(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2012-128811
(P2012-128811A)
(43)公開日　平成24年7月5日(2012.7.5)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 11/34	(2006.01)	G06F	11/34	S	5B042
		G06F	11/34	L	

審査請求 未請求　請求項の数 8　ＯＬ　（全 25 頁）

(21)出願番号　特願2010-282212 (P2010-282212)
(22)出願日　平成22年12月17日 (2010.12.17)

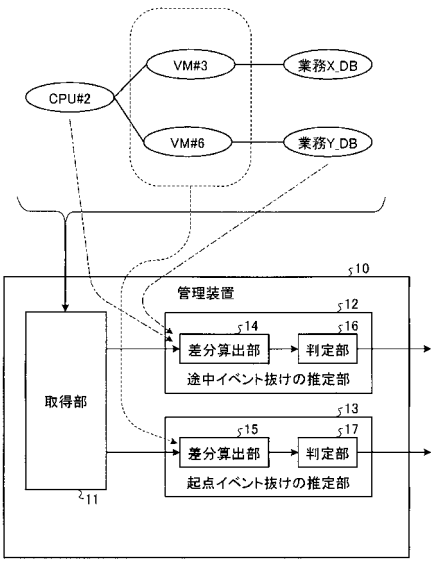
(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　松田　雄一
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

Ｆターム(参考)　5B042 GA12　MA14　MC35　MC40

(54)【発明の名称】管理装置、管理プログラム、および管理方法

(57)【要約】
【課題】イベントの検知に漏れがある場合であってもイベントの依存関係を判定すること。
【解決手段】管理装置１０の途中イベント抜けの推定部１２は、ＣＰＵでのイベントの発生時刻とＤＢでのイベントの発生時刻との差分を求め、差分が閾値Ｔｓ３以内である場合にＤＢのイベントがＣＰＵのイベントに基づくと判定する。また、管理装置１０の起点イベント抜け推定部１３は、複数のＶＭで発生したイベントの発生時刻の差分が閾値Ｔｓ４以内である場合に複数のＶＭのイベントがＣＰＵのイベントに基づくと判定する。このため、管理装置１０は、イベントの検知に漏れがある場合であってもイベントの依存関係を判定することができる。
【選択図】図１３

実施例1にかかる管理装置の説明図



【特許請求の範囲】
【請求項１】
第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して第２のイベントを発生する第２の管理対象と、前記第２の管理対象で発生した前記第２のイベントに依存して第３のイベントを発生する第３の管理対象とを管理する管理装置であって、
前記第１のイベントの発生時刻と前記第３のイベントの発生時刻との差分を求める差分算出部と、
前記差分算出部が算出した差分が所定時間以内である場合に前記第３のイベントが前記第１のイベントに基づくと判定する判定部と
を備えたことを特徴とする管理装置。
【請求項２】
第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して各々第２のイベントを発生する複数の第２の管理対象とを管理する管理装置であって、
前記複数の第２の管理対象で発生した複数の第２のイベントの発生時刻の差分を求める差分算出部と、
前記差分算出部が算出した差分が所定時間以内である場合に前記複数の第２のイベントが前記第１のイベントに基づくと判定する判定部と、
を備えたことを特徴とする管理装置。
【請求項３】
前記複数の第２の管理対象の数と、前記複数の第２の管理対象のうち前記第２のイベントを発生した前記第２の管理対象の数に基づいて信頼度を算出し、
前記判定部は、前記信頼度が所定値以上で、かつ前記差分が所定時間以内である場合に前記複数の第２のイベントが前記第１のイベントに基づくと判定することを特徴とする請求項２に記載の管理装置。
【請求項４】
前記判定部は、前記複数の第２のイベントが前記第１のイベントに基づくと判定した場合に、前記第１のイベントの発生時刻のダミー値を作成することを特徴とする請求項２または３に記載の管理装置。
【請求項５】
第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して第２のイベントを発生する第２の管理対象と、前記第２の管理対象で発生した前記第２のイベントに依存して第３のイベントを発生する第３の管理対象とを管理する管理プログラムであって、
前記第１のイベントの発生時刻と前記第３のイベントの発生時刻との差分を求める差分算出手順と、
前記差分算出手順で算出した差分が所定時間以内である場合に前記第３のイベントが前記第１のイベントに基づくと判定する判定手順と
をコンピュータに実行させることを特徴とする管理プログラム。
【請求項６】
第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して各々第２のイベントを発生する複数の第２の管理対象とを管理する管理プログラムであって、
前記複数の第２の管理対象で発生した複数の第２のイベントの発生時刻の差分を求める差分算出手順と、
前記差分算出手順で算出した差分が所定時間以内である場合に前記複数の第２のイベントが前記第１のイベントに基づくと判定する判定手順と
をコンピュータに実行させることを特徴とする管理プログラム。
【請求項７】
第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して第２のイベントを発生する第２の管理対象と、前記第２の管理対象で発生した前記第２のイベントに依存して第３のイベントを発生する第３の管理対象とを管理する管理方法であって、

前記第１のイベントの発生時刻と前記第３のイベントの発生時刻との差分を求める差分算出ステップと、
前記差分算出ステップで算出した差分が所定時間以内である場合に前記第３のイベントが前記第１のイベントに基づくと判定する判定ステップと
を含んだことを特徴とする管理方法。
【請求項８】
第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して各々第２のイベントを発生する複数の第２の管理対象とを管理する管理方法であって、
前記複数の第２の管理対象で発生した複数の第２のイベントの発生時刻の差分を求める差分算出ステップと、
前記差分算出ステップで算出した差分が所定時間以内である場合に前記複数の第２のイベントが前記第１のイベントに基づくと判定する判定ステップと
を含んだことを特徴とする管理方法。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、管理装置、管理プログラム、および管理方法に関する。
【背景技術】
【０００２】
従来、複数の機能を監視対象として所定期間内に収集したイベント情報をグループとし、パターン定義とイベントグループとの間でイベント情報の発生パターンを照合し、類似するパターン定義グループに予め関連づけられている障害対策情報を抽出する技術が知られている。また、事象ログデータについて、最初の事象と選択された以降のメッセージを事象ログに記憶し、重複したメッセージを記憶対象外とする技術が知られている。
【先行技術文献】
【特許文献】
【０００３】
【特許文献１】国際公開第２００４／０６１６８１号
【特許文献２】特表２００４－５３５０１８号公報
【発明の概要】
【発明が解決しようとする課題】
【０００４】
複数の監視対象について、監視対象間に依存関係がある場合、依存元の監視対象で発生したイベントが依存先の監視対象のイベントを引き起こす場合がある。依存関係にある各監視対象から所定時間内にイベントがあがったことを検知すれば、イベント間に関係があると判断できる。しかし、イベントが発生したことを検知できなかった場合、従来の技術では、依存関係があることを知ることができなかった。
【０００５】
開示の技術は、イベントの検知に漏れがある場合であってもイベントの依存関係を判定することを目的とする。
【課題を解決するための手段】
【０００６】
開示の管理装置、管理プログラム、および管理方法は、一つの態様において、第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して第２のイベントを発生する第２の管理対象と、前記第２の管理対象で発生した前記第２のイベントに依存して第３のイベントを発生する第３の管理対象とを管理する。開示の装置、プログラム、方法は、第１のイベントの発生時刻と前記第３のイベントの発生時刻との差分を求め、差分が所定時間以内である場合に前記第３のイベントが前記第１のイベントに基づくと判定する。
【０００７】

また、開示の管理装置、管理プログラム、および管理方法は、一つの態様において、第１の管理対象と、前記第１の管理対象で発生した第１のイベントに依存して各々第２のイベントを発生する複数の第２の管理対象とを管理する。開示の装置、プログラム、方法は、複数の第２の管理対象で発生した複数の第２のイベントの発生時刻の差分を求め、差分が所定時間以内である場合に前記複数の第２のイベントが前記第１のイベントに基づくと判定する。

