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

特許第4903879号
(P4903879)

(45)発行日　平成24年3月28日(2012.3.28)　(24)登録日　平成24年1月13日(2012.1.13)

(51)Int.Cl.　F I

G06F 9/54	(2006.01)	G06F	9/06	640C
G06F 9/445	(2006.01)	G06F	9/06	610M
G06F 11/00	(2006.01)	G06F	9/06	630B
G06F 11/30	(2006.01)	G06F	11/30	K

請求項の数 11　(全 29 頁)

(21)出願番号　特願2009-539529 (P2009-539529)
(86)(22)出願日　平成19年11月30日 (2007.11.30)
(65)公表番号　特表2010-511940 (P2010-511940A)
(43)公表日　平成22年4月15日 (2010.4.15)
(86)国際出願番号　PCT/US2007/086195
(87)国際公開番号　WO2008/070587
(87)国際公開日　平成20年6月12日 (2008.6.12)
審査請求日　平成22年10月6日 (2010.10.6)
(31)優先権主張番号　11/566,170
(32)優先日　平成18年12月1日 (2006.12.1)
(33)優先権主張国　米国(US)

早期審査対象出願

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)復代理人　100115624
弁理士　濱中　淳宏
(74)復代理人　100129171
弁理士　柿沼　健一
(74)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫

最終頁に続く

(54)【発明の名称】システム解析および管理

(57)【特許請求の範囲】

【請求項１】

プロセッサおよびシステムメモリを有するコンピューティングデバイスで実行する方法であって、

前記プロセッサが、前記コンピューティングデバイスにロードされたプログラムをカタログ化することと、

前記プロセッサが前記プログラムを実行する前に、

前記プロセッサが、実行する前記プログラムについての属性に関してログファイルにクエリーして、前記プログラムの実行に先行して前記システムメモリ内にロードされ且つ前記コンピューティングデバイスに登録されている前記プログラムと当該属性により関連付けられたファイルの最後のロード時刻を獲得し、

前記プロセッサが、前記ログファイルにクエリーして、前記プログラムの実行に先行してシステムメモリ内にロードされ且つ前記コンピューティングデバイスに登録されている前記プログラムと前記属性により関連付けられたファイルの最後の更新時刻を獲得し、

前記プロセッサが、前記最後の更新時刻を、前記コンピューティングデバイスに登録されている前記プログラムに関連付けられたファイルの前記最後のロード時刻と比較し、

前記プロセッサが、比較中に見つけられた不整合を記録することであって、前記プログラムは最後に更新されたファイルで実行されない旨のリポートを含む、ことと

を備えることを特徴とする方法。

【請求項２】

前記プロセッサがカタログ化することは、前記プロセッサが、前記コンピューティングデバイスのオペレーティングシステムに登録されているプログラムのリストを作成することを含むことを特徴とする請求項１に記載の方法。

【請求項３】

前記プロセッサがカタログ化することは、前記プロセッサが、前記プログラムをリスト内に配置することによって前記プログラムを列挙することを含むことを特徴とする請求項１に記載の方法。

【請求項４】

前記プロセッサがカタログ化することは、前記プロセッサが、前記コンピューティングデバイスに登録されているプログラムすべてを検出するようにスキャンすることを含むことを特徴とする請求項１に記載の方法。

【請求項５】

コンピューティングデバイスであって、

プロセッサユニットおよびシステムメモリと、

前記コンピューティングデバイスに登録されている前記プログラムに関連付けられたファイルの最後のロード時刻、および前記コンピューティングデバイスに登録されている前記プログラムに関連付けられたファイルの最後の更新時刻を記録するように構成されたログストレージコンポーネントと、

前記コンピューティングデバイス上にロードされたプログラムをカタログ化し、前記プログラムの実行前に、実行する当該プログラムについての属性に関してログストレージコンポーネントにクエリーして、前記プログラムの実行に先行して前記システムメモリ内にロードされ且つ前記コンピューティングデバイスに登録されている前記プログラムと当該属性により関連付けられたファイルの前記最後の更新時刻を前記最後のロード時刻と比較して前記コンピューティングデバイス内の無効ファイルを検出し、前記無効ファイルをレポートするように構成されたクエリーログコンポーネントであり、ここで、無効ファイルの存在は前記プログラムにおけるアップグレードを無視する結果となり、前記プログラムのアップグレードを無視することは古いファイルから引き続き実行する結果となり、前記レポートすることは前記プログラムが最後に更新されたファイルで実行されない旨をリポートするか、または、当該プログラムに対する当該最後に試みた更新のリトライを試みることを含む、クエリーログコンポーネントと

を備えたことを特徴とするコンピューティングデバイス。

【請求項６】

前記コンピューティングデバイスに登録されている前記プログラムに関連付けられた前記ファイルの前記最後のロード時刻および前記最後の更新時刻を記録するように構成されたアーカイブコレクションコンポーネントをさらに備えたことを特徴とする請求項５に記載のコンピューティングデバイス。

【請求項７】

前記最後のロード時刻および前記最後の更新時刻は、前記最後のロード時刻および前記最後の更新時刻の日付を含むことを特徴とする請求項５に記載のコンピューティングデバイス。

【請求項８】

前記クエリーログコンポーネントは、前記コンピューティングデバイスに登録されているプログラムすべてをスキャンするようにさらに構成されていることを特徴とする請求項５に記載のコンピューティングデバイス。

【請求項９】

コンピュータに登録されているプログラムに関連づけられたファイルおよび設定をカタログ化して列挙することと、

前記コンピュータに登録されているプログラムの実行前に、

実行する前記プログラムについての属性に関してログファイルにクエリーして、前記プログラムの実行に先行して前記システムメモリ内にロードされ且つ前記コンピュータに登

録されているプログラムと当該属性により関連づけられたファイルおよび設定の最後のロード時刻および日付を獲得し、
前記ログファイルにクエリーして、前記プログラムの実行に先行してシステムメモリ内にロードされ且つ前記コンピュータに登録されているプログラムと前記属性により関連づけられたファイルおよび設定の最後の更新時刻および日付を獲得し、
前記最後の更新時刻および日付を、前記コンピュータに登録されている前記プログラムに関連付けられたファイルおよび設定の前記最後のロード時刻および日付と比較して、前記コンピュータに登録されたプログラムにおける不整合を記録し、前記コンピュータに登録されていてかつ最後に更新されたファイルで実行されないプログラムを含む記録を生成する処理を
コンピュータに実行させるプログラムを記録したコンピュータ読み取り可能な記録媒体。
【請求項１０】
前記比較することは、前記コンピュータの登録されているアンインストールされたプログラムに関連付けられたファイルおよび設定を比較して、リークファイルを決定することをさらに含むことを特徴とする請求項９に記載のコンピュータ読み取り可能な記録媒体。
【請求項１１】
前記リークファイルは、前記コンピュータから削除されることを特徴とする請求項１０に記載のコンピュータ読み取り可能な記録媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、システム解析および管理に関する。
【背景技術】
【０００２】
信頼性の高い、安全なコンピューターシステムを構築するための最も重要な挑戦課題は、すべての実行ファイル、構成設定、およびシステムがどのように機能するかを支配する他のデーターを含む、システムの永続状態（ＰＳ：Ｐｅｒｓｉｓｔｅｎｔ　Ｓｔａｔｅ）を管理することである。構成誤り、および他のＰＳ問題は、個々のデスクトップマシーンから大規模なインターネットサービスに及ぶ様々なシステムにわたって、故障およびセキュリティ脆弱性の主な原因の中に入っている。ＰＳ問題は、ハードウェアコンポーネントやプログラミングロジックなどシステム要素内の故障によって引き起こされる問題と共に、システム全体に有害な影響を及ぼす可能性がある。
【０００３】
システムのＰＳを効果的に管理できないことのコストは高いものである。たとえば、ＰＳ問題は、システムのリブートまたはアプリケーションの再起動の後で再現される可能性がある。さらに、ＰＳは、パッチや、アプリケーション関連の更新など、変更によってランタイム中にドリフトする。現在、変更がシステム上で発生したときループを閉じる効果的な方法は存在しない。そのような状況では、既知の問題識別がうまくゆかない場合、かつ後続のシステムのリブート／アプリケーションの再起動によりＰＳ問題を取り除くことができない場合、根本原因のＰＳを識別するために、システムを手動で調べる以外に選択肢はないこともある。
【発明の概要】
【０００４】
根本原因のＰＳを識別するための、システムの手動調査は、潜在的な問題の数が多いため、困難でありコストがかかる。たとえば、障害を有するアプリケーションに影響を及ぼす可能性がある、状態の潜在的なセットは膨大であり、それに応じて、潜在的な根本原因のリストは、システム上の状態のセット全体を含む可能性がある。さらに、特に単一のＰＳ根本原因がない場合にセットのあらゆる可能な組合せについても検討する場合、状況は、おそらくはより悪化するおそれがある。

【０００５】
この概要は、下記詳細な説明でさらに述べるモデルをベースとするライセンス計数（ｌｉｃｅｎｓｅ　ｃｏｕｎｔｉｎｇ）の簡単な概要を紹介するために提供されている。この概要は、特許請求されている主題の本質的な特徴を識別するものではなく、特許請求されている主題の範囲を決定する際に使用するものでもない。
【０００６】
一実施形態では、コンピューティングベースのデバイス（ｃｏｍｐｕｔｉｎｇ－ｂａｓｅｄ　ｄｅｖｉｃｅ）のプログラムがカタログ化および列挙され、コンピューティングベースのデバイスに登録されたプログラムの最後のロード時間が獲得され、コンピューティングベースのデバイスに登録されているプログラムに関連付けられたファイルの最後の修正時間について、最後のロード時間との比較が行われる。
【図面の簡単な説明】
【０００７】
【図１】システム管理のための例示的なアーキテクチャーの図である。
【図２】例示的なコレクションサーバーの図である。
【図３】生成された通知を示す例示的なビジュアルインターフェースの図である。
【図４】第１のプログラムに対する１つのプログラムの実行の依存を示す例示的なビジュアルインターフェースの図である。
【図５】システムのＰＳにおける修正に関連するデーターを取り込むための例示的な方法の図である。
【図６】顕著な変更を分類するための例示的な方法の図である。
【図７】無許可の対話の実行を禁止するための例示的な方法の図である。
【図８】１つまたは複数の拡張性ポイントを検出するための例示的な方法の図である。
【図９】リークエンティティーを検出するための例示的な方法の図である。
【図１０】共通の構成誤りまたは無効のファイルを検出するための例示的な方法の図である。
【図１１】例示的なコンピューター環境の図である。
【発明を実施するための形態】
【０００８】
詳細な説明について、添付の図を参照して述べる。図では、符号の上位桁の数字（１つまたは複数）が、その符号が最初に現れた図を識別する。同じ符号が図面全体を通して使用され、同様の特徴および構成要素を参照する。
【０００９】
これを実装するために、このシステムは、システム内で発生する修正に関連するデーターをレポートする１つまたは複数のコンピュータープログラムまたはエージェントを含む。そのデーターは、ファイルおよび／または設定との対話すべてに関連する情報を含む。そのようなタイプの対話は、レジストリーエントリー、ファイルに対する読取りおよび書込みアクセス、ならびにロードなどバイナリーモジュール対話などのような活動を含む。エージェントは、収集されたデーターをバックエンドサービスにレポートし、そのバックエンドサービスは、ウェブレポート、警報を生成することや、システム管理を行うために他のサービスと一体化することのような活動のために、レポートされた情報を処理する。さらに、処理は、データーが収集される単一のマシーン上でも行うことができる。これは、レポート、警報などの生成を含む。具体的には、システムの永続状態（ＰＳ）が対処されるものであり、ＰＳは、すべての実行ファイル、構成設定、およびシステムがどのように機能するかを支配する他のデーターを含む。永続状態について論じられているが、論じられている技法および方法は、他の種類の状態にも適用可能であることを理解されたい。
【００１０】
レポートされたデーターは、いくつかの目的に使用することができる。たとえば、データーを調べ、開始される対話が設定されたポリシー（ｓｅｔ　ｐｏｌｉｃｙ）と一致している対話であることまたは許可された対話に関連するものであることを検証することがで

