

(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号

特許第7170638号

(P7170638)

(45)発行日 令和4年11月14日(2022.11.14)

(24)登録日 令和4年11月4日(2022.11.4)

(51)国際特許分類

F I

G 0 6 F 16/38 (2019.01)

G 0 6 F 16/38

請求項の数 14 (全23頁)

(21)出願番号	特願2019-525760(P2019-525760)	(73)特許権者	509123208
(86)(22)出願日	平成29年12月1日(2017.12.1)		アビニシオ テクノロジー エルエルシー
(65)公表番号	特表2020-501235(P2020-501235 A)		アメリカ合衆国 0 2 4 2 1 マサチュー
(43)公表日	令和2年1月16日(2020.1.16)		セッツ州 レキシントン スプリング ス
(86)国際出願番号	PCT/US2017/064227	(74)代理人	トリート 2 0 1
(87)国際公開番号	WO2018/102691		100079108
(87)国際公開日	平成30年6月7日(2018.6.7)	(74)代理人	弁理士 稲葉 良幸
審査請求日	令和1年7月10日(2019.7.10)	(74)代理人	100109346
審判番号	不服2021-16157(P2021-16157/J 1)	(74)代理人	弁理士 大貫 敏史
審判請求日	令和3年11月25日(2021.11.25)	(74)代理人	100117189
(31)優先権主張番号	62/428,860	(74)代理人	弁理士 江口 昭彦
(32)優先日	平成28年12月1日(2016.12.1)	(74)代理人	100134120
(33)優先権主張国・地域又は機関		(72)発明者	弁理士 内藤 和彦
	最終頁に続く		クレメンス、デイビッド
			アメリカ合衆国、マサチューセッツ州
			最終頁に続く

(54)【発明の名称】 系統メタデータの生成、アクセス、及び表示

(57)【特許請求の範囲】

【請求項 1】

データ処理機器によって実行される方法であって、

データソースからメタデータの一部を受信することであって、前記メタデータの一部はノード及び辺を記述し、前記辺の少なくとも一部は或るノードに対する別のノードの影響をそれぞれ表し、各辺は単一の方法を有する、受信すること、

前記メタデータの一部を表すデータ構造のインスタンスを生成し、および前記データ構造の前記インスタンスをランダムアクセスメモリ内に記憶することであって、前記データ構造の少なくとも1つのインスタンスは

対応するノードを識別する識別値、

前記対応するノードのそれぞれの特性を表す1つ又は複数の特性値、及び

前記ランダムアクセスメモリのそれぞれの一部に関連するそれぞれのメモリアドレスに対する1つ又は複数のポインタであって、前記メモリアドレスの少なくとも1つは前記データ構造の別の生成されたインスタンスノードのメモリアドレスに対応し、各ポインタは前記データ構造の前記別の生成されたインスタンスノードに含まれる、対応するそれぞれの識別値によって識別されるノードに関連する辺を表す、1つ又は複数のポインタ

を含む、生成すること、

クエリとウォークプランを受信することであって、前記クエリは、少なくとも1つの特定のデータ要素の識別を含み、前記ウォークプランは、辺の種類毎に辺をたどり又は収集するための条件を特定する第2のデータ構造である、こと、及び

10

20

前記特定のデータ要素の系統の表現をコンピュータシステムのディスプレイに表示させるために前記データ構造の少なくとも1つのインスタンスを使用すること、を含む、方法。

【請求項2】

前記ウォークプランが、対応するノードのそれぞれの特性を表す1つ又は複数の特性値に基づいて辺をたどり又は収集するための条件を含む、請求項1に記載の方法。

【請求項3】

前記識別値の少なくとも1つが、前記クエリ内で識別される前記特定のデータ要素に対応するノードを表す、請求項1または2に記載の方法。

【請求項4】

前記データ処理機器が前記クエリに対する応答を返すように構成され、前記応答は前記特定のデータ要素の系統を記述するメタデータを含む、請求項1～3の何れか一項に記載の方法。

【請求項5】

前記特定のデータ要素の系統を記述する前記メタデータがノード及び辺のシーケンスを記述するメタデータを含み、前記シーケンスの前記ノードのうちの1つは前記特定のデータ要素を表す、請求項4に記載の方法。

【請求項6】

前記メタデータの一部がそれぞれのノードに関連するテキスト文字列を含み、前記データ構造が前記テキスト文字列の少なくとも1つを省く、請求項1～5の何れか一項に記載の方法。

【請求項7】

前記データ構造の前記インスタンスを生成することが、系統に関連するメタデータを識別すること及び前記識別したメタデータを前記データ構造によって定められる形式で記憶することを含む、請求項1～6の何れか一項に記載の方法。

【請求項8】

前記クエリを受信することに応答し、
前記クエリによって識別される前記特定のデータ要素に対応する少なくとも1つの識別情報を含む前記データ構造のインスタンスにアクセスすることを含む、前記データ構造を使用して記憶されるデータにアクセスすること、及び

前記データ構造を使用して記憶される前記データに基づいて前記クエリに対する応答を生成することであって、前記応答は系統メタデータを含み、前記系統メタデータはノード及び辺のシーケンスを記述し、前記シーケンスの前記ノードのうちの1つは前記特定のデータ要素を表す、生成すること

を含む、請求項1～7の何れか一項に記載の方法。

【請求項9】

前記データ構造の第1のインスタンスにアクセスすることであって、前記データ構造の前記第1のインスタンスは前記クエリによって識別される前記特定のデータ要素に対応する第1の識別情報によって識別される、アクセスすること、

前記第1の識別情報に関連するポイントにアクセスすることであって、前記ポイントは前記第1の識別情報に関連する、アクセスすること、

前記データ構造の第2のインスタンスにアクセスすることであって、前記データ構造の前記第2のインスタンスは第2の識別情報によって識別され、前記第2の識別情報は前記第1の識別情報に関連する前記ポイントによって参照されるメモリアドレスに記憶されている、アクセスすること、及び

前記データ構造の前記第2のインスタンスのデータを収集すること

を含み、

前記応答が前記第2の識別情報に関連するノードを表すデータを含む、

請求項4または8に記載の方法。

【請求項10】

10

20

30

40

50

スケジュールされた間隔で前記データソースに要求を伝送することを含み、前記メタデータの一部は前記要求の１つに回答して前記データソースから受信される、請求項１～９の何れか一項に記載の方法。

【請求項１１】

実行可能命令を記憶する少なくとも１つの非一時的コンピュータ可読記憶媒体と、前記命令を実行するように構成される１つ又は複数のプロセッサとを含むシステムであって、前記命令を実行することが、請求項１～１０の何れか一項に記載のステップの実施を引き起こす、システム。

【請求項１２】

非一時的コンピュータ可読記憶装置であって、実行されるとき、請求項１～１０の何れか一項に記載のステップを実施する、非一時的コンピュータ可読記憶装置。

【請求項１３】

ソフトウェアプログラムであって、実行されるとき、請求項１～１０の何れか一項に記載のステップを実施する、ソフトウェアプログラム。

【請求項１４】

システムであって、データソースからメタデータの一部を受信するための手段であって、前記メタデータの一部はノード及び辺を記述し、前記辺の少なくとも一部は或るノードに対する別のノードの影響をそれぞれ表し、各辺は単一の方法を有する、受信するための手段と、

前記メタデータの一部を表すデータ構造のインスタンスを生成し、および前記データ構造の前記インスタンスをランダムアクセスメモリ内に記憶するための手段であって、前記データ構造の少なくとも１つのインスタンスは

対応するノードを識別する識別値、

前記対応するノードのそれぞれの特性を表す１つ又は複数の特性値、及び

前記ランダムアクセスメモリのそれぞれの一部に関連するそれぞれのメモリアドレスに対する１つ又は複数のポインタであって、前記メモリアドレスの少なくとも１つは前記データ構造の別の生成されたインスタンスノードのメモリアドレスに対応し、各ポインタは前記データ構造の前記別の生成されたインスタンスノードに含まれる、対応するそれぞれの識別値によって識別されるノードに関連する辺を表す、１つ又は複数のポインタ

を含む、生成するための手段と、

クエリとウォークプランとを受信する手段であって、前記クエリは、少なくとも１つの特定のデータ要素の識別を含み、前記ウォークプランは、辺の種類毎に辺をたどり又は収集するための条件を特定する第２のデータ構造である、手段と、

前記特定のデータ要素の系統の表現をコンピュータシステムのディスプレイに表示させるために前記データ構造の少なくとも１つのインスタンスを使用するための手段と、

を含む、システム。

【発明の詳細な説明】

【技術分野】

【０００１】

技術分野

本願は系統メタデータ、例えばデータ記憶システム内に記憶されるデータ要素の系統を生成し、それにアクセスし、それを表示するためのデータ構造及び方法に関する。

【背景技術】

【０００２】

背景

企業はデータを管理するためにデータウェアハウジング、顧客関係管理、データマイニング等のデータ処理システムを使用する。多くのデータ処理システムにおいて、データは

10

20

30

40

50

データベースファイル、運用システム、フラットファイル、インターネット、他の情報源等の多くの異なるデータソースから中央リポジトリ内にプルされる。多くの場合、データシステム内にロードされる前にデータが変換される。変換は、クレンジング、統合、及び抽出を含む。データ、データのソース、及びデータシステム内に記憶されるデータに生じている変換を追跡するためにメタデータを使用することができる。メタデータ（「データに関するデータ」と呼ばれることもある）とは、他のデータの属性、形式、元、履歴、相互関係等を記述するデータである。メタデータの管理は複雑なデータ処理システムにおいて中心的な役割を果たし得る。

【 0 0 0 3 】

利用者は特定のデータが様々なデータソースからどのように導出されるのかを調査したい場合がある。例えば利用者はデータセット又はデータオブジェクトがどのように生成されたのか、又はどのソースからデータセット又はデータオブジェクトがインポートされたのかを知りたい場合がある。導出元のソースまでデータセットを追跡することをデータ系統追跡（又は「アップストリームデータ系統追跡」）と呼ぶ。利用者は特定のデータセットがどのように使用されているのか、例えばどのアプリケーションが所与のデータセットを読み出したのかを調査したい場合がある（「ダウストリームデータ系統追跡」又は「インパクト解析」と呼ばれる）。利用者は或るデータセットが他のデータセットとどのように関係するのかを知ることに関心がある場合もある。例えば利用者はデータセットが修正されているのかどうかや、どのテーブルが影響を受けるのかを知りたい場合がある。