【発明の効果】

【０００８】

開示の技術によれば、イベントの検知に漏れがある場合であってもイベントの依存関係を判定することができるという効果を奏する。

【図面の簡単な説明】

【０００９】

【図１】図１は、情報管理システム１００の一例を示す説明図である。

【図２】図２は、管理対象から発生するイベントのデータ構造の一例を示す説明図である。

【図３】図３は、実施の形態で用いられるコンピュータのハードウェア構成を示すブロック図である。

【図４】図４は、情報管理装置の機能的構成を示すブロック図である。

【図５】図５は、プロセス割当テーブルの記憶内容の一例を示す説明図である。

【図６】図６は、ＣＰＵ＃１を障害の基点とした場合の依存関係情報を示す説明図である。

【図７】図７は、ＣＰＵ＃２を障害の基点とした場合の依存関係情報を示す説明図である。

【図８】図８は、ＶＭを障害の基点とした場合の依存関係情報を示す説明図である。

【図９】図９は、業務プロセスを障害の基点とした場合の依存関係情報を示す説明図である。

【図１０】図１０は、判定部による判定処理の具体例（その１）を示す説明図である。

【図１１】図１１は、判定部による判定処理の具体例（その２）を示す説明図である。

【図１２】図１２は、統合管理ＤＢの記憶内容の一例を示す説明図である。

【図１３】図１３は、実施例１にかかる管理装置の説明図である。

【図１４】図１４は、途中イベント抜けの具体例の説明図（その１）である。

【図１５】図１５は、途中イベント抜けの具体例の説明図（その２）である。

【図１６】図１６は、起点イベント抜けの具体例の説明図（その１）である。

【図１７】図１７は、起点イベント抜けの具体例の説明図（その２）である。

【図１８】図１８は、本実施の形態にかかる情報管理装置による情報管理処理手順を示すフローチャートである。

【図１９】図１９は、図１８に示した依存関係判定処理の詳細な処理手順を示すフローチャートである。

【図２０】図２０は、図１８に示した障害発生起点の判定処理の詳細な処理手順を示すフローチャートである。

【図２１】図２１は、図２０に示した途中イベント抜け判定処理の詳細について説明するフローチャートである。

【図２２】図２２は、図２０に示した起点イベント抜け判定処理の詳細について説明するフローチャートである。

【図２３】図２３は、起点イベント抜け判定処理の変形例の説明図である。

【図２４】図２４は、起点イベント抜け判定処理の変形例のフローチャートである。

【発明を実施するための形態】

【００１０】

以下に添付図面を参照して、本発明にかかる管理装置、管理プログラム、および管理方法の実施の形態を詳細に説明する。

【００１１】
（情報管理システムの一例）
図１は、情報管理システム１００の一例を示す説明図である。情報管理システム１００は、管理対象装置１０１と管理対象装置１０１を管理する管理機能１０２と統合管理データベース（ＤＢ）１０３を備える。情報管理システム１００は、１台のコンピュータでもよく、複数台のコンピュータで構成してもよい。
【００１２】
まず、管理対象装置１０１について説明する。管理対象装置１０１は、複数種類の管理対象群の集合である。たとえば、管理対象装置１０１をクラウドコンピューティングに適用する場合、ＣＰＵ（Ｃｅｎｔｒａｌ　Ｐｒｏｃｅｓｓｉｎｇ　Ｕｎｉｔ）とＶＭ（Ｖｉｒｔｕａｌ　Ｍａｃｈｉｎｅ：仮想計算機）と業務プロセスの３種類を管理対象とすることができる。
【００１３】
図１では、たとえば、ＣＰＵ１１１としてＣＰＵ＃１、ＣＰＵ＃２、ＶＭ１１２としてＶＭ＃１～ＶＭ＃６、業務プロセス１１３として業務Ｘ用プロセス１１３Ｘ（Ｘ＿Ｗｅｂ，Ｘ＿ＡＰ，Ｘ＿ＤＢ），業務Ｙ用プロセス１１３Ｙ（Ｙ＿Ｗｅｂ，Ｙ＿ＡＰ，Ｙ＿ＤＢ）を管理対象とする。なお、Ｘ＿Ｗｅｂ，Ｙ＿ＷｅｂはＷｅｂサーバとして機能するプログラムである。また、Ｘ＿ＡＰ，Ｙ＿ＡＰはアプリケーションサーバとして機能するプログラムである。Ｘ＿ＤＢ，Ｙ＿ＤＢはデータベースサーバとして機能するプログラムである。
【００１４】
また、図１の例では、ＣＰＵ＃１がＶＭ＃１，ＶＭ＃２，ＶＭ＃４，ＶＭ＃５を制御し、ＣＰＵ＃２がＶＭ＃３，ＶＭ＃６を制御する。また、ＶＭ＃１がＸ＿Ｗｅｂを制御する。また、ＶＭ＃２がＸ＿ＡＰを制御する。また、ＶＭ＃３がＸ＿ＤＢを制御する。また、ＶＭ＃４がＹ＿Ｗｅｂを制御する。また、ＶＭ＃５がＹ＿ＡＰを制御する。また、ＶＭ＃６がＹ＿ＤＢを制御する。
【００１５】
管理対象装置１０１では、ＣＰＵ１１１がＶＭ１１２を制御し、ＶＭ１１２が業務プロセス１１３を制御する。このため、制御主体となる管理対象において障害が発生すると、その障害が原因となって制御対象となる管理対象にも障害が発生する。たとえば、ＣＰＵ＃１で障害が発生すると、ＶＭ＃１，ＶＭ＃２，ＶＭ＃４，ＶＭ＃５にも障害が発生する。同様に、ＶＭ＃１で障害が発生すると、その障害が原因となってＸ＿Ｗｅｂにも障害が発生する。
【００１６】
このように、障害発生に関して、制御対象となる管理対象は、制御主体となる管理対象に依存しているため、制御主体となる管理対象を以後、「依存元管理対象」と称す。また、制御対象となる管理対象を「依存先管理対象」と称す。図１では、ＣＰＵ１１１は、ＶＭ１１２に対して依存元管理対象となり、ＶＭ１１２はＣＰＵ１１１に対して依存先管理対象となる。同様に、ＶＭ１１２は、業務プロセス１１３に対して依存元管理対象となり、業務プロセス１１３はＶＭ１１２に対して依存先管理対象となる。このように、依存元管理対象と依存先管理対象との関わりを、依存関係と称す。
【００１７】
このように、ＣＰＵ１１１は、依存元管理対象にはなるが依存先管理対象にはならず、業務プロセス１１３は、依存先管理対象にはなるが依存元管理対象にはならない。また、ＶＭ１１２は、依存元管理対象にも依存先管理対象にもなり得る。
【００１８】
つぎに、管理機能１０２について説明する。管理機能１０２は、管理対象の種類ごとに管理機能１０２を有する。たとえば、ＣＰＵ１１１に対してはＣＰＵ管理機能１２１、ＶＭ１１２に対してはＶＭ管理機能１２２、業務プロセス１１３に対しては業務管理機能１２３を有する。

【００１９】
ＣＰＵ管理機能１２１は、管理対象装置１０１内のＣＰＵ１１１を管理するソフトウェアである。ＶＭ管理機能１２２は、管理対象装置１０１内のＶＭ１１２を管理するソフトウェアである。業務管理機能１２３は、管理対象装置１０１内の業務プロセス１１３を管理するソフトウェアである。各管理機能１２１～１２３は、それぞれＤＢ１２４～１２６を有し、各々の管理対象から障害や故障、通信状態の監視状態の変化が起こったときに通知されるイベントを収集し、ログとして保存する。
【００２０】
また、管理機能１０２は、統合管理機能１２７を有する。統合管理機能１２７は、管理対象の種類ごとに分散して保存されたイベントを収集して、ログとして統合管理ＤＢ１０３に保存する。本実施の形態では、各管理機能１２１～１２３のＤＢ１２４～１２６に保存されたイベントとの重複保存の低減化を図るため、統合管理ＤＢ１０３に保存するイベントを絞り込む。
【００２１】
具体的には、たとえば、管理者や統合管理機能１２７から見れば、複数のイベントの中から障害が発生している基点から通知される障害イベントが重要である。したがって、ＤＢ１２４～１２６から収集された障害イベントのうち障害箇所となる管理対象を特定するのに必要なイベントをログとして統合管理ＤＢ１０３に保存する。それ以外のイベントは、ＤＢ１２４～１２６に保存されているため、統合管理ＤＢ１０３に保存しなくても、統合管理ＤＢ１０３に保存したイベントを手がかりにして、必要に応じて読み出せばよい。
【００２２】
（イベントのデータ構造の一例）
つぎに、上述した管理対象から発生するイベントのデータ構造について説明する。
【００２３】
図２は、管理対象から発生するイベントのデータ構造の一例を示す説明図である。イベントは、番号項目２０１、タイムスタンプ項目２０２、イベント種類項目２０３、発生箇所項目２０４、警報種類項目２０５、予備項目２０６といった項目を有する。番号項目２０１には、イベントフレームに付けられるシリアル番号が記述される。タイムスタンプ項目２０２には、イベントの発生時刻（たとえば、２００９＿０９＿０５＿１７:５８:２３）が記述される。
【００２４】
イベント種類項目２０３には、イベント種類を識別するフラグ（たとえば、「０」が警報イベント、「１」が品質監視イベント）が記述される。発生箇所項目２０４には、イベントの発生箇所となる管理対象の識別情報（たとえば、ＣＰＵ＃１、ＶＭ＃２、Ｗｅｂ＃１など）が記述される。警報種類項目２０５には、警報の種類に関する識別情報（装置関連、ＶＭ１１２関連、アプリ関連、通信関連、品質関連などの識別情報）が記述される。予備項目２０６には、必要に応じて設定された情報が記述される。
【００２５】
（コンピュータのハードウェア構成）
図３は、実施の形態で用いられるコンピュータのハードウェア構成を示すブロック図である。図３において、コンピュータは、ＣＰＵ３０１と、ＲＯＭ（Ｒｅａｄ‐Ｏｎｌｙ Ｍｅｍｏｒｙ）３０２と、ＲＡＭ（Ｒａｎｄｏｍ Ａｃｃｅｓｓ Ｍｅｍｏｒｙ）３０３と、磁気ディスクドライブ３０４と、磁気ディスク３０５と、光ディスクドライブ３０６と、光ディスク３０７と、ディスプレイ３０８と、インターフェース（Ｉｎｔｅｒｆａｃｅ 以下、「Ｉ／Ｆ」と略する。）３０９と、キーボード３１０と、マウス３１１と、スキャナ３１２と、プリンタ３１３と、を備えている。また、各構成部はバス３００によってそれぞれ接続されている。
【００２６】
ここで、ＣＰＵ３０１は、コンピュータの全体の制御を司る。ＲＯＭ３０２は、ブートプログラムなどのプログラムを記憶している。ＲＡＭ３０３は、ＣＰＵ３０１のワークエ

リアとして使用される。磁気ディスクドライブ３０４は、ＣＰＵ３０１の制御にしたがって磁気ディスク３０５に対するデータのリード／ライトを制御する。磁気ディスク３０５は、磁気ディスクドライブ３０４の制御で書き込まれたデータを記憶する。

【００２７】

光ディスクドライブ３０６は、ＣＰＵ３０１の制御にしたがって光ディスク３０７に対するデータのリード／ライトを制御する。光ディスク３０７は、光ディスクドライブ３０６の制御で書き込まれたデータを記憶したり、光ディスク３０７に記憶されたデータをコンピュータに読み取らせたりする。

【００２８】

ディスプレイ３０８は、カーソル、アイコンあるいはツールボックスをはじめ、文書、画像、機能情報などのデータを表示する。このディスプレイ３０８は、たとえば、ＣＲＴ、ＴＦＴ液晶ディスプレイ、プラズマディスプレイなどを採用することができる。

【００２９】

Ｉ／Ｆ３０９は、通信回線を通じてＬＡＮ（Ｌｏｃａｌ Ａｒｅａ Ｎｅｔｗｏｒｋ）、ＷＡＮ（Ｗｉｄｅ Ａｒｅａ Ｎｅｔｗｏｒｋ）、インターネットなどのネットワークに接続され、このネットワーク３１４を介して他の装置に接続される。そして、Ｉ／Ｆ３０９は、ネットワーク３１４と内部のインターフェースを司り、外部装置からのデータの入出力を制御する。Ｉ／Ｆ３０９には、たとえばモデムやＬＡＮアダプタなどを採用することができる。

【００３０】

キーボード３１０は、文字、数字、各種指示などの入力のためのキーを備え、データの入力をおこなう。また、タッチパネル式の入力パッドやテンキーなどであってもよい。マウス３１１は、カーソルの移動や範囲選択、あるいはウィンドウの移動やサイズの変更などをおこなう。ポインティングデバイスとして同様に機能を備えるものであれば、トラックボールやジョイスティックなどであってもよい。

【００３１】

スキャナ３１２は、画像を光学的に読み取り、コンピュータ内に画像データを取り込む。なお、スキャナ３１２は、ＯＣＲ（Ｏｐｔｉｃａｌ Ｃｈａｒａｃｔｅｒ Ｒｅａｄｅｒ）機能を持たせてもよい。また、プリンタ３１３は、画像データや文書データを印刷する。プリンタ３１３には、たとえば、レーザプリンタやインクジェットプリンタを採用することができる。

【００３２】

（情報管理装置４００の機能的構成）

情報管理装置４００の機能的構成について説明する。図４は、情報管理装置４００の機能的構成を示すブロック図である。情報管理装置４００は、図１に示した統合管理機能１２７に相当する。情報管理装置４００は、取得部４０１と、特定部４０２と、抽出部４０３と、判定部４０４と、決定部４０５と、算出部４０６と、保存部４０７と、を備える。取得部４０１～保存部４０７は、具体的には、たとえば、図３に示したＲＯＭ３０２、ＲＡＭ３０３、磁気ディスク３０５、光ディスク３０７などの記憶装置に記憶されたプログラムをＣＰＵ３０１に実行させることにより、または、Ｉ／Ｆ３０９により、その機能を実現する。

【００３３】

取得部４０１は、管理対象の種類ごとのイベントが格納された管理対象の種類ごとのデータベース群から所定期間内に発生したイベント群を取得する機能を有する。具体的には、たとえば、ＤＢ１２４～１２６に保存されているイベントのタイムスタンプを参照することにより、所定期間内に発生したイベント群を読み出す。

【００３４】

特定部４０２は、取得部４０１によって取得されたイベント群内の各イベントに記述されている発生元の管理対象に関する情報に基づいて、依存関係がある管理対象群を特定する機能を有する。具体的には、たとえば、取得部４０１によって取得された各イベントの