きる。

【００１１】

述べられている、システム管理のためのシステムおよび方法の諸態様は、任意の数の様々なコンピューティングシステム、環境、および／または構成で実装することができるが、システム解析および管理の実施形態については、以下の例示的なシステムアーキテクチャーの状況で述べる。

【００１２】

（例示的なシステム）

図１は、１つまたは複数のプログラム間の対話に関連する情報を収集および解析することができる例示的なコンピューターシステム１００を示す。システム１００は、１つまたは複数のプログラムが動作中である、またはインストールされているコンピューティングベースのデバイス１０２と、コレクションサーバー１０４と、アーカイブコレクション１０６と、レポート１０８とを含む。

【００１３】

１つまたは複数のプログラムおよび／またはファイルシステムまたは設定間の対話に関連する情報は、システム１００内で発生する可能性がある永続状態（ＰＳ）における修正を表す。コンピューティングベースのデバイス１０２は、任意の数のコンピューティングベースのデバイス１０２を含むことができる。たとえば、一実装では、システム１００はまた、数千のオフィスパーソナルコンピューター（ＰＣ）、様々なサーバー、およびいくつかの国全体に広がる他のコンピューティングベースのデバイスであってコンピューティングベースのデバイス１０２として働くすべてを含む会社ネットワークを含むことができる。別法として、他の可能な実装では、システム１００は、限られた数のＰＣを有する家庭ネットワークを含むことができる。コンピューティングベースのデバイス１０２は、ＬＡＮ、ＷＡＮ、または当技術分野で知られている任意の他のネットワーキング技術を含む有線ネットワークおよび／または無線ネットワークを介して、様々な組合せで互いに結合させることができる。

【００１４】

コンピューティングベースのデバイス１０２は、システム１００において関数に機能を装備する（ｉｎｓｔｒｕｍｅｎｔｉｎｇ）、１つまたは複数のコンピューティングベースのデバイス１０２および／またはファイルシステムおよび設定間の対話に関連する情報を取り込むことが可能なエージェント１１０を含むことができる。一実装では、エージェント１１０は、その関数を呼び出すスレッドをインターセプトするように、その関数内でコンピューター可読命令を修正、追加、および／または削除することが可能なスレッドデーターレコーダー（ＴＤＲ）とすることができる。他の可能な実装では、関数に機能を装備することはまた、あるスレッドに関連するデーターの取込みを可能にするある関数内のコンピューター可読命令を実行することをそのスレッドに要求するように、その関数内でコンピューター可読命令を修正、追加、および／または削除することを含む。他の実装では、スレッドに関連するデーターは、スレッドが関連付けられているプログラムに関する情報、スレッドに関連する１つまたは複数の対話、およびスレッドが関連付けられているプログラムの使用に関する情報を含む。ＴＤＲについて論じられているが、インターセプトは必ずしもアルゴリズムすべてに必要とされるものではない可能性があり、したがって、論じられている技法および方法は、必ずしもＴＤＲをベースとするデーター収集に結び付けられない可能性があることを理解されたい。さらに、仮想マシーン（ＶＭ）をベースとする機能装備は、コードが動的に追加される、ＴＤＲをベースとする機能装備と異なる可能性がある。ＶＭでは、この種の収集を行うための、ＶＭインターナルのハードコードされた関数とすることができる。

【００１５】

機能装備された関数は、プログラム／プロセスによって呼び出される可能性がある関数を含むことができる。一実装では、機能装備された関数は、ファイルシステムドライバなど下位のチョークポイント関数、レジストリー関数、新しいプロセスおよび／またはサー

ビスを作成する関数などを含むことができる。

【００１６】

スレッドデーターレコーダーによってスレッドから取り込まれたデーターは、システム１００の挙動を調節するために、またある条件、またはシステム１００の永続状態を調査するために、記憶および／または処理することができる。スレッドデーターレコーダーによってスレッドから取り込むことができるデーターのタイプ、およびスレッドデーターレコーダーの動作については、Ｖｅｒｂｏｗｓｋｉらによる「Ｔｈｒｅａｄ Ｉｎｔｅｒｃｅｐｔｉｏｎ ａｎｄ Ａｎａｌｙｓｉｓ」という名称の、＿＿＿＿＿＿＿＿に出願された米国特許出願第＿＿＿＿＿＿号明細書でより詳細に論じられており、この特許出願は、ここに参照によって組み込む。

【００１７】

コレクションサーバー１０４は、システム１００内で発生している可能性がある修正に関して情報を収集する責任がある。一実装では、エージェント１１０は、対話に関連する情報をコレクションサーバー１０４内で、圧縮されたログとして記憶する。他の実装では、対話に関連する情報は、さらにアーカイブコレクション１０６内にアップロードすることができる。コレクションサーバー１０４またはアーカイブコレクション１０６内で収集された情報の解析が、レポート１０８を生成するために使用される。コレクションサーバー１０４またはアーカイブコレクション１０６上で収集された情報に対して実施された解析の結果として生成されるレポート１０８により、１つまたは複数のコンピューティングベースのデバイス１０２内で発生しつつある可能性がある対話または修正を見抜くことができる。他の実装では、レポート１０８は、ビジュアルインターフェースを介して生成することができる。他の実装では、ビジュアルインターフェースは、コレクションサーバー１０４またはアーカイブコレクション１０６内で記憶された情報の、プログラム可能なデーターアクセスを実施するために、以前に作成されたレポート、および／またはキャッシュされたレポートを取り出し表示するためのブラウザーを介して実装することができる。コレクションサーバー１０４およびアーカイブコレクション１０６は、コレクションサーバー１０４またはアーカイブコレクション１０６として働く単一のデバイスに常駐することも、その一部とすることもできる。

【００１８】

上記のように、エージェント１１０によって収集されコレクションサーバー１０４またはアーカイブコレクション１０６内で記憶された情報を解析し、システム１００で機能していることについての識見を提供することができる。実施される解析は、異常検出、変更管理、異常なシステム活動の管理、セキュリティ脆弱性の識別、無許可のアプリケーションの識別、コンプライアンス監査などを含むことができる。

【００１９】

図２は、エージェント１１０からのデーターを記憶、処理、および／または解析するように構成された、例示的なコレクションサーバー１０６を示す。コレクションサーバー１０６は、１つまたは複数のプロセッサ２０２と、メモリー２０４とを含む。プロセッサ２０２は、たとえばマイクロプロセッサ、マイクロコンピューター、マイクロコントローラ、デジタル信号プロセッサ、中央処理装置、状態機械、論理回路、および／または演算命令に基づいて信号を操作する任意のデバイスを含む。プロセッサ２０２は、諸機能の中でもとりわけ、メモリー２０４内に格納されたコンピューター可読命令をフェッチし実行するように構成される。

【００２０】

メモリー２０４は、当技術分野で知られている任意のコンピューター可読媒体、たとえば揮発性メモリー（たとえば、ＲＡＭ）および／または不揮発性メモリー（たとえば、ＲＯＭ、フラッシュなど）とすることができる。また、メモリー２０４は、プログラム２０６と、データー２０８とを含むことができる。プログラム２０６は、演算の中でもとりわけ、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラム間の対話に関連するデーターに対して、クエリー関連の処理を実施することができる。

さらに、プログラム２０６は、たとえばクエリーモジュール２１０と、通知モジュール２１２と、オペレーティングシステム２１６と、他のアプリケーション２１４とを含む。オペレーティングシステム２１６は、プログラム２０６内のモジュールの１つまたは複数が機能するための動作環境を提供する。

【００２１】

クエリーモジュール２１０は、ログストレージ２１８内に含まれる情報など、エージェント１１０によって収集された情報に対して、クエリーをベースとする演算を実施する。エージェント１１０によって収集された情報はまた、アーカイブコレクション１０６から取出し可能とすることもできる。クエリー２２０は、事前定義されたクエリーなど、複数のクエリーを含む。そのような事前定義されたクエリーは、セキュリティポリシー定義など、システム１００に対して規定することができる１つまたは複数のポリシー定義に関する条件に関係することができる。そのような場合には、クエリーモジュール２１０によって実施することができるいくつかの、またはすべての解析が、そのような事前定義されたポリシー定義、または事前定義されたクエリーに一致している可能性がある。

【００２２】

クエリーモジュール２１０は、クエリー２２０を１つまたは複数の属性に制限することができる。そのような属性は、ファイル名、アプリケーションタイプ、実行の時間などを含むことができる。制限されたクエリーに基づいて機能するとき、クエリーモジュール２１０は、属性の存在を示す値を求めて、ログストレージ２１８および／またはアーカイブコレクション１０６内で記憶されている情報すべてをスキャンする。たとえば、ある個人が、ワードプロセッサなど、ある種のアプリケーションに関係するデーターを求めてアーカイブコレクション１０６を探索したいと望む場合、クエリーモジュール２１０は、そのワードプロセッサによって開始され影響を受けている対話に関連するエントリーまたはイベントを探索する。

【００２３】

クエリー２２０は、システム管理者など、１人もしくは複数の個人または１つもしくは複数のエンティティーによって入力またはプログラムされたクエリーを含むことができる。たとえば、クエリー２２０は、所与のユーザーＩＤに関連する対話すべてを検出するための命令を含むことができる。さらに、クエリー２２０は、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するアプリケーションに関連する対話すべてを検出するための命令を含むことができる。

【００２４】

コレクションサーバー１０４に戻ると、１つまたは複数のコンピューティングベースのデバイス１０２の、ファイルシステムおよび／または設定との対話に関連する情報の解析が、システム１００の機能および／または永続状態を決定するために実施されることになる。クエリーモジュール２１０を使用し、エージェント１１０によって収集された、またログストレージ２１８および／またはアーカイブコレクション１０６内で記憶されている情報に対して、解析を実施することができる。クエリーモジュール２１０は、クエリー２２０内で指定された１つまたは複数のクエリーを使用してログストレージ２１８および／またはアーカイブコレクション１０６の探索を実施することによって、これを実施することができる。クエリー２２０の実行によって生成される結果は、１つまたは複数のコンピューティングベースのデバイス１０２とファイルシステムおよび／または設定との間の対話を示す。

【００２５】

クエリーモジュール２１０は、クエリー２２０の実行の結果として生成された結果に関して通知を発行するように、通知モジュール２１２に指令することができる。通知モジュール２１２によって生成された通知は、データー２０８内の通知２２２内で記憶することができる。また、通知モジュール２１２によって発行された通知は、外部記憶装置のような外部データーベース内で記憶することができる。また、通知モジュール２１２は、クエリーモジュール２１０によって、クエリー２２０の実行の結果として生成された通知を通

信するように指令を受けることができる。

【００２６】

クエリーモジュール２１０は、ログストレージ２１８および／またはアーカイブコレクション１０６を探索し、クエリーモジュール２１０によって実行されるクエリー２２０に関して、１つまたは複数のコンピューティングベースのデバイス１０２間の対話に関連する情報における逸脱を検出する。そのような場合、ある対話に関するそのような逸脱の検出は、やはり通知モジュール２１２によって通知することができ、将来参照するために通知を記憶するために、対応する通知２２２を、システム管理者のような個人、またはコンピューティングシステムに通信することができる。

【００２７】

通知モジュール２１２はまた、通知２２２に関連するコンテキスト情報を提供することが可能である。このコンテキスト情報は、対応する対話に関連付けることができる設定を追加的に指定することができる。コンテキスト情報を、１つまたは複数の段階で、関連の通知２２２にアノテーションすることができる。たとえば、あるレベルは、あるプログラムがインストールされている可能性があるマシーンの数、ファイルの最も一般的なバージョンなどに関する統計情報を提供する。アノテーションの別のレベルは、インストールされているファイルのハッシュ値と、たとえばプログラム名、バージョン情報など属性を示すデーターコレクションとの比較を示す。既知の問題、ベンダなどに関するコメントまたは任意の補助情報を提供することができる、さらに別のレベルのアノテーションが存在する可能性がある。追加のレベルのアノテーションを実装し、それにより、関連付けられた通知２２２に関して追加の属性を指定することもできる。また、通知２２２は、個人、たとえばシステム管理者が通知２２２を見直し、必要とされる場合について適切な行動を起こすことを可能にするビジュアルインターフェースを介して表示することができる。

