(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6555908号
(P6555908)
(45)発行日　令和1年8月7日(2019.8.7)　(24)登録日　令和1年7月19日(2019.7.19)

(51)Int.Cl.　F I
G06F 9/48 (2006.01)　G06F 9/48 100Z
G06F 9/48 370
G06F 9/48 300H

請求項の数 7　(全 16 頁)

(21)出願番号　特願2015-53658(P2015-53658)
(22)出願日　平成27年3月17日(2015.3.17)
(65)公開番号　特開2016-173746(P2016-173746A)
(43)公開日　平成28年9月29日(2016.9.29)
審査請求日　平成30年3月8日(2018.3.8)

(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
(74)代理人　100121511
弁理士　小田　直
(72)発明者　母里　南陽
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　清木　泰

最終頁に続く

(54)【発明の名称】情報処理装置及びその制御方法、プログラム

(57)【特許請求の範囲】
【請求項１】
モジュールを管理する管理部と、前記管理部の管理下にある第１種のモジュールと、前記管理部の管理下にない第２種のモジュールとを有するプロセスの実行処理が行われる情報処理装置であって、
前記情報処理装置は、前記情報処理装置にて動作するオペレーティングシステムによる管理が為されるシグナル通知関数リストと、前記管理部による管理が為される設定データとを備え、
前記情報処理装置は、前記第２種のモジュールがロードされることに伴って、ロードされる前記第２種のモジュールのために定義されている第２の関数の情報を、前記シグナル通知関数リストに登録し、前記管理部に基づいて、前記第１種のモジュールがロードされることに伴って、前記設定データ内の自モジュール登録関数の管理領域に、ロードされる前記第１種のモジュールのために定義されている第１の関数の情報を記憶し、
前記管理部に基づいて、前記情報処理装置は、前記第１種のモジュールがロードされることに伴って、前記シグナル通知関数リストを取得し、取得した前記シグナル通知関数リストに前記第２の関数の情報が既に登録されており、かつ、前記第１種のモジュールの管理のために定義されている関数が登録されていないと判定した場合、前記設定データ内の他モジュール登録関数の管理領域に前記第２の関数の情報を記憶し、取得した前記シグナル通知関数リストに対し、前記第２の関数の情報に代えて第３の関数の情報を登録するものであり、

前記オペレーティングシステムに基づいて、前記情報処理装置は、前記プロセスの実行中に例外が発生した場合に、前記シグナル通知関数リストに登録されている関数の情報を参照し、前記シグナル通知関数リストに前記第３の関数の情報が登録されている場合に前記管理部に対してシグナルを送信し、

前記管理部に基づいて、前記情報処理装置は、前記シグナルの受信に伴って、前記第３の関数を実行するために、前記設定データ内の前記自モジュール登録関数の管理領域に記憶されている前記第１の関数の情報に対応する、前記第１の関数を呼び出し、さらに、前記設定データ内の前記他モジュール登録関数の管理領域に記憶されている前記第２の関数の情報に対応する、前記第２の関数を呼び出すために前記シグナルを、前記第２の関数に対して送信する

ことを特徴とする情報処理装置。

【請求項２】

前記情報処理装置はひとつの前記プロセス内に複数の前記管理部を備え、複数の前記管理部は前記設定データを共有し、

複数の前記管理部のうちの第１の管理部に基づいて、前記情報処理装置は、前記設定データが存在していない場合に前記設定データを新規に作成し、また、前記設定データが存在する場合には前記設定データに、前記第１の管理部のエントリー情報を登録する

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】

前記第１の管理部に基づいて、前記情報処理装置は、前記設定データから、前記第１の管理部のエントリー情報を削除すべきときに、前記設定データに、複数の前記管理部のうちの第２の管理部のエントリー情報が存在する場合には前記設定データを削除せずに、前記設定データにある、前記第１の管理部のエントリー情報を削除する

ことを特徴とする請求項２に記載の情報処理装置。

【請求項４】

前記エントリー情報は、前記自モジュール登録関数の管理領域にて、複数の前記管理部のうちのどの管理部が登録した関数であるかの判断に用いられる情報でもある

ことを特徴とする請求項２または請求項３に記載の情報処理装置。

【請求項５】

前記管理部に基づいて、前記情報処理装置は、前記情報処理装置のリソース情報を取得し、前記リソース情報に従って、前記例外の発生時に呼び出される複数の前記第１の関数間における順序を、前記第１の関数に対応する前記第１種のモジュールの優先度に従う順序に変更する

ことを特徴とする請求項１乃至４のいずれか１項に記載の情報処理装置。

【請求項６】

モジュールを管理する管理部と、前記管理部の管理下にある第１種のモジュールと、前記管理部の管理下にない第２種のモジュールとを有するプロセスの実行処理が行われる情報処理装置にて実行される制御方法であって、

前記情報処理装置は、前記情報処理装置にて動作するオペレーティングシステムによる管理が為されるシグナル通知関数リストと、前記管理部による管理が為される設定データとを備えるものであり、

前記制御方法は、

前記情報処理装置が、前記第２種のモジュールがロードされることに伴って、ロードされる前記第２種のモジュールのために定義されている第２の関数の情報を、前記シグナル通知関数リストに登録する工程と、

前記管理部に基づいて、前記情報処理装置が、前記第１種のモジュールがロードされることに伴って、前記設定データ内の自モジュール登録関数の管理領域に、ロードされる前記第１種のモジュールのために定義されている第１の関数の情報を記憶する工程と、

前記管理部に基づいて、前記情報処理装置が、前記第１種のモジュールがロードされることに伴って、前記シグナル通知関数リストを取得し、取得した前記シグナル通知関数リ

ストに前記第２の関数の情報が既に登録されており、かつ、前記第１種のモジュールの管理のために定義されている関数が登録されていないと判定した場合、前記設定データ内の他モジュール登録関数の管理領域に前記第２の関数の情報を記憶し、取得した前記シグナル通知関数リストに対し、前記第２の関数の情報に代えて第３の関数の情報を登録する工程と、

前記オペレーティングシステムに基づいて、前記情報処理装置が、前記プロセスの実行中に例外が発生した場合に、前記シグナル通知関数リストに登録されている関数の情報を参照し、前記シグナル通知関数リストに前記第３の関数の情報が登録されている場合に前記管理部に対してシグナルを送信する工程と、

