じである場合、ＶＭ２０３ａやＶＭ２０３ｂがメッセージを送信する送信処理を行う（ステップＳ１００２）。ステップＳ１００２の送信処理ではスレッド管理情報４００から送信先のキューＩＤ４０３を取得し、取得したキューＩＤ４０３に対応するキューにメッセージを格納し、当該キューを送出する。例えば、ステップＳ８０２では、ＶＭ２０３ａがメッセージとして画像データの取得要求をキューに格納して当該キューをＶＭ２０３ｂへ送出する。その後、ＶＭ２０３ａ及びＶＭ２０３ｂは処理を終了する。ステップＳ１００１の判別の結果、比較される２つのアプリケーションの識別番号が互いに異なる場合、メッセージの送信を行わずにエラーを操作部１０４の画面等に表示する送信エラー処理を行う（ステップＳ１００３）。その後、ＶＭ２０３ａ及びＶＭ２０３ｂは処理を終了する。

【００３６】

図１０の処理によれば、ＶＭ２０３ａが利用するＶＭスレッド２０５ａに対応するアプリケーションの識別番号と、ＶＭ２０３ｂが利用するＶＭスレッド２０５ｂに対応するアプリケーションの識別番号とが比較される。そして、比較される２つのアプリケーションの識別番号が互いに異なる場合、ＶＭ２０３ａ及びＶＭ２０３ｂの間でメッセージの送信が行われない。これにより、ＶＭ２０３ａが利用するＶＭスレッド２０５ａに対応するアプリケーションの識別番号と、ＶＭ２０３ｂが利用するＶＭスレッド２０５ｂに対応するアプリケーションの識別番号とが同一である場合のみ、メッセージの送信が実行される。その結果、いずれのアプリケーションもがメッセージの送信に基づいて制限無く実行されるのを防止することができる。すなわち、実行を所望しないアプリケーションが実行されるのを防止することができる。

【００３７】

図１１は、図１０のステップＳ１００２の各ＶＭ及び拡張ＶＭシステムサービスの間における処理の流れを説明するための図である。図１１の処理は、ステップＳ８０２におけるメッセージ送信を前提とする。

【００３８】

まず、送信元であるＶＭ２０３ａからメッセージを送信するために拡張ＶＭシステムサービス２０９へメッセージ送信依頼を送出する（ステップＳ１１０１）。このとき、ＶＭ２０３ａは、送信先であるＶＭ２０３ｂが利用するＶＭスレッド２０５ｂのスレッドＩＤと、送信したいメッセージ（画像データの取得要求）とを一緒に拡張ＶＭシステムサービス２０９へ送出する。次いで、拡張ＶＭシステムサービス２０９は、メッセージ送信依頼

を受け取ると、図１０の処理を実行する（ステップＳ１１０２）。その後、送信先であるＶＭ２０３ｂはメッセージを受け取るために拡張ＶＭシステムサービス２０９へメッセージ要求を送出する（ステップＳ１１０３）。このとき、ＶＭ２０３ｂは、自身が利用するＶＭスレッド２０５ｂのスレッドＩＤを拡張ＶＭシステムサービス２０９へ送出する。拡張ＶＭシステムサービス２０９は、メッセージ要求を送出したＶＭ２０３ｂが利用するＶＭスレッド２０５ｂのスレッドＩＤを取得し、スレッド管理情報４００に基づいて取得したスレッドＩＤ４０２に対応するキューＩＤ４０３を確認する。次いで、拡張ＶＭシステムサービス２０９は、確認されたキューＩＤ４０３のキューに格納されているメッセージを取得するメッセージ受信処理を行う（ステップＳ１１０４）。