発生箇所項目２０４には、発生元の管理対象の識別情報が記述されている。この識別情報を手がかりとして、依存関係がある管理対象群を特定する。

【００３５】

たとえば、取得された各イベントの発生箇所項目２０４に、「ＣＰＵ＃２」、「ＶＭ＃３」、「ＶＭ＃６」、「Ｘ＿ＤＢ」、「Ｙ＿ＤＢ」が記述されている場合、「ＣＰＵ＃２」、「ＶＭ＃３」、「ＶＭ＃６」、「Ｘ＿ＤＢ」、「Ｙ＿ＤＢ」を依存関係のある管理対象群として特定する。このような特定部４０２による特定では、プロセス割当テーブルを用いることができる。

【００３６】

図５は、プロセス割当テーブルの記憶内容の一例を示す説明図である。プロセス割当テーブル５００は、番号項目５０１と管理対象項目５０２とを有する。番号項目５０１には、レコード順に昇順の番号が記憶されている。管理対象項目５０２は、管理対象の種類別に分けられている。図５では、ＣＰＵ項目とＶＭ項目と業務プロセス項目に分けられている。このように、プロセス割当テーブル５００は、管理対象装置１０１内部において、ＣＰＵ１１１、ＶＭ１１２、業務プロセス１１３のそれぞれがどのように割り当てられているかを示している。

【００３７】

たとえば、番号１のレコードでは、ＣＰＵ＃１、ＶＭ＃１、Ｘ＿Ｗｅｂが記憶されている。番号１のレコードは、業務プロセス１１３であるＸ＿ＷｅｂはＶＭ＃１に割り当てられており、ＶＭ＃１はＣＰＵ＃１に割り当てられていることを意味する。なお、プロセス割当テーブル５００はあらかじめ管理者によって設定されているものとする。

【００３８】

なお、プロセス割当テーブル５００は、図３に示したＲＯＭ３０２、ＲＡＭ３０３、磁気ディスク３０５、光ディスク３０７などの記憶装置により、その機能を実現する。

【００３９】

図６～図９は、依存関係情報を示す説明図である。依存関係情報とは、ある管理対象で発生した障害がどの範囲まで影響するのかを表現した情報である。障害は、依存元管理対象から依存先管理対象に伝搬するため、依存元管理対象ごとに、依存関係情報が設定される。なお、図６～図９中、楕円は管理対象を示すノードであり、ノード間のリンクは依存関係を示している。すなわち、リンクで結ばれている左側のノードが依存元管理対象であり、右側のノードが依存先管理対象である。したがって、依存関係情報において、左端のノードが障害の基点となる管理対象を示している。

【００４０】

図６および図７は、ＣＰＵ１１１を障害の基点とした場合の依存関係情報を示す説明図である。特に図６は、ＣＰＵ＃１を障害の基点とした場合の依存関係情報６００である。図７は、ＣＰＵ＃２を障害の基点とした場合の依存関係情報７００である。

【００４１】

図８は、ＶＭ１１２を障害の基点とした場合の依存関係情報を示す説明図である。（Ａ）は、ＶＭ＃１を障害の基点とした場合の依存関係情報８０１である。（Ｂ）は、ＶＭ＃２を障害の基点とした場合の依存関係情報８０２である。（Ｃ）は、ＶＭ＃３を障害の基点とした場合の依存関係情報８０３である。

【００４２】

（Ｄ）は、ＶＭ＃４を障害の基点とした場合の依存関係情報８０４である。（Ｅ）は、ＶＭ＃５を障害の基点とした場合の依存関係情報８０５である。（Ｆ）は、ＶＭ＃６を障害の基点とした場合の依存関係情報８０６である。

【００４３】

図９は、業務プロセス１１３を障害の基点とした場合の依存関係情報を示す説明図である。（Ａ）は、Ｘ＿Ｗｅｂを障害の基点とした場合の依存関係情報９０１である。（Ｂ）は、Ｘ＿ＡＰを障害の基点とした場合の依存関係情報９０２である。（Ｃ）は、Ｘ＿ＤＢを障害の基点とした場合の依存関係情報９０３である。

【００４４】
（Ｄ）は、Ｙ＿Ｗｅｂを障害の基点とした場合の依存関係情報９０４である。（Ｅ）は、Ｙ＿ＡＰを障害の基点とした場合の依存関係情報９０５である。（Ｆ）は、Ｙ＿ＤＢを障害の基点とした場合の依存関係情報９０６である。
【００４５】
また、基点となる管理対象（左端のノード）から末端の管理対象（右端のノード）までの経路をルートと称す。この経路はパスとも呼ばれる。たとえば、図６の依存関係情報６００は、｛ＣＰＵ＃１→ＶＭ＃１→Ｘ＿Ｗｅｂ｝、｛ＣＰＵ＃１→ＶＭ＃２→Ｘ＿ＡＰ｝、｛ＣＰＵ＃１→ＶＭ＃４→Ｙ＿Ｗｅｂ｝、｛ＣＰＵ＃１→ＶＭ＃５→Ｙ＿ＡＰ｝の４本のルートを有する。
【００４６】
依存関係情報は、プロセス割当テーブル５００と同様、あらかじめ管理者によって設定されているものとしてもよい。ＸＭＬ（Ｅｘｔｅｎｓｉｂｌｅ Ｍａｒｋｕｐ Ｌａｎｇｕａｇｅ）形式の場合、依存関係情報をツリー構造で表現することができる。このように、あらかじめ設定されている場合、特定部４０２では、取得部４０１によって取得された各イベントの発生箇所項目２０４に記述されている発生元の管理対象の識別情報を手がかりとして、依存関係がある管理対象群としての依存関係情報を特定する。
【００４７】
たとえば、取得したイベント群の発生箇所項目２０４にＣＰＵ１１１に属する識別情報（たとえば、ＣＰＵ＃１）が記述されている場合、依存関係情報の中から図６の依存関係情報６００を特定する。
【００４８】
また、取得したイベント群の発生箇所項目２０４にＶＭ１１２に属する識別情報（たとえば、ＶＭ＃２）が記述されており、かつ、ＣＰＵ１１１に属する識別情報が記述されていない場合、依存関係情報の中から図８の（Ｂ）の依存関係情報８０２を特定する。
【００４９】
さらに、取得したイベント群の発生箇所項目２０４に業務プロセス１１３に属する識別情報（たとえば、Ｘ＿ＤＢ）が記述されており、かつ、ＣＰＵ１１１およびＶＭ１１２に属する識別情報が記述されていない場合、依存関係情報の中から図９の（Ｃ）の依存関係情報９０３を特定する。
【００５０】
また、依存関係情報をあらかじめ設定しておかず、特定部４０２によりプロセス割当テーブル５００から検索することにより、該当する依存関係情報を特定することとしてもよい。具体的には、たとえば、リレーショナルＤＢの内部にプロセス割当テーブル５００を作成しておき、プロセス割当テーブル５００に対して、予め用意したＳＱＬ（Ｓｔｒｕｃｔｕｒｅｄ Ｑｕｅｒｙ Ｌａｎｇｕａｇｅ）の検索式を実行する。これにより、得られる結果セット（テーブル形式）を該当する依存関係情報として特定することができる。
【００５１】
プロセス割当テーブル５００から検索して該当する依存関係情報として特定することにより、あらかじめ依存関係情報を作成する負担がない。また、検索する都度、該当する依存関係情報をメモリに書き出せばよいため、すべての依存関係情報を用意する必要がなく、メモリ使用量の削減を図ることができる。
【００５２】
なお、依存関係情報６００，７００，８０１～８０６，９０１～９０６は、図３に示したＲＯＭ３０２、ＲＡＭ３０３、磁気ディスク３０５、光ディスク３０７などの記憶装置により、その機能を実現する。
【００５３】
また、図４において、抽出部４０３は、依存関係がある管理対象群で発生したイベント群の中から、依存元管理対象で発生した第１のイベントと依存元管理対象に依存する依存先管理対象で発生した第２のイベントとの組み合わせを抽出する機能を有する。

【００５４】
具体的には、たとえば、該当する依存関係情報内の各リンクの両端のノードの組み合わせを抽出する。たとえば、図６の依存関係情報６００の場合、｛ＣＰＵ＃１，ＶＭ＃１｝，｛ＶＭ＃１，Ｘ＿Ｗｅｂ｝，｛ＣＰＵ＃１，ＶＭ＃２｝，｛ＶＭ＃２，Ｘ＿ＡＰ｝，｛ＣＰＵ＃１，ＶＭ＃４｝，｛ＶＭ＃４，Ｙ＿Ｗｅｂ｝，｛ＣＰＵ＃１，ＶＭ＃５｝，｛ＶＭ＃５，Ｙ＿ＡＰ｝の８個の組み合わせが抽出される。
【００５５】
判定部４０４は、抽出部４０３によって抽出された組み合わせごとに、第１のイベントの発生時刻と第２のイベントの発生時刻との差分により、第１のイベントと第２のイベントとの依存関係の有無を判定する機能を有する。
【００５６】
具体的には、たとえば、抽出部４０３によって抽出された組み合わせの一方の管理対象で発生したイベントの発生時刻をそのタイムスタンプから読み出す。同様に、他方の管理対象で発生したイベントの発生時刻をそのタイムスタンプから読み出す。そして、両タイムスタンプの差分を算出する。
【００５７】
差分は、両タイムスタンプの時間差の絶対値とする。通常、依存元管理対象で発生したイベントが依存先管理対象で発生したイベントよりも先に検出されるが、何らかの原因で依存先管理対象で発生したイベントが先に検出されることもある。このため、両タイムスタンプの時間差の絶対値を差分とする。そして、判定部４０４は、差分がしきい値Ｔｓ以内の場合、両イベント間に障害の依存関係ありと判定する。一方、差分がしきい値Ｔｓ以内ではない場合、両イベント間に障害の依存関係なしと判定する。
【００５８】
図１０は、判定部４０４による判定処理の具体例（その１）を示す説明図である。ここでは、図８の（Ａ）に示した依存関係情報８０１から得られた組み合わせ｛ＶＭ＃１，Ｘ＿Ｗｅｂ｝を例に挙げ、ＶＭ＃１では時刻Ｔ１でイベントＥ１が発生し、Ｘ＿Ｗｅｂでは時刻Ｔ２でイベントＥ２が発生したものとする。
【００５９】
（Ａ）では、差分｜Ｔ２－Ｔ１｜≦Ｔｓとなるため、イベントＥ１，Ｅ２は障害の依存関係ありと判定される。（Ｂ）では、差分｜Ｔ２－Ｔ１｜＞Ｔｓとなるため、イベントＥ１，Ｅ２は障害の依存関係なしと判定される。
【００６０】
図１１は、判定部４０４による判定処理の具体例（その２）を示す説明図である。ここでは、図７に示した依存関係情報７００から得られた４個の組み合わせ｛ＣＰＵ＃２，ＶＭ＃３｝，｛ＶＭ＃３，Ｘ＿ＤＢ｝，｛ＣＰＵ＃２，ＶＭ＃６｝，｛ＶＭ＃６，Ｙ＿ＤＢ｝を例に挙げる。また、ＣＰＵ＃２では時刻Ｔ１でイベントＥ１が発生し、ＶＭ＃３では時刻Ｔ２１でイベントＥ２１が発生し、Ｘ＿ＤＢでは時刻Ｔ３１でイベントＥ３１が発生し、ＶＭ＃６では時刻Ｔ２２でイベントＥ２２が発生し、Ｙ＿ＤＢでは時刻Ｔ３２でイベントＥ３２が発生したものとする。
【００６１】
また、ＣＰＵ１１１とＶＭ１１２との間のしきい値ＴｓをＴｓ１とし、ＶＭ１１２と業務プロセス１１３との間のしきい値ＴｓをＴｓ２とする。しきい値Ｔｓ１，Ｔｓ２は、管理者が自由に設定でき、Ｔｓ１＝Ｔｓ２でもよく、Ｔｓ１≠Ｔｓ２でもよい。
【００６２】
本例では、４個の組み合わせ｛ＣＰＵ＃２，ＶＭ＃３｝，｛ＶＭ＃３，Ｘ＿ＤＢ｝，｛ＣＰＵ＃２，ＶＭ＃６｝，｛ＶＭ＃６，Ｙ＿ＤＢ｝が抽出されるため、それぞれ差分｜Ｔ２１－Ｔ１｜，｜Ｔ３１－Ｔ２１｜，｜Ｔ２２－Ｔ１｜，｜Ｔ３２－Ｔ２２｜を算出し、対応するしきい値Ｔｓ１，Ｔｓ２以内であるかを判定することとなる。図１１の例では、すべての差分｜Ｔ２１－Ｔ１｜，｜Ｔ３１－Ｔ２１｜，｜Ｔ２２－Ｔ１｜，｜Ｔ３２－Ｔ２２｜が対応するしきい値Ｔｓ１，Ｔｓ２以内である。したがって、イベントＥ１，Ｅ２