前記管理部に基づいて、前記情報処理装置が、前記シグナルの受信に伴って、前記第３の関数を実行するために、前記設定データ内の前記自モジュール登録関数の管理領域に記憶されている前記第１の関数の情報に対応する、前記第１の関数を呼び出し、さらに、前記設定データ内の前記他モジュール登録関数の管理領域に記憶されている前記第２の関数の情報に対応する、前記第２の関数を呼び出すために前記シグナルを、前記第２の関数に対して送信する工程と、を有する

ことを特徴とする制御方法。

【請求項７】

請求項６に記載した制御方法の各工程をコンピュータに実行させる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、情報処理装置における、プロセス終了時の例外処理の技術に関する。

【背景技術】

【０００２】

情報処理装置が実行するプログラムにおいて、所定範囲から逸脱した値の入力や、メモリのアクセス違反等が原因で、予期せぬ例外が発生する場合がある。例えば、プログラム実行中にプロセスが最終的に異常終了してしまい、メモリに保持されていたデータが失われることがある。ログ情報やレジストリの情報等のように、作業中に保存されていなかったデータの消失が起こり得る。その対策としては、オペレーティングシステム（以下、ＯＳとも記す）が提供する仕組みを使用することで、プロセスが異常終了する前に例外を検知し、プロセスのダンプ情報を取得することが可能である。これは、プログラマが例外の発生原因を究明し、プログラムを修正する場合に有効な手段である。

【０００３】

従来のプロセス異常終了時における例外処理法として、あるモジュールがロードされた際、例外発生時に実施したい処理を定義した関数をＯＳに対して登録しておく方法がある。ＯＳは、プロセスが異常を検知した際、登録された関数にシグナルを送る。例えばダンプ情報の取得処理を定義した関数をＯＳに登録しておくことで、ダンプ情報が取得可能となる。例外処理が必要になった際にシグナルを受けて例外処理を実行する際の登録方法として、特許文献１に開示された方法がある。これは、プログラマのシグナル設定に要する負担を軽減させ、意図したとおりの例外処理を確実にかつ容易に行わせることを目的としている。１つのモジュールから他のモジュールに制御が移る際、登録されていた例外処理の内容を退避し、制御が戻る際に退避しておいた情報を元に戻す処理が行われる。

【先行技術文献】

【特許文献】

【０００４】

【特許文献１】特開平７－３３４３７７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に開示の技術では、シグナルを受けて例外処理を実行する際の関数の呼び出し制御について言及されていない。例えば、プログラムにその作成者以外の者が開発したモジュール（以下、他モジュールともいう）が含まれている場合の例外処理を想定する。プログラムは複数のモジュールから構成されているのが一般的である。プログラムの規模が大きくなるほど、多人数や多くの組織が関与し、単独のプログラマまたはチームが全てのモジュールを実装する場合は少なくなる。他モジュールとしては、インタフェースの一部だけが公開されているが、その他に関して非公開であるために制御できないモジュールと連結してプログラムが構成される場合がある。そのため、例外処理時にて、制御下にないモジュールの関数が先に呼び出されてしまうと、このモジュールの制御によっては、プログラムの作成者の開発したモジュールにシグナルが送信されない場合があり得る。このような場合、一部のモジュールのダンプ情報しか取得できない可能性がある。
【０００６】
本発明は、例外発生時に、実行プロセス内でロードされているモジュールの例外処理を実行することができる情報処理装置とその制御方法、プログラムの提供を目的とする。
【課題を解決するための手段】
【０００７】
本発明に係る情報処理装置は、モジュールを管理する管理部と、前記管理部の管理下にある第１種のモジュールと、前記管理部の管理下にない第２種のモジュールとを有するプロセスの実行処理が行われる情報処理装置であって、前記情報処理装置は、前記情報処理装置にて動作するオペレーティングシステムによる管理が為されるシグナル通知関数リストと、前記管理部による管理が為される設定データとを備える。前記情報処理装置は、前記第２種のモジュールがロードされることに伴って、ロードされる前記第２種のモジュールのために定義されている第２の関数の情報を、前記シグナル通知関数リストに登録し、前記管理部に基づいて、前記第１種のモジュールがロードされることに伴って、前記設定データ内の自モジュール登録関数の管理領域に、ロードされる前記第１種のモジュールのために定義されている第１の関数の情報を記憶し、前記管理部に基づいて、前記情報処理装置は、前記第１種のモジュールがロードされることに伴って、前記シグナル通知関数リストを取得し、取得した前記シグナル通知関数リストに前記第２の関数の情報が既に登録されており、かつ、前記第１種のモジュールの管理のために定義されている関数が登録されていないと判定した場合、前記設定データ内の他モジュール登録関数の管理領域に前記第２の関数の情報を記憶し、取得した前記シグナル通知関数リストに対し、前記第２の関数の情報に代えて第３の関数の情報を登録するものである。前記オペレーティングシステムに基づいて、前記情報処理装置は、前記プロセスの実行中に例外が発生した場合に、前記シグナル通知関数リストに登録されている関数の情報を参照し、前記シグナル通知関数リストに前記第３の関数の情報が登録されている場合に前記管理部に対してシグナルを送信し、前記管理部に基づいて、前記情報処理装置は、前記シグナルの受信に伴って、前記第３の関数を実行するために、前記設定データ内の前記自モジュール登録関数の管理領域に記憶されている前記第１の関数の情報に対応する、前記第１の関数を呼び出し、さらに、前記設定データ内の前記他モジュール登録関数の管理領域に記憶されている前記第２の関数の情報に対応する、前記第２の関数を呼び出すために前記シグナルを、前記第２の関数に対して送信する。

【発明の効果】
【０００８】
本発明によれば、例外発生時に、実行プロセス内でロードされているモジュールの例外処理を実行することができる。
【図面の簡単な説明】
【０００９】
【図１】本発明の実施形態にかかる情報処理装置の構成例を示すブロック図である。