【００３９】

図１２は、外部装置としてのホストＰＣに格納された拡張アプリケーションをＭＦＰにインストールするときの処理の流れを説明するための図である。

【００４０】

まず、ホストＰＣ１２００がＭＦＰ１００に対してインストール画面表示要求を送出する（ステップＳ１２０１）。ＭＦＰ１００は、インストール画面表示要求を受け付け（ステップＳ１２１１）、さらに、インストール画面表示用ＨＴＭＬコンテンツを送信する（ステップＳ１２１２）。ホストＰＣ１２００は、受信したインストール画面表示用ＨＴＭＬコンテンツをホストＰＣ１２００が有する表示部の画面に表示する（ステップＳ１２０２）。さらに、ホストＰＣ１２００は、ＭＦＰ１００へインストールされる拡張アプリケーションを圧縮してＭＦＰ１００へインストールファイルとしてアップロードする（ステップＳ１２０３）。ＭＦＰ１００は、アップロードされたインストールファイルを受信し（ステップＳ１２１３）、さらに、圧縮されたインストールファイルを展開する（ステップＳ１２１４）。次いで、ＭＦＰ１００は、展開されたインストールファイルのプログラムコードをバイトコード化し（ステップＳ１２１５）、バイトコード化されたプログラムコードの自身への書込を開始する（ステップＳ１２１６）。次いで、ＭＦＰ１００は、インストールファイルのインストールが完了したか否かを判別し（ステップＳ１２１７）、インストールが完了すれば、インストール完了通知をホストＰＣ１２００に送信する（ステップＳ１２１８）。インストール完了通知を受信したホストＰＣ１２００は、インストール完了画面を表示部の画面に表示する（ステップＳ１２０４）。その後、ホストＰＣ１２００及びＭＦＰ１００は処理を終了する。

【００４１】

図１２の処理によれば、インストールファイルのプログラムコードがバイトコード化され、バイトコード化されたプログラムコードがＭＦＰ１００へ書き込まれる。すなわち、インストールファイルのプログラムコードがデバイス依存のコードであるバイトコードに変換されるため、プログラムコードの書込先がどのようなデバイスであっても、プログラムコードは当該デバイスに書込可能なコードに変換される。その結果、実行環境（デバイス）に関係なく拡張アプリケーションをインストールすることができる。

【００４２】

以上、本発明について実施の形態を用いて説明したが、本発明は上述した実施の形態に限定されるものではない。

【００４３】

本発明は、上述の実施の形態の１以上の機能を実現するプログラムを、ネットワークや記憶媒体を介してシステムや装置に供給し、そのシステム又は装置のコンピュータの１つ以上のプロセッサーがプログラムを読み出して実行する処理でも実現可能である。また、本発明は、１以上の機能を実現する回路（例えば、ＡＳＩＣ）によっても実現可能である。

