(19)日本国特許庁(JP)

(12)特許公報(B2)

(11)特許番号
特許第7079091号
(P7079091)

(45)発行日　令和4年6月1日(2022.6.1)

(24)登録日　令和4年5月24日(2022.5.24)

(51)国際特許分類
Ｇ０６Ｆ　9/48 (2006.01)

ＦＩ
Ｇ０６Ｆ　9/48　３００Ｊ

請求項の数　5　（全30頁）

(21)出願番号	特願2017-532738(P2017-532738)
(86)(22)出願日	平成27年12月7日(2015.12.7)
(65)公表番号	特表2018-501566(P2018-501566A)
(43)公表日	平成30年1月18日(2018.1.18)
(86)国際出願番号	PCT/US2015/064163
(87)国際公開番号	WO2016/099965
(87)国際公開日	平成28年6月23日(2016.6.23)
審査請求日	平成30年12月7日(2018.12.7)
審判番号	不服2021-670(P2021-670/J1)
審判請求日	令和3年1月18日(2021.1.18)
(31)優先権主張番号	14/572,745
(32)優先日	平成26年12月16日(2014.12.16)
(33)優先権主張国・地域又は機関	米国(US)

(73)特許権者　314015767
マイクロソフト テクノロジー ライセンシング，エルエルシー
アメリカ合衆国 ワシントン州 ９８０５２ レッドモンド ワン マイクロソフト ウェイ
(74)代理人　100118902
弁理士　山本 修
(74)代理人　100106208
弁理士　宮前 徹
(74)代理人　100196508
弁理士　松尾 淳一
(74)代理人　100173565
弁理士　末松 亮太
(72)発明者　ピーコック，アンドリュー・ジェイ

最終頁に続く

(54)【発明の名称】　ジョブ・スケジューリングおよび監視

(57)【特許請求の範囲】
【請求項１】
方法であって、
以下のアクトを実行するためにメモリに格納されたコンピュータ実行可能命令を実行するように構成された少なくとも１つのプロセッサを使用するステップを含み、前記アクトが、
グラフィカル・ユーザ・インタフェースを介してデータ変換ジョブの選択を検出するアクトと、
前記ジョブの選択を検出したことに応答して、ジョブ間におけるデータ依存性の分析に基づいて１つ以上の関連ジョブを自動的に判定するアクトであって、
前記データ依存性の分析に基づく前記１つ以上の関連ジョブが、選択されたジョブに依存して生成されたデータに関連付けられる１つ以上のジョブと、前記選択されたジョブが依存するデータに関連付けられる１つ以上のジョブとを含み、
前記データ依存性が、ジョブを頂点とし、データ集合を、前記ジョブを接続する有向エッジとする有向グラフとして表現され、
前記選択されたジョブに依存するジョブを識別するために、前記選択されたジョブから順方向に分析が進み、また、前記選択されたジョブが依存するジョブを識別するために、逆方向に分析が進む、アクトと、
前記選択されたジョブおよび前記１つ以上の関連ジョブを、前記グラフィカル・ユーザ・インタフェース上に表示された他のジョブとは視覚的に区別して提示するアクトと、
実行したジョブおよび実行が成功すると予測されたジョブを、失敗したジョブまたは失

敗すると予測されたジョブから差別するために、実行ステータスを取り込むアクトと、
選択された失敗ジョブ・ランおよび１つ以上の関連失敗ジョブ・ラン、または、前記選択失敗ジョブ・ランを識別する信号を受け取った後に失敗ジョブ・ランに対する依存性に基づいて失敗すると予測されたランを強調するアクトと、
を含む、方法。
【請求項２】
請求項１記載の方法であって、更に、
ジョブ実行と併せてコンピュータ・リソース利用度を監視するアクトと、
１つ以上のそれぞれのジョブと整列して、前記リソース利用度の可視化を提示するアクトと、
を含む、方法。
【請求項３】
請求項１または２記載の方法であって、更に、
ジョブ・ランの実行が成功かまたは失敗かを判定するアクトと、
失敗した実行ランを、成功した実行ランとは異なって提示するアクトと、
を含む、方法。
【請求項４】
請求項１から３の何れか一項記載の方法であって、更に、正常な実行に失敗したジョブ・ランに関して受け取った信号に基づいて、ジョブ・ランの実行を再スケジューリングするアクトを含む、方法。
【請求項５】
請求項１から４の何れか一項記載の方法であって、更に、ジョブとデータ集合、ならびに前記ジョブとデータ集合の間の接続の表現を含む図において、前記選択されたジョブおよび前記１つ以上の関連ジョブを提示するアクトを含む、方法。
【発明の詳細な説明】
【背景技術】
【０００１】
[0001] 貴重な知見を収集するための膨大な量のデータ、即ち、いわゆるビッグ・データの処理は、最初にデータを変換しなければならない。データは、１つ以上のジョブの作成、スケジューリング、および実行によって、ビジネス・インテリジェンス・エンドポイントによる公開または消費のために、ダッシュボードのような使用可能な形態に変換される。このコンテキストでは、ジョブとは１つ以上の変換処理を含むデータに対する作業の単位である。通例、ジョブは、データ開発者、データ・アーキテクト、ビジネス・インテリジェンス・アーキテクト等によって手作業でコード化される。続いて、ジョブをスケジューリングし実行することができる。
【発明の概要】
【課題を解決するための手段】
【０００２】
[0002] 以下に紹介するのは、開示する主題の内いくつかの態様の基本的な理解を得るための簡略化した摘要である。この摘要は広範な全体像ではない。これは、主要な／必須のエレメントを特定することや、特許請求する主題の範囲を明確に定めることを意図するのでもない。その唯一の目的は、後に紹介する更に詳細な説明に対する序文として、いくつかの概念を単純化した形態で紹介することである。
【０００３】
[0003] 端的に説明すると、本開示はジョブ・スケジューリングおよび監視に関する。実行のためにジョブをスケジューリングすることができ、ユーザがジョブ・スケジュールおよび実行を視認して対話処理することを可能にする可視化を生成する(produce)ことができる。１つの態様によれば、ジョブの選択に応答して、１つ以上の関連ジョブを、データ依存性に基づいて自動的に決定することができる。続いて、選択されたジョブ、および関連ジョブを強調することができる。また、例えば、ジョブの成功および失敗した実行を区

別できるように、実行ステータスも判定し提示することができる。更に、失敗したジョブ・ランの選択に応答して、１つ以上の関連失敗ジョブ・ラン、または失敗すると予測されたジョブ・ランを識別することができる。続いて、選択された失敗ジョブ・ラン、および関連失敗ジョブ・ラン、または失敗すると予測されたジョブ・ランを強調することができる。