【図２】情報処理装置のハードウェア構成の一例を示すブロック図である。
【図３】モジュール管理部が管理する設定データの構造例を示す図である。
【図４】ＯＳが管理するシグナル通知関数リストの構造例を示す図である。
【図５】第一実施形態における、モジュールロード時にモジュール管理部が行う処理の一例を示すフローチャートである。
【図６】第一実施形態における、モジュールロード時のデータ遷移の一例を示す図である。
【図７】第一実施形態における、例外発生時にモジュール管理部が行う例外処理の一例を示すフローチャートである。
【図８】第二実施形態にかかる情報処理装置の構成例を示すブロック図である。
【図９】第二実施形態における、モジュール管理部が行う関数情報登録処理の一例を示すフローチャートである。
【図１０】第二実施形態における、モジュール管理部が行う設定データ削除処理の一例を示すフローチャートである。
【図１１】第三実施形態における、例外発生時にモジュール管理部が行う処理の一例を示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、本発明の各実施形態について図面を用いて説明する。図１から図４を参照して各実施形態に共通する事項を説明した後、各実施形態の詳細をそれぞれ説明する。
図１は、本発明の実施形態にかかる情報処理装置１のシステム構成例を示すブロック図である。情報処理装置１は各種プログラムを実行するための実行装置であり、オペレーティングシステム（ＯＳ）２上で複数のプロセス３が動作する。ＯＳ２はハードウェアやリソース管理を行う基本ソフトウェア群であり、代表例として、Ｍｉｃｒｏｓｏｆｔ　Ｗｉｎｄｏｗｓ（登録商標）やＬｉｎｕｘ（登録商標）等がある。プロセス３は実行プログラムの単位であり、例えば、複数のモジュール４～６、モジュール管理部７、ランタイムライブラリ８から構成されているものとする。図１に示すモジュール４から６は、モジュールＡからＣをそれぞれ示す。これらのモジュールは、プロセス３よりも小さな実行プログラムの単位を表す。
【００１１】
本実施形態では「自モジュール」と「他モジュール」の２種類のモジュールが存在する。「自モジュール」とは、モジュール管理部７が管理可能なモジュールであることを表す、第１種のモジュールである。一方、「他モジュール」とは、モジュール管理部７の管理外のモジュールであることを表す、第２種のモジュールである。「他モジュール」には、プログラムの作成者以外の者が開発したモジュールが含まれる。図１に例示するように、モジュールＡは他モジュールであり、ランタイムライブラリ８との間で情報の送受を直接行う。これに対して、モジュールＢとモジュールＣは自モジュールであり、モジュール管理部７との間で情報の送受を行う。
【００１２】
モジュール管理部７は、自モジュールを管理する機能を有する。ここでいう管理とは主に、各モジュールの例外処理時に呼び出す関数をＯＳ２へ登録することや、プロセス３が異常を検知した際の各モジュールの例外処理を実行することを意味する。またモジュール管理部７は設定データ９を管理する。モジュール管理部７が管理する設定情報としての設定データ９の詳細については図３を用いて後述する。
【００１３】
ランタイムライブラリ８は、複数のプログラム間で共通する機能等をまとめたモジュールであり、プログラムの実行に必須のライブラリである。ランタイムライブラリ８は、開発されたプログラムを実行する際に連結して利用され、ＯＳ２との間で情報のやり取りを行う。
【００１４】

図２は、情報処理装置１のハードウェア構成例を示すブロック図である。ＣＰＵ（中央演算処理装置）２１は、所定のプログラムを読み出して解釈および実行する。ＲＯＭ（リード・オンリ・メモリ）２６の内部のプログラム領域に記憶されたプログラムや、ハードディスク２３からＲＡＭ（ランダム・アクセス・メモリ）２２にロードされたＯＳ２、汎用アプリケーション等のプログラムの実行処理が行われる。ＲＡＭ２２は、ＣＰＵ２１の主メモリ、ワークエリア等として機能する記憶デバイスである。ハードディスク２３は、ブートプログラム、種々のアプリケーション、フォントデータ、ユーザファイル、電子原稿ファイル等を記憶する記憶媒体である。本実施形態にかかるプログラムは、ハードディスク２３に記憶されており、ＲＡＭ２２に読み出されてＣＰＵ２１によって実行される。

【００１５】

ディスプレイコントローラ２４はディスプレイ（表示部）の表示制御を行う。ネットワークコントローラ２５は、ネットワークに接続された機器との通信処理の制御を行う。外部記憶ドライブ２７はメディア（記録媒体）２８に関するデータの読み出しと書き込みを制御する。キーボードコントローラ２９は、ユーザのキー操作やポインティングデバイス等の操作による操作指示信号の入力処理を行う。

【００１６】

図３は、モジュール管理部７が管理する設定データ９の構造例を示す概念図である。設定データ９は、モジュール管理部７が、例外処理時に呼び出す関数の登録処理や例外処理の実行時に参照し、また更新するデータ構造を有する。設定データ９は、プロセスごとに１つ存在する。設定データ９の記憶領域についてはファイル、メモリ、データベース等のように、データを記憶する機能を有していればよく、特に限定されない。設定データ９は、例えば第１および第２の管理領域を有する。第１の管理領域３１はプロセス情報管理領域であり、第２の管理領域３２はモジュール管理部情報管理領域である。

【００１７】

第１の管理領域３１は、プログラムの実行により動作中のプロセス３に関する情報を管理するための領域である。情報処理装置１にて動作する各プロセスは、プロセスＩＤ３３とプロセス名３４が付与される。第１の管理領域３１には、これら２つの情報が格納される。モジュール管理部７はプロセスＩＤ３３とプロセス名３４を参照することで、自身がどのプロセスで動作しているかを特定することが可能となる。

【００１８】

第２の管理領域３２は、プロセス３内で動作しているモジュール管理部７に関する情報を管理するための領域であり、例外処理を行う際に呼び出される関数について情報を記憶している。第２の管理領域３２は複数の登録関数管理領域を有する。例えば、第２の管理領域３２は、自モジュール登録関数管理領域（以下、自関数管理域と略称する）３５と、他モジュール登録関数管理領域（以下、他関数管理域と略称する）３６という２つの領域を有する。

【００１９】

自関数管理域３５は、自モジュール、すなわちモジュール管理部７の管理下にあるモジュールから依頼された、例外処理時に呼び出されるべき関数の情報が記憶される領域である。一方、他関数管理域３６は、他モジュール、すなわちモジュール管理部７の管理下にないモジュールがＯＳ２に登録していた、例外処理時に呼び出されるべき関数の情報が記憶される領域である。これらの領域には、関数ＩＤ３７と関数ポインタ３８がそれぞれ記憶される。関数ＩＤ３７は、ＯＳ２によって付与された、登録された関数を一意に識別するための識別子を表す。関数ポインタ３８は、登録された関数がロードされているメモリの先頭アドレスを表す。例外処理の発生時には、自関数管理域３５や他関数管理域３６の記憶情報を参照して、関数を呼び出す処理が実行される。

【００２０】

図４はＯＳ２が管理するシグナル通知関数リスト１０について構造の一例を示す。シグナル通知関数リスト１０は、例外が発生した際にシグナルが送信される関数のリストを管理するデータであり、各プロセスに１つのデータが存在する。シグナル通知関数リスト１