【 0 0 0 4 】

一種のメタデータである系統は、データ系統に関する質問（例えば「所与の値がどこから来たのか？」、「出力値はどのように計算されたのか？」や、「どのアプリケーションがこのデータを作り出し、このデータに依存するのか？」）に対する回答を利用者が得ることを可能にする。利用者は提案された修正（例えば「この部分が変更された場合、他の何処が影響を受けるのか？」や「このソースフォーマットが変更される場合、どのアプリケーションが影響を受けるのか？」）の結果を理解することができる。利用者は技術上のメタデータ及びビジネス上のメタデータの両方が関与する回答への質問も得ることができる（例えば「どのグループがこのデータを作成し使用する責任を負うのか？」、「このアプリケーションを最後に変更したのは誰か？」や、「どのような変更を彼らは加えたのか？」）。

【 0 0 0 5 】

これらの質問に対する回答は複雑なデータ処理システムを解析し、その問題に対処するのに助けることができる。例えば或るデータ要素が異常値を有する場合、任意の数の過去の入力又はデータ処理ステップがその異常値に関与している可能性がある。従って、関心のあるデータ要素を表す視覚要素、並びに関心のあるデータ要素に影響を及ぼす又は関心のあるデータ要素の影響を受ける他のデータ要素を表す視覚要素を含む図の形式で系統を利用者に提示する場合がある。利用者はその図を見て、関心のあるデータ要素に影響を及ぼす他のデータ要素及び／又は変換を視覚的に識別することができる。一例として、この情報を使用することにより、利用者はデータ要素及び／又は変換の何れかが異常値の原因であり得るかどうかを知り、問題が発見される場合は根底にあるデータ処理ステップの何れかを訂正する（又は訂正するためにフラグを立てる）ことができる。別の例として、この情報を使用することにより、利用者はシステムの一部に必須であり得る（例えばそのためシステムからそれらを除くことで関心のあるデータ要素に影響を受けることになる）任意のデータ要素若しくは変換、及び／又はシステムの一部に必須ではない可能性がある（例えばそのためシステムからそれらを除くしても関心のあるデータ要素は影響を受けない）データ要素若しくは変換を識別することができる。

【 発明の概要 】

【 課題を解決するための手段 】

【 0 0 0 6 】

概要

10

20

30

40

50

とりわけ本発明者らはデータ処理機器によって実行される方法を記載し、その方法は、データソースからメタデータの一部を受信することであって、メタデータの一部はノード及び辺を記述し、辺の少なくとも一部は或るノードに対する別のノードの影響をそれぞれ表し、各辺は単一の方法を有する、受信すること、メタデータの一部を表すデータ構造のインスタンスを生成することであって、データ構造の少なくとも1つのインスタンスは対応するノードを識別する識別値、対応するノードのそれぞれの特性を表す1つ又は複数の特性値、及びそれぞれの識別値に対する1つ又は複数のポインタを含み、各ポインタは対応するそれぞれの識別値によって識別されるノードに関連する辺を表す、生成すること、データ構造のインスタンスをランダムアクセスメモリ内に記憶すること、少なくとも1つの特定のデータ要素の識別を含むクエリを受信すること、及び特定のデータ要素の系統の表現をコンピュータシステムのディスプレイに表示させるためにデータ構造の少なくとも1つのインスタンスを使用することを含む。

10

【0007】

これらの技法は、方法として、システムとして、及び/又はコンピュータ可読記憶装置上に記憶されるコンピュータプログラム製品としてを含む幾つかのやり方で実装することができる。

【0008】

これらの技法の態様は以下の利点の1つ又は複数を含み得る。系統メタデータは、系統メタデータに対するクエリに応答するとき速度及び効率を得るように設計された専用データ構造を使用して記憶することができる。系統メタデータはメモリ内に記憶することができる、そのため系統メタデータを記憶するコンピュータシステムは、系統メタデータがメモリ内に記憶されていない場合（例えば系統メタデータがハードディスク又は別の種類の記憶技法の中に記憶されそこからアクセスされる場合）よりも速く系統メタデータに対するクエリに応答することができる。本明細書に記載する技法を使用し、系統データを他の技法よりもはるかに速く、例えば500倍速く取得することができる。

20

【図面の簡単な説明】

【0009】

図面の説明

【図1A】メタデータの処理環境を示す。

【図1B】メタデータの処理環境を示す。

【図1C】メタデータの処理環境を示す。

【図1D】メタデータの処理環境を示す。

【図2A】メタデータの閲覧環境内で表示される情報の一例を示す。

【図2B】メタデータの閲覧環境内で表示される情報の一例を示す。

【図2C】メタデータの閲覧環境内で表示される情報の一例を示す。

【図2D】メタデータの閲覧環境内で表示される情報の一例を示す。

【図2E】メタデータの閲覧環境内で表示される情報の一例を示す。

【図3】データ構造の一例を示す。

【図4】ウォークプランの一例を示す。

【図5A】専用データ構造によって定められる形式で系統メタデータを記憶するための手続きを表す流れ図を示す。

40

【図5B】系統メタデータを表示させるための手続きを表す流れ図を示す。

【図6】専用データ構造の形式で記憶される系統メタデータを横断するための手続きを表す流れ図を示す。

【発明を実施するための形態】

【0010】

様々な図面の中の同様の参照番号は同様の要素を示す。

【0011】

説明

メタデータへのアクセスを管理するシステムは特定のデータ要素の系統を要求する利用

50

者からクエリを受信し、それに応答してデータ要素の系統を表す図を送ることができる。データ要素が相対的に大量のデータを記憶するデータ記憶システムに属する場合、メタデータへのアクセスを管理するシステムはデータ要素の系統を処理し、対応する図を生成するために大量の処理時間を費やす必要があり得る。しかし、系統メタデータを処理することに充てられその種の処理に最適化されるシステムを導入することにより、その処理を加速しより効率的にすることができる。従って本明細書は、専用のシステムを使用しない場合よりも概して高速且つ効率的なやり方で系統メタデータを処理し記憶するために専用のシステムを使用する技法を記載する。

【0012】

図1Aは、系統メタデータを記憶し、それを環境内の他のシステムに提供する系統サーバ102を含むメタデータ処理環境100を示す。メタデータ処理環境100は、メタデータを得るための要求に概して応答するメタデータサーバ104も含む。メタデータサーバ104は、メタデータデータベース108内に記憶されるメタデータ106にアクセスできる。メタデータ106は、メタデータデータベース108にメタデータ112A~Cを継続的に与えるデータソース110A~Cから来る。例えばデータソース110A~Cは、リレーショナルデータベース、フラットファイル、ネットワークソース等の任意の組合せとすることができる。

10

【0013】

使用中、メタデータサーバ104は利用者118によって操作されるユーザ端末116から受信されるクエリ114に応答する。例えばユーザ端末116は、パーソナルコンピュータ、ラップトップコンピュータ、タブレット装置、スマートフォン等の計算装置とすることができる。一部の例では、例えばメタデータサーバ104がネットワーク上でデータへのアクセスを提供するように構成され、ウェブブラウザとインタフェース可能なウェブサーバを含む又はウェブサーバと通信する場合、ユーザ端末116がウェブブラウザ等のネットワークベースのユーザアプリケーションを動作させる。概して、本明細書に記載のコンピュータシステム間の対話の多くは、インターネット又はその種のネットワーク上で一般に使用される通信プロトコルを使用する同様の通信ネットワークを使用して行われ得る。

20

【0014】

メタデータサーバ104は、複数の種類のメタデータに対するクエリに応答するように構成される。一種のメタデータは系統メタデータなので、メタデータサーバ104は、メタデータデータベース108内に記憶される特定のデータ要素の系統を記述するメタデータ106にアクセスすることにより、特定のデータ要素、例えばデータソース110A~Cの1つによって記憶されるデータ要素120の系統を要求するクエリ114を処理することができる。次いでメタデータサーバ104はユーザ端末116に系統メタデータ122、例えば系統図形式の系統メタデータを提供することができる(図2A~図2Eに関して以下で説明する)。

30

【0015】

一部の例では、系統メタデータに関するクエリ114を処理することは、メタデータサーバ104の相対的に大量の処理時間を必要とし及び/又は相対的に大量の処理資源を使用するタスクである。例えばクエリ114を処理するために、メタデータサーバ104はメタデータデータベース108内に記憶されるメタデータ106にアクセスする必要がある。この例では、必要なメタデータの全てにアクセスするために、メタデータサーバ104がメタデータデータベース108に対するクエリを生成する処理資源を費やす必要がある。更に、メタデータデータベース108にクエリを伝送し、応答を待つプロセスはレイテンシ、例えば通信ネットワークレイテンシを発生させる。加えて一部の例では、系統図を作成するのに必要なメタデータを抽出するために、メタデータデータベース108から受信されるメタデータをメタデータサーバ104が処理しなければならない。例えばメタデータデータベース108は系統メタデータ以外に様々な種類のメタデータを記憶するので、メタデータデータベース108から受信されるメタデータは関心のあるデータ要

40

50

素の系統に直接関係しない情報を含む場合があり、そのため系統に関係しない情報を識別し除去するために追加の処理時間が使われる。

【 0 0 1 6 】

一部の実装形態では、例えばメタデータ処理環境 1 0 0 の性能を改善するために、系統サーバ 1 0 2 を使用してメタデータサーバ 1 0 4 に系統メタデータを提供する。系統サーバ 1 0 2 は、系統サーバを使用しない技法よりも概して高速且つ効率的にアクセス可能な形式で系統メタデータ 1 2 4 を記憶する専用のシステムである。具体的には、系統サーバ 1 0 2 は、系統メタデータに対するクエリに応答するとき速度及び効率を得るように設計された専用データ構造を使用して系統メタデータ 1 2 4 を記憶する。特定のデータ構造を使用して記憶される全てのデータが同じやり方で構成されるように、データ構造はデータの構成を定める。データ構造の技法については図 3 に関して以下でより詳細に説明する。