【０００４】

[0004] 以上のおよび関係する目的に付随して、本明細書では、特許請求する主題のある種の例示的な態様について、以下の説明および添付図面に関係付けて説明する。これらの態様は、主題を実施できる種々の方法を示し、その全ては特許請求する主題の範囲内に該当することを意図している。他の利点および新規な特徴は、以下の詳細な説明を図面と合わせて検討することから明白になるであろう。

【図面の簡単な説明】

【０００５】

【図１】図１は、ジョブ・システムのブロック図である。

【図２】図２は、代表的なユーザ・インタフェース・コンポーネントのブロック図である。

【図３】図３は、代表的なスケジュール・コンポーネントのブロック図である。

【図４】図４は、代表的なスケジューラ・コンポーネントのブロック図である。

【図５】図５は、代表的な監視コンポーネントのブロック図である。

【図６】図６は、ジョブ先導（job driven)スケジューリングおよび監視のためのインタフェースの例証的なスクリーンショットである。

【図７】図７は、関連ジョブを区別するインタフェースの例証的なスクリーンショットである。

【図８】図８は、関連ジョブ失敗を区別するインタフェースの例証的なスクリーンショットである。

【図９】図９は、図解ビュー(diagram view)を生成し関連するジョブおよびデータを区別するインタフェースの例証的なスクリーンショットである。

【図１０】図１０は、データ先導スケジューリングおよび監視のためのインタフェースの例証的なスクリーンショットである。

【図１１】図１１はジョブ・システムのブロック図である。

【図１２】図１２は、ジョブをスケジューリングおよび監視する方法のフロー・チャート図である。

【図１３】図１３は、関連ジョブを区別する方法のフロー・チャート図である。

【図１４】図１４は、成功および失敗のジョブ実行を区別する方法のフロー・チャート図である。

【図１５】図１５は、スケジューリング・チェーンのトラブルシューティングを容易にする方法のフロー・チャート図である。

【図１６】図１６は、関連ジョブおよびデータ集合を区別するフロー・チャート図である。

【図１７】図１７は、ジョブのデータ先導(data driven)処理方法のフロー・チャート図である。

【図１８】図１８は、本開示の態様に適した動作環境を示す模式ブロック図である。

【発明を実施するための形態】

【０００６】

[0023] 以下の詳細は、全体的に、ジョブ・スケジューリングおよび監視に関する。ジョブは、少なくとも１つのデータ変換処理に対応する。１つ以上のジョブをディスプレイ上のインタフェースにおいて、例えば、ジョブ実行のスケジューリングおよび監視と併せて提示することができる。例えば、ジョブをタイムライン・ビュー上で提示し、いつジョブが実行されることになっているか、いつジョブが実行されたか、そしてジョブに対する実行時間の長さを示すことができる。ジョブを選択することができ、その後ジョブ間におけるデータ依存性に基づいて関連ジョブを自動的に識別し、ユーザに提示する。関連ジョブは、選択されたジョブに依存するジョブ、および／または選択されたジョブが依存するジ

ョブを含むことができる。これは、ユーザがジョブ・スケジューリングに関する情報を効率的に取得し対話処理するのを補助する。データ依存性情報に加えて、実行ステータスも取り込み、実行したジョブおよび実行が成功すると予測されたジョブを、失敗したジョブまたは失敗すると予測されたジョブから差別(differentiate)することができる。更に、失敗したジョブ・ラン(job run)を選択することができ、その後、失敗した関連ジョブ・ラン、または失敗すると予測された関連ジョブ・ランを、データ依存性、ステータス、およびスケジュール情報に基づいて自動的に識別することができる。続いて、選択された失敗ジョブ・ラン、関連失敗ジョブ・ラン、または失敗すると予測されたジョブ・ランを強調することができる。その結果、スケジューリング・チェーンをトラブルシューティングする便利な方法が得られる。更に、追加のビューも少なくとも依存性データを活用することができる。例えば、ジョブまたはジョブ・パイプラインの図が依存性情報を使用して、選択ジョブによって利用されるジョブおよびデータ集合を含むジョブ系統、ならびにその選択ジョブに依存するジョブおよびデータ・ソースを識別することができる。これらおよびその他の態様は、少なくとも、ユーザが効率的にジョブ実行をスケジューリングおよび監視するときに補助し、更にエラーを低減する。

【０００７】

[0024] 本開示の種々の態様について、これより、添付図面を参照しながら更に詳しく説明する。図面では、同様の符号は全体的に同様のまたは対応するエレメントを指す。しかしながら、図面およびそれに関係する詳細な説明は、特許請求する主題を特定の開示される形態に限定することを意図するのではない。逆に、特許請求する主題の主旨および範囲に該当する全ての修正、均等、および代替物を包含することを意図している。

【０００８】

[0025] 最初に図１を参照すると、ジョブ・システム１００が示されている。ジョブ・システム１００は、データ変換処理(data transformation operation)を含むジョブのスケジューリング、実行、および監視手段を設ける。ジョブ・システムは、ユーザ・インタフェース・コンポーネント１１０、データ・ストア１２０、スケジューラ・コンポーネント１３０、実行コンポーネント１４０、および監視コンポーネント１５０を含む。ユーザ・インタフェース・コンポーネント１１０は、ユーザがジョブを視認し、指定し、制御することを可能にするように構成されている。１つの実例(one instance)では、ユーザ・インタフェース１１０はジョブまたはジョブ・パイプラインを定めるメカニズムを設けるように構成される。この定義の一部として、ジョブが依存する１つ以上のデータ集合、およびジョブによって生成された出力データ集合を含む、１つ以上の関係を指定することができる。データ依存性データを含むジョブをデータ・ストア１２０に保存することができる。データ・ストア１２０はコンピュータ読み取り可能記憶媒体である。スケジューラ・コンポーネント１３０は、ジョブに関して指定されたスケジュール、およびジョブ間の関係を尊重して、実行のためにジョブをスケジューリングするように構成されている。スケジュールは、表示のためにユーザ・インタフェースに供給することができ、ユーザはこのスケジュールと対話処理して、特定の情報を取得し、必要に応じて、スケジュールを修正することができる。スケジューラ・コンポーネント１３０は、実行コンポーネント１４０によるジョブ処理(job processing)を開始する。実行コンポーネント１４０は、ジョブをコンピュータ上で実行することを可能にするソフトウェアおよびハードウェア・リソースの集合体を含むことができる。ジョブ実行の結果は、データ・ストア１２０に格納することができる。更に、監視コンポーネント１５０は、実行コンポーネント１４０によるジョブ実行を監視することができる。例えば、監視コンポーネント１５０は、実行が失敗したかまたは成功であったかというようなジョブ・ステータスを識別することができる。加えて、監視コンポーネント１５０は、ジョブを処理することに関するコンピュータ・リソース利用度を取得することができる。監視コンポーネント１５０によって取得されたデータおよび情報は、データ・ストア１２０に格納し、提示のためにユーザ・インタフェース・コンポーネント１１０に利用可能にすることができる。