０の記憶領域についてはファイル、メモリ、データベース等、データを記憶する機能を有していれば特に限定されない。シグナル通知関数リスト１０の登録や更新については、プロセス３内の各モジュールまたはモジュール管理部７が、ランタイムライブラリ８を介して、ＯＳ２との間で通信することで可能である。シグナル通知関数リスト１０では主に、関数ＩＤ３７と関数ポインタ３８の２つの情報が管理される。図４に示す例では、全部で３つの関数が登録されている。例外処理が発生するとリストの先頭から関数が順次実行される。つまり、この場合には、まず関数ＩＤ３７が「００１」である１番目の関数が呼び出される。この関数の実行が完了し、次の制御で関数ＩＤ３７が「００２」である２番目の関数が呼び出され、その次に３番目の関数が呼び出されるという仕組みとなっている。

【００２１】

［第一実施形態］

以下に、本発明の第一実施形態を説明する。本実施形態にて、モジュール管理部７が行う一連の処理例について、図５に示すフローチャートを参照して説明する。本実施形態では図１に示すように、プロセス３内で３つのモジュールＡ、モジュールＢ、モジュールＣがロードされてそれぞれ実行される場合について説明する。以下では、モジュールがロードされる順番については、他モジュールであるモジュールＡ、自モジュールであるモジュールＢ、自モジュールであるモジュールＣの順であるものとする。さらに、モジュールＡ、モジュールＢ、モジュールＣにはそれぞれ関数Ａ、関数Ｂ、関数Ｃが定義されており、これらの関数は例外発生時に行われる処理（例えばダンプ情報の取得処理）が定義されているものとする。また、モジュール管理部７には関数Ｄが定義されており、例外発生時に呼び出される自モジュールの登録済み関数を順番に呼び出す処理が実行される。

【００２２】

図５において、あるモジュールがロードされ、モジュール管理部７が起動すると、Ｓ１に進む。Ｓ１にてモジュール管理部７は、例外発生時に呼び出したい関数の登録依頼をモジュールから受信する。その際、モジュール管理部７は設定データ９のインスタンスを新規に作成し、受信した関数の情報（関数情報）を設定データ９に登録する。このとき、モジュール管理部７が登録依頼を受けたモジュールは、モジュール管理部７の管理下にあるとする。つまり、このモジュールは自ジュールであり、関数情報の登録依頼を受けたモジュール管理部７は自関数管理域３５への登録を行う。

【００２３】

次のＳ２においてモジュール管理部７は、ランタイムライブラリ８を介してＯＳ２からシグナル通知関数リスト１０を取得する。そしてＳ３においてモジュール管理部７は、Ｓ２で取得したシグナル通知関数リスト１０内に関数が既に登録されているか否かを判定する。シグナル通知関数リスト１０内に関数が既に登録されている場合、Ｓ４に処理を進め、関数が登録されていない場合にはＳ６に移行する。

【００２４】

Ｓ４においてモジュール管理部７は、シグナル通知関数リスト１０に登録されている関数が自身の関数（管理下にある関数）であるかどうかを判断する。判断対象となる関数が自身の関数ではない場合、Ｓ５に進み、モジュール管理部７は、既に登録されていた関数情報を設定データ９に登録し、Ｓ６に進む。このとき、シグナル通知関数リスト１０に既に登録されていた関数はモジュール管理部７の管理下にない他モジュールの関数である。このため、モジュール管理部７は他関数管理域３６への登録を行う。一方、Ｓ４においてモジュール管理部７が、判断対象の関数が自身の関数であると判断した場合、この関数を再度登録する必要は無い。このため、特に何もせずに処理を終了する。

【００２５】

Ｓ６においてモジュール管理部７は、シグナル通知関数リスト１０に自身の関数を登録する。自モジュールから登録依頼を受けた関数の場合、関数の内容は、設定データ９内の自関数管理域３５に記憶されている関数リストにある。この場合、自身の関数が呼び出されると、この関数リストに登録されている関数が順番に呼び出されてＣＰＵ２１によって実行される。

【００２６】
以上のように、プロセス３内でモジュールがロードされる度に、モジュール管理部７はＯＳ内に登録されているシグナル通知関数リスト１０を取得する。自身の関数以外で登録されている関数が存在している場合にモジュール管理部７は該関数リストの情報を記憶手段に記憶し、代わりに自身の関数の情報を登録する。これにより、プロセス３で例外が発生した際に、必ずモジュール管理部７が定義する関数が呼び出されることになる。
【００２７】
以下、図６を参照して具体的に説明する。図６は、本実施形態におけるモジュールロード時の設定データ９及びシグナル通知関数リスト１０のデータ遷移の一例を示す。図６（Ａ－１）～（Ａ－３）は、設定データ９内の第２の管理領域３２に関するデータ遷移を例示する。図６（Ａ－１）に示すように、初期状態において自関数管理域３５および他関数管理域３６のいずれにもデータが存在しない状態である。また図６（Ｂ－１）～（Ｂ－３）は、シグナル通知関数リスト１０のデータ遷移を例示する。図６（Ｂ－１）に示すように、初期状態は登録されている関数リストの情報が存在しない状態である。
【００２８】
まずモジュールＡがロードされるものとする。モジュールＡは他モジュールであり、モジュール管理部７を介さないため、第２の管理領域３２内のデータに変更は発生しない。次にモジュールＢがロードされるものとする。モジュールＢは自モジュールであるため、モジュール管理部７に対して関数Ｂの登録依頼を行う。これにより、図５のＳ１にてモジュール管理部７は、設定データ９内の自関数管理域３５に関数Ｂの情報を記憶する。その後、図５のＳ２においてモジュール管理部７はシグナル通知関数リスト１０を取得する。この結果、図６（Ｂ－２）に示すように、他モジュールＡの関数Ａが登録されているため、図５のＳ３からＳ４に処理を進める。Ｓ４からＳ５に進んでモジュール管理部７は、図６（Ａ－２）に示すように、他関数管理域３６に関数Ａの情報を記憶する。図５のＳ６でモジュール管理部７は自身の関数Ｄをシグナル通知関数リスト１０に登録する。
【００２９】
最後にモジュールＣがロードされるものとする。モジュールＣは自モジュールであり、モジュールＢの場合と同様にモジュール管理部７に対して関数Ｃの登録依頼を行う。これにより、図６（Ａ－３）に示すようにモジュール管理部７は、図５のＳ１において設定データ９内の自関数管理域３５に関数Ｃの情報を記憶する。またモジュールＢのロード時と同様、図５のＳ２においてシグナル通知関数リスト１０を取得する処理が行われるが、図６（Ｂ－３）に示すようにモジュール管理部７自身の関数Ｄが既に登録されている。このため、図６（Ａ－３）に示すように、他関数管理域３６の情報の更新は行われない。
【００３０】
図６（Ｂ－１）～（Ｂ－３）を参照すると、登録されている関数リストが存在しない状態である図６（Ｂ－１）の初期状態から、モジュールＡがロードされた際、図６（Ｂ－２）の状態に遷移する。これは、モジュールＡが他モジュールであり、モジュール管理部７を介さずにシグナル通知関数リスト１０への登録が行われるからである。つまり、シグナル通知関数リスト１０には関数Ａの情報が登録される。
【００３１】
次に、モジュールＢのロード時にモジュール管理部７は、図５のＳ６においてシグナル通知関数リスト１０の情報を更新する。このため、図６（Ｂ－３）に示すように、シグナル通知関数リスト１０にはモジュール管理部７自身の関数Ｄが登録される。最後にモジュールＣがロードされる際、モジュールＢのロード時と同様にモジュール管理部７はシグナル通知関数リスト１０の情報を参照するが、登録されている関数はモジュール管理部７自身が登録した関数である。そのため、シグナル通知関数リスト１０の更新は行われず、図６（Ｂ－３）の状態のままである。
【００３２】
以上のように、図６ではモジュールＡ、Ｂ、Ｃが順次にロードされた時のデータ遷移について説明した。他方、モジュールがアンロードされる際の処理は、ロード時とは逆のデ