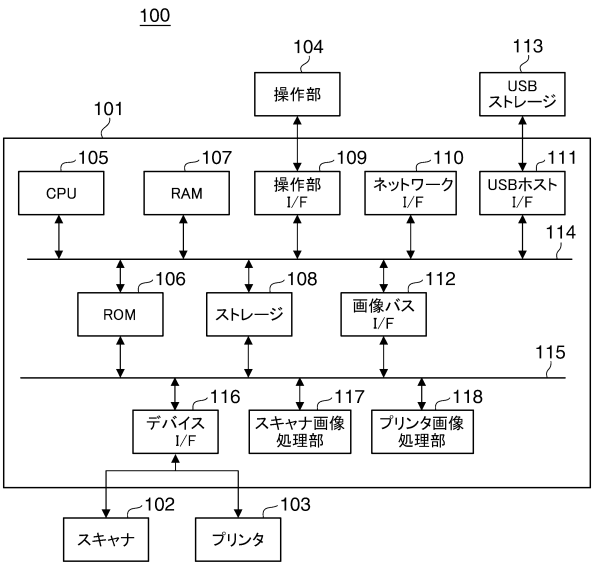
【符号の説明】

【００４４】

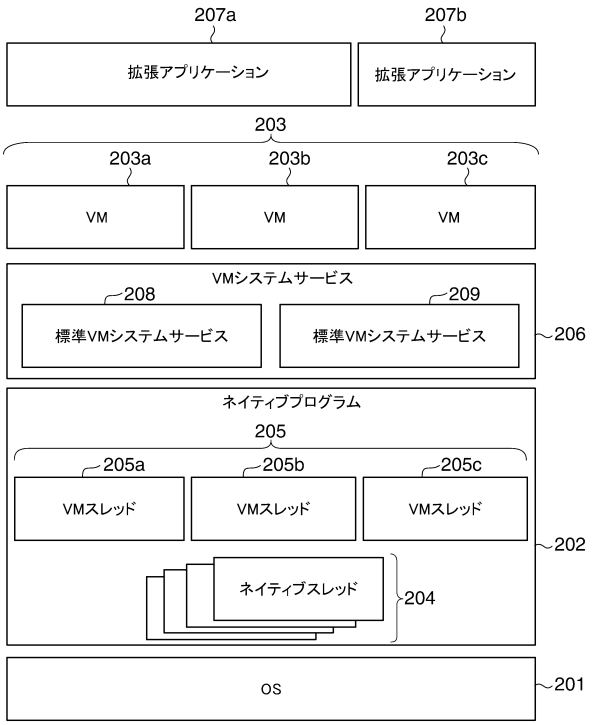
１００　ＭＦＰ
１０５　ＣＰＵ

２０１　ＯＳ
２０３　ＶＭ
２０５　ＶＭスレッド
２０６　ＶＭシステムサービス
２０７ａ,２０７ｂ　拡張アプリケーション
４００　スレッド管理情報

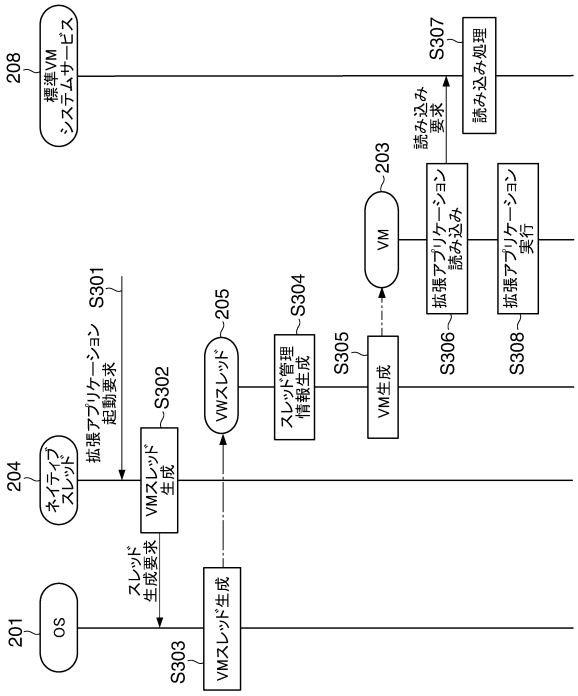
【図１】



【図２】



【図３】

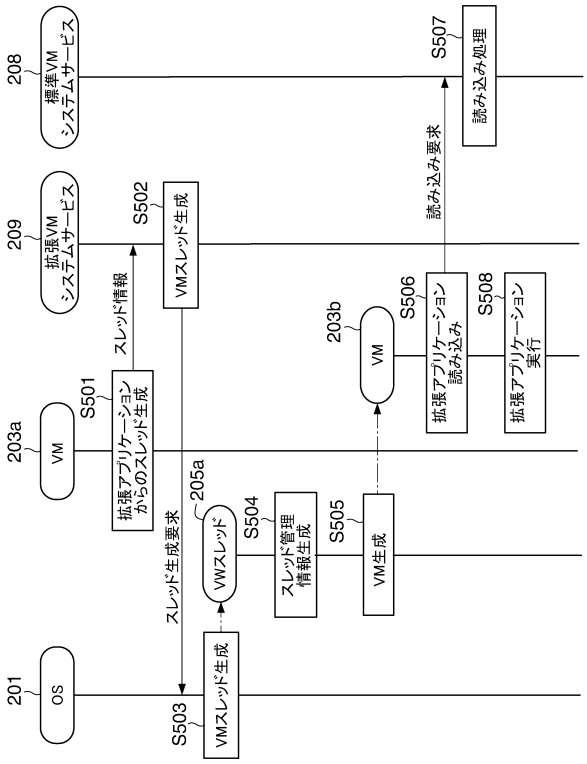


【図４】

400

401	402	403
アプリケーションID	スレッドID	キューID
アプリケーションA	VMスレッド1	キュー1
アプリケーションA	VMスレッド2	キュー2
アプリケーションB	VMスレッド3	キュー3