10

【 0 0 1 7 】

使用中、系統メタデータ 1 2 8 を取得するために系統サーバ 1 0 2 がメタデータデータベース 1 0 8 にクエリ 1 2 6 を伝送する。系統サーバ 1 0 2 は、理想的には広範な多数の、例えばデータソース 1 1 0 A ~ C によって記憶されるデータ要素 1 2 0 の殆ど又は全てに関する系統メタデータを記憶する。このようにして、系統サーバ 1 0 2 はクエリが行われ得るデータ要素 1 2 0 の殆どについて系統のクエリに応答することができる。系統サーバ 1 0 2 がメタデータデータベース 1 0 8 から系統メタデータ 1 2 8 を受信すると、系統サーバ 1 0 2 は記憶済みの系統メタデータ 1 2 4 を含む自らのデータ構造を更新する。一部の例では、相対的に最新の多数の系統メタデータを記憶するために、系統サーバ 1 0 2 がメタデータデータベース 1 0 8 に新たなクエリ 1 2 6 を定期的に、例えば毎時、毎日、又は別の間隔で送信する。例えばこの間隔は、例えば系統サーバ 1 0 2 によって保持されるスケジュールデータに対応するスケジュール済みの間隔であり得る。

20

【 0 0 1 8 】

図 1 B でより詳細に示すように、メタデータサーバ 1 0 4 は系統メタデータに対するクエリ 1 1 4 (例えばユーザ端末 1 1 6 上で系統図を表示するために使用可能な系統メタデータに対するクエリ)を受信し、そのクエリ 1 1 4 を系統サーバ 1 0 2 に提供することができる。次いで系統サーバは、クエリ 1 1 4 に応答して系統メタデータ 1 2 2 を返すことができる。メタデータサーバ 1 0 4 は、系統メタデータをメタデータデータベース 1 0 8 から取得すること等の他の技法を使用して取得される系統メタデータと比較し、受信した系統メタデータ 1 2 2 をユーザ端末 1 1 6 に提供する前に準備するのに処理時間をさほど費やす必要がなく、処理資源をさほど使用する必要がない。

30

【 0 0 1 9 】

図 1 C 及び図 1 D は、系統サーバ 1 0 2 及びメタデータサーバ 1 0 4 の要素並びにそれらがどのように対話するのかを示す。上記のように、メタデータサーバ 1 0 4 は特定のデータ要素の系統を要求するクエリ 1 1 4 を受信する。クエリ 1 1 4 は、系統が要求された特定のデータ要素(例えば図 1 A のデータ要素 1 2 0 のうちの 1 つ)を識別する。メタデータサーバ 1 0 4 はデータ要素の識別を使用して、そのデータ要素に関係する系統メタデータを集めるために使用可能な 1 組のウォークプラン 1 3 2 からウォークプラン 1 3 0 を選択する。ウォークプラン 1 3 0 は、1 組の系統メタデータを特定のやり方で横断(「ウォーク」)する方法を記述するデータ構造(例えば X M L 文書等のタグ付き部分を含む構造化文書)である。一部の例では、データ要素のデータの種類の基づいてウォークプランを選択することができる。例えば特定のデータの種類のウォークプラン 1 3 2 のうちの特定のものに関連し得る(例えばメタデータサーバ 1 0 4 にとってアクセス可能な関連のインデックス内に記憶される関連)。ウォークプラン 1 3 2 については図 4 に関して以下で詳細に説明する。

40

【 0 0 2 0 】

ウォークプラン 1 3 0 が選択されると、メタデータサーバ 1 0 4 が系統サーバ 1 0 2 にクエリ 1 1 4 及びウォークプラン 1 3 0 を伝送する。それに応答し、系統サーバ 1 0 2 が、自らの系統メタデータのデータ構造 1 3 4 の中からクエリ 1 1 4 に関連する系統メタデ

50

ータを識別する。データ構造 134 は、システムメタデータに対するクエリに応答するのに必要な如何なるデータも省くことなしに、システムメタデータを含むのに必要な記憶空間量を最小化するやり方で構成されるシステムメタデータの表現である。従って、システムサーバ 102 は典型的には自らのデータ構造 134 内に記憶されるデータを使用して、クエリ 114 に応答するためにメタデータサーバ 104 が必要とするシステムメタデータの全てをメタデータサーバ 104 に提供することができる。データ構造 134 については図 3 に関して以下で詳細に説明する。

【0021】

一部の実装形態では、データ構造 134 が高速アクセス（例えばデータの高速の読出し及び書込み）のためにシステムサーバ 102 のメモリ 135 内にロードされる。メモリの一例はランダムアクセスメモリである。ランダムアクセスメモリは、同じサイズ（例えばバイトやワード）の他の任意の項目とほぼ同じ時間で各データ項目にアクセスできるようなやり方でデータ項目を記憶する。対照的に、磁気ディスク等の他の種類のデータ記憶域は、ディスクの現在の物理的状態（例えば磁気リード/ライトヘッドの位置）にもよるが、一部のデータ項目にアクセスするのに他のデータ項目よりも長い時間がかかることを生じさせる物理的制約を有する。ランダムアクセスメモリ内に記憶されるデータ項目は、典型的にはそのデータ項目に固有のアドレスに又は少数のデータ項目間で共有されるアドレスに記憶される。ランダムアクセスメモリは典型的には揮発性であり、そのためランダムアクセスメモリがアクティブ電源から切断される（例えばコンピュータシステムが受電でなくなる）場合、ランダム内に記憶されるデータが失われる。対照的に、磁気ディスク及び他の一部の種類のデータ記憶域は不揮発性であり、アクティブ電源なしでもデータを保持する。

【0022】

システムサーバ 102 はデータ構造 134 をメモリ 135 内に記憶するので、システムサーバ 102 はデータ構造をメモリ 135 内に記憶しない技法よりも速くシステムメタデータを読み書きすることができる。具体的には、データ使用量を最小化するやり方でデータ構造 134 が構成される。例えばデータ構造 134 は、メタデータサーバ 104 から得られる元のシステムメタデータ内にあるテキスト文字列等のデータを省くことができる。従ってシステムサーバ 102 の使用中、データ構造 134 の全て、例えばシステムメタデータを表すデータの全てをメモリ 135 内に記憶することができる。コンピュータシステムは、（例えばアドレス指定の制限により）所与の時間に使用可能なランダムアクセスメモリの量に関する制約を概して有する。更に、ランダムアクセスメモリは他の種類のデータ記憶域（例えば磁気ディスク）よりも 1 バイト当たり高価である傾向がある。従ってランダムアクセスメモリを使用した場合、データ構造 134 は特定のコンピュータシステム上でのその合算したサイズについて上限を有する可能性がある。従って、本明細書に記載の技法（例えば図 3 に関して以下で説明する技法）はデータ構造のサイズを最小化するが、クエリ 114 によって要求され得る系統に関する情報は保つ。

【0023】

一部の実装形態では、メタデータサーバ 104 がシステムメタデータ 137（例えば図 1A に示すメタデータデータベース 108 から受信されるシステムメタデータ）も記憶する。しかしメタデータサーバ 104 は、例えばシステムサーバ 102 のデータ構造を使用しないので、自らの記憶済みのシステムメタデータ 137 の殆どをランダムアクセスメモリ内に記憶しない。従ってメタデータサーバ 104 も幾らかのシステムメタデータ 137 を記憶しても、メタデータサーバ 104 はメタデータサーバ 104 にローカルに記憶されていない任意のメタデータを得るためにシステムサーバ 102 のシステムメタデータ 124 にアクセスすることができる。システムサーバ 102 が使用されていない場合、幾らかのシステムメタデータ 137 を記憶するメタデータサーバ 104 はメタデータデータベース 108（図 1A）内に記憶されるシステムメタデータに概してアクセスし、そのことには上記のようにシステムサーバ 102 を使用することと比べて性能の不利益があり得る。

【0024】

10

20

30

40

50

ここではランダムアクセスメモリを主たる例として使用したが、他の種類のメモリも系統サーバ102と共に使用することができる。例えば別の種類のメモリはフラッシュメモリである。ランダムアクセスメモリと異なり、フラッシュメモリは不揮発性である。しかし、フラッシュメモリは典型的にはデータ項目にアクセスすることに対する制約を有する。一部の種類のフラッシュメモリは、個々にアクセス可能なデータ項目とは対照的に、データ項目の集合（例えばデータ項目のブロック）が一度にアクセス可能な最小データ単位であるやり方で構成される。例えば一部の種類のフラッシュメモリ上でデータ項目を削除するために、全ブロックを削除しなければならない。残りのデータ項目を保つには、残りのデータ項目をフラッシュメモリに再書き込みすることができる。

【0025】

系統サーバ102はウォークプラン130を使用してデータ構造134を横断し、クエリ114に回答するデータ構造内に記憶される系統メタデータを収集する。図1Dに示すように、次いで系統サーバ102が系統メタデータ139を含む応答138をメタデータサーバ104に送り返す。メタデータサーバ104は系統メタデータ139を使用してクエリ114に対する自らの応答140を生成することができる。応答140は幾つかある形式のうちの1つを取り得る。一部の例では応答140が、系統サーバ102から受信される同じ系統メタデータ139を例えば最小限の後処理を伴う形式で含む。一部の例では、メタデータサーバ104が系統メタデータ139に対する後処理を実行する。例えば人間が読めない符号化形式で系統メタデータ139が受信される場合、例えばメタデータサーバ104は系統メタデータ139の形式を人間が読める形式に変更することができる。一部の例では、メタデータサーバ104が系統メタデータ139に基づいて系統図を生成し、系統図を表すデータを応答140に組み込む。（図2A～図2Eに関して以下で詳細に説明するように）例えば応答140が系統図である場合、一部の例では応答140がユーザ端末116（図1A）に伝送される。一部の例では、応答140が中間システムに伝送されてからユーザ端末に伝送され及び/又はユーザ端末に伝送するのに適した形式に処理される。

【0026】

図2Aは、メタデータの閲覧環境内で表示される情報の一例を示す。一部の例では、メタデータの閲覧環境がユーザ端末、例えば図1Aに示すユーザ端末116上で実行されるインタフェースである。図2Aの例では、メタデータの閲覧環境がデータ系統図200Aに関する情報を表示する。メタデータの閲覧環境の一例は、利用者（例えば図1Aに示す利用者118）がメタデータを視覚化し編集することを可能にするウェブベースのアプリケーションである。メタデータの閲覧環境を使用し、利用者は標準のウェブブラウザを使用して企業内の何処からでもメタデータを探索し、解析し、管理することができる。メタデータオブジェクトのそれぞれの種類は1つ又は複数のビュー又は視覚的表現を有する。図2Aのメタデータの閲覧環境は、標的要素206Aに関する系統図を示す。