ータ遷移となる。例えば図６（Ａ－３）および（Ｂ－３）に示すように、モジュールＡ、Ｂ、Ｃの全てがロードされている状態を想定する。この状態においてモジュールＣがアンロードされた場合、第２の管理領域３２は図６（Ａ－３）から図６（Ａ－２）の状態へ遷移する。その際、シグナル通知関数リスト１０は図６（Ｂ－３）から（Ｂ－２）の状態へ遷移する。

【００３３】

次に、図７を参照して、本実施形態における例外処理を説明する。図７は、例外発生時にモジュール管理部７が行う例外処理の一例を示すフローチャートである。

例外の発生はプロセス３内のランタイムライブラリ８によって検知される。例外の発生が検知されるとＯＳ２は、シグナル通知関数リスト１０に登録されている関数に対して順次シグナルを送信する。シグナル通知関数リスト１０にモジュール管理部７の関数が登録されていた場合、Ｓ１１においてモジュール管理部７は、ＯＳ２が送信したシグナルを受信する。次のＳ１２において、登録されていた関数の処理が実行される。本実施形態の場合、モジュール管理部７によって定義されている関数Ｄが呼び出される。関数Ｄについては、自関数管理域３５に登録されている関数リストを順次に呼び出す処理内容である。このため、本実施形態では図６（Ａ－３）で示すように関数Ｂ、関数Ｃの順で実行される。Ｓ１２の完了後、Ｓ１３に処理を進め、モジュール管理部７は設定データ９内の他関数管理域３６に関数が登録されているか否かを判定する。他関数管理域３６に関数が登録されている場合、Ｓ１４に進み、他関数管理域３６に関数が登録されていない場合、Ｓ１５に移行する。

【００３４】

Ｓ１４においてモジュール管理部７は、他関数管理域３６に登録されている関数を呼び出す。その際、Ｓ１１で受信したシグナルと同一のシグナルが当該関数に送信される。本実施形態では図６（Ａ－３）で示すように関数Ａの処理が実行される。そしてＳ１５に進む。Ｓ１５にて設定データ９の削除処理が実行される。設定データ９の削除処理は、プロセス３の終了後にメモリ上やファイルとしてデータが残存してしまうことを避ける目的で実行される。Ｓ１５の処理後にはランタイムライブラリ８によってプロセス３の終了処理が実行される。

【００３５】

以上のようにプロセス内のあるモジュールがロードされた際、モジュール管理部７はシグナルの受信登録が行われている関数の情報を記憶領域に記憶し、モジュール管理部自身の管理下にある関数を上書きする。これにより、プロセスで例外が発生した際に関数呼び出しの制御が可能となり、例外発生時に全ての関数に対して同一のシグナルを通知することができ、プロセス異常終了時のダンプ情報を取得可能となる。

【００３６】

本実施形態では、管理下にある第１種のモジュール（自モジュール）と、管理下にない第２種のモジュール（他モジュール）が区別して管理される。例外処理においては、第２種のモジュールの関数が登録されていた場合、当該関数の情報を記憶手段に記憶する処理が実行され、当該関数に代えて、第１種のモジュールの関数の処理を実行させる関数の登録が行われる。関数の登録では、例えば第１種のモジュールの関数を呼び出す処理を行う関数、または第１種のモジュールの関数の情報が関数リストに記載される。

【００３７】

本実施形態では、モジュール管理部がプロセス内に１つだけ存在する場合を例示して、モジュールのロード時及び例外発生時のモジュール管理部の処理とデータ遷移の様子を説明した。モジュール管理部７はモジュールがロードされる際に、シグナル通知関数リスト１０を監視し、他モジュールの関数が登録済みであるかどうかを判断する。他モジュールの関数がシグナル通知関数リスト１０に登録されている場合には、自モジュールの関数の処理を実行させる関数に置き換えることで、例外発生時にモジュール管理部７の関数が呼び出されることになる。また、設定データ９内では、自モジュールと他モジュールとで管理領域が区別されて情報管理が行われるので、自モジュールの関数の呼び出し後に他モジ

ュール関数の呼び出しが可能となる。これにより、自モジュールの例外処理を行えるとともに、他モジュールの例外処理をも行うことが可能となる。

【００３８】

［第二実施形態］

次に、本発明の第二実施形態における情報処理装置の構成及び処理等について説明する。本実施形態では、１つのプロセス内に複数のモジュール管理部が存在する場合を例示して説明を行う。尚、第一実施形態の場合と同様の構成要素については既に使用した符号を用いることにより、それらの詳細な説明を省略し、主に相違点を説明する。このような説明の省略の仕方は後述の実施形態においても同じである。