【０００９】

[0026] 図２は、代表的なユーザ・インタフェース・コンポーネント１１０を更に詳しく示す。ユーザ・インタフェース・コンポーネント１１０は、オーサ・コンポーネント２１０とスケジュール・コンポーネント２２０とを含む。オーサ・コンポーネント２１０は、ユーザが、ジョブと、１組の１つ以上の関連ジョブを含むパイプラインとを指定することを可能にするように構成され、第１ジョブの出力が、必要に応じて、第２ジョブに入力を供給する。実施形態によれば、オーサ・コンポーネント２１０は、図式的にジョブおよびパイプラインをオーサリングするためのインタラクティブ視覚作業空間またはキャンバスを提供することができる。例えば、データ集合を円筒形で表し、そのデータ集合を消費して修正データ集合を生成するジョブを表す立体に、矢印によって接続することができる。本質的に、ユーザはデータ集合とジョブとの間の関係の図を描くことができる。この結果、関係を理解し最終的にパイプラインを指定することに関して時間を節約する直観的な体験が得られる。他の態様によれば、オーサ・コンポーネント２１０をコード・エディタとして具体化することができ、プログラム・コードまたは１つ以上のダイアログ・ボックスを受け入れて、ジョブとジョブ間の関係とを指定する。スケジュール・コンポーネント２２０は、実行のためにジョブをスケジューリングし実行を監視することに関して、可視化を提示するように構成されている。

【００１０】

[0027] 図３に注意を向けると、明確化および理解を容易にするために、代表的なスケジュール・コンポーネント２２０が更に詳細に示されている。スケジュール・コンポーネント２２０は、ビュー・コンポーネント３１０、更新コンポーネント３２０、関連ジョブ・コンポーネント３３０、修正コンポーネント３４０、およびログ・コンポーネント３５０を含む。ビュー・コンポーネント３１０は、実行のためにスケジューリングされたジョブの内、少なくとも１つのビューを提示するように構成されている。１つの態様によれば、ビューは、実行時間によって順序付けられ、スケジューリングされた実行および完了した実行を含む、１組のジョブを可視化するタイムラインに対応することができる。また、このようなタイムラインは、ジョブ実行の長さを可視化することも可能にすることができる。

【００１１】

[0028] 更新コンポーネント３２０は、ビュー・コンポ３１０によって生成された可視化を更新する、またはジョブ実行に関して最新にするように構成されている。例えば、更新コンポーネント３２０は、実行され終えたジョブを反映するように、可視化を変更することができる。一実施形態では、更新コンポーネント３２０はビュー・コンポーネント３１０と相互作用して、現在の時刻を表すラインを含ませ、スケジューリングされているが実行されていないジョブをグレーに着色する、またはラインの後ろで操作対象から外す(gray out)ことができる。加えて、更新コンポーネント３２０は、実行に成功したジョブ、および失敗したジョブまたは失敗することが予測できるジョブを識別し、区別するために利用することができる。例えば、実行に成功したジョブを緑に着色することができ、一方失敗は赤でその意味を表す(signify)ことができる。

【００１２】

[0029] 特定の実施形態によれば、ビュー・コンポーネント３１０および更新コンポーネント３２０が協働して、タイムラインと、コンピュータ・リソース利用度の可視化とを含む分割ビューを提示することができる。第１部分では、スケジューリングされたジョブのタイムラインを、以上で説明したように提示することができる。第２部分では、例えば、第１部分の下に、グラフのような可視化を提示し、ジョブ実行と併せて利用されたリソースを表すことができる。この実例では、更新コンポーネント３２０は、リソース利用度およびジョブ実行に関するデータを取得し、それに応じてグラフを更新することができる。

【００１３】

[0030] 関連ジョブ・コンポーネント３３０は、ジョブ間の関係に基づいてジョブを区別するように構成されている。関係は、ジョブと、選択ジョブ(select job)が依存するデータ、または選択ジョブに依存するデータとを含むことができる。１つの実例では、選択されたジョブ(selected job)は、この選択されたジョブに関連するジョブを判定し、視覚的

にこれらの関連ジョブを、ビュー・コンポーネント３１０によって行われる視覚化において区別し始めることができる。一例として、ジョブを選択する信号、または可視化におけるその表現を受け取った後、選択されたジョブに関連するジョブを、データ依存度に基づいて判定し、強調することができる。関連ジョブは、ジョブ間の関係を保存するデータ・ストア１２０から識別することができる。一実施形態によれば、ジョブを頂点として含み、データ集合を、ジョブ同士を接続する有向エッジとして含む有向グラフを保存することができる。関連ジョブを判定するために、このグラフを前方および後方に、選択されたジョブを表す頂点から横断することができ、前方への横断（例えば、選択されたジョブから下流側）で、ジョブ、および選択されたジョブに依存するデータ集合を取り込み、後方の横断（選択されたジョブから上流側）で、ジョブ、および選択されたジョブが依存するデータ集合を識別する。ジョブ・ラン(job run)、即ち、ジョブのインスタンスも、特性(characteristic)またはプロパティ(property)に基づいて区別することができる。例えば、ユーザが、実行を成功させるために、失敗したジョブ・ランを選択した場合、この失敗したまたは失敗すると予測された、選択されたジョブ・ランに関連する他のジョブ・ランを自動的に識別し、ビューにおいて強調することができる。このようなジョブ・ランは、ジョブ間のデータ依存性、ジョブに関連付けられたステータス（例えば、失敗、成功、．．．）、および実行スケジュールに関する情報に基づいて識別することができる。ここで、ジョブ・ランが失敗すると予測することができるのは、それが、正常な実行に失敗した(that has failed to execute successfully)ジョブ・ランに依存する場合である。