１，Ｅ３１，Ｅ２２，Ｅ３２は依存関係ありと判定される。

【００６３】

加えて、判定部４０４は、途中のイベントや起点のイベントが抜けた場合についても、依存関係を判定する。この判定部４０４の処理動作の具体例については後述する。

【００６４】

また、図４に戻って、決定部４０５は、判定部４０４によって判定された判定結果に基づいて、イベント群のうち、依存先管理対象にならない依存元管理対象で発生したイベントを保存対象イベントに決定する機能を有する。

【００６５】

具体的には、判定部４０４によって組み合わせのすべてにおいて依存関係有りと判定された場合、依存先管理対象にならない依存元管理対象で発生したイベントを保存対象イベントに決定する。たとえば、依存関係情報において左端のノードとなる管理対象は、依存先管理対象にならない依存元管理対象であるため、依存関係情報において左端のノードとなる管理対象が障害の起点となる。したがって、依存関係情報において左端のノードとなる管理対象で発生したイベントを保存対象イベントに決定する。

【００６６】

たとえば、図１０の（Ａ）に示した例では、ＶＭ＃１で発生したイベントＥ１が保存対象イベントに決定される。したがって、２個のイベントＥ１，Ｅ２のうち、決定部４０５によりイベントＥ１が保存対象イベントとなるため、双方のイベントを保存する場合に比して５０％の削減効果が得られる。

【００６７】

また、図１１に示した例では、ＣＰＵ＃２で発生したイベントＥ１が保存対象イベントに決定される。したがって、５個のイベントＥ１，Ｅ２１，Ｅ３１，Ｅ２２，Ｅ３２を保存する場合に比して、８０％の削減効果が得られる。

【００６８】

なお、決定部４０５は、判定部４０４によって依存関係なしと判定された場合、依存関係なしと判定されたイベント群を保存対象イベントに決定することとなる。たとえば、図１０の（Ｂ）では、イベントＥ１，Ｅ２とは依存関係なしと判定されたため、イベントＥ１，Ｅ２を保存対象イベントに決定することとなる。

【００６９】

また、算出部４０６は、組み合わせの総数と第１のイベントおよび第２のイベントが抽出された組み合わせの数に基づいて、保存対象イベントに関する信頼度を算出する機能を有する。ここで、信頼度とは、判定部４０４による依存関係ありと判定された判定結果の信頼性を評価する指標値である。たとえば、組み合わせの総数を分母とし、第１のイベントおよび第２のイベントが抽出された組み合わせの数を分子とした値を信頼度とする。

【００７０】

たとえば、図１０の（Ａ）の場合は、組み合わせは｛ＶＭ＃１，Ｘ＿Ｗｅｂ｝の１個であるため、組み合わせの総数は１である。また、ＶＭ＃１で発生したイベントＥ１およびＸ＿Ｗｅｂで発生したイベントＥ２が抽出されるため、第１のイベントおよび第２のイベントが抽出された組み合わせの数は１である。したがって、信頼度は１／１となる。同様に、図１１の場合も、信頼度は４／４である。

【００７１】

また、決定部４０５は、算出部４０６によって算出された信頼度に基づいて、保存対象イベントを決定することとしてもよい。たとえば、しきい値となる所定信頼度Ｐを設定しておく。所定信頼度Ｐは管理者が自由に設定することができる。

【００７２】

そして、算出部４０６で算出された信頼度が所定信頼度Ｐ以上である場合は、判定部４０４で依存関係ありと判定されたイベント群のうち依存先管理対象にならない依存元管理対象で発生したイベント（障害の基点となるイベント）を保存対象イベントに決定する。一方、算出部４０６で算出された信頼度が所定信頼度Ｐ未満である場合は、判定部４０４

で依存関係ありと判定されたイベント群を保存対象イベントに決定する。
【００７３】
たとえば、所定信頼度ＰをＰ＝７０％とした場合、図１０の（Ａ）の例の信頼度１／１は、所定信頼度Ｐ以上となるため、イベントＥ１が保存対象イベントに決定される。また、図１１の例の信頼度４／４は、所定信頼度Ｐ以上となるため、イベントＥ１が保存対象イベントに決定される。
【００７４】
図４に戻って、保存部４０７は、決定部４０５によって決定された保存対象イベントに関する情報をＤＢ４０８に保存する機能を有する。具体的には、たとえば、保存対象イベントに記述されている番号、タイムスタンプ、イベント種類、発生箇所、警報種類、予備といった情報をレコードとして統合管理ＤＢ１０３に保存する。
【００７５】
図１２は、統合管理ＤＢ１０３の記憶内容の一例を示す説明図である。なお、保存部４０７は、保存対象イベントに記述されている情報をすべて保存することとしてもよいが、少なくとも番号と発生箇所が保存されていればよい。番号と発生箇所が保存されていれば、ＤＢ１２４～１２６から検索可能である。
【００７６】
また、保存部４０７は、算出部４０６によって算出された信頼度も保存することとしてもよい。この場合、信頼度は、統合管理ＤＢ１０３の予備項目２０６に保存することができる。
【実施例１】
【００７７】
（構成の説明）
図１３は、実施例１にかかる管理装置の説明図である。図１３に示した管理装置１０は、図１に示した統合管理機能１２７の一部分であり、この例ではＣＰＵ＃２、ＶＭ（Ｖｉｒｔｕａｌ　Ｍａｃｈｉｎｅ：仮想計算機）＃３，６、業務Ｘ＿ＤＢ、業務Ｙ＿ＤＢを管理しているものとする。
【００７８】
ＣＰＵ＃２は、管理装置１０にとって第１の管理対象である。また、ＶＭ＃３，６は管理装置１０にとって第２の管理対象であり、業務Ｘ＿ＤＢと業務Ｙ＿ＤＢは管理装置１０にとって第３の管理対象である。
【００７９】
ＣＰＵ＃２とＶＭ＃３，６との間にはそれぞれ依存関係がある。この依存関係において、ＣＰＵ＃２は依存元であり、ＶＭ＃３，６は依存先である。すなわち、ＣＰＵ＃２に異常が発生すると、ＶＭ＃３，６にも異常が発生する場合がある。
【００８０】
また、ＶＭ＃３と業務Ｘ＿ＤＢとの間には依存関係がある。この依存関係において、ＶＭ＃３は依存元であり、業務Ｘ＿ＤＢは依存先である。すなわち、ＶＭ＃３に異常が発生すると、業務Ｘ＿ＤＢにも異常が発生する場合がある。
【００８１】
同様に、ＶＭ＃６と業務Ｙ＿ＤＢとの間には依存関係がある。この依存関係において、ＶＭ＃６は依存元であり、業務Ｙ＿ＤＢは依存先である。すなわち、ＶＭ＃６に異常が発生すると、業務Ｙ＿ＤＢにも異常が発生する場合がある。
【００８２】
したがって、ＣＰＵ＃２とＶＭ＃３，６との依存関係、ＶＭ＃３と業務Ｘ＿ＤＢとの依存関係、ＶＭ＃６と業務Ｙ＿ＤＢとの依存関係によって、ＣＰＵ＃２の異常が起点となってＶＭ＃３，６、業務Ｘ＿ＤＢ、業務Ｙ＿ＤＢに異常が発生することが考えられる。
【００８３】
依存元の管理対象で発生したイベントと依存先の管理対象で発生したイベントをそれぞれ検知し、イベントの発生時刻の差が所定時間以内であれば、管理対象の依存関係によっ

て引き起こされた依存関係のあるイベント群であると判定することができる。このようなイベント間の依存関係は、イベントの管理に利用できる。一例として、依存元のイベントは依存先のイベントよりも重要度が高いとし、起点のイベントを選択的に収集、保存する場合がある。

【００８４】

このようにイベントの依存関係を知ることは重要であるので、イベントの検知に漏れがあった場合にもイベントの依存関係を判定することは有用である。

【００８５】

そこで、開示の管理装置１０は、管理対象からイベントを取得する取得部１１に加え、途中イベント抜けの推定部１２および起点イベント抜けの推定部１３を有する。

【００８６】

途中イベント抜けの推定部１２は、差分算出部１４と判定部１６を有する。差分算出部１４は、第１の管理対象であるＣＰＵ＃２におけるイベントの発生時刻と第３の管理対象である業務Ｘ＿ＤＢ，業務Ｙ＿ＤＢにおけるイベントの発生時刻との差分を求める。判定部１６は、差分算出部１４が算出した差分が所定時間以内である場合に、業務Ｘ＿ＤＢ，業務Ｙ＿ＤＢにおけるイベントがＣＰＵ＃２のイベントに基づくと判定する。

【００８７】

また、起点イベント抜けの推定部１３は、差分算出部１５と判定部１７を有する。差分算出部１５は、複数の第２の管理対象であるＶＭ＃３，６でそれぞれ発生したイベントについて、発生時刻の差分を求める。判定部１７は、差分算出部１５が算出した差分が所定時間以内である場合にＶＭ＃３，６で発生したイベントがＣＰＵ＃２のイベントに基づくと判定する。

【００８８】

（途中イベント抜けの具体例）

途中イベント抜けの推定部１２の動作について具体例を挙げて説明する。図１４は、途中イベント抜けの具体例の説明図（その１）であり、図１５は、途中イベント抜けの具体例の説明図（その２）である。図１４，図１５では、ＣＰＵ＃２、ＶＭ＃３、業務Ｘ＿ＤＢのルートをＡルート、ＣＰＵ＃２、ＶＭ＃６、業務Ｙ＿ＤＢのルートをＢルートとする。

【００８９】

図１４に示した例では、ＣＰＵ＃２から時刻Ｔ１に発生したイベントＥ１の通知があり、業務Ｘ＿ＤＢから時刻Ｔ３１に発生したイベントＥ３１の通知があがっている。また、ＶＭ＃６から時刻Ｔ２２に発生したイベントＥ２２の通知があがり、業務Ｙ＿ＤＢから時刻Ｔ３２に発生したイベントＥ３２の通知があがっている。しかし、ＶＭ＃３からはイベントの通知があがっていない。

【００９０】

Ｂルートでは、時刻Ｔ１と時刻Ｔ２２との差分が閾値Ｔｓ１以下であることから、イベントＥ２２がイベントＥ１に依存していると判定できる。また、時刻Ｔ２２と時刻Ｔ３２との差分が閾値Ｔｓ２以下であることからイベントＥ３２がイベントＥ２２に依存していると判定できる。