【００３９】

図８は、本実施形態における情報処理装置の構成例を示すブロック図である。第一実施形態との相違点は、１つのプロセス３にてモジュール管理部７が複数のインスタンスとして存在している点である。図８では、モジュール管理部７Ａ（モジュール管理部Ａ）とモジュール管理部７Ｂ（モジュール管理部Ｂ）を例示する。モジュール管理部ＡはモジュールＢ（モジュール５）とモジュールＣ（モジュール６）を管理し、モジュール管理部ＢはモジュールＤ（モジュール１１）とモジュールＥ（モジュール１２）を管理しているものとする。また設定データ９については、１つのプロセス３内に複数のモジュール管理部が存在する場合でもインスタンスは１つである。モジュール管理部ごとに設定データ９を独立して管理する方法の場合、各モジュール管理部が管理しているモジュール群の例外発生時の呼び出し順序が独立してしまい制御できない。このため、複数のモジュール管理部が１つの設定データ９を共有する構成となっている。

【００４０】

図８の構成の場合、設定データ９は１つであるため、例えばモジュール管理部Ａが既に起動しているにもかかわらず、モジュール管理部Ｂが設定データ９を新規作成してしまうと既存の設定データ９内に予め存在していた情報が失われてしまう。この場合、例外発生時に一部の関数が呼び出されない可能性がある。また、例えばモジュール管理部Ｂがアンロードされる際、まだモジュール管理部Ａが起動しているにもかかわらず設定データ９を削除してしまうと、設定データ９内の情報が失われてしまう。そこで本実施形態では、１つのプロセス３内に複数のモジュール管理部７が存在する場合において、設定データ９の更新および削除処理について、図９および図１０を参照して説明する。

【００４１】

図９は、モジュール管理部７が行う関数情報登録処理の一例を示すフローチャートである。図９に示す処理は図５のＳ１の処理に関し、プロセス３内のあるモジュールがロードされた際に、例えばモジュール管理部７Ａによって実行される。

まずＳ２１においてモジュール管理部７Ａは、設定データ９が既に存在するか否かを判定する。設定データ９が既に存在するということは、既に他のモジュール管理部７Ｂがロードされ、設定データ９を作成したということになる。このため、設定データ９が既に存在する場合、設定データ９の新規作成は行わずにＳ２３に進む。一方、設定データ９が存在しない場合には、モジュール管理部７Ａは自身が最初にロードされたモジュール管理部であるとみなし、Ｓ２２に移行する。Ｓ２２では最初にロードされたモジュール管理部７Ａが設定データ９を新規に作成する。そしてＳ２３に進む。

【００４２】

Ｓ２３でモジュール管理部７Ａは、設定データ９内に自身のエントリー情報を登録する。エントリー情報とは、プロセス３内にロードされているモジュール管理部のインスタンスに関する情報である。エントリー情報は、複数のモジュール管理部のうち、第１のモジュール管理部が、第２のモジュール管理部の存否、つまり第２のモジュール管理部が既に起動されて存在しているかどうかを判定できるようにするために利用される。またエントリー情報は、自関数管理域３５において、どのモジュール管理部が登録した関数であるかを判断するための情報としても必要である。モジュール管理部のエントリー情報としては、例えばモジュールＩＤとモジュール名が設定データ９内に登録される。その後、Ｓ２４

においてモジュール管理部７Ａは、自モジュールから登録依頼があった関数の情報を設定データ９内の自関数管理域３５に追記する。その際には、どのモジュール管理部が設定した情報かを判別できるように、モジュール管理部自身のエントリー情報を付与する処理が行われる。

【００４３】

次に図１０を参照して、本実施形態におけるモジュール管理部が行う設定データ９の削除処理例を説明する。図１０のフローチャートに示す処理は、図７のＳ１５の処理に関するものであり、プロセス３にて例外発生時に、モジュール管理部７によって実行される。

まずＳ３１において、例えばモジュール管理部７Ａは、設定データ９中に登録されているモジュール管理部のエントリー情報を参照する。その中に他のモジュール管理部７Ｂのエントリー情報が存在するか否かをモジュール管理部７Ａが判定する。他のモジュール管理部７Ｂのエントリー情報が存在するということは、まだ他のモジュール管理部７Ｂがロードされている状態である。よって、この場合には設定データは削除せず、Ｓ３３に進む。Ｓ３３でモジュール管理部７Ａは自身のエントリー情報を削除する。一方、他のモジュール管理部７Ｂのエントリー情報が存在しない場合、自身以外にロードされているモジュール管理部は存在していない。よってＳ３１からＳ３２に移行し、モジュール管理部７Ａは設定データ９の全体を削除する。

【００４４】

以上のように、本実施形態では、複数のモジュール管理部が同一プロセス内に存在し、１つの設定データ９を参照する際、他のモジュール管理部が起動中か否かについて判断される。判断結果にしたがって設定データ９の新規作成、削除等の処理が実行される。本実施形態によれば、設定データ９の上書きや削除により、登録されている関数の情報が失われる可能性がないため、適切な例外処理を実行することができる。

【００４５】

［第三実施形態］

次に、第三実施形態における情報処理装置の処理について説明する。第一実施形態ではプロセス内での例外発生時、登録されていた関数を呼び出す処理について説明した。ここで一例として、例外発生時にメモリ容量が充分でない状態で関数の呼び出しを実施しようとする場合を想定する。この場合に、関数の呼び出しが多すぎたとき等にスタックオーバーフローが発生すると、全ての関数の呼び出しが完了しないままプロセスが終了してしまう可能性がある。このような場合、他より重要なモジュールのダンプ情報が取得できず、その他のモジュールのダンプ情報しか取得できていないという事態に陥ることが懸念される。そこで第三実施形態では、例えばパーソナルコンピュータ（ＰＣ）等の情報処理装置にてそのリソース状況が圧迫されている場合において、重要度の高い関数から順番に呼び出しを行う処理について説明する。

【００４６】

図１１のフローチャートを参照して、本実施形態における例外発生時にモジュール管理部７が行う処理の一例を説明する。図１１において、Ｓ１１からＳ１５の各ステップに示す処理は、第一実施形態にて図７で説明した通りである。よって、Ｓ１１からＳ１５の各ステップについての詳細な説明は割愛し、相違点であるＳ４１からＳ４４の処理を中心に説明する。

【００４７】

まずランタイムライブラリ８が例外を検知し、ＯＳ２からシグナルが送信され、Ｓ１１でモジュール管理部７がシグナルを受信すると、Ｓ４１に進む。Ｓ４１でモジュール管理部７は、情報処理装置１のリソース情報を取得する。リソース情報とは、例えばＣＰＵ２１、ＲＡＭ２２、ハードディスク２３等の使用率である。次のＳ４２においてモジュール管理部７は、Ｓ４１で取得したリソース情報の値を所定の閾値と比較する。この閾値については、全ての関数の呼び出しが行えない可能性のあるリソース状況に応じて設定される。その設定方法として、予め固定値を設定する方法と、登録された関数の数等に応じて動的に決定される可変値を設定する方法がある。Ｓ４２にてリソース情報の値が閾値を超え