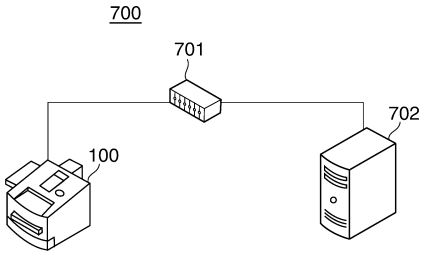
【図５】



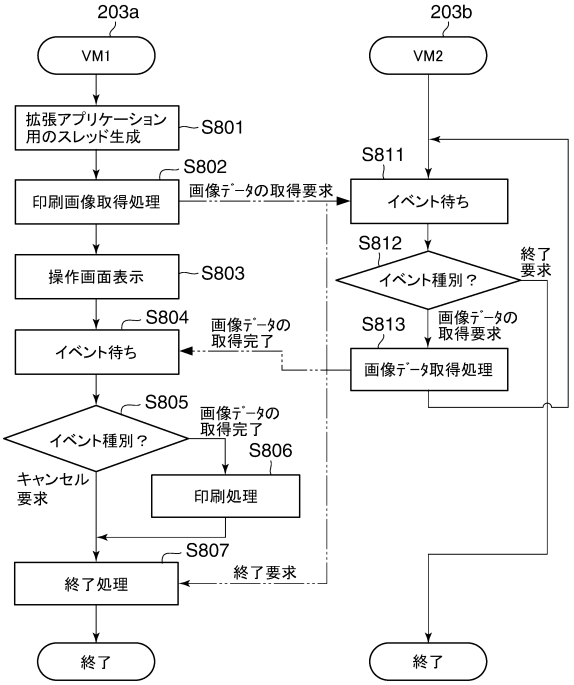
【図６】

601	602	603	604
実行ファイル	実行関数	優先度	スタックサイズ
file1	funcA	10	1024

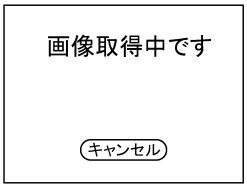
【図７】



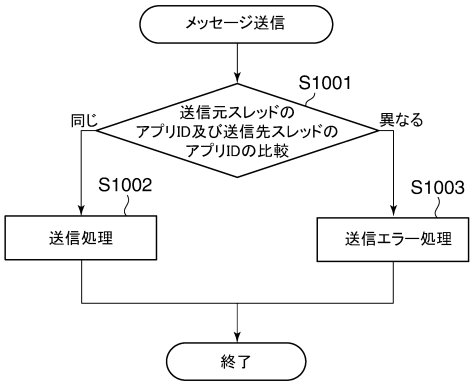
【図８】



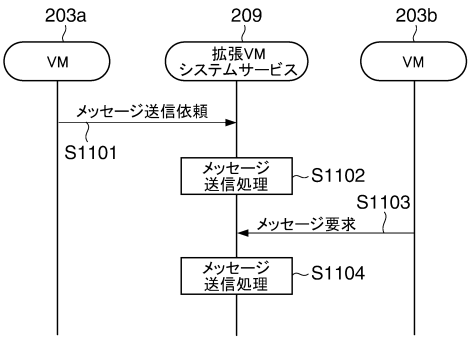
【図９】



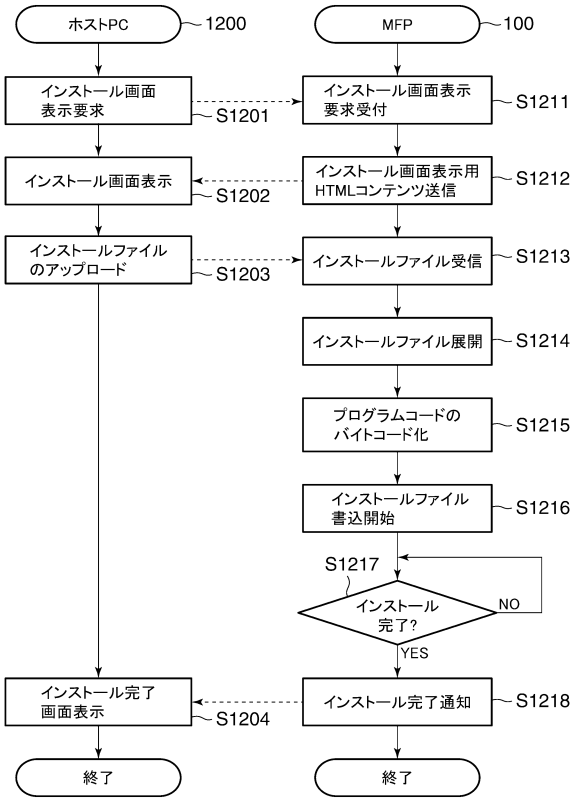
【図１０】



【図１１】



【図１２】



フロントページの続き

(56)参考文献 特表２０１３－５０４８０７（ＪＰ，Ａ）
笹田 耕一，Ｒｕｂｙ用マルチ仮想マシンによる並列処理の実現，情報処理学会論文誌 論文誌トランザクション ２０１１（平成２３）年度 ２ ［ＣＤ－ＲＯＭ］，日本，一般社団法人情報処理学会，２０１２年 ４月１５日，第5巻,第2号，pp.25-42

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４８
Ｇ０６Ｆ ９／４４
Ｇ０６Ｆ ９／４５５