【００９１】

しかし、Ａルートでは、ＶＭ＃３からイベントの通知がないため、ＶＭ＃３のイベント通知を利用した障害の依存関係の判定ができない。

【００９２】

これに対し、途中イベント抜けの推定部１２は、図１５に示すように第１の管理対象であるＣＰＵ＃２のイベント発生時刻と第３の管理対象である業務Ｘ＿ＤＢのイベント発生時刻から障害の依存関係を判定する閾値Ｔｓ３を用いて判定を行う。すなわち、途中イベント抜けの推定部１３は、時刻Ｔ１と時刻Ｔ３１との差分が閾値Ｔｓ３以下であれば、途中のＶＭ＃３からイベントの通知が無くともイベントＥ３１がイベントＥ１に依存していると判定できる。

【００９３】
（起点イベント抜けの具体例）
起点イベント抜けの推定部１３の動作について具体例を挙げて説明する。図１６は、起点イベント抜けの具体例の説明図（その１）であり、図１７は、起点イベント抜けの具体例の説明図（その２）である。図１６，図１７では、ＣＰＵ＃２、ＶＭ＃３、業務Ｘ＿ＤＢのルートをＡルート、ＣＰＵ＃２、ＶＭ＃６、業務Ｙ＿ＤＢのルートをＢルートとする。
【００９４】
図１６に示した例では、ＶＭ＃３から時刻Ｔ２１に発生したイベントＥ２１の通知があがり、業務Ｘ＿ＤＢから時刻Ｔ３１に発生したイベントＥ３１の通知があがっている。また、ＶＭ＃６から時刻Ｔ２２に発生したイベントＥ２２の通知があがり、業務Ｙ＿ＤＢから時刻Ｔ３２に発生したイベントＥ３２の通知があがっている。しかし、ＣＰＵ＃２からはイベントの通知があがっていない。
【００９５】
Ａルートでは、時刻Ｔ２１と時刻Ｔ３１との差分が閾値Ｔｓ２以下であることからイベントＥ３１がイベントＥ２１に依存していると判定できる。また、Ｂルートでは、時刻Ｔ２２と時刻Ｔ３２との差分が閾値Ｔｓ２以下であることからイベントＥ３２がイベントＥ２２に依存していると判定できる。
【００９６】
しかし、ＣＰＵ＃２からのイベント通知がないため、ＶＭ＃３，６がイベントの起点であるように見える。
【００９７】
これに対し、起点イベント抜けの推定部１３は、第２の管理対象であるＶＭ＃３，６のイベント発生時刻から障害の依存関係を判定する閾値Ｔｓ４を用いて判定を行う。すなわち、起点イベント抜けの推定部１３は、時刻Ｔ２１と時刻Ｔ２２との差分が閾値Ｔｓ４以下であれば、起点のＣＰＵ＃２からイベントの通知が無くともイベントＥ２１，２２がＣＰＵ＃２のイベントに依存していると判定できる。
【００９８】
起点イベント抜けの推定部１３による判定には、第３の管理対象からのイベントをさらに用いてもよい。具体的には、図１７の例では、Ａルートにおいて、時刻Ｔ２１と時刻Ｔ３１との差分が閾値Ｔｓ２以下であることからイベントＥ３１はイベントＥ２１に依存している。また、Ｂルートにおいて、時刻Ｔ２２と時刻Ｔ３２との差分が閾値Ｔｓ２以下であることからイベントＥ３２はイベントＥ２２に依存している。このように、ＣＰＵ＃２を起点とする２つのルートで共に第２の管理対象のイベントと第３の管理対象のイベントに依存関係があるため、ＣＰＵ＃２がイベントの起点であると判定する。
【００９９】
複数の第２のイベントが第１のイベントに基づくと判定した場合、第１のイベントの発生時刻の値について判定部１７は、ダミーの値を作成する。具体的には、第２のイベントの発生時刻から所定時間を減算した値を第１のイベントの発生時刻とすることができる。第２のイベントの発生時刻から減算する時間は、任意の値を用いることができる。一例としてＴｓ１を用いてもよい。
【０１００】
以上説明してきたように、本実施例１では、管理装置１０は、第１のイベントの発生時刻と第３のイベントの発生時刻との差分を求め、差分が閾値Ｔｓ３以内である場合に第３のイベントが第１のイベントに基づくと判定する。また、管理装置１０は、複数の第２の管理対象で発生した複数の第２のイベントの発生時刻の差分が閾値Ｔｓ４以内である場合に複数の第２のイベントが第１のイベントに基づくと判定する。このため、本実施例に開示した管理装置１０は、イベントの検知に漏れがある場合であってもイベントの依存関係を判定することができる。
【０１０１】

（情報管理処理手順）
つぎに、図４に示した情報管理装置４００による管理処理手順について説明する。
【０１０２】
図１８は、本実施の形態にかかる情報管理装置４００による情報管理処理手順を示すフローチャートである。まず、情報管理装置４００は、初期設定として対象期間を指定し（ステップＳ１８０１）、対象期間内で開始区間となる対象区間を設定する（ステップＳ１８０２）。そして、情報管理装置４００は、対象区間内にイベントがあるか否かをＤＢ１２４～１２６を参照することで判断する（ステップＳ１８０３）。
【０１０３】
対象区間内にイベントがある場合（ステップＳ１８０３，Ｙｅｓ）、情報管理装置４００は、取得部４０１により、対象区間内のイベントをＤＢ１２４～１２６から取得する（ステップＳ１８０４）。そして、特定部４０２により、取得イベントに該当する依存関係情報を特定する（ステップＳ１８０５）。
【０１０４】
つぎに、判定部４０４による依存関係判定処理（ステップＳ１８０６）および決定部４０５による障害発生基点の判定処理（ステップＳ１８０７）を実行する。そして、障害発生基点の判定処理（ステップＳ１８０７）で判定された基点のイベントを保存対象イベントとしてＤＢ４０８（統合管理ＤＢ１０３）に保存する（ステップＳ１８０８）。
【０１０５】
このあと、対象期間が終了したか否かを判断する（ステップＳ１８０９）。対象期間が終了していない場合（ステップＳ１８０９，Ｎｏ）、対象区間をシフトして（ステップＳ１８１０）、次区間を対象区間とし、ステップＳ１８０３に戻る。現区間と次区間との間でイベントが通知される場合もあるため、次区間は、現区間と一部重複して設定することとしてもよい。
【０１０６】
また、ステップＳ１８０３において、対象区間内にイベントがない場合（ステップＳ１８０３，Ｎｏ）、ステップＳ１８０９に移行する。また、ステップＳ１８０９において、対象期間が終了した場合（ステップＳ１８０９，Ｙｅｓ）、一連の管理処理を終了する。
【０１０７】
図１９は、図１８に示した依存関係判定処理（ステップＳ１８０６）の詳細な処理手順を示すフローチャートである。まず、情報管理装置４００は、ステップＳ１８０５において特定された依存関係情報の中に、依存関係判定が未処理のルートがあるか否かを判断する（ステップＳ１９０１）。未処理のルートがない場合（ステップＳ１９０１，Ｎｏ）、障害発生基点の判定処理（ステップＳ１８０７）に移行する。
【０１０８】
一方、未処理のルートがある場合（ステップＳ１９０１，Ｙｅｓ）、情報管理装置４００は、未処理のルートを選択する（ステップＳ１９０２）。たとえば、図１１の依存関係情報７００の場合、｛ＣＰＵ＃２→ＶＭ＃３→Ｘ＿ＤＢ｝、｛ＣＰＵ＃２→ＶＭ＃６→Ｙ＿ＤＢ｝の２本のルートから未処理のルートを選択することとなる。
【０１０９】
そして、情報管理装置４００は、選択ルートの中に未処理の連結ノードの組み合わせがあるか否かを判断する（ステップＳ１９０３）。連結ノードの組み合わせとは、依存関係がある管理対象群で発生したイベント群の中から、依存元管理対象で発生した第１のイベントと依存元管理対象に依存する依存先管理対象で発生した第２のイベントとの組み合わせである。すなわち、リンクによって連結しあうノードの組み合わせである。未処理の連結ノードの組み合わせがない場合（ステップＳ１９０３，Ｎｏ）、ステップＳ１９０１に移行する。
【０１１０】
一方、未処理の連結ノードの組み合わせがある場合（ステップＳ１９０３，Ｙｅｓ）、情報管理装置４００は、未処理の連結ノードの組み合わせを選択する（ステップＳ１９０

４）。たとえば、図６の依存関係情報６００の場合、８個の組み合わせ｛ＣＰＵ＃１，ＶＭ＃１｝，｛ＶＭ＃１，Ｘ＿Ｗｅｂ｝，｛ＣＰＵ＃１，ＶＭ＃２｝，｛ＶＭ＃２，Ｘ＿ＡＰ｝，｛ＣＰＵ＃１，ＶＭ＃４｝，｛ＶＭ＃４，Ｙ＿Ｗｅｂ｝，｛ＣＰＵ＃１，ＶＭ＃５｝，｛ＶＭ＃５，Ｙ＿ＡＰ｝の中から未処理の連結ノードの組み合わせを選択することとなる。

【０１１１】

つぎに、情報管理装置４００は、選択組み合わせの総数を計数するカウンタＣａ（初期値はＣａ＝０）をインクリメントする（ステップＳ１９０５）。そして、情報管理装置４００は、選択された連結ノードの組み合わせにおいて、イベントが不足しているか否かを判断する（ステップＳ１９０６）。イベントが不足していない場合（ステップＳ１９０６，Ｎｏ）、情報管理装置４００は、選択された連結ノードの組み合わせ内の各管理対象からのイベントのタイムスタンプを読み出して、差分を算出する（ステップＳ１９０７）。

【０１１２】

そして、情報管理装置４００は、差分がしきい値Ｔｓ１あるいはＴｓ２以内であるか否かを判断し（ステップＳ１９０８）、しきい値Ｔｓ１あるいはＴｓ２以内である場合（ステップＳ１９０８，Ｙｅｓ）、依存関係が成立したこととなり、ステップＳ１９０３に戻る。一方、しきい値Ｔｓ１あるいはＴｓ２以内でない場合（ステップＳ１９０８，Ｎｏ）、依存関係が不成立となり、情報管理装置４００は、依存関係の不成立数を計数するカウンタＣｃ（初期値はＣｃ＝０）をインクリメントする（ステップＳ１９０９）。そして、ステップＳ１９０３に戻る。

【０１１３】

一方、ステップＳ１９０６において、イベント不足であると判断された場合（ステップＳ１９０６，Ｙｅｓ）、情報管理装置４００は、イベントの不足が１つであるかを判定する（ステップＳ１９１０）。

【０１１４】

この結果、イベントの不足数が１つではない場合（ステップＳ１９１０，Ｎｏ）、情報管理装置４００は、イベントが２つとも欠落するケースの数を示すカウンタＣｄ（初期値はＣｄ＝０）をインクリメントし（ステップＳ１９１７）、ステップＳ１９０３に戻る。

【０１１５】

一方、ステップＳ１９１０において、イベントの不足数が１つであると判定した場合（ステップＳ１９１０，Ｙｅｓ）、情報管理装置４００は、イベント不足の連結ノード数を計数するカウンタＣｂ（初期値はＣｂ＝０）をインクリメントする（ステップＳ１９１１）。