ているか否かについて判定処理が行われ、閾値を超えている場合にはＳ４３に進み、閾値を超えていない場合にはＳ１２に移行する。

【００４８】

Ｓ４３においてモジュール管理部７は、各モジュールの重要度の情報を取得する。具体的には、モジュールごとに予め重要度が設定されているものとし、例えばモジュール管理部７は、各モジュールの重要度を表す値を本ステップで参照する。そしてＳ４４においてモジュール管理部７は、設定データ９を更新し、自関数管理域３５に登録されている関数を重要度が高い順番に並べ替えて呼び出し順序を変更する。例えば、重要度を表す値が大きいほどモジュールの重要度が高い場合には、重要度を表す値に従って降順に関数リストのソーティング処理が実行される。そしてＳ１２以降に処理を進める。Ｓ１２では関数リスト順に関数の処理が実行されるため、重要度（つまり、優先度）の高い関数から順番に呼び出しが行われる。

【００４９】

本実施形態では、例外発生時の情報処理装置のリソース状況（切迫度や余裕度等）に応じて、関数の呼び出し順序を制御することで、より重要度の高いモジュールの情報を優先的に取得することができる。重要度の設定に関する基準としては、例えば開発段階でのバグやエラー等が発生する可能性が高いモジュール順や、規模の大きいモジュール順に序列が決定されるものとする。したがって、仮に全てのダンプ情報が取得できなかった場合でも、例外の発生原因の解析に有用な情報の取得の可能性を高めることができる。

【００５０】

［その他の実施形態］

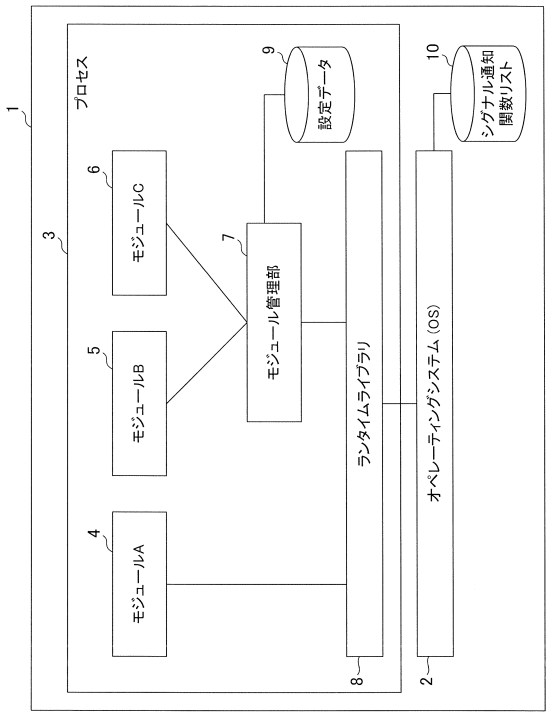
本発明は、上述の実施形態の１以上の機能を実現するプログラムを、ネットワーク又は記憶媒体を介してシステム又は装置に供給し、そのシステム又は装置のコンピュータにおける１つ以上のプロセッサーがプログラムを読出し実行する処理でも実現可能である。また、１以上の機能を実現する回路（例えば、ＡＳＩＣ）によっても実現可能である。