【００１４】

[0031] 修正コンポーネント３４０は、ジョブ・スケジューリングの修正を可能にするように構成されている。修正コンポーネント３４０は、ビュー・コンポーネント３１０と一緒に動作して、ジョブ実行スケジューリングの修正に関するユーザ入力を取得するように構成されている。例えば、修正コンポーネント３４０は、提示されているインタラクティブ・ビューに関してユーザから信号として受け取った、１つ以上のジェスチャに基づいて、実行のためにジョブの再スケジューリングを少なくとも開始することができる。非限定的な一例によれば、ユーザは、正常な実行に失敗したジョブのインスタンスを選択すること、および／または実行を再スケジューリングするために、右クリックまたはドラッグおよびドロップのような、何らかの追加のジェスチャを実行することができる。

【００１５】

[0032] ログ・コンポーネント３５０は、実行ログの取得および表示を可能にするように構成されている。一実施形態によれば、ログ・コンポーネント３５０は、重ね合わせ検索ペインまたはパネルによって、検索メカニズムを設けることができる。このシナリオでは、ユーザは検索を指定および提出し、実行ログに関する結果を受け取ることができる。他の実施形態によれば、ログ・コンポーネント３５０は例えば、特定のジョブ・ランの選択に基づいて、クエリーを自動的に生成および提出することができる。ログ・ファイル取得に関連付けられた特定のジェスチャに応答して、選択されたジョブ・ランに対応する結果を戻すことができる。このように、ユーザは失敗の通知から素早く、例えば、出発点(point of origin)に達することができる。あるイベントでは、実行ログをアクセス可能にすることにより、失敗の解明のようなトラブルシューティングが容易になる。

【００１６】

[0033] 図４は、特定の一実施態様による代表的なスケジューラ・コンポーネント１３０を示す。スケジューラ・コンポーネント１３０は、出力スライス選択コンポーネント４１０、依存性－期間判定コンポーネント４２０および依存性評価コンポーネント４３０、ならびに実行開始コンポーネント４４０を含む。このコンテキストでは、データ集合は、データ処理動作を取り込む(capture)ジョブまたはアクティビティによって生成または消費され、時間軸上におけるデータの集合体に対応する。具体的には、データ集合は、データ・スライスと呼ばれるデータ片(pieces)を時間期間と相関付ける。別の言い方をすると、データ集合は、データ・スライスの連続体で構成される。各データ・スライスは、特定のスライス長、ならびに開始時刻および終了時刻を有することができる。

【００１７】
[0034] 出力スライス選択コンポーネント４１０は、生成するデータ・スライスを決定するように構成されている。この決定は、アクティビティ期間におけるステータス、および動作ポリシーに基づく。各データ・スライスは、対応するデータの可用性を示すステータスを有することができる。ステータスは、多数の値を取ることができ、データが未だ生成されていないことを意味する「実行保留」(pending execution)、スライスが生成されつつあることを示す「進行中」、データを消費する準備ができていることを示す「準備完了」、およびデータを生成する１つ以上の試みが失敗したことを意味する「実行失敗」(failed execution)を含む。出力スライス選択コンポーネント４１０は、ジョブがその出力を生成する時間枠を指定するアクティブ期間内における「実行保留」ステータスのデータ・スライスを識別する。これらのデータ・スライスは、ステータスおよびアクティブ期間によって選別された(filter)データ・スライスに対するクエリーを実行することによって識別することができる。更に、出力スライス選択コンポーネント４１０は、更に、ポリシーに基づいて、スライスの実行、またはデータ・スライスの生成を指令することができる。実行保留ステータスのスライスを実際の経過時間(wall clock time)と比較し、何らかの順序で取り出すことができる。例えば、ポリシーは、例えば、いつステータスが「実行保留」に設定されたかに基づいて、最も古いスライスを最初に生成することまたは最も新しいスライスを最初に生成することを指示することができる。最終的に、出力スライス選択コンポーネント４１０は、生成する１つのスライス、出力スライスを識別する。
【００１８】
[0035] 依存性－期間判定コンポーネント４２０は、依存期間を判定するように構成されている。依存性期間とは、識別された出力スライスを生成しなければならない入力データのデータ時間範囲である。依存性期間は、ジョブに関して定められた依存性情報の一部である。例えば、ジョブが、第１ソースからの３時間にわたるデータに対して動作し、次いで第２ソースからの１時間にわたるデータに対して動作するようにジョブを指定することができる。したがって、全てのデータ集合が時間毎の(hourly)スケジュールを有する場合（例えば、スライスが１時間である）、第１ソースからの３時間のデータおよび第２ソースからの１時間のデータは、１時間の出力スライス・データを生成することが要求される。
【００１９】
[0036] 依存性評価コンポーネント４３０は、出力スライスの依存性が満たされているか否か判定するように構成されている。依存性評価コンポーネント４３０は、直前に判定された依存性期間内において入力データ・スライスを識別することができる。更に、各入力スライスのステータスを取得し、各スライスのステータスが「準備完了」であるか否か判定を行う。これが意味するのは、スライスが消費の準備ができているということである（例えば、スライスの生成に成功した、または外部に利用可能にされた）。依存性期間内の入力スライスが「準備完了」ステータスである場合、依存性が満たされていることになる。そうでない場合、依存性は満たされていない。
【００２０】
[0037] 実行開始コンポーネント４４０は、一旦依存性状態が満たされたなら、出力スライスを生成するためにジョブの実行を開始するように構成されている。言い換えると、実行コンポーネント１４０上でアクティビティ実行がトリガされ、選択された出力データ・スライスの生成が開始する。実行が開始された後、出力データ・スライスのステータスは「実行保留」から「進行中」に変わる。最終出力スライスの生成に成功した場合、出力ステータスを「準備完了」に設定することができる。これが意味するのは、スライスをその入力として消費する下流アクティビティがこの時点でこのスライスを使用できるということである。実行が失敗した場合、判定ポリシーに基づく回数だけ再試行することができる。この時間中、ステータスを「再指向」に設定することができ、これは直前の失敗および実行の再試行を意味する。続いて実行が成功しなかった場合、ステータスを「実行失敗」に設定することができる。更に、各実行をラン・レコード(run record)に記録することができる。ラン・レコードは、出力データ・スライスと関連付けられる。