【００２８】

クエリーモジュール２１０は、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムとファイルシステムおよび／または設定との間の対話によりシステム１００内で発生する顕著な変更を検出するために使用することができる。顕著な変更は、あるプログラム、オペレーティングシステム、会計のような特定のビジネスタスクを達成するために使用されるプログラム、および他のプログラムの予期しない実行により生じる可能性があるシステムのＰＳに対する変更または修正を含む。システムのＰＳに対するそのような顕著な変更は、システム性能のチョーク状態（ｃｈｏｋｅｄ　ｐｅｒｆｏｒｍａｎｃｅ　ｏｆ　ｓｙｓｔｅｍ）、セキュリティ問題などのような望ましくない状況を防止するために許可を受け、制御される。また、ＰＳにおいて発生する変更すべてが顕著な変更であるわけではないことに留意されたい。

【００２９】

顕著な変更は、識別子によってアノテーションし、割り振られたアノテーションに従って分類することができる。顕著な変更のアノテーションは、クエリーモジュール２１０によって分類規則を指定することによって実施することができる。分類規則内で指定されたパラメータに基づいて、適切なパラメータが、その顕著な変更に関するある属性に関連付けられる。たとえば、クエリーモジュール２１０は、分類規則内に含まれるサブストリングの、各合致したものを、各顕著な変更内に含まれる修正の名前またはタイプに関連付ける。顕著な変更に対する分類は、複数の値に基づいて割り当てることができる。たとえば、そのような場合、より高い優先順位を有する分類サブストリングに対して合致したものが、より低い優先順位を有する分類サブストリングより優先される。次いで、より高い優先順位を有する分類サブストリングは、相対的な顕著な変更に対する関連分類として決定力を持つもの（ｄｅｔｅｒｍｉｎａｔｅ）となる。

【００３０】

顕著な変更は、以下の分類の少なくとも１つまたは複数としてラベル付けすることによって分類することができる。すなわち、

・問題（Ｐｒｏｂｌｅｍ）：既知の問題、または現在のＰＳの存在または除去からの結果

を示す。
・インストール（Ｉｎｓｔａｌｌ）：インストールまたはアップグレードの結果としてのＰＳにおける変更を示す。
・設定（Ｓｅｔｔｉｎｇ）：構成設定または構成ＰＳに対して行われた変更を示す。
・コンテンツ（Ｃｏｎｔｅｎｔ）：ウェブページ、画像、テキストデーター、およびユーザーデーターを示す。
・管理変更（Ｍａｎａｇｅｍｅｎｔ　Ｃｈａｎｇｅ）：システム上で動作するそのシステムの管理に責任があるプログラムに対して行われたインストール、パッチ適用（ｐａｔｃｈｉｎｇ）、または構成変更を示す。
・無許可（ｕｎａｕｔｈｏｒｉｚｅｄ）：無許可の、もしくは禁止されたアプリケーションのインストール、または禁止された値を含む構成変更を示す。
・ユーザー活動（Ｕｓｅｒ　Ａｃｔｉｖｉｔｙ）：ユーザーのログインまたはウィンドウアプリケーションを動作させたことの結果としてのＰＳにおける変更を示す。
・ノイズ（Ｎｏｉｓｅ）：一時的なＰＳ、またはキャッシュされたＰＳを示す。
・未知（Ｕｎｋｎｏｗｎ）：未分類のＰＳを示す。

【００３１】

顕著な変更をさらに分類し、それらを他の変更から判別可能なものとするために、追加のアノテーションを提供することができる。

【００３２】

クエリーモジュール２１０はまた、システム１００上で動作するプログラムのステータスを、許可されたもの、または無許可のものとして決定するために使用することができる。これは、許可されたプロセスまたはプログラムだけがシステム１００上で動作するべきであるという必要に基づいている。クエリーモジュール２１０は、クエリー２２０内で指定された属性とシステムのＰＳにおける特定の変更または修正を定義する属性を比較することによって、システム１００上で動作するプログラムのステータスを、許可されたもの、または無許可のものとして決定する。たとえば、クエリーモジュール２１０は、あるアプリケーションタイプを無許可のプログラムとして指定するクエリー２２０を実行する。クエリー２２０の実行の結果として得られる結果は、指定されたアプリケーションタイプの実行に応答して発生したＰＳにおける変更に関して情報を含む。クエリーモジュール２１０は、それらの結果を得る際に、無許可のプログラムの実行またはアクションによって誘発された変更としてそのような結果にマーク付けする。

【００３３】

クエリーモジュール２１０は、承認済みプログラムおよび／または未承認のプログラムの事前定義されたリスト内で指定された属性を、システムのＰＳにおける特定の変更または修正を定義する属性と比較することができる。現在の場合におけるリストは、特定の数の承認済みプログラム、または未承認のプログラムを含むことができる。事前定義されたリスト内で未承認として識別されているプログラムと同様のものである、システム１００上で動作するプログラムは、無許可のプログラムとしてマーク付けされる。

【００３４】

事前定義されたリスト内で指定された承認済みプログラムおよび／または未承認のプログラムはまた、プログラムの性質および／または様々な特性の属性を示す、ラベルなど追加の情報を含むことができる。そのような追加の情報の例には、「承認済み（ａｐｐｒｏｖｅｄ）」「タイプ（ｔｙｐｅ）」「カテゴリ（ｃａｔｅｇｏｒｙ）」「機能（ｆｕｎｃｔｉｏｎ）」「製品情報（ｐｒｏｄｕｃｔ　ｉｎｆｏｒｍａｔｉｏｎ）」「製造者情報（ｍａｎｕｆａｃｔｕｒｅｒ　ｉｎｆｏｒｍａｔｉｏｎ）」および「製品説明（ｐｒｏｄｕｃｔ　ｄｅｓｃｒｉｐｔｉｏｎ）」など、ラベルが含まれる。たとえば、「承認済み」としてラベル付けされているプログラムは、システム１００内の１つまたは複数のコンピューティングベースのデバイス１０２上で動作するための許可されたプログラムと見なされ、「カテゴリ」ラベルは、そのプログラムの所期の使用を指定する。

【００３５】

あるプログラムによって初めて実施される変更または修正は、デフォルトで承認されず、「無許可」としてマーク付けされる。たとえば、あるプログラムによる変更または修正を初めて検出したとき、クエリーモジュール２１０は、そのプログラム、およびその関連する対話を「無許可」としてマーク付けする。「無許可」としてマーク付けされたプログラムは、たとえばシステム管理者の見直し用に、または必要な場合に診断を実施するために、または未解決の承認用に、通知モジュール２１２が通知することができる。承認が得られた場合には、承認済みプログラムはそのプログラムの属性を示す適切なラベルにさらに関連付けられ、承認済みプログラムおよび／または未承認のプログラムを含む事前定義されたリストにそのプログラムを追加することができる。

【００３６】

クエリーモジュール２１０はまた、拡張性ポイント（ＥＰ／ｅｘｔｅｎｓｉｂｉｌｉｔｙ　ｐｏｉｎｔ）を検出することができる。ＥＰは、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムまたはオペレーティングシステムに関連付けられた命令の動的ロードおよび実行を示す対話である。たとえば、１つまたは複数のコンピューティングベースのデバイス１０２上で動作する、ワードプロセッサ、スプレッドシートアプリケーションなど第１のプログラムが起動したとき、その第１のプログラムは、第１のプログラムの動作に追加の機能をもたらす、アドオンプログラムなど他のプログラムに関連付けられた命令をトリガすることもある。このようにして、第１のプログラムの動作により、第１のプログラムとファイルシステムの間の対話、および第１のプログラムの動作に追加の機能をもたらす他のプログラムとファイルシステムの間の対話を含めて、様々な対話が生成される可能性がある。そのような情報により、第１のプログラムがインストールされていたシステムの機能を見抜くことができ、またそのようなインストールによってシステムにもたらされる可能性がある影響を推測するための見識が得られる。

【００３７】

第１のプログラムの動作の結果として生成される様々な対話に関連する情報は、たとえばエージェント１１０がインターセプトおよびコピーすることができる。様々な対話に関連するイベント情報は、圧縮されたログとして、ログストレージ２１８および／またはアーカイブコレクション１０６内で記憶することができる。イベント情報は、圧縮ストレージ内で記憶されるが、必ずしも圧縮ストレージが使用されなくてもよいことを理解されたい。しかし、圧縮ストレージの使用により、ストレージがあまり空間を占有しなくなることによって、システムがよりスケーラブルなものになる。記憶されたイベント情報は、第１のプログラムおよび他のプログラムに関連する、ファイルシステムとの対話を検出するために、システム管理者などエンティティーによる、またはクエリーモジュール２１０による見直しを受けることができる。このようにして、第１のプログラムの動作に関連する場合、他のプログラムを検出することができる。

【００３８】

クエリーモジュール２１０はまた、第１のプログラムについての直接ＥＰ（ｄｉｒｅｃｔ　ＥＰ）を検出するために使用することができる。たとえば、クエリーモジュール２１０は、（１）第１のプログラムの実行前に、実行するためにシステムメモリー内にロードされている様々なプログラムに関連し、かつ（２）第１のプログラムを参照する、または第１のプログラムの動作に関連付けられる対話を分離することによって、直接ＥＰを検出することができる。

【００３９】

例示的な一実装では、クエリーモジュール２１０は、第１のプログラムの実行前に、実行するためにシステムメモリー内にロードされている様々なプログラムに関連する対話に関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることによって、第１のプログラムについての潜在的な直接ＥＰを識別することができる。たとえば、クエリーモジュール２１０は、第１のプログラムの実行前に、１秒など所与の時間範囲内で、実行するためにシステムメモリー内にロードされている様々なプログ

ラムに関連する対話に関してクエリーすることができる。クエリーモジュール２１０は、第１のプログラムを参照する、または第１のプログラムの動作に関連付けられる対話に関して潜在的なＥＰにクエリーすることによって、潜在的なＥＰから、第１のプログラムについての直接ＥＰを識別することができる。直接ＥＰは、他のデーター２２４内で記憶することができる。

【００４０】

クエリーモジュール２１０はまた、間接ＥＰ（ｉｎｄｉｒｅｃｔ　ＥＰ）を検出するために使用することができる。たとえば、上記の第１のプログラムの例に戻ると、クエリーモジュール２１０は、直接ＥＰを参照する、または直接ＥＰに関連付けられる対話に関してログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。そのような対話は、間接ＥＰと称することができる。間接ＥＰは、他のデーター２２４内で記憶することができる。

【００４１】

クエリーモジュール２１０はまた、直接ＥＰを監視することによって、悪意のあるソフトウェアアプリケーションの存在を検出するために使用することができる。悪意のあるソフトウェアアプリケーションには、通常の状況下ではプログラムに関連付けられないはずである「スパイウェア」「トロイの木馬」「ワーム」「ウィルス」などを含めることができる。たとえば、クエリーモジュール２１０は、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムについてのＥＰを、そのプログラムが悪意のあるソフトウェアの不在時にコンピューティングベースのデバイス１０２上で動作していたとき見出された同じプログラムについての制御ＥＰに突き合わせて比較することができる。それらのＥＰと制御ＥＰとの差をクエリーモジュール２１０および／またはシステム管理者などエンティティーが調べ、それらの差が、そのプログラムと共に動作する悪意のあるソフトウェアの存在を示しているかどうか判定することができる。ＥＰを使用して見つけられた悪意のあるソフトウェアは、クエリーモジュール２１０、システム管理者などが、影響を受けたコンピューティングベースのデバイス１０２から除去することができる。