【0027】

例えばこの系統図は、メタデータサーバ104（図1A）内に記憶されるメタデータオブジェクトを表すデータ及び/又は処理ノードに関する端から端までの系統、つまり所与の開始オブジェクトが依存するオブジェクト（そのソース）及び所与の開始オブジェクトが影響を及ぼすオブジェクト（その標的）を表示する。この例では、メタデータオブジェクトの2つの例であるデータ要素202Aと変換204Aとの間のつながりが示されている。これらのメタデータオブジェクトは図中のノードによって表現されている。データ要素202Aは例えばデータセット、データセット内のテーブル、テーブル内のカラム、及びファイル、メッセージ、レポート内のフィールドを表し得る。変換204Aの一例は、単一のデータ要素出力がどのように作成されるのかを記述する実行ファイル（executable）の要素である。ノード間のつながりはメタデータオブジェクト間の関係に基づく。

【0028】

図2Bは、図2Aに示す同じ標的要素206Aに関する対応する系統図200Bを示すが、各要素202Bがグループ化され、コンテキストに基づいてグループ内に表示されて

10

20

30

40

50

いる。例えばデータ要素 2 0 2 B は、データセット 2 0 8 B（例えばテーブル、ファイル、メッセージ、レポート）、（グラフ、計画、プログラム等の実行ファイル及びそれらが作用するデータセットを含む）アプリケーション 2 1 0 B、及びシステム 2 1 2 B 内でグループ化されている。システム 2 1 2 B は、データ及びデータを処理するアプリケーションの機能上のグループ化であり、システムはアプリケーション及びデータグループ（例えばデータベース、ファイルグループ、メッセージングシステム、データセットのグループ）で構成される。変換 2 0 4 B が、実行ファイル 2 1 4 B、アプリケーション 2 1 0 B、及びシステム 2 1 2 B 内でグループ化される。グラフ、計画、プログラム等の実行ファイルはデータセットを読み書きする。既定でどのグループを展開し、どのグループを折り畳むのかをパラメータが設定することができる。不要な詳細度を除去することにより、かかるパラメータの設定は利用者が自分にとって重要なグループの詳細だけを見ることを可能にする。

10

【 0 0 2 9 】

データシステムの計算を行うためにメタデータの閲覧環境を使用することは幾つかの理由から有用である。例えばデータ要素と変換との間の関係を計算し示すことは、報告された値が所与のフィールドレポートに関してどのように計算されたのかを利用者が突き止めるのを助け得る。利用者はどのデータセットが特定の種類のデータを記憶するのか、及びどの実行ファイルがそのデータセットに対して読み書きするのかを見ることが出来る。ビジネス用語の場合、データ系統図はどのデータ要素（例えばカラムやフィールド）が特定のビジネス用語（例えば企業内での定義）に関連するのかを示すことができる。

20

【 0 0 3 0 】

メタデータの閲覧環境内で示されるデータ系統図はインパクト解析でも利用者を助けることができる。とりわけ利用者は、データセットにカラム又はフィールドが追加される場合にどのダウストリーム実行ファイルが影響を受けるのか、及び誰が通知される必要があるのかを知りたい場合がある。インパクト解析は所与のデータ要素がどこで使用されるのかを明らかにすることができ、そのデータ要素を変更する分岐（ramification）も明らかにすることができる。同様に、利用者は実行ファイル内の変更によってどのデータセットが影響を受けるのか、又は製作から特定のデータベーステーブルを除去しても大丈夫かどうかを見ることができる。

【 0 0 3 1 】

30

データ系統図を生成するためにメタデータの閲覧環境を使用してデータシステムの計算を行うことはビジネス用語を管理するのに有用である。例えば或る企業内の従業員にとって、その企業の全体にわたるビジネス用語の意味、それらの用語間の関係、及びそれらの用語が指すデータについて一致をみることが望ましい場合が多い。ビジネス用語を整合的に使用することは企業データの透明性を高めることができ、ビジネス要件の伝達を助ける。従って、ビジネス用語の基礎を成す物理データを何処で見つけることができるのか、及びどのビジネスロジックが計算に使用されるのかを知ることが重要である。

【 0 0 3 2 】

データノード間の関係を見ることはメタデータを管理し維持する際にも有用であり得る。例えば利用者は誰がメタデータ片を変更したのか、メタデータ片のソース（又は「レコードのソース」）は何なのか、又は外部ソースからメタデータをロードし若しくはリロードするときどのような変更が加えられたのかを知りたい場合がある。メタデータを維持する際、指定の利用者がメタデータオブジェクト（ビジネス用語等）を作成できるようにすること、メタデータオブジェクトの特性（他のオブジェクトに対するオブジェクトの記述や関係等）を編集できるようにすること、又は使われなくなったメタデータオブジェクトを削除できるようにすることが望ましい場合がある。

40

【 0 0 3 3 】

メタデータの閲覧環境はオブジェクトの幾つかのグラフィカルビューを提供し、利用者がメタデータを探索し解析することを可能にする。例えば利用者はシステム及びアプリケーションのコンテンツを閲覧し、任意のオブジェクトの詳細を探索することができ、上記

50

のデータ系統解析やインパクト解析等の様々な種類の依存性解析を利用者が容易に行うことを可能にするデータ系統ビューを使用してオブジェクト間の関係も見ることができる。オブジェクトの階層構造も見ることができ、特定のオブジェクトについて階層構造を検索することができる。オブジェクトが見つかり、そのオブジェクトに利用者が容易に戻ることを可能にするブックマークをオブジェクトについて作成することができる。

【0034】

適切な許可により、利用者はメタデータの閲覧環境内でメタデータを編集することができる。例えば利用者はオブジェクトの記述を更新し、ビジネス用語を作成し、オブジェクト間の関係を定め（ビジネス用語をレポート内のフィールドやテーブル内のカラムにリンクすること等）、オブジェクトを移動し（例えば或るアプリケーションから別のアプリケーションにデータセットを移動すること）、又はオブジェクトを削除することができる。

10

【0035】

図2Cでは、標的要素206Aに関する対応する系統図200Cを示すが、標的データ要素206Aの計算に関与しているアプリケーションに合わせて分解能のレベルが設定されている。とりわけアプリケーション202C、204C、206C、208C、及び210Cを示しており、それはそれらのアプリケーションだけが標的データ要素206Aに関する計算に直接関与するからである。（例えば図中に更に多くの又は少ない詳細を表示するために）利用者が異なる分解能のレベルで系統図の任意の部分を見たい場合、利用者は対応する展開/折畳ボタン212Cを活性化することができる。

【0036】

20

図2Dは、異なる分解能のレベルでの対応する系統図200Dを示す。この例では展開/折畳ボタン212Cが利用者によって活性化されており、メタデータの閲覧環境が今度も同じ系統図を表示しているが、アプリケーション202C内のデータセット214D及び実行ファイル216Dを表示するためにアプリケーション202Cが展開されている。

【0037】

図2Eは、異なる分解能のレベルでの対応する系統図200Eを示す。この例ではカスタム展開によって展開される全てのものを表示することに利用者が決めている。最終的なデータソースである（例えばアップストリームシステムを有さない）任意のフィールド又はカラムが展開される。加えて、特定のフラグが立てられているフィールドも展開される。この例では、系統内の重要な中間点にあるデータセット及びフィールド上に特定のフラグが立てられており、1つのカラムは系統が表示されているカラムである。

30

【0038】

系統の他の例が、参照によりその全体を本明細書に援用する「VISUALIZING RELATIONSHIPS BETWEEN DATA ELEMENTS AND GRAPHICAL REPRESENTATIONS OF DATA ELEMENT ATTRIBUTES」と題された米国特許出願第12/629,466号の中で記載されている。

【0039】

メタデータの閲覧環境内で要素及び関係を見ることは、それらを表すノードの各々に関連する情報を追加することによってより有用にされ得る。関連情報をノードに追加する1つの例示的なやり方は特定のノード上に情報をグラフィカルにオーバーレイすることである。これらのグラフィックはノードによって表されるデータの幾らかの値又は特性を示すことができ、メタデータデータベース内の任意の特性であり得る。この手法には、通常は共通点のない2つ以上の情報片（データのノードとノードによって表されるデータの特性との間の関係）を組み合わせ、有用な情報を「脈絡内に」おこうとする利点がある。例えばデータノード間の関係の視覚的表現と共に、メタデータの品質、メタデータの新しさ、レコード情報のソース等の特性を表示することができる。この情報の一部は表形式でアクセス可能とすることができるが、データの様々なノード間の関係と共にデータの特性を見ることの方が利用者にとって役立つ可能性がある。利用者は、メタデータの閲覧環境内のデータ要素及び/又は変換ノード上にデータのどの特性を表示するのかを選ぶことができる。どの特性を表示するのかは既定のシステム設定に従って設定することもできる。

40

50

【 0 0 4 0 】

図 1 A に関して上記で説明したように、系統サーバ 1 0 2 はデータ構造 1 3 4 を使用して系統メタデータをメモリ（例えばランダムアクセスメモリ）内に記憶する。図 3 はデータ構造 3 0 0 の一例を示す。使用中、系統サーバ 1 0 2 はデータ構造 3 0 0 の多くのインスタンスを含む。データ構造のインスタンスは、データ構造によって定められるやり方でフォーマットされるデータの集合（例えばビットの集合）である。本明細書に記載のデータ構造 3 0 0 のインスタンスは「ノード」と呼ぶ場合がある。

【 0 0 4 1 】

データ構造 3 0 0 の各インスタンスはメタデータオブジェクト、例えば図 2 A に示すデータ要素 2 0 2 A 又は変換 2 0 4 A の 1 つを表す。一部の例では、データ構造 3 0 0 の各インスタンスは系統図、例えば図 2 A ~ 図 2 E に示す図 2 0 0 A ~ 2 0 0 E の中で表示され得るノードを表す。

【 0 0 4 2 】

使用中、系統サーバ 1 0 2 はデータ構造に固有のメモリ位置 3 0 2 に各データ構造 3 0 0 を記憶する。各データ構造 3 0 0 は典型的には他のデータ構造のメモリ位置を指す。