【０１１６】

ステップＳ１９１１の後、情報管理装置４００は、２つのイベントの組み合わせが揃ったかを判定し（ステップＳ１９１２）、揃っていなければ（ステップＳ１９１２，Ｎｏ）ステップＳ１９０３に戻る。

【０１１７】

一方、２つのイベントの組み合わせが揃った場合（ステップＳ１９１２，Ｙｅｓ）、情報管理装置４００は、起点イベント抜け判定用カウンタＭｄ（初期値はＭｄ＝０）をインクリメント（ステップＳ１９１３）する。その後、イベントのタイムスタンプを読み出して、差分を算出する（ステップＳ１９１４）。

【０１１８】

そして、情報管理装置４００は、差分がしきい値Ｔｓ３以内であるか否かを判断し（ステップＳ１９１５）、しきい値Ｔｓ３以内である場合（ステップＳ１９１５，Ｙｅｓ）、ステップＳ１９０３に戻る。一方、しきい値Ｔｓ３以内でない場合（ステップＳ１９１５，Ｎｏ）、依存関係が不成立となり、情報管理装置４００は、カウンタＣｃ（初期値はＣｃ＝０）に２を加え（ステップＳ１９１６）、ステップＳ１９０３に戻る。

【０１１９】

図２０は、図１８に示した障害発生起点の判定処理（ステップＳ１８０７）の詳細な処

理手順を示すフローチャートである。まず、情報管理装置４００は、カウンタＭｄが正の値であるかを判定する（ステップＳ２００１）。

【０１２０】

カウンタＭｄが正の値でない場合（ステップＳ２００１，Ｎｏ）、情報管理装置４００は、途中イベント抜け判定処理（ステップＳ２００２）を行って保存処理（ステップＳ１８０８）に移行する。一方、カウンタＭｄが正の値である場合（ステップＳ２００１，Ｙｅｓ）、情報管理装置４００は、起点イベント抜け判定処理（ステップＳ２００３）を行って保存処理（ステップＳ１８０８）に移行する。

【０１２１】

図２１は、図２０に示した途中イベント抜け判定処理（ステップＳ２００２）の詳細について説明するフローチャートである。情報管理装置４００は、（Ｃａ－Ｃｄ）／ＣａがＰ以上であるかを判定する（ステップＳ２１０１）。ここで、Ｐは、信頼度を示す所定の値であり、任意の値を設定することができる。

【０１２２】

（Ｃａ－Ｃｄ）／ＣａがＰ未満である場合（ステップＳ２１０１，Ｎｏ）、情報管理装置４００は、障害発生の基点が判定不可能であるとして（ステップＳ２１０５）、カウンタをリセットし（ステップＳ２１０６）、保存処理（ステップＳ１８０８）に移行する。

【０１２３】

一方、（Ｃａ－Ｃｄ）／ＣａがＰ以上である場合（ステップＳ２１０１，Ｙｅｓ）、情報管理装置４００は、Ｃａ－ＣｄでＣａを更新し（ステップＳ２１０２）、１－Ｃｃ／Ｃａが１であるかを判定する（ステップＳ２１０３）。

【０１２４】

１－Ｃｃ／Ｃａ＝１である場合（ステップＳ２１０３，Ｙｅｓ）、情報管理装置４００は、最上位のノードが障害発生の起点であると判定して（ステップＳ２１０４）、カウンタをリセットし（ステップＳ２１０６）、保存処理（ステップＳ１８０８）に移行する。

【０１２５】

一方、１－Ｃｃ／Ｃａ＝１でない場合（ステップＳ２１０３，Ｎｏ）、情報管理装置４００は、障害発生の起点が判定不可能であるとして（ステップＳ２１０５）、カウンタをリセットし（ステップＳ２１０６）、保存処理（ステップＳ１８０８）に移行する。

【０１２６】

図２２は、図２０に示した起点イベント抜け判定処理（ステップＳ２００３）の詳細について説明するフローチャートである。情報管理装置４００は、１－Ｃｃ／（Ｃａ－Ｃｄ－Ｃｂ）がＰ以上であるかを判定する（ステップＳ２２０１）。ここで、Ｐは、信頼度を示す所定の値であり、任意の値を設定することができる。

【０１２７】

１－Ｃｃ／（Ｃａ－Ｃｄ－Ｃｂ）がＰ未満である場合（ステップＳ２２０１，Ｎｏ）、情報管理装置４００は、障害発生の起点が判定不可能であるとして（ステップＳ２２０４）、カウンタをリセットし（ステップＳ２２０５）、保存処理（ステップＳ１８０８）に移行する。

【０１２８】

一方、１－Ｃｃ／（Ｃａ－Ｃｄ－Ｃｂ）がＰ以上である場合（ステップＳ２２０１，Ｙｅｓ）、情報管理装置４００は、イベント発生時刻Ｔ２ｎ（ｎは自然数）の最小値と最大値の差がＴｓ４未満であるかを判定する（ステップＳ２２０２）。

【０１２９】

イベント発生時刻Ｔ２ｎの最小値と最大値の差がＴｓ４未満である場合（ステップＳ２２０２，Ｙｅｓ）、情報管理装置４００は、最上位のノードが障害発生の起点であると判定して（ステップＳ２２０３）、カウンタをリセットし（ステップＳ２２０５）、保存処理（ステップＳ１８０８）に移行する。

【０１３０】

一方、イベント発生時刻Ｔ２ｎの最小値と最大値の差がＴｓ４以上である場合（ステッ

プＳ２２０２，Ｎｏ）、情報管理装置４００は、障害発生の起点が判定不可能であるとして（ステップＳ２２０４）、カウンタをリセットし（ステップＳ２２０５）、保存処理（ステップＳ１８０８）に移行する。

【０１３１】

すなわち、ステップＳ２２０２の処理では、全てのイベント（Ｔ２１～Ｔ２ｎ）がＴｓ４未満の時間の間で発生した場合に、最上位のノードが障害発生の起点であると判定する。

【０１３２】

変形例として、イベント発生時刻Ｔ２ｎのそれぞれに対して、すべてのイベント発生時刻Ｔ２ｎの中での最小値を引いた差｜Ｔ２ｎ－Ｔｍｉｎ｜がＴｓ４未満である条件を満たす連結ノードの割合が所定比率Ｒ以上である場合に、最上位のノードが障害発生の起点であると判定することもできる。

【０１３３】

図２３は、起点イベント抜け判定の変形例の説明図である。図２３に示した例では、ＣＰＵ＃１にＶＭ＃１，ＶＭ＃２，ＶＭ＃４，ＶＭ＃５が接続しており、ＶＭ＃１，ＶＭ＃２，ＶＭ＃４，ＶＭ＃５からイベント通知があがっている。また、ＶＭ＃１には業務Ｘ＿Ｗｅｂが接続し、ＶＭ＃２には業務Ｘ＿ＡＰが接続し、ＶＭ＃４には業務Ｙ＿Ｗｅｂが接続し、ＶＭ＃５には業務Ｙ＿ＡＰが接続している。そして、業務Ｘ＿Ｗｅｂ、業務Ｘ＿ＡＰ、業務Ｙ＿Ｗｅｂ、業務Ｙ＿ＡＰからもイベント通知があがっている。

【０１３４】

ＣＰＵ＃１からＶＭ＃１を経由して業務Ｘ＿Ｗｅｂに至るルートがＣルートであり、ＶＭ＃１は時刻Ｔ２１にイベントＥ２１を上げ、業務Ｘ＿Ｗｅｂは時刻Ｔ３１にイベントＥ３１を上げている。

【０１３５】

また、ＣＰＵ＃１からＶＭ＃２を経由して業務Ｘ＿ＡＰに至るルートがＤルートであり、ＶＭ＃２は時刻Ｔ２２にイベントＥ２２を上げ、業務Ｘ＿ＡＰは時刻Ｔ３２にイベントＥ３２を上げている。

【０１３６】

ＣＰＵ＃１からＶＭ＃４を経由して業務Ｙ＿Ｗｅｂに至るルートがＥルートであり、ＶＭ＃４は時刻Ｔ２３にイベントＥ２３を上げ、業務Ｙ＿Ｗｅｂは、時刻Ｔ３３にイベントＥ３３を上げている。

【０１３７】

ＣＰＵ＃１からＶＭ＃５を経由して業務Ｙ＿ＡＰに至るルートがＦルートであり、ＶＭ＃５は時刻Ｔ２４にイベントＥ２４を上げ、業務Ｙ＿ＡＰは、時刻Ｔ３４にイベントＥ３４を上げている。

【０１３８】

イベントＥ２１～Ｅ２４の発生時刻Ｔ２１～２４の最小値、すなわち最も早くイベントがあがった時刻がＴ２１である場合、情報管理装置４００は、各イベント発生時刻からＴ２１を引いた値がＴｓ４未満であるかを判定する。したがって、図２３の例では、Ｔ２１－Ｔ２１，Ｔ２２－Ｔ２１，Ｔ２３－Ｔ２１，Ｔ２４－Ｔ２１について、Ｔｓ４未満であるかを判定する。

【０１３９】

例えば、Ｔ２１－Ｔ２１，Ｔ２２－Ｔ２１，Ｔ２４－Ｔ２１がＴｓ４未満、Ｔ２３－Ｔ２１がＴｓ４以上であり、Ｒが０．７０である場合、４つのルートＣ～Ｆのうち、ルートＣ，Ｄ，Ｆの３ルートがＴｓ４未満を満たすので、３／４＝０．７５＞Ｒとなり、ＣＰＵ＃１が障害の起点であると判定することができる。

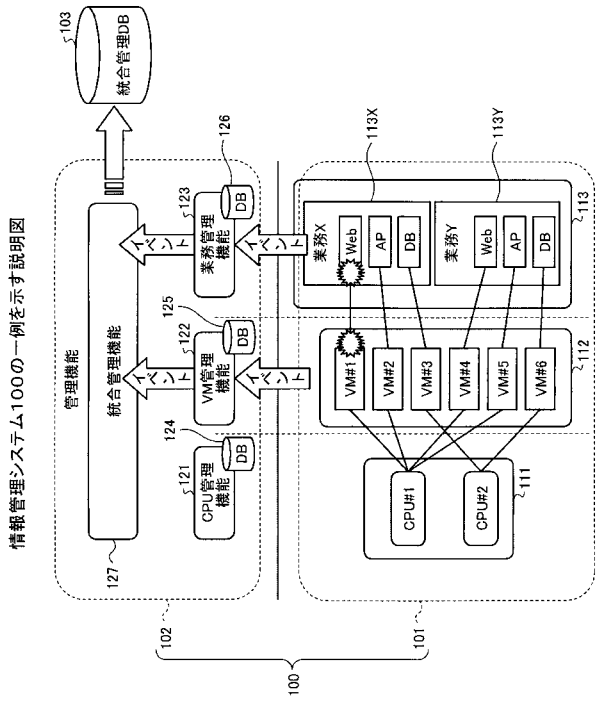
【０１４０】

図２４は、起点イベント抜け判定の変形例のフローチャートである。情報管理装置４００は、１－Ｃｃ／（Ｃａ－Ｃｄ－Ｃｂ）がＰ以上であるかを判定する（ステップＳ２４０１）。ここで、Ｐは、信頼度を示す所定の値であり、任意の値を設定することができる。