【００２１】
[0038] スケジューラ・コンポーネント１３０の動作に関して更に明確化および理解を促進するために、いくつかの例証的なシナリオについて説明する。最初に、データ変換処理を含むジョブが１入力および１出力を有するというシナリオについて検討する。ここでは、実行は単純である。時間が経過するに連れて、現在のラン時刻前にステータスが「実行保留」であったデータ・スライスを取り上げ、依存性期間が準備完了である場合このデータ・スライスを生成する。次に、ジョブが１入力および複数の出力に関して処理する(operate)シナリオについて検討する。この場合、全ての出力からのデータ・スライスの連合体が、生成されるスライスのプールとして使用される。出力データ・スライスに対するステータスの変化は、全ての出力データ集合に起こる。実行の残りの部分は、直前の場合と変わらない。次に、複数の入力および複数の出力があるジョブについて検討する。この場合も、全ての出力からのデータ・スライスの連合体が、生成されるスライスのプールとして使用される。「実行保留」出力スライスが生成されるために、依存性期間を判定し、アクティブ期間内における全てのスライスが「準備完了」ステータスを有する場合、このジョブを実行して出力スライスを求める。出力データ・スライスに対するステータスの変化は、全ての出力データ集合に起こり、実行は最初のシナリオと同じである。

【００２２】
[0039] 図５は、代表的な監視コンポーネント１５０を更に詳しく図示する。監視コンポーネント１５０は、ステータス・コンポーネント５１０と実行記録コンポーネント５２０とを含む。ステータス・コンポーネント５１０は、ジョブのステータス、および／またはこれらが生成するデータを監視する。例えば、ステータス・コンポーネントは、ジョブ実行による出力データの生成が成功したか、またはジョブ実行が出力データを生成するのを失敗したか監視することができる。既に注記したように、実行の失敗または成功を、ユーザ・インタフェースを介して、ユーザに提示することができる。また、ステータス・コンポーネント５１０は、とりわけ、データを生成したジョブが、実行保留、進行中、または消費のために準備完了であるときを含む追加のステータス情報も監視することができる。実行記録コンポーネント５２０は、ジョブ実行中におけるコンピュータ・リソース利用度に関するデータを取得するように構成されている。１つの実例では、この情報は、オペレーティング・システムまたは同様の制御メカニズムによって要求され、取得することができる。リソース利用度は、その後、ユーザ・インタフェースによって、タイムラインと、このタイムラインと整列されたデータ利用度とを含む分割ビューを提示するために利用することができる。

【００２３】
[0040] 図６から図１０は、ジョブ・スケジューリングおよび監視に関してユーザ・インタフェース・コンポーネント１１０によって生成される種々の可視化を示す例証的なスクリーンショットである。これらのスクリーンショットは、本開示の態様に関する明確化および理解を補助することを意図するのであり、特許請求する主題をこれらに限定することを意図するのではない。尚、提示するスクリーンショットは一実施態様を図示するに過ぎないことは認められよう。グラフィック・エレメントおよびテキストの種々の他の組み合わせおよび配列も可能であると考えられ、添付する請求項の範囲に該当することを意図している。更に、ジョブ・スケジューリングに関するユーザの理解を補助するために、可視化と併せて種々の音響も採用できることは理解されよう。一例として、そして限定ではなく、ジョブまたはデータの選択時に、あるいは実行失敗の検出時に、音を鳴らすことができる。

【００２４】
[0041] 図６は、ユーザ・インタフェース・コンポーネント１１０によって生成することができるインタフェース６００のスクリーンショットである。図示のように、インタフェース６００は、３つのパネル、即ち、ソース・パネル６１０、公開パネル(published panel)６２０、およびスケジュール・パネル６３０を含む。ソース・パネル６１０は、複数の利用可能なデータ集合を提示し、そこからソースを追加または削除することができる。

尚、ソース・パネル６１０内に図示されるデータ集合は任意のデータ・ソースが可能であることは認められてしかるべきである。例えば、あるデータ集合をオンプレミス・データと関連付けることができ、一方他のデータ・ソースをネットワークまたはクラウド・データ・ストアと関連付けることができる。更に、データ集合は実質的にあらゆる構造またはフォーマットにすることができる。公開パネル６２０は、所望の変換が実行された後に、公開されたデータ・ソースまたは消費可能なデータ・ソースの視覚表現を示す(provide)。

【００２５】

[0042] スケジュール・パネル６３０は、ジョブ実行のスケジュール、および監視対象の実行結果(monitored results of execution)を可視化する。更に具体的には、スケジュール・パネル６３０は、ジョブの実行開始および終了時刻を含むガン・チャートとして表示されるタイムライン・ビュー６４０と、ジョブ実行と整列してリソース消費を表す線グラフを含むリソース利用度ビュー６５０とを含む分割ビューを提示する。このグラフに基づけば、ジョブ・スケジューリングを決定するのは容易である。例えば、ここでは、重複除去処理(remove duplicate operation)を実行する第１ジョブは、毎日実行するようにスケジューリングされ、条件付き分割(conditional split)を実行する第２ジョブは１日おきに実行するようにスケジューリングされる。尚、ライン６６０は実行に関して現在の時刻も表すことを注記しておく。このラインよりも前にあるジョブは、既に実行されたジョブであり、このライン上(over)にあるジョブは、今後のある時点における実行のためにスケジューリングされている。この区別を強調するために、スケジューリングされているが未だ実行されていない処理(operation)はグレーで図示されており、言い換えると、これらの処理は選択対象から除外されている。一旦これらが実行されると、その処理はもはや選択対象から除外されない。更に、ジョブ・ランに関する色の違いが、追加の情報を表すことができる。ここでは、例えば、黒く着色されたジョブは失敗した実行を示す。１つの態様によれば、ユーザは、黒く着色されたジョブ・ランを選択し、このジョブの実行を再スケジューリングすることができる。

【００２６】

[0043] 図７は、ユーザ・インタフェース・コンポーネント１１０によって生成することができるインタフェース７００のスクリーンショットである。図６のインタフェース６００と同様、インタフェース７００は、前述のように、ソース・パネル６１０、公開パネル６２０、およびタイムライン・ビュー６４０とリソース利用度ビュー６５０とを含むスケジュール・パネル６３０を含む。しかしながら、インタフェース７００は、ジョブの選択、および選択に関連するジョブの強調を示す。ここでは、第２ジョブ７１０は、条件付き分割を実行し、例えば、クリック、タッチ、または他のジェスチャによってユーザによって選択される。ジョブが選択された後、この選択されたジョブに関連するジョブが自動的に識別され、視覚的に他のジョブと異なって提示される。この例では、重複除去処理を実行する第１ジョブ７１２、販売データの消去(cleansing)を実行する第５ジョブ７１４、連合を実行する第７ジョブ７１６、およびソート処理を実行する第８ジョブ７１８が関連ジョブとして識別され、他のジョブに関して強調されている。具体的には、選択されたジョブ、および関連ジョブは白い背景と共に提示され、一方他の全ての関連のないジョブは選択対象から除外されている。関連ジョブは、選択されたジョブに依存するジョブ、および選択されたジョブが依存するジョブを含む。ここでは、第５ジョブ７１４、第７ジョブ７１６、および第８ジョブ７１８が、選択されたジョブに依存し、一方選択されたジョブは第１ジョブ７１２に依存する。これらの依存性は、ジョブの位置に基づいて識別することができ、選択されたジョブの後に提示されるジョブは、選択されたジョブに依存し、一方選択されたジョブの前に位置するジョブは、選択されたジョブが依存するジョブである。