【００４２】

他の実装では、クエリーモジュール２１０は、ＥＰの検出に応答して、通知モジュール２１２によって通知２２２を生成するために使用することができる。他の実装では、生成される通知２２２は、システム管理者のような個人が、さらに解析するために、または必要とされる診断を実施するために通知２２２を見直すことを容易にするビジュアルインターフェースを介してさらに閲覧する、または取り出すことができる。

【００４３】

図３は、可能な実装の１つにおける生成された通知２２２を示す例示的なビジュアルインターフェース３００を示す。図の実装では、ビジュアルインターフェース３００は、第１のプログラム（たとえば、ウェブブラウザー）によって、その実行時に実施されたダウンロードを示す。ビジュアルインターフェース３００は、セグメント３０２および３０４内で、第１のプログラム（図３に示されている例では、特に「ＭＳＮ　Ｓｅａｒｃｈ　Ｔｏｏｌｂａｒ」および「Ｗｉｎａｍｐ　Ｍｅｄｉａ　Ｐｌａｙｅｒ」）によって、その実行中にダウンロードされたプログラムのリストを示す。図では、セグメント３０６内において、セグメント３０２、３０４内に見られるプログラムのダウンロードにより、第１のプログラム、およびセグメント３０２、３０４内に示されているプログラム以外のプログラムに対応するプログラムファイルもまた作成されることがわかる。したがって、ビジュアルインターフェース３００の形態にある視覚的表現は、第１のプログラムを実行、ダウンロード、および／またはインストールする間にシステム１００の１つまたは複数のコンピューティングベースのデバイス１０２上に気付かずにインストールされるプログラムのリストを提供する。

【００４４】

セグメント３０６はまた、第１のプログラム以外のプログラムのインストールまたは実

行の結果としての影響またはシステム１００のＰＳにおける修正を示すことができる。第１のプログラム以外の１つまたは複数のプログラムのさらなる実行は、第１のプログラムの実行に依存する可能性がある。たとえば、図のように、「ＭＳＮ　Ｓｅａｒｃｈ　Ｔｏｏｌｂａｒ」は、第１のプログラムのプログラムファイルの実行時にアクティブ化される可能性がある。この作用に対する決定は、第１のプログラムに対応するＥＰを検出することによって実施することができる。第１のプログラムに関連するＥＰを監視することにより、第１のプログラムの実行に依存する他のプログラムのプログラム実行のインスタンスを検出し、必要な場合、是正措置をとることができる。

【００４５】

また、第１のプログラムの実行に依存する他のプログラムの実行のインスタンスは、図４に示されている別のビジュアルインターフェース４００を介して表示させることができる。図４は、セグメント４０２として示されている第１のプログラム、たとえば「ｉｅｘｐｌｏｒｅｒ．ｅｘｅ」の実行により、セグメント４０４として示されているＷｉｎａｍｐの実行が誘発され、それにより、セグメント４０６として示されている「ｅｍｕｓｉｃ．ｅｘｅ」がさらに実行されることを示す。そのような図から、第１のプログラムに関連するＥＰの検出を詳細に、視覚的手段を介して実施することができる。

【００４６】

クエリーモジュール２１０は、リーク（ｌｅａｋｅｄ）ＰＳを検出するために使用することができる。リークファイルは、それらのファイルまたはレジストリー設定を作成したプログラムがアンインストールされた後で、システム１００などシステム上に残されるファイルまたはレジストリー設定を含む。また、インストールの結果として作成された可能性があるが、インストールプロセスが完了した後で削除できなかったファイルまたは設定、たとえば一時ファイルを含むこともできる。さらに、プログラムのランタイム中（すなわち、インストール後）に生成されるＰＳなど、別のクラスのＰＳがリークする可能性がある。これらの例は、最初のインストールの後で別途インストールされるプログラムに対する初めての使用または実行時に生成される可能性がある状態である。

【００４７】

リークファイルを検出するために、クエリーモジュール２１０は、システム１００上にロードされた各プログラムに関連するインストールファイルおよび設定の変更をカタログ化し、それらの変更を、プログラムの使用ならびに最初のインストールを介して追跡することができる。その後で、そのプログラムがアンインストールされた場合、インストールファイルおよび構成またはレジストリー設定の、対応するカタログを呼び戻すことができ、システム１００を検査し、インストールファイルおよびレジストリー設定すべてが除去された、またはリセットされたことを確認することができる。コンピューティングベースのデバイス１０２上のリークファイルを検出するために、クエリーモジュール２１０は、１つまたは複数のコンピューティングベースのデバイス１０２全体にわたってスキャンを実行し、コンピューティングベースのデバイス１０２上にインストールされた、アプリケーションなどプログラムすべてを検出することによって、インストールファイルをカタログ化する。

【００４８】

クエリーモジュール２１０はまた、１つまたは複数のコンピューティングベースのデバイス１０２のオペレーティングシステムのインストーラデーターベース内に登録されているプログラムすべてのリストを獲得することができる。インストーラデーターベースの例には、考慮中のコンピューティングベースのデバイス１０２上にインストールされたプログラムのポピュレートされたリストを生成するコンポーネントが含まれる。

【００４９】

クエリーモジュール２１０は、レジストリー構成または設定に関して、またコンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムのリストを列挙するために、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーする。次いで、クエリーモジュール２１０は、ログストレージ２１

８および／またはアーカイブコレクション１０６をスキャンし、すべてのＰＳに一般化することができる、コンピューティングベースのデバイス１０２上にインストールされたプログラムすべてのファイルおよびレジストリーエントリーを列挙することができる。ファイルおよびレジストリーエントリーを列挙するために、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２上にインストールされたプログラムの１つまたは複数の属性、たとえばプログラムＩＤに対応するファイルおよびレジストリーエントリーすべてに関して、クエリーすることができる。

【００５０】

コンピューティングベースのデバイス１０２上のあるファイルまたは設定が、コンピューティングベースのデバイス１０２上にインストールされたプログラムのプログラムＩＤに対応するファイルおよびレジストリーエントリー内に含まれない場合には、クエリーモジュール２１０は、そのファイルまたは設定がリークファイルであると推定することができる。リークファイルは、クエリーモジュール２１０、またはオペレーティングシステムを含む様々な他のプログラム、システム管理者などが除去することができる。

【００５１】

検出されたリークファイル（ＰＳ）は、個人、たとえばシステム管理者がリークファイルを見直し、必要とされる場合について適切な行動を起こすことを可能にするビジュアルインターフェースを介して表示することができる。さらに、表示されたリークファイル（ＰＳ）および関連する情報は、将来参照するために、外部記憶コレクション、たとえば外部データーベース内で記憶することができる。リークＰＳリストは、主アプリケーションが除去されたとき、リーク状態をシステムによって自動的に除去するために使用することができる。また、このリークＰＳリストは、システム上の各ＰＳをオーナーアプリケーションに関連付けるために使用することもできる。

【００５２】

クエリーモジュール２１０は、共通の構成誤りや古いソフトウェアバージョンなどを含めて、変更されたファイル、設定、または無効のモジュールによる無効のプロセスを検出することができる。無効のプロセスは、たとえばソフトウェアアップグレードにより、ディスク上の実行可能ファイル、プログラムファイル、または設定を置き換えた後で、影響を受けたプロセスが再起動できなかったとき発生する。その結果、無効のプロセスが見出されるコンピューティングベースのデバイスは、アップグレードを無視し、古い実行可能ファイル、プログラムファイル、または設定に基づいて引き続き実行されることになる。

【００５３】

無効のプロセスを検出するために、クエリーモジュール２１０は、ログストレージ２１８および／またはアーカイブコレクション１０６内で記憶されているプログラムの対話に関連する情報にクエリーすることができる。クエリーモジュール２１０は、１つまたは複数のコンピューティングベースのデバイス１０２上にインストールされたプログラムの最後のロード時間に関してログストレージ２１８および／またはアーカイブコレクション１０６にクエリーする。また、クエリーモジュール２１０は、インストールされているソフトウェアに関連付けられたファイルまたはレジストリー設定の最後のロード時間に関してログストレージ２１８および／またはアーカイブコレクション１０６にクエリーする。例示的な一実装では、クエリーモジュール２１０は、ソフトウェアと共にインストールされたソフトウェア関連のダイナミックリンクライブラリ（ＤＬＬ）に関連付けられたファイルまたはレジストリー設定の最後のロード時間に関してログストレージ２１８および／またはアーカイブコレクション１０６にクエリーする。また、クエリーモジュール２１０は、コンピューティングベースのデバイス上にインストールされたソフトウェアの最後の修正の時間または日付に関してログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。そのような修正は、たとえば、最後の既知のバージョンのインストール済みソフトウェアに関連付けられた１つまたは複数のファイルまたはプログラム設定に対して実施されたアクセスを含む。

【００５４】

ソフトウェアの最後のロード時間が、ソフトウェアの最後の既知の修正の時間または日付より長い場合には、最後にロードされた更新を使用しないソフトウェアに起因する不整合が発生するおそれがある。そのような不整合は、クエリーモジュール２１０によって検出された場合、システム管理者など個人によって記録（ｎｏｔｅ）および是正することができる。

【００５５】

検出された無効のファイルは、個人、たとえばシステム管理者が無効のファイルを見直し、必要とされる場合について適切な行動を起こすことを可能にするビジュアルインターフェースを介して表示することができる。表示された無効のファイルおよび関連する情報は、将来参照するために、外部記憶コレクション、たとえば外部データーベース内で記憶することができる。

【００５６】

クエリーモジュール２１０は、「マルウェア」「スパイウェア」「トロイの木馬」「ウィルス」などソフトウェアを含めて、既知の不当なプログラムの発生を検出することができる。これを行うために、クエリーモジュール２１０は、１つまたは複数のコンピューティングベースのデバイス１０２のメモリー内の、実行するためにロードされたプログラムに関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーし探索することができる。次いで、メモリー内の、実行するためにロードされたプログラムを、たとえばクエリーモジュール２１０によって、既知の不当なプログラムのリストに突き合わせて比較することができる。

【００５７】

たとえば、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２上のメモリー内の、実行するためにロードされたプログラムの発生を、プログラムＩＤなど、それらのプログラムに関連付けられた識別子に基づいて検出することができる。次いで、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２上のメモリー内の、実行するためにロードされたプログラムの識別子を、既知の不当なプログラムの、プログラムＩＤなど識別子のリストに突き合わせて比較することができる。メモリー内の、実行するためにロードされたプログラムの識別子が、既知の不当なプログラムの識別子と合致した場合、クエリーモジュール２１０は、メモリー内の、実行するためにロードされたそのプログラムの、コンピューティングベースのデバイス１０２からの除去を実施することができる。可能な一実装では、既知の不当なプログラムの識別子のリストは、少なくとも部分的には、システム管理者が入力することができる。

【００５８】

クエリーモジュール２１０によって検出された不当なプログラムは、システム管理者など個人がその不当なプログラムを見直し、それらを除去するために適切な行動を起こすことを可能にするビジュアルインターフェースを介して表示することができる。表示された不当なプログラムおよび関連する情報は、将来参照し、同じまたは同様の不当なプログラムを検出するために、外部記憶コレクション、たとえば外部データーベース内で記憶することができる。

【００５９】

識別子が関連付けられていない、１つまたは複数のコンピューティングベースのデバイス１０２上の未確認プログラムを、クエリーモジュール２１０によって検出し、その未確認プログラムが不当なプログラムか否か確認するためにシステム管理者にレポートすることができる。システム管理者は、レポートの形態の未確認プログラムのリストを見直すことによって未確認プログラムの性質を調べることができる。システム管理者による見直しには、未確認プログラムの目的、未確認プログラムの、他のプログラムに対する依存を調べること、および未確認プログラムが不当なものであるかどうか判定するために調べることを含めることができる。さらに、システム管理者は、未確認プログラムが不当なものであるかどうか判定するために、その未確認プログラムの特性と同様の特性を有するプログラムとの過去の経験を見直すことができる。