【０１４１】
１－Ｃｃ／（Ｃａ－Ｃｄ－Ｃｂ）がＰ未満である場合（ステップＳ２４０１，Ｎｏ）、情報管理装置４００は、障害発生の基点が判定不可能であるとして（ステップＳ２４１０）、カウンタをリセットし（ステップＳ２４２２）、保存処理（ステップＳ１８０８）に移行する。
【０１４２】
一方、１－Ｃｃ／（Ｃａ－Ｃｄ－Ｃｂ）がＰ以上である場合（ステップＳ２４０１，Ｙｅｓ）、情報管理装置４００は、イベント発生時刻Ｔ２１～Ｔ２ｎ（ｎは自然数）の最小値をＴｍｉｎとする（ステップＳ２４０２）。
【０１４３】
つぎに、情報管理装置４００は、変数ｉを１とする（ステップＳ２４０３）。そして、Ｔ２ｉ－Ｔｍｉｎ＜Ｔｓ４であるかを判定する（ステップＳ２４０４）。Ｔ２ｉ－Ｔｍｉｎ＜Ｔｓ４が成立すれば（ステップＳ２４０４，Ｙｅｓ）、情報管理装置４００は、カウンタＣｅをインクリメントする（ステップＳ２４０５，Ｙｅｓ）。カウンタＣｅの初期値は０である。ただしＴ２ｉは、Ｓ２４０１の条件を満たした連結ノードで発生した第１のイベントの発生時刻のみを対象とする。
【０１４４】
ステップＳ２４０５の後、または、Ｔ２ｉ－Ｔｍｉｎ＜Ｔｓ４が成立しない場合（ステップＳ２４０４，Ｎｏ）、情報管理装置４００は、ｉ＝αであるかを判定する（ステップＳ２４０６）。ここで、α＝Ｃａ－Ｃｂ－Ｃｃ－Ｃｄとする。
【０１４５】
ｉ＝αでなければ（ステップＳ２４０６，Ｎｏ）、情報管理装置４００は、ｉをインクリメントし（ステップＳ２４０７）、ステップＳ２４０４に戻る。ｉ＝αである場合（Ｓ２４０６，Ｙｅｓ）、情報管理装置４００は、Ｃｅ／αが所定比率Ｒ以上であるかを判定する（Ｓ２４０８）。
【０１４６】
Ｃｅ／αが所定比率Ｒ以上である場合（ステップＳ２４０８，Ｙｅｓ）、情報管理装置４００は、最上位のノードが障害発生の起点であると判定して（ステップＳ２４０９）、カウンタをリセットし（ステップＳ２４１１）、保存処理（ステップＳ１８０８）に移行する。
【０１４７】
一方、Ｃｅ／αが所定比率Ｒ未満である場合（ステップＳ２４０８，Ｎｏ）、情報管理装置４００は、障害発生の起点が判定不可能であるとして（ステップＳ２４１０）、カウンタをリセットし（ステップＳ２４１１）、保存処理（ステップＳ１８０８）に移行する。
【０１４８】
以上説明してきたように、本実施例では、情報管理装置４００は、第１のイベントの発生時刻と第３のイベントの発生時刻との差分から第３のイベントが第１のイベントに基づくと判定することができる。また、情報管理装置４００は、複数の第２の管理対象で発生した複数の第２のイベントの発生時刻の差分から複数の第２のイベントが第１のイベントに基づくと判定することができる。このため、情報管理装置４００は、イベントの検知に漏れがある場合であってもイベントの依存関係を判定することができる。
【０１４９】
加えて、情報管理装置４００は、障害の起点となるイベントを保存することができるため、重要なイベントを選択的に保存可能である。
【０１５０】
障害の基点となるイベントが保存できていれば、そのイベントが持つ情報をキーにして、依存関係情報を参照して依存関係が伝搬する管理対象からのイベントを、ＤＢ１２４～１２６から検索することができる。したがって、保存データ量の削減とイベント検索の効率化を図ることができる。また、障害の起点となるイベントがわかれば、当該イベントを

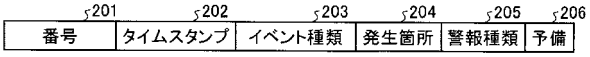
発生した管理対象を容易に特定できるため、メンテナンスの容易化も図ることができる。
【０１５１】
さらに、保存対象イベントとともに信頼度を保存することで、管理者がデータベース（統合管理ＤＢ１０３）を参照する際に、信頼度に応じて、ＤＢ１２４～１２６を検索するかしないかの判断指標とすることができる。
【０１５２】
また、本実施の形態では、障害イベントや監視イベントを通知するものであれば管理対象とすることができる。たとえば、クラウドコンピューティングにおいて、ネットワーク構成またはサーバ、クライアント、さらにその中間に存在する論理レイヤを示した管理対象として適用することができる。
【０１５３】
この場合、たとえば、クラウドコンピューティング環境で利用されるサーバやクライアント、それらをつなぐネットワークなどを監視するシステムにおいて、膨大なイベントをログとして保存しなければならないストレージを装備するシステムに有効である。
【０１５４】
なお、本実施の形態で説明した管理方法は、予め用意されたプログラムをパーソナル・コンピュータやワークステーション等のコンピュータで実行することにより実現することができる。本情報管理プログラムは、ハードディスク、フレキシブルディスク、ＣＤ－ＲＯＭ、ＭＯ、ＤＶＤ等のコンピュータで読み取り可能な記録媒体に記録され、コンピュータによって記録媒体から読み出されることによって実行される。また本情報管理プログラムは、インターネット等のネットワークを介して配布してもよい。
【符号の説明】
【０１５５】
１０　管理装置
１１　取得部
１２　途中イベント抜けの推定部
１３　起点イベント抜けの推定部
１４，１５　差分算出部
１６，１７　判定部
１００　情報管理システム
１０１　管理対象装置
１０２　管理機能
１１３　業務プロセス
１２７　統合管理機能
４００　情報管理装置
４０１　取得部
４０２　特定部
４０３　抽出部
４０４　判定部
４０５　決定部
４０６　算出部
４０７　保存部
５００　プロセス割当テーブル
６００，７００，８０１～８０６，９０１～９０６　依存関係情報

【図１】



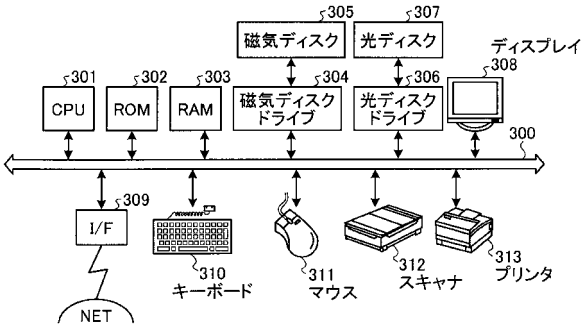
【図２】

管理対象から発生するイベントのデータ構造の一例を示す説明図



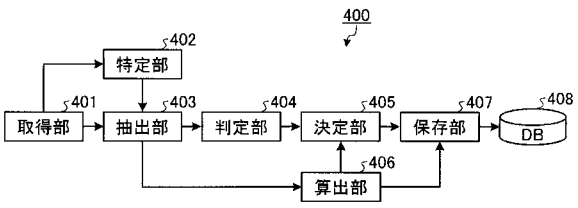
【図３】

実施の形態で用いられるコンピュータのハードウェア構成を示すブロック図



【図４】

情報管理装置の機能的構成を示すブロック図



【図５】

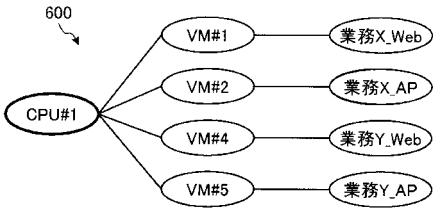
プロセス割当テーブルの記憶内容の一例を示す説明図

500

501 番号	502 管理対象		
	CPU	VM	業務プロセス
1	CPU#1	VM#1	業務X_Web
2	CPU#1	VM#2	業務X_AP
3	CPU#2	VM#3	業務X_DB
4	CPU#1	VM#4	業務Y_Web
5	CPU#1	VM#5	業務Y_AP
6	CPU#2	VM#6	業務Y_DB
…	…	…	…

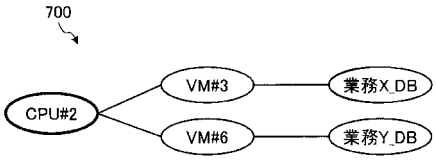
【図６】

CPU＃１を障害の基点とした場合の依存関係情報を示す説明図



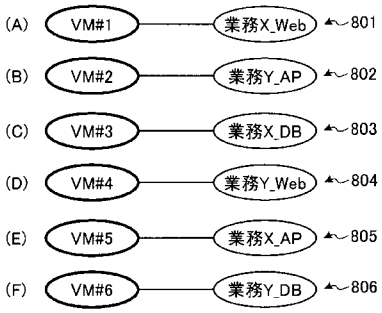
【図７】

CPU＃２を障害の基点とした場合の依存関係情報を示す説明図



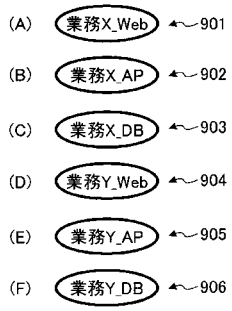
【図８】

VMを障害の基点とした場合の依存関係情報を示す説明図



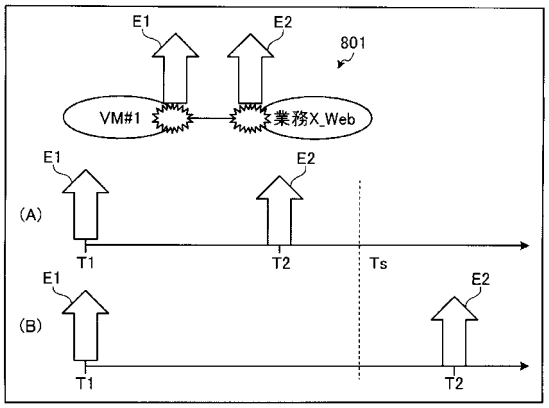
【図９】

業務プロセスを障害の基点とした場合の依存関係情報を示す説明図



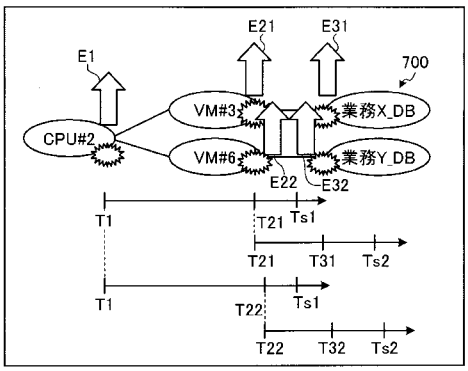
【図１０】

判定部による判定処理の具体例(その1)を示す説明図



【図１１】

判定部による判定処理の具体例(その2)を示す説明図



【図１２】

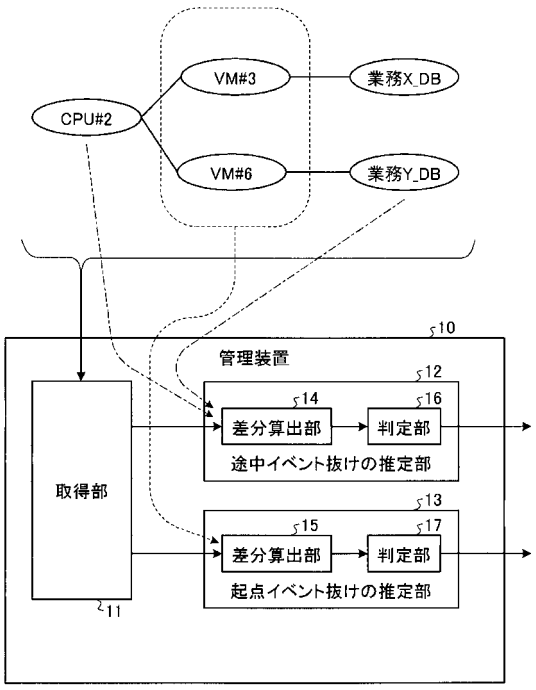
統合管理DBの記憶内容の一例を示す説明図

103

番号	タイムスタンプ	イベント種類	発生箇所	警報種類	予備
0005	2009/12/01/17:26	0	CPU#2	1003	…
0029	2009/12/02/09:02	1	VM#1	2010	…
⋮	⋮	⋮	⋮	⋮	