【符号の説明】

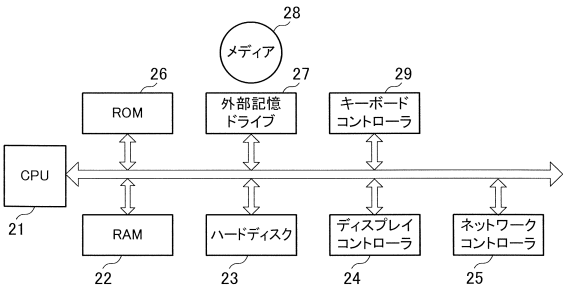
【００５１】

１　情報処理装置
２　オペレーティングシステム
７　モジュール管理部
９　設定データ
１０　シグナル通知関数リスト

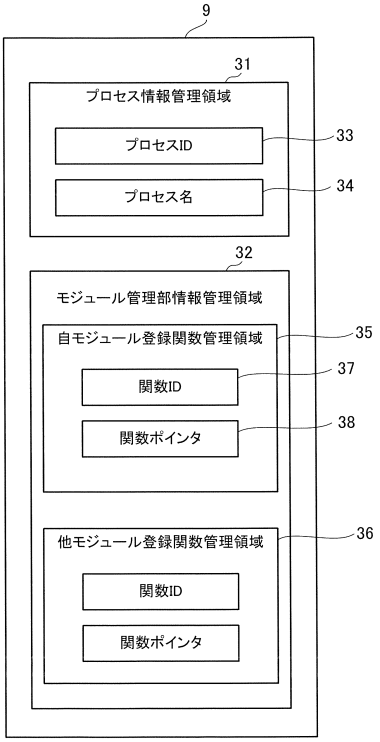
【図１】



【図２】



【図３】

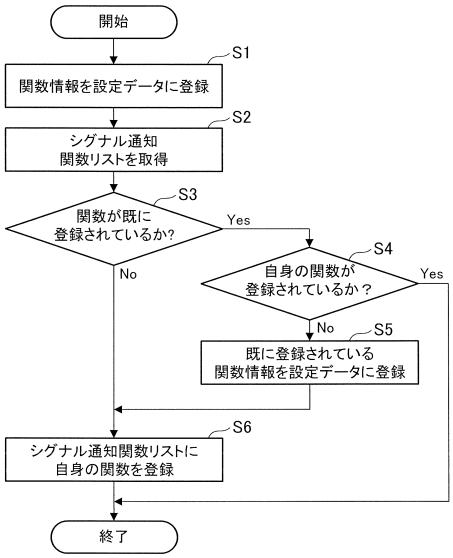


【図４】

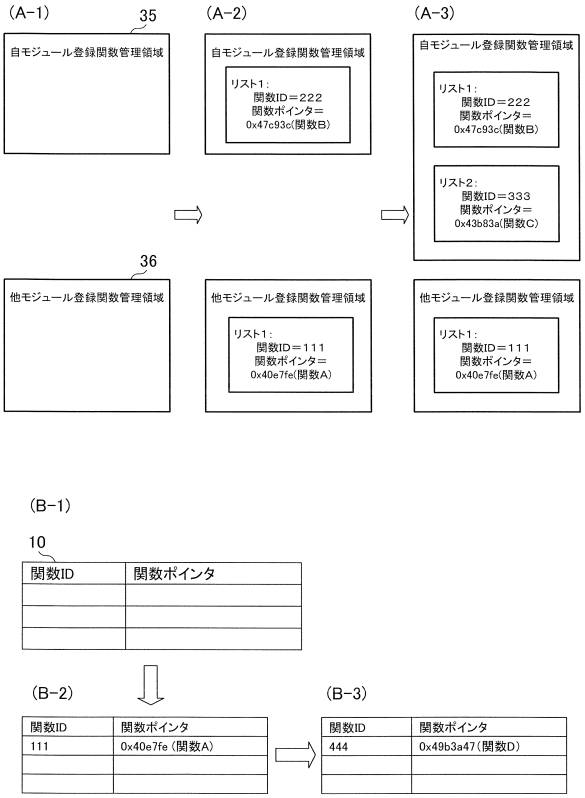
10

37 関数ID	38 関数ポインタ
001	0x40e7fe
002	0x47c93c
003	0x43b83a

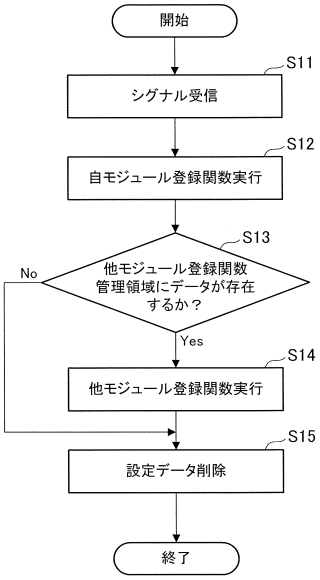
【図５】



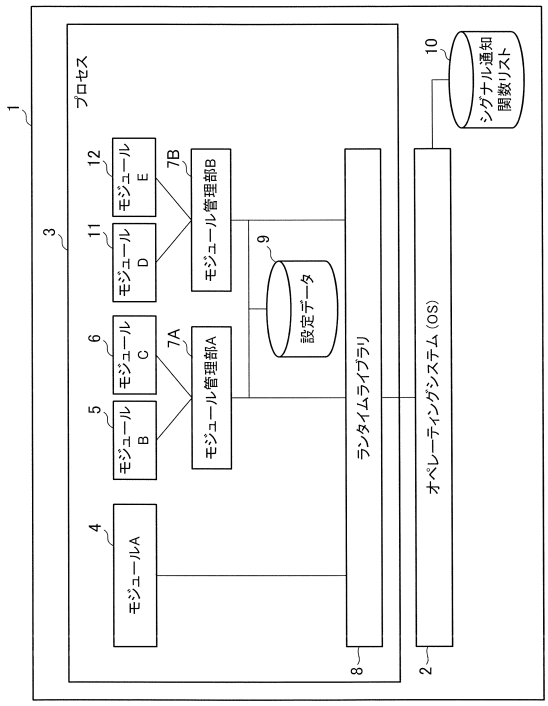
【図６】



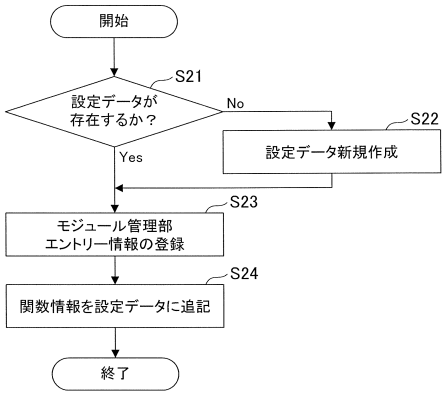
【図７】



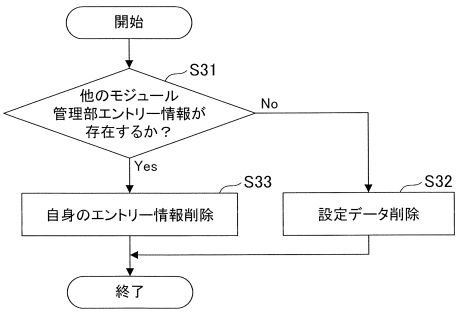
【図８】



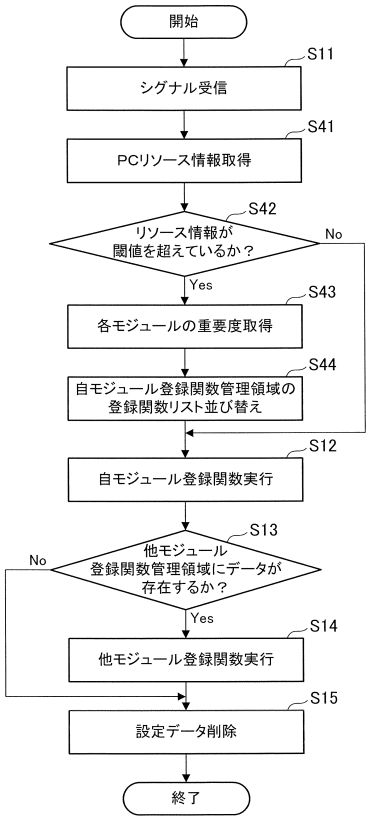
【図９】



【図１０】



【図１１】


開始
S11
シグナル受信
S41
PCリソース情報取得
S42
リソース情報が
閾値を超えているか？
No
Yes
S43
各モジュールの重要度取得
S44
自モジュール登録関数管理領域の
登録関数リスト並び替え
S12
自モジュール登録関数実行
S13
他モジュール
登録関数管理領域にデータが
存在するか？
No
Yes
S14
他モジュール登録関数実行
S15
設定データ削除
終了

フロントページの続き

(56)参考文献 特開平１０－２７１４８０（ＪＰ，Ａ）
特開平０７－３３４３７７（ＪＰ，Ａ）
特開平０４－０９７４３５（ＪＰ，Ａ）
米国特許第０６５９４７７４（ＵＳ，Ｂ１）
米国特許第０５５２６４８５（ＵＳ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ ９／４４－ ９／５４
Ｇ０６Ｆ１１／０７
Ｇ０６Ｆ１１／２８－１１／３６
Ｇ０６Ｆ ８／００－ ８／３８
Ｇ０６Ｆ ８／６０－ ８／７７
Ｇ０６Ｆ ９／００－ ９／０６
Ｇ０６Ｆ ９／３０－ ９／３５５
Ｂ４１Ｊ２９／００－２９／７０
Ｈ０４Ｎ １／００