【 0 0 4 3 】

データ構造 3 0 0 は幾つかのフィールドで構成される。フィールドとはデータの集合、例えばデータ構造 3 0 0 のインスタンスを作るビットのサブセットである。識別情報フィールド 3 1 0 は、データ構造 3 0 0 のインスタンスに関する一意識別情報を表すデータを含む。種類フィールド 3 1 2 は、データ構造 3 0 0 の対応するインスタンスによって表されるメタデータオブジェクトの種類を表すデータを含む。一部の例では、種類は「データ要素」や「変換」等であり得る。一部の例では、種類フィールド 3 1 2 がデータ構造 3 0 0 のインスタンス内に含まれる前進辺及び後退辺の数も示す。特性フィールド 3 1 4 は、データ構造 3 0 0 の対応するインスタンスによって表されるメタデータオブジェクトの様々な特性をそれぞれ表す。特性フィールド 3 1 4 の例は、メタデータオブジェクトを識別するテキストラベルを含む「名前」フィールド、及びメタデータオブジェクトのサブタイプ、例えばメタデータオブジェクトがファイルオブジェクトを表すのか、実行ファイルオブジェクトを表すのか、データベースオブジェクト又は別のサブタイプを表すのかを示す「サブタイプ」フィールドを含み得る。他の種類の特性も使用することができる。概して、種類フィールド 3 1 2 及び特性フィールド 3 1 4 は系統サーバ 1 0 2 の特定のインスタンスについてカスタマイズすることができ、本明細書で挙げる例に制限されない。

【 0 0 4 4 】

データ構造は、前進辺 3 1 6 A ~ C 及び後退辺 3 1 6 D ~ F を表すフィールドも含む。これらの辺フィールド 3 1 6 A ~ F は、系統サーバ 1 0 2 がデータ構造からデータ構造に「ウォーク」し、系統メタデータを集めるときデータ構造のデータを収集することを可能にする。最も広い意味で、部分データを「収集する」ことに言及する場合、将来のアクション（例えば収集したデータを伝送すること）に関係するものとしてデータの一部を識別することを意味する。データの一部を収集することはデータを複製すること、例えば将来のアクションに使用されるバッファ又は待ち行列にデータを複製することを含む場合がある。

【 0 0 4 5 】

それぞれの辺フィールド 3 1 6 A ~ F は、ポインタフィールド 3 2 0 A ~ B を含む。ポインタフィールド 3 2 0 A ~ B は、それぞれのメモリ位置 3 2 2 A ~ B のアドレスを記憶する。概して、ポインタフィールド 3 2 0 A ~ B によって参照されるメモリ位置 3 2 2 A ~ B は、データ構造 3 0 0 の別のインスタンスを記憶するメモリの部分を指す。このようにして、メタデータオブジェクトを表すデータ構造の或るインスタンスが、他のメタデータオブジェクトを表すデータ構造の 1 つ又は複数の他のインスタンスに「リンク」される。従って辺 3 1 6 A ~ D は、例えば図 2 A ~ 図 2 E の系統図の例 2 0 0 A ~ E の中で示すメタデータオブジェクト間の関係に対応し得る。例えば前進辺 3 1 6 A は、メタデータオブジェクト（例えばデータ構造 3 0 0 のこのインスタンスによって表されるメタデータオ

10

20

30

40

50

プロジェクト)が別のメタデータオブジェクト(例えばメモリ位置322Aにあるデータ構造のインスタンスによって表されるメタデータオブジェクト)に有する影響を表す。別の例として、後退辺316Dは、別のメタデータオブジェクト(例えばメモリ位置322Bにあるデータ構造のインスタンスによって表されるメタデータオブジェクト)がデータ構造300のこのインスタンスのメタデータオブジェクトに対して有する影響を表す。

【0046】

それぞれの辺フィールド316A~Fは1つ又は複数のフラグ324も含む。フラグ324は、その関連する辺に関する情報のインジケータである。一部の例では、フラグ324の1つが、複数のあり得る種類から選択される関連する辺の種類を示し得る。多くの種類の辺があり得る。例えば一部の種類の辺は、入力/出力辺(或るオブジェクトからの出力及び別のオブジェクトへの入力を表す)、要素/データセット辺(要素と要素が属するデータセットとの間の関連を表す)、及びアプリケーション/親辺(実行可能アプリケーションとアプリケーションに関連するデータセットも含むコンテナ等のコンテナとの間の関連を表す)である。

10

【0047】

データ構造300の要素の多くは典型的には相対的に少量のデータを使用する。例えば識別情報フィールド310、種類フィールド312、及び特性フィールド314に関連するデータは合わせても数バイト、例えば32バイトにしかない可能性がある。これらのフィールドは僅か数ビットの中に一般に使用される情報を符号化し、例えばノードについて8個のあり得る種類しかない場合、種類フィールド312は僅か3ビット長とすることができる。ノードの種類を表すテキストの文字列等のより複雑なデータは使用する必要がない。更に、メモリ位置322A~Cに関連するデータは、典型的にはデータ構造300をインスタンス化するソフトウェアを実行するコンピュータシステムの種類に関連するメモリアドレスの長さと同じデータ量である。従って、システムメタデータを記憶するための他の技法によって使用されるデータと比較し、データ構造300の殆どの又は全てのインスタンスが相対的に少量のデータを合計で使用し得る。

20

【0048】

図4は、ウォークプラン400の一例を示す。図1Cに関して上記で説明したように、ウォークプラン400は典型的にはメタデータサーバ104によって記憶される。使用中、メタデータサーバ104はシステムメタデータを要求するときシステムサーバ102にウォークプランを提供する。

30

【0049】

ウォークプラン400は、システムサーバ102の記憶済みのデータ構造134を横断するときシステムサーバ102によって使用される情報を記述する。概して、システムメタデータ、例えば特定のメタデータオブジェクトに係るシステムメタデータに対するクエリを受信した場合、それに応答して全ての種類のシステムメタデータを返す必要はない。一部の例では、クエリにもよるが、一部の種類の辺に関連するシステムメタデータをクエリに応答したものではないことを理由に返す必要がない場合がある。

【0050】

従ってウォークプラン400は、システムサーバ102によって記憶されるシステムメタデータによって表される辺の種類の中にあり得る辺の種類ごとにレコード402A~Cを含む。レコード402Aは、レコード402Aに対応する辺の種類を示すデータを含む辺の種類フィールド404を含む。レコード402Aは、前進方向410に関してフォローフラグ406、ノード収集フラグ408、及び辺収集フラグ409も含み、後退方向416に関してフォローフラグ412、ノード収集フラグ414、及び辺収集フラグ415も含む。

40

【0051】

フォローフラグ406、412は、システムサーバ102が自らのデータ構造134を横断するときこの辺の種類の辺をたどるべきかどうかを示す。言い換えれば、前進方向410のフォローフラグ406はシステムサーバ102が、図3を参照してデータ構造300のインスタンスの前進辺フィールド316Aのポインタフィールド320Aによって識別される

50

メモリ位置 3 2 2 A にアクセスすべきかどうかを示す。同様に、後退方向 4 1 6 のフォローフラグ 4 1 2 はシステムサーバ 1 0 2 が、図 3 を参照してデータ構造 3 0 0 のインスタンスの後退辺フィールド 3 1 6 D のポインタフィールド 3 2 0 b によって識別されるメモリ位置 3 2 2 b にアクセスすべきかどうかを示す。

【 0 0 5 2 】

ノード収集フラグ 4 0 8、4 1 4 は、システムサーバ 1 0 2 が自らのデータ構造 1 3 4 を横断するとき、この辺の種類によって指される「ノード」と呼ばれることがあるデータ構造 3 0 0 (図 3) のインスタンスを収集すべきかどうかを示す。データ構造 3 0 0 のインスタンスを収集することに言及する場合、クエリを処理するシステムサーバ 1 0 2 (図 1 A) がクエリを処理することに対応して返されるデータにインスタンス (又はノード) のデータが追加されることを意味する。従ってノードが収集される場合、データ構造 3 0 0 のインスタンスによって表されるメタデータオブジェクトに関連するデータは、システムサーバ 1 0 2 によって返されるシステムメタデータの中にある。

10

【 0 0 5 3 】

辺収集フラグ 4 0 9、4 1 5 は、システムサーバ 1 0 2 が (例えばデータ構造 3 0 0 のインスタンスのポインタフィールド 3 2 0 A に対応する) 辺を収集すべきかどうかを示す。辺が収集される場合、辺を表すデータはシステムサーバ 1 0 2 によって返されるシステムメタデータの中にある。一部の実装形態では、辺がノード間のデータフローを表さない場合はその辺を収集しなくても良い。例えば辺は、 (或るノードによって表される) データオブジェクトと (別のノードによって表している) データオブジェクトのコンテナとの間の関連を表すことができる。このようにして、ウォークプラン 4 0 0 内でノード収集フラグ 4 0 8、4 1 4 及び辺収集フラグ 4 0 9、4 1 5 を使用することにより、システムメタデータ内に含めるために収集される場合もされない場合もあるノードを様々なやり方で互に関連させることができ、ノードはシステムメタデータ内に含めるために収集される場合もされない場合もある様々なデータを表し得る。

20

【 0 0 5 4 】

一部の实装形態では、使用中、ウォークプラン 4 0 0 を 1 つ又は複数の X M L (拡張マーク付け原語) 文書形式で表すことができる。X M L 文書は「タグ」によって分けられた部分の集合である。タグは典型的にはラベル (例えばタグの種類を識別するラベル) を含み、1 つ又は複数の属性も含み得る。タグは開始タグ及び終了タグの形式で、開始タグが対応する終了タグと対にされるようにもたらされることがある。このようにしてタグは階層的とすることができ、そのため例えば別のタグの開始タグと終了タグとの対の間にタグを配置することにより、タグが他のタグの中に「ネスト」される。

30

【 0 0 5 5 】

X M L 文書形式のウォークプランの一例を以下に示す :

```
lineageServerPlan direction= " both " conditionalOnArg= " !autoFilterEnabled " replacesQueries= " walk "
useEdge name= " DE-Tr "
condition special= " ExeInterfaceCallStack " /
condition special= " ControlFilter " /
condition special= " Summarization " /
/useEdge
useEdge name= " Tr-DE "
condition special= " ExeInterfaceCallStack " /
condition special= " ControlFilter " /
condition special= " Summarization " /
/useEdge
useEdge name= " DE-DS " direction= " forward " collectEdge= " false "
conditionalOnArg= " walkDSlevel " /
useEdge name= " Tr-Exe " direction= " forward " collectEdge= " false "
```

40

50

```

conditionalOnArg= " walkDSlevel " /
  useEdge name= " DS-Exe " conditionalOnArg= " walkDSlevel "
  condition special= " ExeInterfaceCallStack " /
  condition special= " DSLevelIfNoDE " /
  /useEdge
  useEdge name= " Exe-DS " conditionalOnArg= " walkDSlevel "
  condition special= " ExeInterfaceCallStack " /
  condition special= " DSLevelIfNoDE " /
  /useEdge
  useEdge name= " DE-DS " direction= " backward " collectEdge= " false "
  condition special= " DSLevelIfNoDE " /
  /useEdge
  useEdge name= " Tr-Exe " direction= " backward " collectEdge= " false "
  condition special= " DSLevelIfNoDE " /
  /useEdge
/lineageServerP1an