【００２７】

[0044] 図８は、ジョブ・ラン失敗の選択に応答してユーザ・インタフェース・コンポーネント１１０によって提示することができるインタフェース８００のスクリーンショットである。図７および図８のスクリーンショットと同様、インタフェース８００は、既に論じたように、ソース・パネル６１０、公開パネル６２０、およびタイムライン・ビュー６

４０とリソース利用度ビュー６５０とを含むスケジュール・パネル６３０を含む。また、インタフェース８００は、正常な実行に失敗したジョブの特定のランの選択も示す。これは、べた塗りの黒で表現されている(capture)。ここでは、条件付き分割ジョブのランを８１０において示す。失敗したジョブ・ランを選択すると、この選択されたジョブ・ランが失敗した原因になった可能性がある他の関連失敗ジョブ・ラン、あるいは失敗した可能性があるまたは失敗すると予測された他のジョブ・ランを、選択されたジョブ・ランの失敗に基づいて識別する動作がトリガされる。これらの関連ジョブは、ジョブ間において記録された依存性、実行ステータス（例えば、失敗、成功）、およびスケジュールに基づいて識別することができる。８２０に示すように、依存性、ステータス、スケジュール情報の分析時に、８１０において記された、選択された「条件付き分割」ジョブ・ランの失敗のあり得る原因として、重複除去ジョブのランを識別することができる。更に、８３０に示すように、「販売変換消去」ジョブのランの失敗も、「条件付き分割」ジョブの失敗の結果として予測することができる。更にまた、８４０に記すように、「連合」ジョブのランの失敗は、「条件付き分割」ジョブおよび「販売変換消去」ジョブの一方または双方の失敗に基づいて予測することができる。同様に、８５０に示すように、「ソート」ジョブの失敗は、「条件付き分割」ジョブ、「データ変換消去」ジョブ、または「連合」ジョブの失敗に応じて予測することができる。インタフェース８００は、選択された失敗ジョブ・ランおよび関連する失敗ジョブ・ランを、他のジョブ・ランから区別する。言い換えると、選択されたおよび関連失敗ジョブ・ランは強調される。ここでは、選択された失敗ジョブ・ランおよび関連失敗ジョブ・ラン以外の全てのジョブ・ランは、選択対象から除外される。勿論、とりわけ、逆、異なる色、異なる字体、異なるサイズも、差別化の目的で利用することができる。この便利なメカニズムは、ユーザがスケジューリング・チェーンをトラブルシュートするのを助ける。更にその上、失敗ジョブ・ランの１つに関する選択または他のジェスチャのときに、ジョブ実行に関連する実行ログまたはログ・ファイルの検索を可能にすることができるダイアログ検索ペインを提示することができる（図示せず）。このように、ユーザは素早く失敗の識別およびログ・ファイル内の開始点(point of origin)から移行することができる。更に、ユーザは、ジョブの失敗したランおよび実行の再スケジュールに関して、選択またはジェスチャを行うことができる。例えば、ユーザは、実行を再スケジュールするために、失敗ジョブ・ランをドラッグし他の時点にドロップすることができる。他の例として、ユーザは失敗ジョブ・ラン上で右クリックして、ユーザがジョブを再スケジューリングすることを可能にするダイアログ・ボックスを表示させる(bring up)ことができる。

【００２８】

[0045] 図９は、ユーザ・インタフェース・コンポーネント１１０によって生成することができるインタフェース９００のスクリーンショットを示す。以前のスクリーンショットと同様、インタフェース９００は、複数の利用可能なデータ集合を提示しそこからソースを追加または削除することを可能にするソース・パネル６１０と、所望の変換が実行された後に、公開または消費可能なデータ集合の視覚表現を示す公開パネル６２０とを含む。更に、インタフェース９００は作業空間パネル９１０も含む。作業空間パネル９１０は、図によって、ジョブおよびパイプラインの視覚オーサリング(visual authoring)を可能にする。例えば、ユーザは、ソース・パネル６１０からデータ集合の視覚表現をドラッグおよびドロップすることによって、データ集合を取得することができる。次に、例えば、データ集合表現から、ジョブの立体表現まで矢印を描くことによって、このデータ集合を、以前にオーサリングしたジョブ（例えば、データ・プレビューによって自動的に作成され、および／または手作業でコード化される）に接続し、このデータ集合が入力を供給し、ジョブがこの入力を消費して１つ以上のデータ変換処理(operation)（例えば、ソート、グループ化、旋回、分割、フィルタリング．．．）を実行することを示すことができる。加えて、変換された出力の表現をジョブの表現に、作業空間上でリンクすることができる。その結果、データ・ソースから入力を受け取り、ジョブの１つ以上の変換処理(operation)の適用(application)を反映する新たなデータ・ソースを出力するジョブの図が表示さ

れる。インタフェース９００は、このジョブ・オーサリング用の図解ビュー(diagram view)と、ジョブ・スケジューリングおよび監視用のタイムライン・ビューとの間における移行を可能にする。更に、この図解ビュー内において、スケジューリングおよび監視データを作業空間パネル９１０に提示することができる。９２０に示すように、「ソート」ジョブの表現は、スケジューリング・データとリソース利用度のグラフとを含む分割ビューを含む。更に、作業空間パネル９１０内に提示されたジョブ表現、またはタイムラインに関して以前に提示されたジョブ表現を選択する信号を受け取った後、関連ジョブおよびデータ集合を判定し強調する。ここでは、スクリーンショットは、９３０において「条件付き分割」ジョブの選択を表す。選択が受け取られた後、ジョブおよびデータに関して収集された依存性データを利用して、選択されたジョブが依存するジョブおよびデータ、ならびに選択されたジョブに依存するジョブおよびデータを識別することができる。言い換えると、上流および下流ジョブならびにデータを含むジョブ系統を判定し、続いて表示することができる。更に、関連データおよびジョブを、他のデータおよびジョブに関して、視覚的に区別するまたは強調することができる。ここでは、関連ジョブおよびデータを白に着色し、一方、他のデータおよびジョブをグレーに着色する、即ち、選択対象から外す。