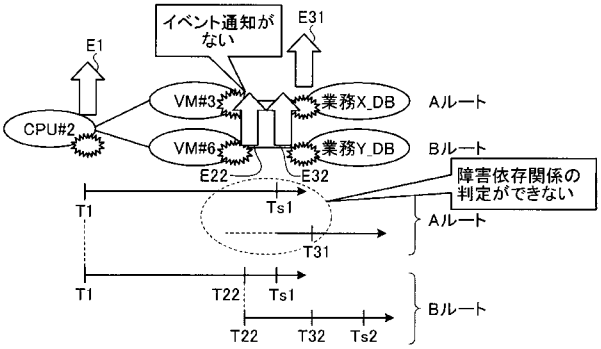
【図１３】

実施例1にかかる管理装置の説明図



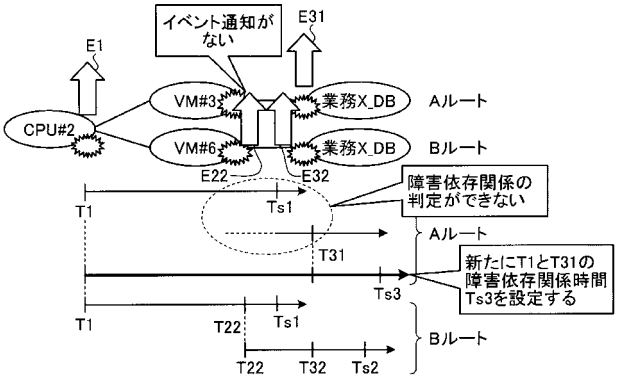
【図１４】

途中イベント抜けの具体例の説明図（その1）



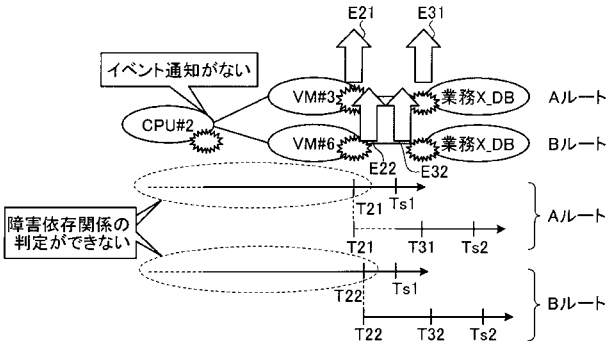
【図１５】

途中イベント抜けの具体例の説明図（その2）



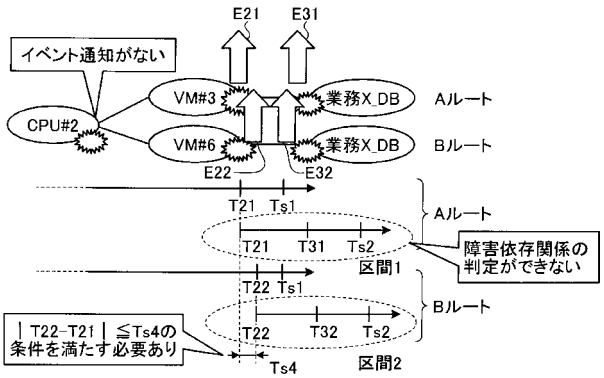
【図１６】

起点イベント抜けの具体例の説明図（その1）



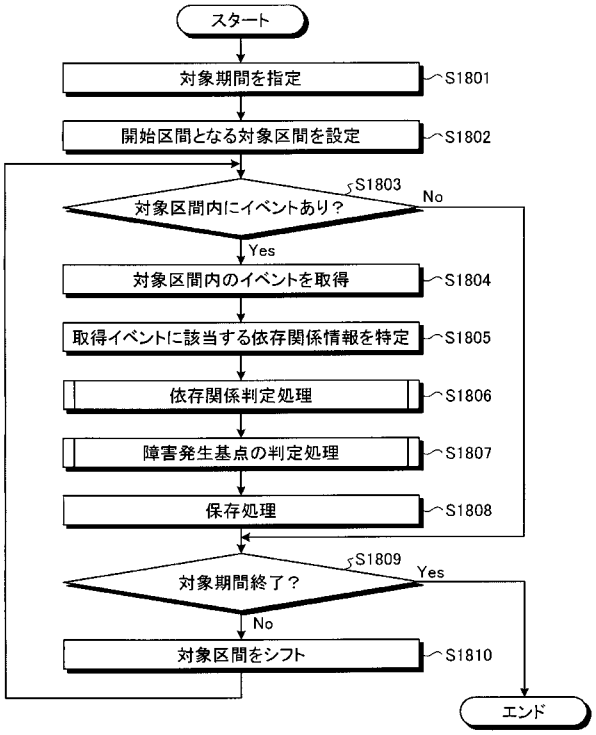
【図１７】

起点イベント抜けの具体例の説明図（その2）



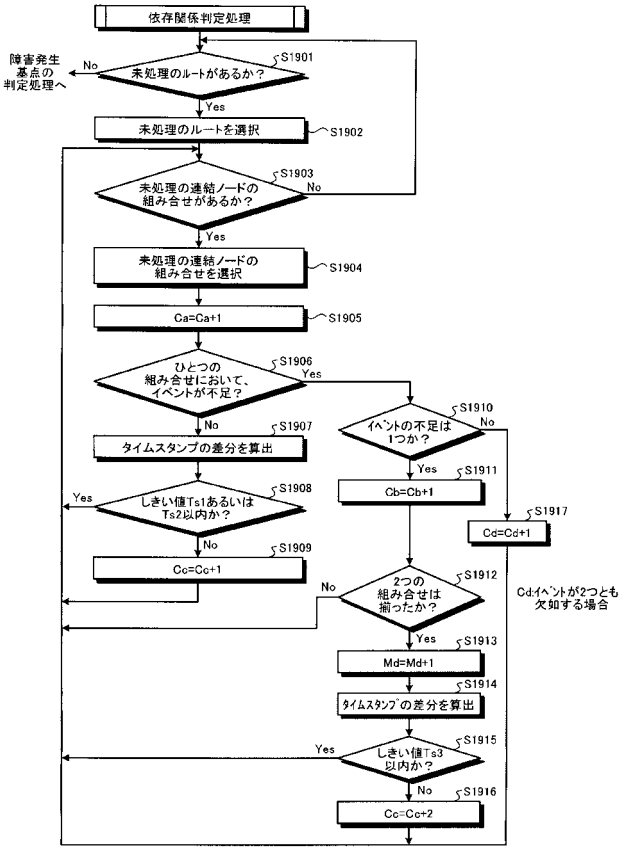
【図１８】

本実施の形態にかかる情報管理装置による
情報管理処理手順を示すフローチャート



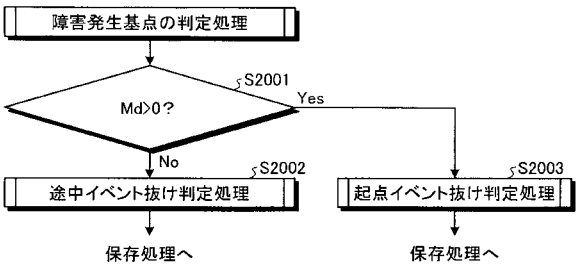
【図１９】

図18に示した依存関係判定処理の詳細な処理手順を示すフローチャート



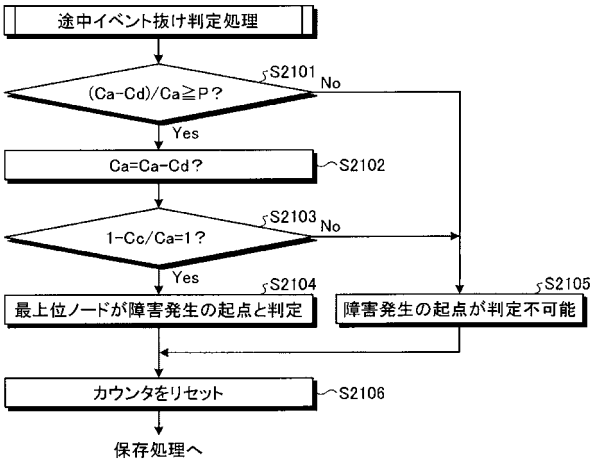
【図２０】

図18に示した障害発生起点の判定処理の詳細な処理手順を示すフローチャート



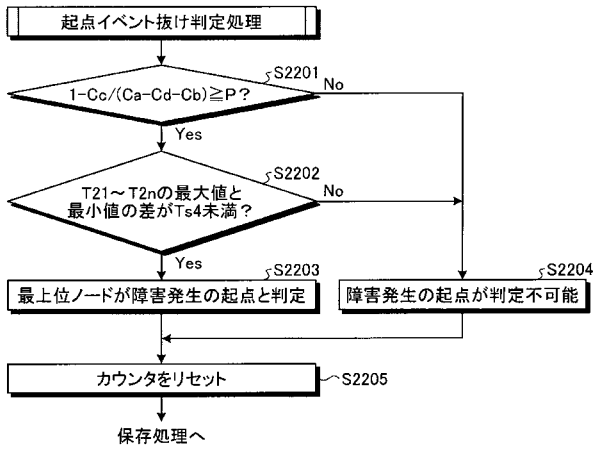
【図２１】

図20に示した途中イベント抜け判定処理の
詳細について説明するフローチャート



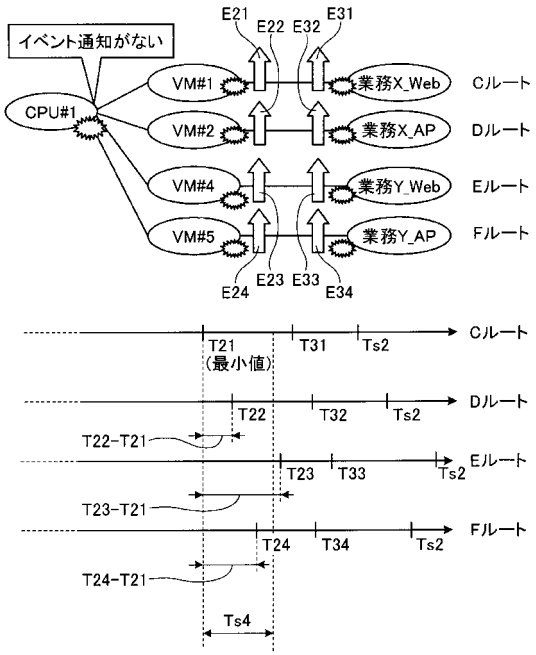
【図２２】

図20に示した起点イベント抜け判定処理の詳細について説明するフローチャート



【図２３】

起点イベント抜け判定の変形例の説明図



【図２４】

起点イベント抜け判定の変形例のフローチャート