```

10

【 0 0 5 6 】

この例では、「useEdge」タグは所与の種類の辺に関する情報を指定する。各「useEdge」タグはレコード（例えばウォークプラン 4 0 0 のレコード 4 0 2 A ~ C）に対応し得る。「name」属性は辺の種類（例えば辺の種類 4 0 4）を指定し、「direction」属性は方向（例えば前進方向 4 1 0 や後退方向 4 1 6）を指定し、「collectEdge」属性は辺を収集するかどうかを指定する（例えば収集フラグ 4 0 8、4 1 4）。他のタグも使用することができる。例えば上記の例の中で示した「condition special」タグは、指定の辺の種類の辺がたどられる場合に実行されるカスタム規則を指定するために使用される。一部の例では、辺をたどるべきかどうか及び / 又は収集すべきかどうかを判定するための条件をカスタム規則が指定し得る。

20

【 0 0 5 7 】

図 5 A は、専用データ構造、例えば図 3 に示すデータ構造 3 0 0 によって定められる形式でシステムメタデータを記憶するための手続き 5 0 0 を表す流れ図を示す。手続き 5 0 0 は、例えば図 1 A に示すシステムサーバ 1 0 2 のコンポーネントによって実行され得る。

30

【 0 0 5 8 】

この手続きは、メタデータソースにシステムメタデータを要求する（5 0 2）。例えばメタデータソースは図 1 A に示すメタデータデータベース 1 0 8 とすることができる。この要求は定期的な間隔又は半定期的な間隔、例えば毎時、1 0 分ごと、1 分ごと、又は他の任意の間隔で行われる要求であり得る。一部の例では、この要求がイベント、例えばメタデータソースにおいて新たなメタデータを入手できるという通知等のイベントに応答して行われ得る。

【 0 0 5 9 】

システムメタデータは典型的には各ノードがメタデータオブジェクトを表すようにノード及び辺を記述し、辺は或るノードに対する別のノードの単向の影響をそれぞれ表し、例えばそのため各辺は単一の方法を有する。

40

【 0 0 6 0 】

一部の例では、例えばシステムサーバ 1 0 2 がシステムメタデータを表す 1 組の最初のデータ構造をまだ生成していない場合、この要求はデータソースによって記憶されている全てのシステムメタデータに対する要求である。一部の例では、例えばシステムサーバ 1 0 2 が 1 組の既存の記憶済みデータ構造を更新している場合、この要求は前回の要求から追加され又は変更されているシステムメタデータに対する要求である。

【 0 0 6 1 】

この手続きはメタデータソースからデータ、例えばシステムメタデータを受信する（5 0 4）。例えばシステムメタデータは、メタデータオブジェクト及びメタデータオブジェクト間の

50

関係を表すデータであり得る。

【 0 0 6 2 】

この手続きはデータ構造、例えば図 3 に示すデータ構造 3 0 0 のインスタンスを生成する (5 0 6)。例えばデータ構造は、メタデータソースから受信されるデータに対応する情報を含み得る。一部の例では、データ構造の各インスタンスがメタデータソースから受信されるそれぞれのノードに対応する。データ構造は識別値、例えばデータ構造のインスタンスに対応するノードを識別する識別値のためのフィールドを含み得る。データ構造は、データ構造のインスタンスに対応するノードの特性を表す特性フィールドも含み得る。データ構造は、他のノードの識別値へのポインタも可能であり、そのためポインタはそれぞれの識別値に対応するノードへの辺を表す。

10

【 0 0 6 3 】

この手続きはデータ構造を記憶する (5 0 8)。例えばデータ構造はメモリ、例えば図 1 C に示すメモリ 1 3 5 内に記憶することができる。一部の例では、データ構造がランダムアクセスメモリ内に記憶される。データ構造は系統メタデータを記憶するために使用されるので、系統に関係しない任意のデータ (例えばメタデータソースに記憶されている他の種類のメタデータ) を省き、データ構造を記憶するのに必要なデータ量を減らすことができる。

【 0 0 6 4 】

使用中、この手続きは例えば定期的にスケジュールされた次の間隔で、メタデータソースに系統メタデータを要求すること (5 0 2) に戻る。

20

【 0 0 6 5 】

図 5 B は、系統メタデータを表示させるための手続き 5 2 0 を表す流れ図を示す。手続き 5 2 0 は、例えば図 1 A に示す系統サーバ 1 0 2 のコンポーネントによって実行され得る。概して系統サーバは特定のデータ要素、例えばメタデータオブジェクトの系統を記述するメタデータを含む、クエリに対する応答を返すように構成される。一部の例では、メタデータはノード及び辺のシーケンスを記述し、そのシーケンスのノードのうちの 1 つは特定のデータ要素を表す。一部の例では、図 5 A に関して上記で説明した手続き 5 0 0 によって記憶される系統メタデータにアクセスするために手続き 5 2 0 が使用される。

【 0 0 6 6 】

この手続きはクエリ、例えば系統メタデータに対するクエリを受信する (5 2 2)。一部の例では、このクエリは系統メタデータが要求されるメタデータオブジェクトを識別する。

30

【 0 0 6 7 】

一部の実装形態では、このクエリは系統の種類の識別、及び識別される系統の種類にどの種類の辺が関連するのかを識別するウォークプランを含む。一部の例では、ウォークプランは、対応するノードのそれぞれの特性を表す 1 つ又は複数の特性値に基づいて辺をたどり又は収集するための条件を含む。ウォークプラン 4 0 0 の一例を図 4 に示す。

【 0 0 6 8 】

この手続きは系統メタデータを集める (5 2 4)。例えば、受信クエリのメタデータオブジェクトを表すノードにアクセスしそれを収集することができ、他のノードを収集するために辺 (例えばメモリ位置へのポインタ) を横断することができる。系統メタデータを集めることについては図 6 に関して以下でより詳細に説明する。

40

【 0 0 6 9 】

この手続きは、集めた系統メタデータを伝送する (5 2 6)。例えば、集めた系統メタデータはクエリを発行したコンピュータシステムに伝送することができる。

【 0 0 7 0 】

集めた系統メタデータの伝送後、集めた系統メタデータがコンピュータシステム、例えば図 1 A に示すユーザ端末 1 1 6 上で表示させられる場合がある (5 2 8)。例えば系統メタデータは、図 2 A ~ 図 2 E に示す系統図 2 0 0 A ~ 2 0 0 E 等の系統図の形式で表示することができる。

50

【 0 0 7 1 】

図 6 は、専用データ構造、例えば図 3 に示すデータ構造 3 0 0 のインスタンスの形式で記憶される系統メタデータを横断するための手続き 6 0 0 を表す流れ図を示す。手続き 6 0 0 は、例えば図 1 A に示す系統サーバ 1 0 2 のコンポーネントによって実行され得る。

【 0 0 7 2 】

この手続きはクエリ及びウォークプラン、例えば図 1 C に示すクエリ 1 1 4 及びウォークプラン 1 3 0 を受信する (6 0 2)。この手続きは、クエリ 1 1 4 によって参照されるメタデータオブジェクトを表す初期ノード (例えば図 3 に示すデータ構造 3 0 0 のインスタンス) にアクセスアクセスする (6 0 4)。例えば初期ノードは、メタデータオブジェクトに関連するデータを記憶する識別情報フィールド 3 1 0 (図 3) によって識別され得る。次いでその初期ノードが「現在の」ノードとして使用され、待ち行列から現在のノードが選択され、現在のノードに操作が適用されるこのプロセスの再帰的な部分が始まる。言い換えれば、初期ノードは待ち行列の最初のノードとして待ち行列内に配置され、この手続きが実行されるにつれて他のノードが続いて待ち行列に追加される。

10

【 0 0 7 3 】

この手続きは、現在のノード内に残っている前進辺ポインタ (例えばまだアクセスされていない前進辺ポインタ) があるかどうかを判定する (6 0 6)。かかる前進辺ポインタがある場合、この手続きはまだアクセスされていない次のポインタにアクセスし (6 0 8)、例えばポインタのメモリ位置にアクセスしてそのメモリ位置に記憶されているデータを取得する。この手続きは、例えばポインタに関連する辺の種類に従い、(図 4 に関して上記で説明した) ウォークプランに基づいてそのポインタにおけるノードを「ウォーク」(例えば処理) するかどうかを判定する (6 1 0)。ウォークしない場合、この手続きは別のポインタにアクセスする (6 0 8)。ウォークする場合、この手続きはそのポインタにおけるノードを収集するかどうかを判定する (6 1 1)。収集する場合、この手続きはクエリに応答して返されるノードのデータを記憶し (6 1 2)、次いでそのポインタにアクセスできるようにそのノードを待ち行列内に配置する (6 1 4)。収集しない場合、この手続きはノードを待ち行列内に入れるだけである。

20

【 0 0 7 4 】

現在のノード内の前進辺ポインタの全てが横断されると、この手続きは現在のノード内に残っている後退辺ポインタがあるかどうかを判定する (6 1 6)。かかる後退辺ポインタがある場合、この手続きは次の後退辺ポインタにアクセスする (6 0 8)。

30

【 0 0 7 5 】

前進辺ポインタ又は後退辺ポインタが残っていない場合、この手続きは待ち行列内にノードがどれが残っているかどうかを判定する (6 1 8)。残っている場合、この手続きは待ち行列内の次のノードにアクセスし (6 2 0)、待ち行列内の次のノードを現在のノードとして使用して上記の操作を実行する。ノードが残っていない場合、この手続きは他のシステムに伝送するために収集済みのデータを準備する (6 2 2)。例えば収集済みのデータは伝送されるので、特定の形式で構成することができる。別の例として、収集済みのデータ内の符号化データを復号することができる。例えば符号化値を含むデータフィールドは値に対応するテキスト文字列に変換することができる。

40

【 0 0 7 6 】

データを伝送する準備ができると、例えば図 5 B のデータ伝送 5 2 6 に関して説明したようにデータを伝送することができる。

【 0 0 7 7 】

本明細書に記載したシステム及び技法は、例えば適切なソフトウェア命令を実行するプログラム可能計算システムを使用して実装することができ、書替え可能ゲートアレイ (F P G A) 等の適切なハードウェアによって実装することができ、又は何らかの混成形式で実装することができる。例えばプログラムされた手法では、1 つ又は複数のプログラムされた計算システム又はプログラム可能な計算システム (分散、クライアント / サーバ、又はグリッド等の様々なアーキテクチャのものであり得る) 上で実行される 1 つ又は複数の