【００６０】
未確認プログラムが不当なものであるとシステム管理者が決定した場合、システム管理者は、その未確認プログラムの、コンピューティングベースのデバイス１０２からの除去を実施することができる。たとえば、システム管理者は、未確認プログラムそれ自体を除去することができ、または、システム管理者は、コンピューティングベースのデバイス１０２の諸要素に、未確認プログラムを除去するように指令することができる。
【００６１】
さらに、システム管理者は、生成されたレポートまたは１つもしくは複数の通知２２２に基づいて、プログラムＩＤなど識別子を未確認プログラムに割り当て、その識別子を不当なプログラムのリスト上に含めることができる。このようにして、未確認プログラムがコンピューティングベースのデバイス１０２上で再び現れた場合、関連付けられた識別子に基づいて、それを不当なプログラムとして迅速に識別することができる。さらに、未確認プログラムの除去は、コンピューティングベースのデバイス１０２の諸要素、エージェント１１０などによって実施することができる。
【００６２】
クエリーモジュール２１０によって検出された未確認プログラム、およびそれらの関連するプロセスは、個人、たとえばシステム管理者がその未確認プログラムを見直し、それらを除去するために適切な行動を起こすことを可能にするビジュアルインターフェースを介して表示することができる。表示された未確認プログラムおよび関連する情報は、将来参照し、同じまたは同様の不当なプログラムを検出するために、外部記憶コレクション、たとえば外部データーベース内で記憶することができる。さらに、望ましくない変更をも、識別および／または追跡することができる。
【００６３】
クエリーモジュール２１０は、ネットワークドライブまたはリムーバブルロケーションに対するコピーファイルを、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムによってそのようなロケーションへの書込みを拒否することにより、阻止することができる。また、クエリーモジュール２１０は、そのような書込みを将来防止するように、監査する目的で、実施された以前のそのような書込み拒否を見直すことができる。
【００６４】
（例示的な方法）
スレッドインターセプトおよび解析のための例示的な方法について、図１から図４を参照して述べる。これらの例示的な方法については、コンピューター実行可能命令の一般的な状況で述べることができる。一般に、コンピューター実行可能命令は、特定の機能を実施する、または特定の抽象データー型を実装するルーチン、プログラム、オブジェクト、コンポーネント、データー構造、プロシージャ、モジュール、関数などを含むことができる。また、この方法は、通信ネットワークを介してリンクされる遠隔処理デバイスによって機能が実施される分散コンピューティング環境内で実施することができる。分散コンピューティング環境では、コンピューター実行可能命令は、メモリー記憶装置を含めて、ローカルと遠隔両方のコンピューター記憶媒体内に位置する可能性がある。
【００６５】
図５は、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムおよび／またはファイルシステムおよび設定間の対話に関連する情報を取り込む、および収集するための例示的な方法５００を示す。この方法が述べられる順序は、限定するものとして解釈しないものとし、任意の数の、述べられている方法ブロックを、この方法または代替の方法を実施するために任意の順序で組み合わせることができる。さらに、個々のブロックは、本明細書に述べられている本主題の精神および範囲から逸脱することなしに、この方法から削除することができる。さらに、この方法は、任意の好適なハードウェア、ソフトウェア、ファームウェア、またはそれらの組合せで実装することができる。

【００６６】
ブロック５０２では、システム上で動作する、または実行されるプログラムに関連する情報またはデーターがインターセプトされる。一実装では、前記情報は、プログラム／プロセスにより、修正された関数コードを含めて機能を装備された関数が呼び出されたとき収集される。たとえば、コンピューター可読命令を修正し、１つまたは複数の関数に、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムおよび／またはファイルシステムおよび設定間の対話に関連するデーターを取り込むように指令することができる。一実装では、仮想機械が、元のコードが実行されるのを解釈したときデーター収集ロジックを直接適用する。この技法は、元のコードの修正を必要としないことになる。同様に、これは、プロセッサハードウェア内に直接実装することができる。
【００６７】
エージェント１１０など、エージェントは、システム１００内で１つまたは複数の関数に機能を装備することができる。１つまたは複数の関数には、それらの１つまたは複数の関数に関連するコンピューター可読命令を修正することによって機能を装備することができる。
【００６８】
スレッドデーターレコーダーなどエージェント１１０は、システム１００内で、修正された関数を呼び出すスレッドをインターセプトすることができる。それらのスレッドが関連付けられるプログラムは、プログラムレイヤ、ミドルウェアレイヤ、オペレーティングシステムレイヤなど、いくつかの動作レイヤの１つで動作している可能性がある。そのプログラムが対話しようと試みている可能性があるファイルシステムは、（データーファイル、実行可能ファイルなど）ファイル、（構成設定またはレジストリー設定など）設定情報などを含むことができる。
【００６９】
ブロック５０４では、１つまたは複数のコンピューティングベースのデバイス１０２上で動作するプログラムの実行に関連する様々な情報またはデーターが、メモリーロケーション内で収集され、またはメモリーロケーションにコピーされる。修正された関数によって開始される対話を含めて、プログラムの、ファイルシステムおよび／または設定との対話に関連する情報は、メモリーロケーション内にコピーされ送られる。たとえば、エージェント１１０は、それらの対話に関連するすべての、または選択されたデーターをコピーし、そのデーターを、コレクションサーバー１０４などメモリーロケーション内で記憶することができる。対話に関連するデーターは、機能を装備された関数によって開始された対話に関する情報を含むことができる。
【００７０】
ブロック５０６では、メモリーロケーション内で記憶されたデーターが圧縮される。一実装では、圧縮されたデーターを別のメモリーロケーション内で記憶することができる。たとえば、圧縮されたデーターは、コレクションサーバー１０４および／またはアーカイブコレクション１０６内のログストレージ２１８内で記憶することができる。
【００７１】
ブロック５０８では、圧縮されたデーターが、解析のために定期的に更新される。圧縮されたデーターは、コレクションサーバー１０４に、またはコレクションサーバー１０４として働くメモリーロケーションにアップロードすることができる。圧縮されたデーターを解析のためにアップロードする周期（ｐｅｒｉｏｄｉｃｉｔｙ）は、変わる可能性がある。一実装では、圧縮されたデーターは、指定された時間間隔の後でアップロードされる。他の実装では、圧縮されたデーターは、圧縮されたデーターが事前定義されたメモリーの閾値限界を超えたときアップロードされる。
【００７２】
図６は、顕著な変更を分類するための例示的な方法６００を示す。顕著な変更は、あるプログラム、オペレーティングシステム、会計のような特定のビジネスタスクを達成するために使用されるプログラム、および他のプログラムの予期しない実行により生じる可能

性がある変更または修正を含む。この方法が述べられる順序は、限定するものとして解釈しないものとし、任意の数の、述べられている方法ブロックを、この方法または代替の方法を実施するために任意の順序で組み合わせることができる。さらに、個々のブロックは、本明細書に述べられている本主題の精神および範囲から逸脱することなしに、この方法から削除することができる。さらに、この方法は、任意の好適なハードウェア、ソフトウェア、ファームウェア、またはそれらの組合せで実装することができる。

【００７３】

ブロック６０２では、様々なパラメータ値によって属性が示される分類規則が指定される。たとえば、分類規則は、その分類規則を定義するパラメータ値と共に、クエリーモジュール２１０が指定することができる。

【００７４】

ブロック６０４では、分類規則を定義するパラメータが、ある顕著な変更の性質および特性を定義する１つまたは複数の属性に関連付けられる。クエリーモジュール２１０は、分類規則を定義するパラメータを、その顕著な変更を特徴付ける属性に関連付ける。１つまたは複数のパラメータ値を、考慮中の顕著な変更を特徴付ける属性に関連付けることにより、１組の予想分類が得られる。たとえば、クエリーモジュール２１０は、分類規則内に含まれるサブストリングの、各合致したものを、各顕著な変更内に含まれるＰＳ名に関連付ける。

【００７５】

ブロック６０６では、予想分類の１つまたは複数に優先順位値が割り当てられる。クエリーモジュール２１０は、優先順位値を、予想分類の１つまたは複数に割り当てることができる。たとえば、より長い期間の間発生する特定の顕著な変更には、より高い優先順位値が割り当てられることになる。

【００７６】

ブロック６０８では、最も高い優先順位値を有する予想分類が、考慮中の顕著な変更に割り振られる。一実装では、クエリーモジュール２１０は、予想分類に割り振られる最も高い優先順位値を決定し、その分類を、考慮中の顕著な変更に割り当てる。

【００７７】

図７は、１つまたは複数のコンピューティングベースのデバイス１０２上の－たとえばシステム管理者によって定義された－無許可の対話の実行を禁止するための例示的な方法７００を示す。無許可の対話の例には、そのようなアクションを実施することが許可されていないエンティティーまたはプログラムによる、ファイルシステムに対して実施される読取りおよび／または書込みアクションが含まれる。

【００７８】

この方法が述べられる順序は、限定するものとして解釈しないものとし、任意の数の、述べられている方法ブロックを、この方法または代替の方法を実施するために任意の順序で組み合わせることができる。さらに、個々のブロックは、本明細書に述べられている本主題の精神および範囲から逸脱することなしに、この方法から削除することができる。さらに、この方法は、任意の好適なハードウェア、ソフトウェア、ファームウェア、またはそれらの組合せで実装することができる。

【００７９】

ブロック７０２では、システム上で動作するプログラムに関連する情報が受け取られる。この情報は、プログラムとファイルシステムおよび／または構成設定との間の対話に関連付けられる。クエリーモジュール２１０は、プログラムによって１つまたは複数のコンピューティングベースのデバイス１０２上で実施される対話に関する情報に関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。クエリーの実行によって得られる情報は、１つまたは複数の属性によって特徴付けられる。

【００８０】

ブロック７０４では、システム上で動作するプログラムの属性が、事前定義されたリス

ト内に含まれる複数の承認済みプログラム／プロセスおよび未承認のプログラム／プロセスの属性と比較される。たとえば、クエリーモジュール２１０は、プログラムの属性、たとえばプログラムタイプと、事前定義されたリスト内に含まれるプログラムの属性を比較する。

【００８１】

ブロック７０６では、諸属性が、ある未承認のプログラム／プロセスまたは対話の属性に対応するかどうか判定される。たとえば、システム１００上で動作するプログラムの属性が未承認の対話に関連付けられた属性に対応する場合（すなわち、ブロック７０６からの「はい」の経路）、プログラムに関連する対話は、進むことが許されない（すなわち、ブロック７０８）。あるいは、システム１００上で動作するプログラムの属性が未承認の対話に関連付けられた属性に対応しない場合（すなわち、ブロック７０６からの「いいえ」の経路）、それらのプログラムに関連する対話は、進むことが許される（すなわち、ブロック７１０）。

【００８２】

図８は、１つまたは複数のコンピューティングベースのデバイス１０２上にインストールされたプログラムの１つまたは複数の拡張性ポイント（ＥＰ）を検出するための例示的な方法８００を示す。ＥＰは、コンピュータープリケーションの動的ロードおよび実行を制御する対話を含む。この方法が述べられる順序は、限定するものとして解釈しないものとし、任意の数の、述べられている方法ブロックを、この方法または代替の方法を実施するために任意の順序で組み合わせることができる。さらに、個々のブロックは、本明細書に述べられている本主題の精神および範囲から逸脱することなしに、この方法から削除することができる。さらに、この方法は、任意の好適なハードウェア、ソフトウェア、ファームウェア、またはそれらの組合せで実装することができる。