【００２９】

[0046] 図１０は、ユーザ・インタフェース・コンポーネント１１０によって生成することができるインタフェース１０００のスクリーンショットを示す。図６のインタフェース６００と同様、インタフェース１０００は、複数の利用可能なデータ集合を提示しそこからソースを追加または削除することを可能にするソース・パネル６１０と、所望の変換が実行された後に、公開または消費可能なデータ集合の視覚表現を示す公開パネル６２０とを含む。また、インタフェース１０００は、タイムライン・ビューとリソース利用度ビュー６５０とを含む分割ビューを含むスケジュール・パネル６３０も含む。しかしながら、ここでは、タイムライン・ビュー１０２０は、ジョブ実行についての処理時間ではなく、データ・スライス時間に関して提示されている。言い換えると、タイムライン・ビュー１０２０は、処理の時間ではなくデータの時間を示す。例えば、イベントに対して時間単位の処理(hourly processing)が実行されることになっているシナリオについて検討する。１時間分の(hour worth of)イベントを取り込むことによって処理が開始し、続いて、何らかの変換処理(operation)を実行する。更に具体的には、午前９時から午前１０時までの１時間の処理は、午前１０時１５分に開始することができる。続いて、午前１１時に何らかの集計を実行することができ、午後１２時に他のデータとの併合が実行されてもよく、午後１時に結果が完成するのでもよい。つまり、タイムライン・ビュー１０２０は、ジョブを実行する(run)のにどの位かかるのかに関するジョブ実行時間ではなく、データの時間、およびある時間期間（例えば、１日、１週間等）にわたってそれがどのように処理されるかについてのビューを表示する(provide)。また、スケジュール・パネル６３０は、データ処理に関する現在の時刻を表すライン６６０も含む。このラインよりも前のデータは、既に生成されたデータを表し、このラインよりも後のデータは、今後のある時点における生成のためにスケジューリングされたデータを表す。この区別を強調するために、スケジューリングされているが未だ生成されていないデータをグレーで表す。即ち、言い換えると、その処理は選択対象から除外される。データが生成された後、このデータはもはや選択対象から外されない。更に、データに関する色の違いが、追加の情報を表すこともできる。例えば、黒く着色されたデータは、関連するデータの生成失敗を示す。１つの態様によれば、ユーザは、黒く着色されたデータを選択し、このデータを生成するための実行を再スケジューリングすることができる。図示しないが、データに関する色または他の視覚的特徴によって、とりわけ、「実行保留」、「準備完了」、「進行中」、および「失敗」のようなデータ・ステータスを含む、 他の情報を伝えることもできる。

【００３０】

[0047] 図１１はジョブ・システム１１００を表す。システム１１００は、データ変換ジョブを含む入力を受け取り、このデータ変換ジョブのビューを生成する手段を設けるビュー生成手段１１１０を含む。ビューの生成には、ハードウェア、ソフトウェア、またはハ

ードウェアおよびソフトウェアの組み合わせを採用することができる。ビューとは、データ変換ジョブを含むまたはこれに関連する可視化である。ビュー生成手段１１１０によって生成されたビューは、提示のためにディスプレイに供給することができる。１つの実例では、ビュー生成手段１１１０は、グラフィカル・ユーザ・インタフェースであること、またはその一部を形成することができる。一実施形態によれば、生成されたビューは、データ変換ジョブ、または１組の１つ以上の関連ジョブを含むジョブ・パイプラインの図であることができ、第１ジョブの出力が、必要に応じて、第２ジョブに入力を供給する。

【００３１】

[0048] 選択手段１１２０は、ビューにおいてジョブの選択を指示する信号を生成するメカニズムである。選択手段１１２０を実装するためには、ハードウェア、ソフトウェア、またはソフトウェアおよびハードウェアの組み合わせを利用することができる。ユーザは、入力メカニズムを使用してジョブを選択するまたそうでなければ特定することができ、選択手段１１２０はこの入力を受け取り、例えば、ユーザ入力の位置をビュー内におけるジョブの位置と比較することによって、選択されたジョブを特定することができる。種々の入力メカニズムがユーザによって使用することができ、タッチ・パッド、マウス、タッチ・スクリーン、カメラ、またはマイクロフォンを含むが、これらに限定されるのではない。

【００３２】

[0049] 関連ジョブ手段１１３０は、選択されたジョブに関連するジョブを自動的に識別するメカニズムを設ける。選択手段１１２０によって選択されたジョブが供給されると、関連ジョブ手段は、ジョブに関する依存性情報を受け取る、引き出す、あるいはそれ以外では入手するまたは取得することができる。続いて、関連ジョブ手段は、例えば、選択されたジョブを調べ、選択されたジョブに関連するジョブを識別することによって、関連ジョブを識別することができる。ここで、関連ジョブとは、選択されたジョブに依存するジョブ、または選択されたジョブが依存するジョブである。１つの態様によれば、依存性は、ジョブが依存する入力データおよびジョブによって生成された出力データを含むジョブ・データ依存性に関して取り込むことができる。一実施形態では、依存性データを有向グラフで表現することができる。有向グラフは、ジョブを頂点として含み、データ集合を、ジョブ同士を接続する有向エッジとして含む。関連ジョブを決定するためには、順方向または逆方向に、選択されたジョブを表す頂点からグラフを横断することができる。ここで、順方向に横断すると（例えば、選択されたジョブから下流側）、選択されたジョブに依存するジョブおよびデータ集合を取り込み、逆方向に横断すると（例えば、選択されたジョブから上流側）、選択されたジョブが依存するジョブおよびデータ集合を識別することになる。関連ジョブ手段１１３０は、関連ジョブ（およびデータ集合）を識別するために、ハードウェア、ソフトウェア、またはハードウェアおよびソフトウェアの組み合わせを、例えば、コンピュータ実行可能ソフトウェア・コンポーネントまたはファームウェアとして採用することができる。