50

コンピュータプログラム内の手続きをソフトウェアが含むことができ、かかる計算システムは少なくとも1個のプロセッサ、(揮発性及び/又は不揮発性のメモリ及び/又は記憶要素を含む)少なくとも1つのデータ記憶システム、(少なくとも1つの入力装置又はポートを使用して入力を受け付け、少なくとも1つの出力装置又はポートを使用して出力を行うための)少なくとも1つのユーザインタフェースをそれぞれ含む。ソフトウェアは、例えばデータフローグラフの設計、構成、及び実行に関するサービスを提供するより大きいプログラムの1つ又は複数のモジュールを含み得る。プログラムのモジュール(例えばデータフローグラフの要素)は、データリポジトリ内に記憶されるデータモデルに従うデータ構造又は他の組織化されたデータとして実装され得る。

【0078】

揮発性記憶媒体や不揮発性記憶媒体又は他の任意の非一時的媒体の中に具体化されるもの等、或る期間(例えばダイナミックRAM等のダイナミックメモリ装置のリフレッシュ期間の間の時間)にわたって媒体の物理的特性(例えば表面ビット及びランド、磁区、電荷)を使用してソフトウェアを非一時的な形式で記憶することができる。命令をロードすることに備えて、ソフトウェアは、(例えば汎用又は専用の計算システム又は計算装置によって読み出し可能な)CD-ROMや他のコンピュータ可読媒体等の有形の非一時的媒体上に与えることができ、又はソフトウェアがそこで実行される計算システムの有形の非一時的媒体までネットワークの通信媒体上で届けられ得る(例えば伝搬信号内に符号化される)。処理の一部又は全てが専用コンピュータ上で、又はコプロセッサ、書替え可能ゲートアレイ(FPGA)、専用の特定用途向け集積回路(ASIC)等の専用ハードウェアを使用して実行され得る。処理は分散式に実装されても良く、その場合ソフトウェアによって指定される計算の異なる部分が異なる計算要素によって実行される。本明細書に記載した処理を実行するために記憶装置媒体がコンピュータによって読み出されるときコンピュータを構成し動作させるために、そのような各コンピュータプログラムは、好ましくは汎用又は専用のプログラム可能コンピュータによってアクセス可能な記憶装置のコンピュータ可読記憶媒体(例えばソリッドステートメモリ又は媒体、磁気媒体、光学媒体)上に記憶され又はかかるコンピュータ可読記憶媒体にダウンロードされる。本発明のシステムは、コンピュータプログラムで構成される有形の非一時的媒体として実装されと考えられる場合もあり、そのように構成される媒体は本明細書に記載した処理ステップの1つ又は複数を実行するようにコンピュータを特定の且つ既定のやり方で動作させる。

【0079】

本発明の幾つかの実施形態を説明してきた。それでもなお、上記の説明は添付の特許請求の範囲によって定める本発明の範囲を限定するのではなく例示することを目的としていることを理解すべきである。従って他の実施形態も以下の特許請求の範囲に含まれる。例えば本発明の範囲から逸脱することなしに様々な修正を加えることができる。加えて、上記のステップの一部は順序に依存しない場合があり、従って説明したのとは異なる順序で実行することができる。