【００８３】

ブロック８０２では、先行の対話（すなわち、第１のプログラムの実行前に、実行するためにシステムメモリー内にロードされている様々なプログラムに関連する対話）が検査される。たとえば、クエリーモジュール２１０は、第１のプログラムの実行前に、実行するために１つまたは複数のコンピューティングベースのデバイス１０２のシステムメモリー内にロードされている様々なプログラムに関連する対話に関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることによって、第１のプログラムについての潜在的な直接拡張性ポイント（ＥＰ）を識別することができる。クエリーモジュール２１０は、第１のプログラムの実行前に、２秒など所与の時間範囲内で、実行するためにメモリー内にロードされている様々なプログラムに関連する対話に関してクエリーすることができる。

【００８４】

ブロック８０４では、実行するためにコンピューティングベースのデバイスのロードされている第１のプログラムのファイル名を参照する先行の対話を見つけるために、検査が実施される。たとえば、クエリーモジュール２１０は、第１のプログラムを参照する様々なプログラムに関連する、またはコンピューティングベースのデバイス１０２上での第１のプログラムの実行に関連する対話に関してクエリーすることができる。クエリーモジュール２１０は、第１のプログラムのファイル名、第１のプログラムのプログラムＩＤなど、様々な属性を含む対話に関してクエリーすることができる。

【００８５】

ブロック８０６では、第１のプログラムのファイル名を参照する先行の対話が、直接ＥＰとしてフラグされる。たとえば、クエリーモジュール２１０は、第１のプログラムを参照する、または第１のプログラムの実行に関連する先行の対話すべてに関してクエリーすることによって、第１のプログラムに関する直接ＥＰを識別することができる。

【００８６】

図９は、あるプログラムを１つまたは複数のコンピューティングベースのデバイス１０２からアンインストールした結果として残されているリークエンティティーを検出するた

めの例示的な方法９００を示す。この方法が述べられる順序は、限定するものとして解釈しないものとし、任意の数の、述べられている方法ブロックを、この方法または代替の方法を実施するために任意の順序で組み合わせることができる。さらに、個々のブロックは、本明細書に述べられている本主題の精神および範囲から逸脱することなしに、この方法から削除することができる。さらに、この方法は、任意の好適なハードウェア、ソフトウェア、ファームウェア、またはそれらの組合せで実装することができる。

【００８７】

ブロック９０２では、コンピューティングベースのデバイスおよび／またはシステム上にロードされている各プログラムに関連するインストールファイルおよび設定の変更がカタログ化され列挙される。列挙は、コンピューティングベースのデバイスのオペレーティングシステムに登録されているプログラムのリストを作成することを含む。

【００８８】

たとえば、システム１００をスキャンし、コンピューティングベースのデバイス上にインストールされているすべてのプログラム、ならびにそれらのコンピューティングベースのデバイス上のそれらのプログラムに関連付けられたすべてのオペレーティングシステムインストールファイルを検出することができる。コンピューティングベースのデバイス上にインストールされているすべてのプログラム、および／またはそれらのプログラムに関連付けられたすべてのオペレーティングシステムインストールファイルは、それらをリスト内に配置することによって列挙することができる。

【００８９】

クエリーモジュール２１０は、システム１００全体にわたってスキャンを実行し、１つまたは複数のコンピューティングベースのデバイス１０２上にインストールされているプログラムすべてを検出し、コンピューティングベースのデバイス１０２上のオペレーティングシステムインストールファイルすべてをカタログ化し列挙する。たとえば、クエリーモジュール２１０は、システム１００内の１つまたは複数のコンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムすべてに関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。見つけられたプログラムは、クエリーモジュール２１０、エージェント１１０など、様々なデバイスによってカタログ化し列挙することができる。さらに、見つけられたプログラムに関連付けられた、コンピューティングベースのデバイス１０２上のオペレーティングシステムインストールファイルすべてを、クエリーモジュール２１０、エージェント１１０など、様々なデバイスによってカタログ化し列挙することができる。

【００９０】

ブロック９０４では、アンインストールされたプログラムに関連付けられたファイルおよびレジストリー設定を含めて、コンピューティングベースのデバイスおよび／またはシステム上に存在する永続状態（ＰＳ）が列挙される。これは、アンインストールされているプログラム用のファイルおよびレジストリー設定を含む、コンピューティングベースのデバイスおよび／またはシステム上にインストールされていたプログラムすべてのファイルおよびレジストリー設定を求めて、コンピューティングベースのデバイスおよび／またはシステム上のメモリーをスキャンすることを含むことができる。たとえば、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２上にインストールされていたプログラムすべての、プログラムＩＤなど識別子に対応するファイルおよびレジストリー設定すべてを得るために、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。

【００９１】

ブロック９０６では、オペレーティングシステムに登録されているプログラムに関連付けられたファイルおよびレジストリー設定が、コンピューティングベースのデバイス１０２および／またはシステム１００上にインストールされていたプログラムのファイルおよびレジストリー設定に突き合わせて比較される。たとえば、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２のオペレーティングシステムに登録されて

いるプログラムに関連付けられた列挙済みのファイルおよびレジストリー設定の、プログラムＩＤなど識別子を、コンピューティングベースのデバイス１０２上にインストールされていたプログラムすべてのファイルまたはレジストリー設定の識別子に突き合わせて比較することができる。

【００９２】

ブロック９０８では、両リスト上のプログラムに関連付けられたファイルおよびレジストリー設定を考慮に入れないことができる。コンピューティングベースのデバイス１０２および／またはシステム１００からアンインストールされているプログラムに対応するファイルおよびレジストリー設定を表す、残りのファイルおよびレジストリー設定を、リークファイルとしてアノテーションすることができ、コンピューティングベースのデバイス１０２および／またはシステム１００から除去することができる。たとえば、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムに関連付けられたファイルおよびレジストリー設定の、プログラムＩＤなど識別子を、コンピューティングベースのデバイス１０２上にインストールされていたプログラムに関連付けられたファイルおよびレジストリー設定の識別子に相関させることができる。相関されていないプログラムに関連付けられたファイルおよびレジストリー設定は、クエリーモジュール２１０によってリークファイルと命名することができ、クエリーモジュール２１０、エージェント１１０など、諸要素によってコンピューティングベースのデバイス１０２から除去することができる。

【００９３】

図１０は、１つまたは複数のコンピューティングベースのデバイス１０２上にインストールされている、共通の構成誤りや旧ソフトウェアバージョンなどを含む無効のファイルを検出するための例示的な方法１０００を示す。無効のファイルは、たとえばソフトウェアアップグレードにより、ディスク上の実行可能ファイルを置き換えた後で、影響を受けたプロセスが再起動できなかったとき発生する。その結果、無効のプロセスが見出されるコンピューティングベースのデバイス１０２は、アップグレードを無視し、古いファイルから引き続き実行されることになる。この方法が述べられる順序は、限定するものとして解釈しないものとし、任意の数の、述べられている方法ブロックを、この方法または代替の方法を実施するために任意の順序で組み合わせることができる。さらに、個々のブロックは、本明細書に述べられている本主題の精神および範囲から逸脱することなしに、この方法から削除することができる。さらに、この方法は、任意の好適なハードウェア、ソフトウェア、ファームウェア、またはそれらの組合せで実装することができる。

【００９４】

ブロック１００２では、コンピューティングベースのデバイスおよび／またはシステム上にロードされているプログラムがカタログ化され列挙される。一実装では、列挙は、コンピューティングベースのデバイスのオペレーティングシステムに登録されているプログラムのリストを作成することを含む。たとえば、システムをスキャンし、そのシステム内のコンピューティングベースのデバイス上にインストールされているプログラムすべてを検出することができる。コンピューティングベースのデバイス上にインストールされているすべてのプログラムは、それらをリスト内に配置することによって列挙することができる。

【００９５】

可能な一実装では、クエリーモジュール２１０は、システム１００全体にわたってスキャンを実行し、１つまたは複数のコンピューティングベースのデバイス１０２上で登録されているプログラムすべてを検出し、そのコンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムすべてをカタログ化し列挙する。たとえば、クエリーモジュール２１０は、システム１００内の１つまたは複数のコンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムすべてに関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。見つけられたプログラムは、クエリーモジュール２１０

、エージェント１１０など、様々なデバイスおよび／またはエンティティーによってカタログ化し列挙することができる。

【００９６】

ブロック１００４では、コンピューティングベースのデバイスおよび／またはシステム上で登録されているプログラムすべての最後のロード時間、ならびにコンピューティングベースのデバイスおよび／またはシステム上で登録されているプログラムに関連付けられたファイルについての最後のロード時間が獲得される。たとえば、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２上に登録されているプログラムの最後のロード時間、および／またはシステム１００内のコンピューティングベースのデバイス１０２上に登録されているプログラムと共にインストールされた、システムダイナミックリンクライブラリ（ＤＬＬ）などファイルの最後のロード時間に関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。

【００９７】

ブロック１００６では、コンピューティングベースのデバイスおよび／またはシステム上で登録されているプログラムに関連付けられたファイルまたは設定の最後の修正時間が獲得され、それらのプログラムの最後のロード時間と比較される。たとえば、クエリーモジュール２１０は、コンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムの最後の修正の時間または日付に関して、ログストレージ２１８および／またはアーカイブコレクション１０６にクエリーすることができる。クエリーモジュール２１０は、その最後の修正の時間または日付を、そのプログラムの最後のロード時間と比較することができる。

【００９８】

ブロック１００８では、比較中に見つけられた不整合が記録される。たとえば、コンピューティングベースのデバイスおよび／またはシステム上で登録されているプログラムの最後のロード時間が、そのプログラムの最後の既知の修正の時間または日付より遅い場合、そのプログラムがその最後の修正に応答しなかった可能性がある。そのような場合には、エラーレポートを、ユーザーまたはシステム管理者などエンティティーに対して発行し、そのプログラムが、最後に試みられた修正に応答していないことをレポートすることができる。別法として、そのプログラムの最後に試みられた修正を再試行しようと試みることができる。

【００９９】

例示的な一実装では、クエリーモジュール２１０は、システム１００内のコンピューティングベースのデバイス１０２のオペレーティングシステムに登録されているプログラムの最後のロード時間と最後の修正時間の両方に関して、クエリーすることができる。クエリーモジュール２１０は、最後のロード時間と最後の修正時間を比較することができ、そのプログラムの最後のロード時間がそのプログラムの最後の修正時間より遅い場合、クエリーモジュール２１０は、エラーレポートを、ユーザーまたはシステム管理者などエンティティーに対して発行し、そのプログラムが、最後に試みられた修正に応答していないことをレポートすることができる。他の実装では、クエリーモジュール２１０はまた、そのプログラムの最後に試みられた修正を再試行しようと試みることができる。

【０１００】

（例示的なコンピューター環境）

図１１は、本明細書に述べられている技法を実施するために使用することができる、また、本明細書に述べられている諸要素を全体的に、または部分的に表すことができる例示的な一般コンピューター環境１１００を示す。コンピューター環境１１００は、コンピューティング環境の一例にすぎず、コンピューターおよびネットワークアーキテクチャーの使用または機能の範囲についてどんな限定をも暗示するものではない。コンピューター環境１１００は、例示的なコンピューター環境１１００内に示されている任意の１つの構成要素または構成要素の組合せに関してどんな依存または要件をも有していると解釈するべきではない。

【０１０１】
コンピューター環境１１００は、コンピューター１１０２の形態にある汎用のコンピューティングベースのデバイスを含む。コンピューター１１０２は、たとえば、デスクトップコンピューター、ハンドヘルドコンピューター、ノートブックコンピューターまたはラップトップコンピューター、サーバーコンピューター、ゲームコンソールなどとすることができる。コンピューター１１０２の構成要素には、それだけには限らないが、１つまたは複数のプロセッサまたは処理装置１１０４、システムメモリー１１０６、プロセッサ１１０４を含む様々なシステム構成要素をシステムメモリー１１０６に結合するシステムバス１１０８を含めることができる。
【０１０２】
システムバス１１０８は、メモリーバスまたはメモリーコントローラ、周辺バス、ＡＧＰ（ａｃｃｅｌｅｒａｔｅｄ ｇｒａｐｈｉｃｓ ｐｏｒｔ）、および様々なバスアーキテクチャーのいずれかを使用するプロセッサバスまたはローカルバスを含めて、いくつかのタイプのバス構造のいずれかの１つまたは複数を表す。例として、そのようなアーキテクチャーには、ＩＳＡ（Ｉｎｄｕｓｔｒｙ Ｓｔａｎｄａｒｄ Ａｒｃｈｉｔｅｃｔｕｒｅ）バス、ＭＣＡ（Ｍｉｃｒｏ Ｃｈａｎｎｅｌ Ａｒｃｈｉｔｅｃｔｕｒｅ）バス、ＥＩＳＡ（Ｅｎｈａｎｃｅｄ ＩＳＡ）バス、ＶＥＳＡ（Ｖｉｄｅｏ Ｅｌｅｃｔｒｏｎｉｃｓ Ｓｔａｎｄａｒｄｓ Ａｓｓｏｃｉａｔｉｏｎ）ローカルバス、およびメザニンバスとしても知られるＰＣＩ（Ｐｅｒｉｐｈｅｒａｌ Ｃｏｍｐｏｎｅｎｔ Ｉｎｔｅｒｃｏｎｎｅｃｔｓ）バスを含めることができる。
【０１０３】
コンピューター１１０２は、典型的には、様々なコンピューター可読媒体を含む。そのような媒体は、コンピューター１１０２によってアクセス可能であり、揮発性媒体と不揮発性媒体、取外し式媒体と非取外し式媒体を共に含む任意の使用可能な媒体とすることができる。
【０１０４】
システムメモリー１１０６は、ランダムアクセスメモリー（ＲＡＭ）１１１０など揮発性メモリー、および／または読出し専用メモリー（ＲＯＭ）１１１２など不揮発性メモリーの形態にあるコンピューター可読媒体を含む。基本入出力システム（ＢＩＯＳ）１１１４は、起動中などにコンピューター１１０２内の要素間で情報を転送するのを助ける基本ルーチンを含み、ＲＯＭ１１１２内に記憶されている。ＲＡＭ１１１０は、典型的には、処理装置１１０４にとって直ちにアクセス可能である、かつ／または処理装置１１０４によって現在処理されているデーターおよび／またはプログラムを含む。
【０１０５】
コンピューター１１０２はまた、取外し式／非取外し式の、揮発性／不揮発性コンピューター記憶媒体を含むことができる。例として、図１１は、非取外し式、不揮発性の磁気媒体（図示せず）との間で読出しおよび書込みを行うためのハードディスクドライブ１１１６、取外し式、不揮発性の磁気ディスク１１２０（たとえば、フロッピー（登録商標）ディスク）との間で読出しおよび書込みを行うための磁気ディスクドライブ１１１８、ならびにＣＤ－ＲＯＭ、ＤＶＤ－ＲＯＭまたは他の光媒体など、取外し式、不揮発性光ディスク１１２４との間で読出しおよび／または書込みを行うための光ディスクドライブ１１２２を示す。ハードディスクドライブ１１１６、磁気ディスクドライブ１１１８、および光ディスクドライブ１１２２はそれぞれ、１つまたは複数のデーター媒体インターフェース１１２６によってシステムバス１１０８に接続される。別法として、ハードディスクドライブ１１１６、磁気ディスクドライブ１１１８、および光ディスクドライブ１１２２は、１つまたは複数のインターフェース（図示せず）によってシステムバス１１０８に接続することができる。
【０１０６】
ディスクドライブや、それらの関連コンピューター可読媒体は、コンピューター可読命令、データー構造、プログラムモジュール、コンピューター１１０２用の他のデーターの

不揮発性記憶を実現する。この例は、ハードディスク１１１６、取外し式磁気ディスク１１２０、取外し式光ディスク１１２４を示しているが、磁気カセットまたは他の磁気記憶装置、フラッシュメモリーカード、ＣＤ－ＲＯＭ、ＤＶＤ（ｄｉｇｉｔａｌ　ｖｅｒｓａｔｉｌｅ　ｄｉｓｋ）または他の光記憶装置、ランダムアクセスメモリー（ＲＡＭ）、読出し専用メモリー（ＲＯＭ）、電気的消去可能なプログラム可能読出し専用メモリー（ＥＥＰＲＯＭ）など、コンピューターによってアクセス可能な、データーを記憶することができる他のタイプのコンピューター可読媒体をも使用し、例示的なコンピューティングシステムおよび環境を実施することができることを理解されたい。

【０１０７】

任意の数のプログラムモジュールを、ハードディスク１１１６、磁気ディスク１１２０、光ディスク１１２４、ＲＯＭ１１１２および／またはＲＡＭ１１１０上に記憶することができ、プログラムモジュールには、例としてオペレーティングシステム１１２７、１つまたは複数のアプリケーションプログラム１１２８、他のプログラムモジュール１１３０、プログラムデーター１１３２が含まれる。そのようなオペレーティングシステム１１２７、１つまたは複数のアプリケーションプログラム１１２８、他のプログラムモジュール１１３０、およびプログラムデーター１１３２（またはそれらの何らかの組合せ）のそれぞれは、分散ファイルシステムをサポートする常駐コンポーネントのすべてまたは一部を実装することができる。

【０１０８】

ユーザーは、キーボード１１３４およびポインティングデバイス１１３６（たとえば、マウス）など入力デバイスを介して、コマンドや情報をコンピューター１１０２に入力することができる。他の入力デバイス１１３８（特に図示せず）には、マイクロフォン、ジョイスティック、ゲームパッド、衛星パラボラアンテナ、シリアルポート、スキャナなどが含まれることがある。これらの、また他の入力デバイスは、システムバス１１０８に結合される入力／出力インターフェース１１４０を介して処理装置１１０４に接続されるが、パラレルポート、ゲームポート、ユニバーサルシリアルバス（ＵＳＢ）など、他のインターフェースおよびバス構造によって接続することもできる。

【０１０９】

また、モニタ１１４２または他のタイプのディスプレイデバイスを、ビデオアダプタ１１４４などインターフェースを介して、システムバス１１０８に接続することができる。モニタ１１４２に加えて、他の出力周辺デバイスには、入力／出力インターフェース１１４０を介してコンピューター１１０２に接続することができるスピーカ（図示せず）およびプリンタ１１４６など諸構成要素を含めることができる。

【０１１０】

コンピューター１１０２は、遠隔のコンピューティングベースのデバイス１１４８など、１つまたは複数の遠隔コンピューターとの論理接続を使用してネットワーク環境内で動作することができる。例として、遠隔のコンピューティングベースのデバイス１１４８は、パーソナルコンピューター、可搬型コンピューター、サーバー、ルータ、ネットワークコンピューター、ピアデバイスまたは他の共通ネットワークノードなどとすることができる。遠隔のコンピューティングベースのデバイス１１４８は、コンピューター１１０２に関して本明細書に述べられている要素および特徴の多数またはすべてを含むことができる可搬型コンピューターとして示されている。

【０１１１】

コンピューター１１０２と遠隔コンピューター１１４８の間の論理接続は、ローカルエリアネットワーク（ＬＡＮ）１１５０および一般的なワイドエリアネットワーク（ＷＡＮ）１１５２として示されている。そのようなネットワーク環境は、事務所、全社コンピュータネットワーク、イントラネット、およびインターネットでごく普通のものである。

【０１１２】

コンピューター１１０２は、ＬＡＮネットワーク環境内で実装されたとき、ネットワークインターフェースまたはアダプタ１１５４を介してローカルネットワーク１１５０に接

続される。コンピューター１１０２は、ＷＡＮネットワーク環境内で実装されたとき、典型的にはモデム１１５６、またはワイドネットワーク１１５２を介して通信を確立するための他の手段を含む。コンピューター１１０２の内蔵または外付けとすることができるモデム１１５６は、入力／出力インターフェース１１４０または他の適切な機構を介してシステムバス１１０８に接続することができる。図のネットワーク接続は例示的なものであり、コンピューター１１０２とコンピューター１１４８の間で通信リンクを確立する他の手段を使用することができることを理解されたい。

【０１１３】

コンピューティング環境１１００と共に示されているものなどネットワーク環境では、コンピューター１１０２またはその一部分に関連して示されているプログラムモジュールは、遠隔メモリー記憶装置内で記憶されてもよい。例として、遠隔アプリケーションプログラム１１５８が、遠隔コンピューター１１４８のメモリーデバイス上で常駐する。例示のために、アプリケーションプログラム、およびオペレーティングシステムなど他の実行可能プログラムコンポーネントが離散的なブロックとして本明細書に示されているが、そのようなプログラムおよびコンポーネントは、様々な時にコンピューティングベースのデバイス１１０２の様々な記憶構成要素内に常駐し、コンピューターのデータープロセッサによって実行されることを理解されたい。

【０１１４】

本明細書では、様々なモジュールおよび技法について、１つまたは複数のコンピューターまたは他のデバイスによって実行される、プログラムモジュールなどコンピューター実行可能命令の一般的な状況で述べることができる。一般に、プログラムモジュールは、特定のタスクを実行する、または特定の抽象データー型を実装するルーチン、プログラム、オブジェクト、コンポーネント、データー構造などを含む。典型的には、プログラムモジュールの機能は、様々な実施形態で望ましいように組み合わせる、または分配することができる。

【０１１５】

これらのモジュールの実装物は、何らかの形態のコンピューター可読媒体上に記憶する、またはそれを渡って送信することができる。コンピューター可読媒体は、コンピューターによってアクセスすることができる任意の使用可能な媒体とすることができる。限定ではなく例として、コンピューター可読媒体は、「コンピューター記憶媒体」および「通信媒体」を含むことができる。

【０１１６】

「コンピューター記憶媒体」には、コンピューター可読命令、データー構造、プログラムモジュール、または他のデーターなど情報を記憶するための任意の方法または技法で実施される揮発性／不揮発性、取外し式／非取外し式媒体が含まれる。コンピューター記憶媒体には、それだけには限らないが、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、フラッシュメモリーもしくは他のメモリー技術、ＣＤ－ＲＯＭ、ＤＶＤ（ｄｉｇｉｔａｌ　ｖｅｒｓａｔｉｌｅ　ｄｉｓｋ）もしくは他の光記憶装置、磁気カセット、磁気テープ、磁気ディスク記憶装置もしくは他の磁気記憶装置、または所望の情報を記憶するために使用することができ、コンピューターによってアクセスすることができる任意の他の媒体が含まれる。

【０１１７】

別法として、このフレームワークの各部分は、ハードウェア、あるいはハードウェア、ソフトウェア、および／またはファームウェアの組合せで実施することができる。たとえば、１つまたは複数の特定用途向け集積回路（ＡＳＩＣ）またはプログラマブルロジックデバイス（ＰＬＤ）を、このフレームワークの１つまたは複数の部分を実施するように設計またはプログラムすることができる。

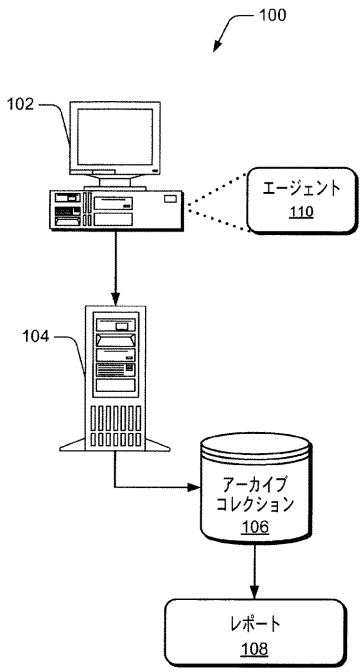
【０１１８】

（結び）

システム管理および解析の実施形態について、構造的特徴および／または方法に特有の言葉で述べたが、添付の特許請求の範囲の主題は、必ずしも述べられている特定の特徴ま

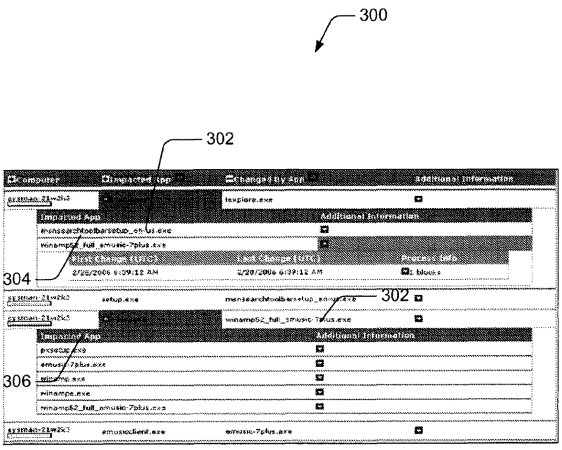
たは方法に限定されないことを理解されたい。それどころか、これらの特定の特徴および方法は、システム管理および解析の例示的な実装として開示されている。