10

20

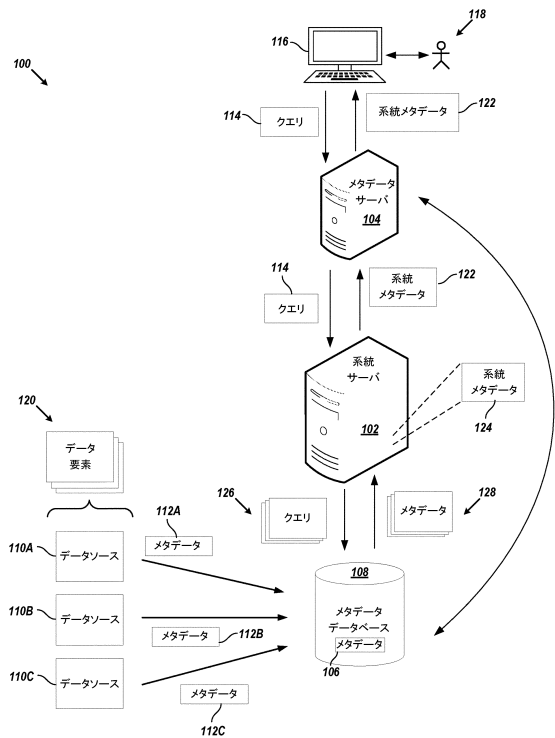
30

40

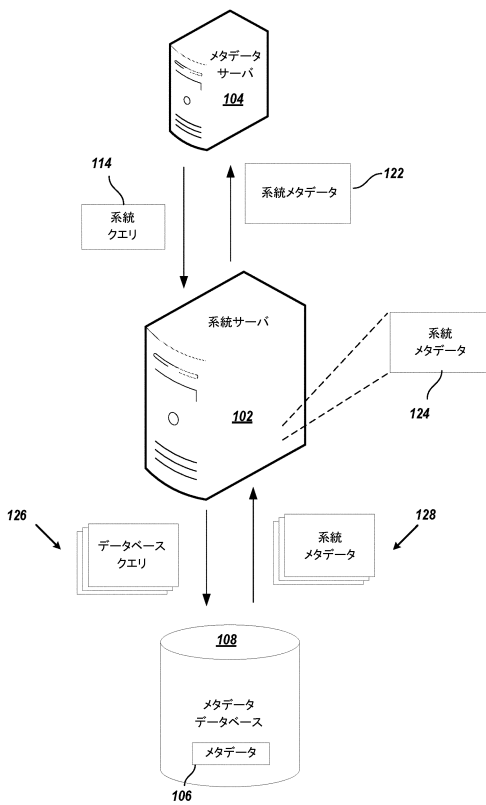
50

【図面】

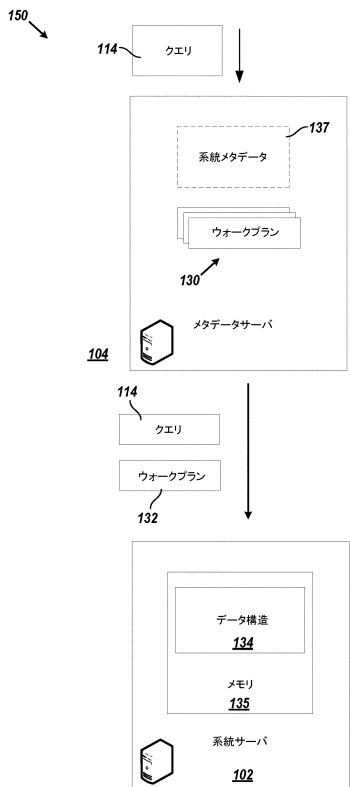
【図 1 A】



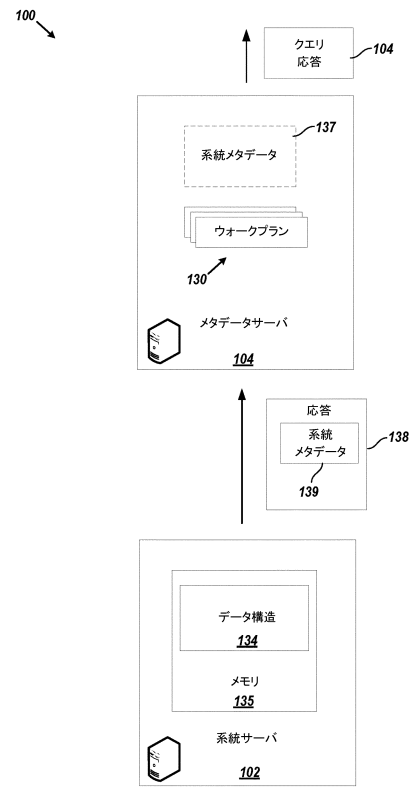
【図 1 B】



【図 1 C】



【図 1 D】



10

20

30

40

50

【図 2 A】

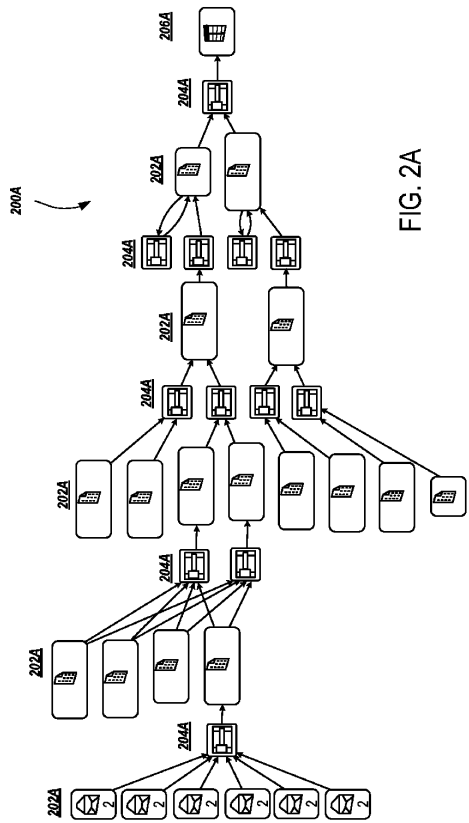


FIG. 2A

【図 2 B】

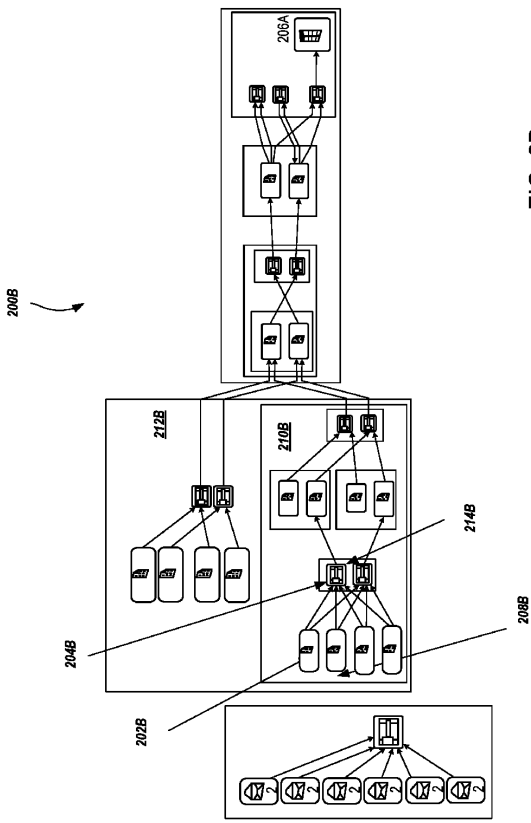
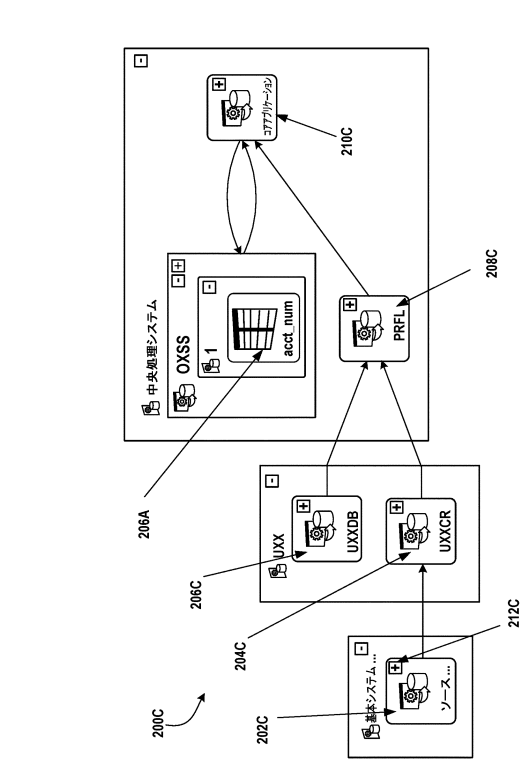
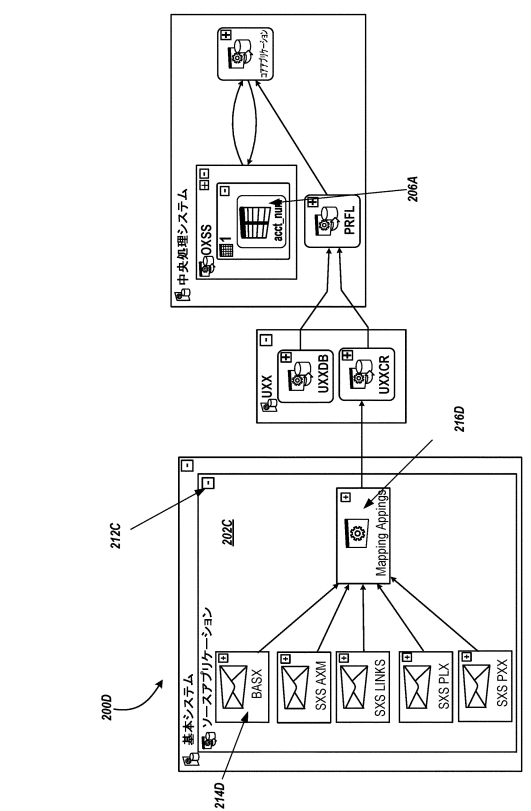


FIG. 2B

【図 2 C】



【図 2 D】



10

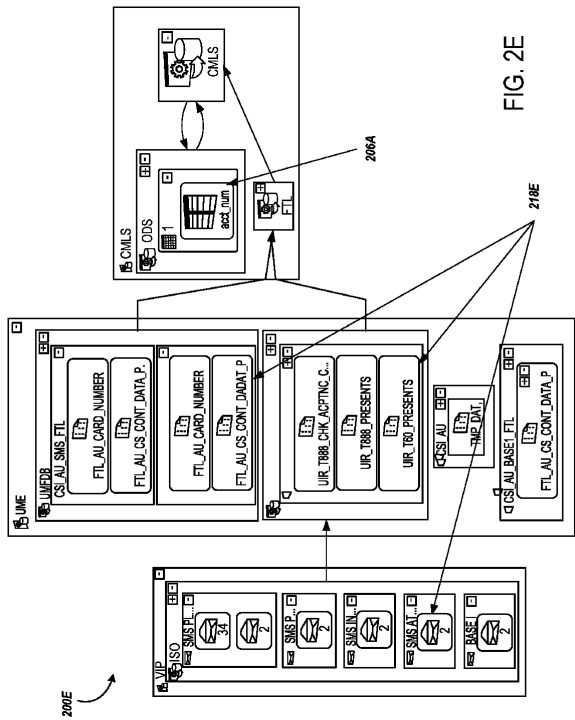
20

30

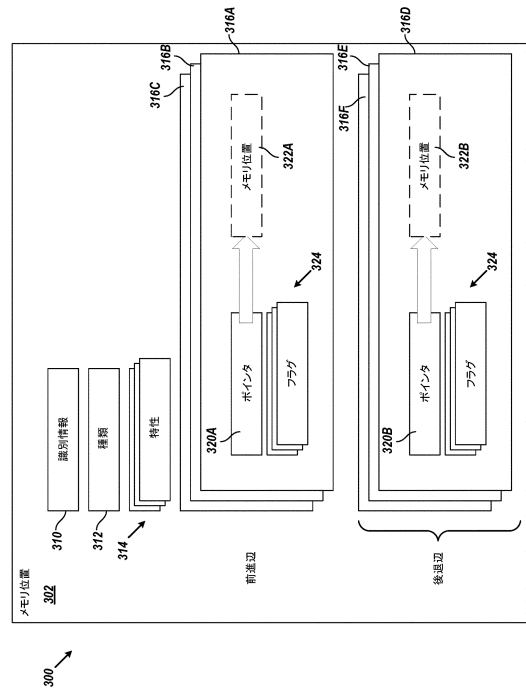
40

50

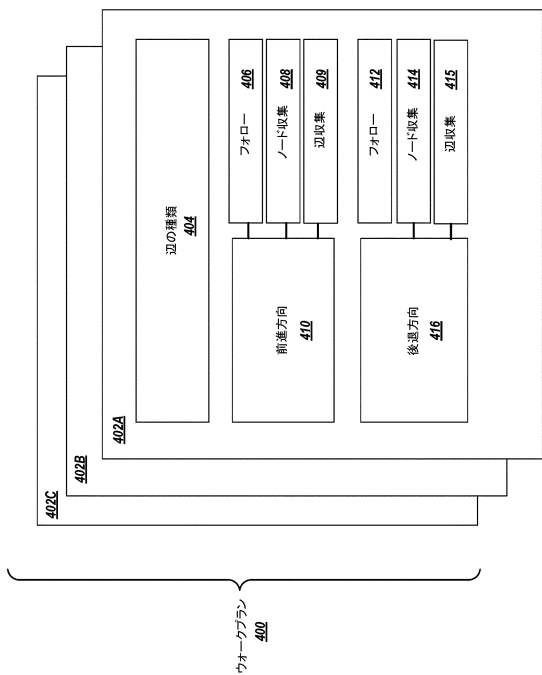
【図 2 E】



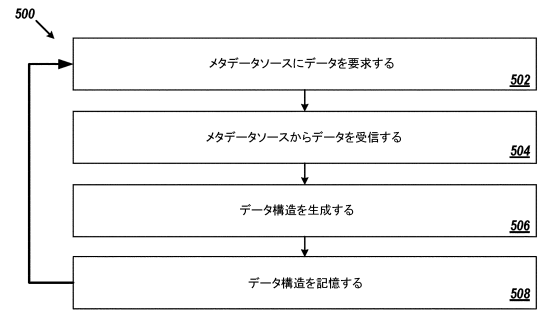
【図 3】



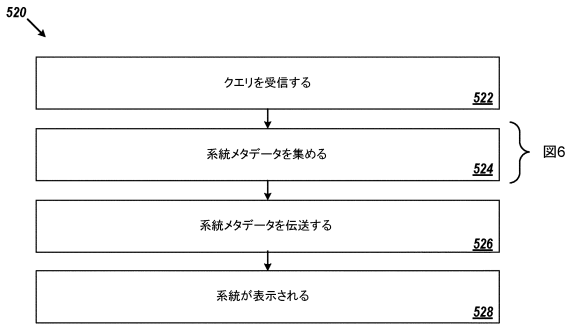
【図 4】



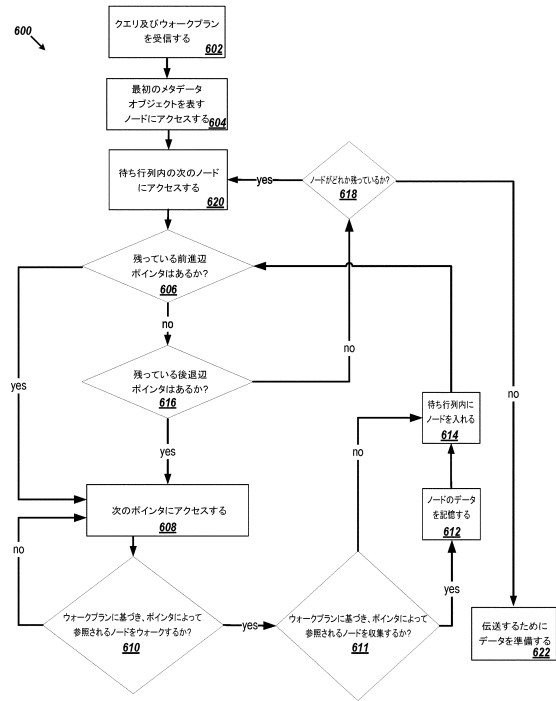
【図 5 A】



【 図 5 B 】



【 図 6 】



10

20

30

40

50

フロントページの続き

米国(US)

早期審査対象出願

0 2 1 4 0 , ケンブリッジ , ミドルセックス ストリート 1 8

(72)発明者 ラディヴォイエヴィック , ドウサン

アメリカ合衆国 , マサチューセッツ州 0 1 8 4 5 , ノース アンドーバー , ウェイランド サークル 1 4 0

(72)発明者 ガラルノー , ネイル

アメリカ合衆国 , マサチューセッツ州 0 2 1 3 2 , ウエスト ロクスベリー , シュルーズベリーロード 2

合議体

審判長 須田 勝巳

審判官 林 毅

審判官 篠原 功一

(56)参考文献 米国特許出願公開第 2 0 0 9 / 0 2 1 6 7 2 8 (U S , A 1)

特表 2 0 1 1 - 5 1 7 3 5 2 (J P , A)

(58)調査した分野 (Int.Cl. , D B 名)

G06F 16/38