【図１】

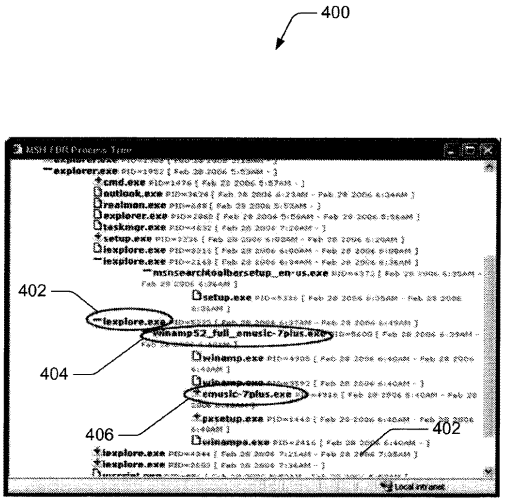


【図２】

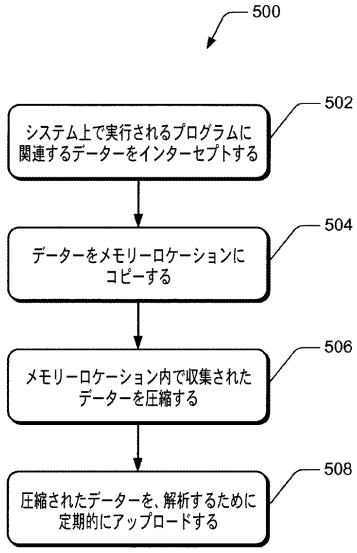
【図３】


300
302
304
306
302

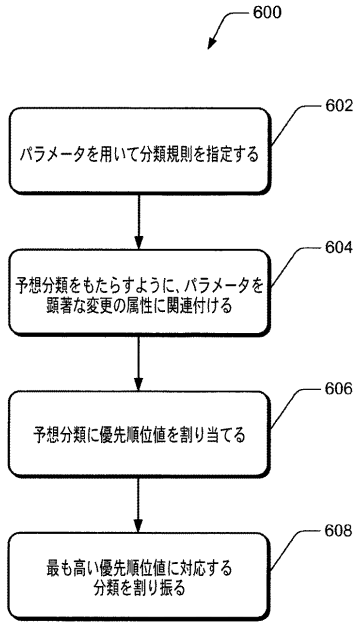
【図４】


400
402
404
406
402

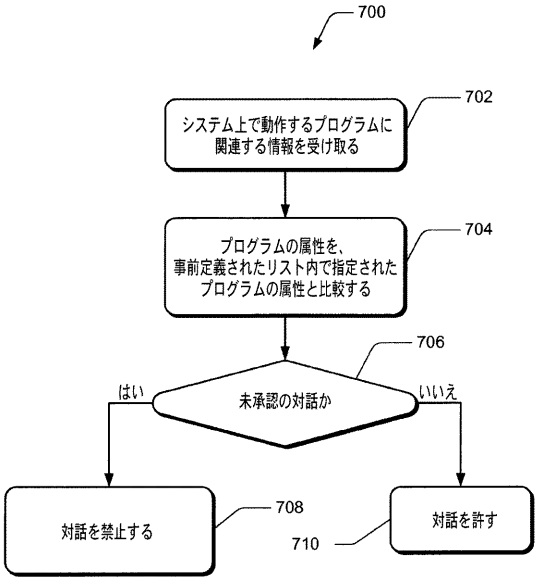
【図５】


500
システム上で実行されるプログラムに関連するデーターをインターセプトする
502
データーをメモリーロケーションにコピーする
504
メモリーロケーション内で収集されたデーターを圧縮する
506
圧縮されたデーターを、解析するために定期的にアップロードする
508

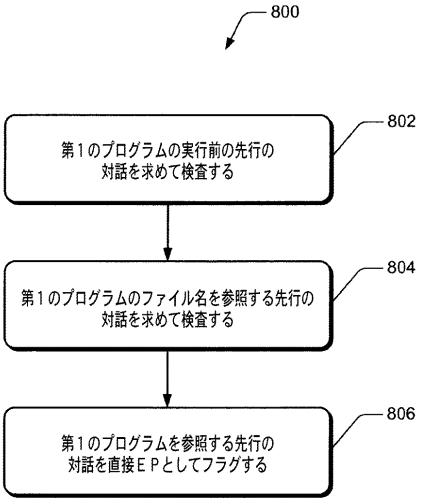
【図６】


600
パラメータを用いて分類規則を指定する
602
予想分類をもたらすように、パラメータを顕著な変更の属性に関連付ける
604
予想分類に優先順位値を割り当てる
606
最も高い優先順位値に対応する分類を割り振る
608

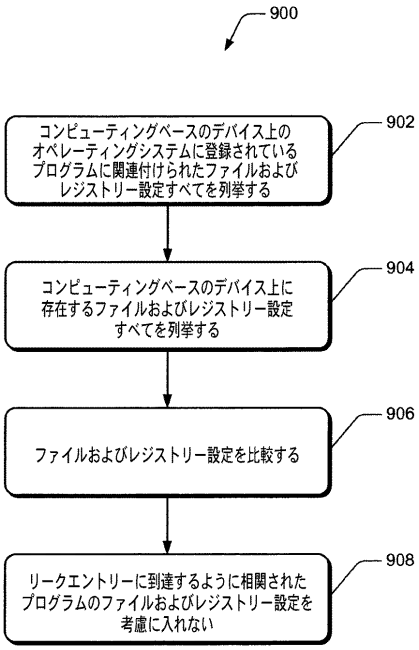
【図７】


700
システム上で動作するプログラムに関連する情報を受け取る
702
プログラムの属性を、事前定義されたリスト内で指定されたプログラムの属性と比較する
704
未承認の対話か
706
はい
いいえ
対話を禁止する
708
対話を許す
710

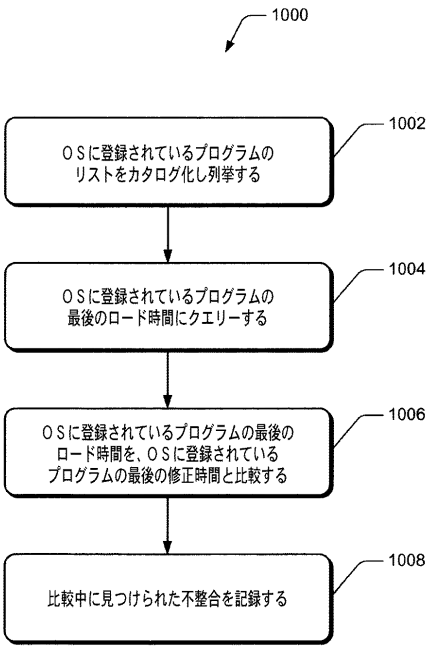
【図８】


800
第１のプログラムの実行前の先行の対話を求めて検査する
802
第１のプログラムのファイル名を参照する先行の対話を求めて検査する
804
第１のプログラムを参照する先行の対話を直接ＥＰとしてフラグする
806

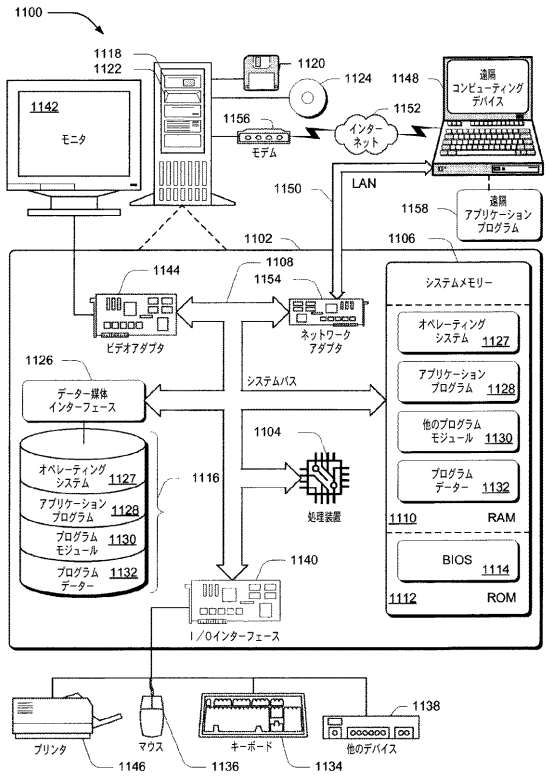
【図９】


900
コンピューティングベースのデバイス上のオペレーティングシステムに登録されているプログラムに関連付けられたファイルおよびレジストリー設定すべてを列挙する
902
コンピューティングベースのデバイス上に存在するファイルおよびレジストリー設定すべてを列挙する
904
ファイルおよびレジストリー設定を比較する
906
リークエントリーに到達するように相関されたプログラムのファイルおよびレジストリー設定を考慮に入れない
908

【図１０】


1000
ＯＳに登録されているプログラムのリストをカタログ化し列挙する
1002
ＯＳに登録されているプログラムの最後のロード時間にクエリーする
1004
ＯＳに登録されているプログラムの最後のロード時間を、ＯＳに登録されているプログラムの最後の修正時間と比較する
1006
比較中に見つけられた不整合を記録する
1008

【図１１】


1100
1118
1122
1120
1124
1148
遠隔
コンピューティング
デバイス
1142
モニタ
1156
モデム
インター
ネット
1152
1150
LAN
1158
遠隔
アプリケーション
プログラム
1102
1106
1144
1108
1154
システムメモリー
ビデオアダプタ
ネットワーク
アダプタ
オペレーティング
システム 1127
アプリケーション
プログラム 1128
他のプログラム
モジュール 1130
プログラム
データー 1132
1126
データー媒体
インターフェース
システムバス
1104
1116
処理装置
1110 RAM
BIOS 1114
1112 ROM
1140
I／Oインターフェース
1138
プリンタ
マウス
キーボード
他のデバイス
1146
1136
1134

フロントページの続き

(72)発明者 チャド ヴェルボウスキー
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 ジュハン リー
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 シャオガン リュー
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 ルーシー ローセブ
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内
(72)発明者 イーミン ワン
アメリカ合衆国 ９８０５２ ワシントン州 レッドモンド ワン マイクロソフト ウェイ マイクロソフト コーポレーション インターナショナル パテンツ内

審査官 多胡 滋

(56)参考文献 特開平１１－２３２１０９（ＪＰ，Ａ）
特開２００５－０６３２５９（ＪＰ，Ａ）
特開平０６－２２２９３３（ＪＰ，Ａ）
特開２００１－１９５２３８（ＪＰ，Ａ）
特開２００４－１２７２５３（ＪＰ，Ａ）
特開２００４－１６４３５１（ＪＰ，Ａ）
服部彩子，本当にパソコンを活用するための基礎から学ぶアップデート，日経パソコン，日本，日経ＢＰ社，２００２年 9月 2日，第416号，pp.102-113
南雲徹，ＷｉｎＡｄｖｉｓｏｒ５，Ｉ／Ｏ，日本，株式会社工学社，１９９８年 2月 1日，第23巻、第2号，pp.91-92

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 9/54
G06F 9/445
G06F 11/00
G06F 11/30