【００３３】

[0050] ビュー更新手段１１４０は、ビューに変更を行うことを可能にするメカニズムである。１つの実例では、ビュー更新手段は、ハードウェア、ソフトウェア、またはハードウェアおよびソフトウェアの組み合わせを含み、選択されたジョブ、１つ以上の関連ジョブ、および必要に応じて、関連データ集合を、関連ジョブ手段１１３０から直接、またはある場所（例えば、メモリ、ストレージ．．．）から間接的に受け取り、引き出し、あるいはそれ以外では入手しまたは取得し、選択されたジョブならびに関連ジョブおよびデータ・ソースを、ビュー内に提示された他のジョブおよびデータ・ソースから視覚的に区別する。言い換えると、ビュー更新手段１１４０は、選択されたジョブならびに関連ジョブおよびデータ・ソースをビュー内において強調することができる。

【００３４】

[0051] スケジューリング手段１１５０は、ジョブ実行に関するスケジューリング機能を設ける。ハードウェア、ソフトウェア、またはハードウェアおよびソフトウェアの組み合

わせとして具体化され、スケジューリング手段１１５０は、１つ以上のジョブを受け取り、引き出し、あるいはそれ以外では入手しまたは取得し、ジョブの定義またはそれに関連しスケジューリング情報を提供する情報にしたがって、ジョブ実行のためのスケジュールを生成することができる。例えば、ジョブが毎日の実行を指令する(dictate)する場合、ジョブまたはジョブ・ランを毎日スケジューリングすることができる。同様に、ジョブが１日おきの実行を指定する場合、ジョブまたはジョブ・ランを１日おきにスケジューリングすることができる。勿論、スケジュールは、ジョブの入力データが利用可能になった後に実行のためにジョブをスケジューリングするように、種々のデータ依存性を考慮することができる。

【００３５】

[0052] スケジューリング手段１１５０は、ビュー生成手段に利用可能なスケジュールを作ることができる。これに応じて、ビュー生成手段は、スケジュールのビューを、スケジューリングされたジョブのタイムラインとして生成することができる。この実例では、選択手段１１２０は、スケジュールのタイムライン・ビューからジョブの選択を可能にすることができる。続いて、関連ジョブ手段は、選択されたジョブに関連するジョブを判定することができ、ビュー更新手段１１４０は、選択されたジョブおよび関連ジョブを強調するために、スケジュールのタイムライン・ビューを更新することができる。

【００３６】

[0053] 実行手段１１６０は、スケジューリング手段１１５０によって供給されたジョブ、またはそれ以外で処理可能(available)にされたジョブを実行する。実行手段１１６０は、ハードウェア、ソフトウェア、またはハードウェアおよびソフトウェアの組み合わせを採用して、１つ以上の入力データ集合に対してデータ変換ジョブを実行し、このジョブによって指定されたデータ変換処理の適用を反映する１つ以上の出力データ集合を生成することができる。実行手段１１６０は、第１指定位置から入力データを読み出し、続いて出力データを第２指定位置に書き込むことができ、これらの位置はメモリまたは記憶デバイスの位置とすることができる。

【００３７】

[0054] 監視手段１１７０は、実行手段１１６０によるジョブの実行を監視するハードウェア、ソフトウェア、またはハードウェアおよびソフトウェアの組み合わせを含むメカニズムである。一実施形態によれば、監視手段１１７０は、ジョブが実行に成功したか、または正常な実行に失敗したか検出する、またそうでなければ判定することができる。これは、実行の成功または失敗に関する潜在的なメッセージ(potential message)を含む、実行手段１１６０の出力を監視することによって行うことができる。１つの実例では、監視手段１１７０は、ジョブの実行がスケジューリングされた通りに開始されたか否か、およびジョブの出力が、実行が成功したかまたは失敗したかに関して生成されたか否か判定することができる。他の実施形態では、監視手段１１７０はジョブ実行に関するリソース利用度を監視することができる。１つの実例では、監視手段１１７０に、プロセッサ、メモリ、ストレージ、およびネットワークの利用度を含むリソースを直接監視するメカニズムを実装することができる。代わりに、監視手段１１７０がこのような情報を要求して、オペレーティング・システム、あるいはリソース利用度を監視する他のシステムまたはコンポーネントから受け取ることもできる。

【００３８】

[0055] 監視手段１１７０は、取得したデータまたは情報を、ビューを生成するときに使用するために、ビュー生成手段１１１０に利用可能にすることができる。１つの実例では、ビュー生成手段１１１０は、スケジューリングされたジョブ・ランが成功したかまたは失敗したか、監視手段１１７０によって提供される情報に基づいて示す、スケジュールのタイムライン・ビューを生成することができる。この場合、選択手段１１２０が失敗した実行ラン(failed execution run)の選択を知らせるメカニズムを設けることができる。続いて、関連ジョブ手段１１３０が関連ジョブ・ランを識別し、更に特定すれば、失敗した関連ジョブ・ラン、または失敗することが予測できる関連ジョブ・ランを、失敗したジョ

ブ・ランに対する依存性に基づいて識別することができる。一実施形態によれば、データ依存性情報に加えて、関連ジョブ手段は、とりわけ、ジョブが実行に成功したかまたは正常な実行に失敗したかを含む処理情報(operation information)を受け取ることができる。１つの実例では、この動作情報を依存性情報と共に格納することができる。あるいは、動作情報を別個に格納しアクセスすることもできる。ビュー更新手段１１４０は、続いて、選択された失敗ジョブ・ラン、および失敗した、または失敗すると予測された関連ジョブ・ランを強調することができる。加えて、ビュー生成手段１１１０は、監視手段１１７０によって提供されたリソース利用度情報の可視化を、ジョブ・ランと整列させて生成することができる。例えば、ビュー生成手段１１１０はリソース利用度を表す線グラフのようなグラフを生成する。

【００３９】

[0056] 前述のシステム、アーキテクチャ、環境等は、様々なコンポーネント間における相互作用に関して説明した。尚、このようなシステムおよびコンポーネントは、本明細書において指定されたコンポーネントまたはサブコンポーネント、指定されたコンポーネントまたはサブコンポーネントの一部、および／または追加のコンポーネントを含むことができることは認められてしかるべきである。また、サブコンポーネントは、親コンポーネント内に含まれるのではなく、通信可能に他のコンポーネントに結合されたコンポーネントとして実装することもできる。更にまた、１つ以上のコンポーネントおよび／またはサブコンポーネントを１つのコンポーネントに組み込んで、総合機能(aggregate functionality)を設けることもできる。システム、コンポーネント、および／またはサブコンポーネント間の通信は、プッシュ・モデルおよび／またはプル・モデルのいずれかにしたがって行うことができる。また、コンポーネントは、簡潔さのために本明細書では具体的に説明しなかったが当業者には知られている１つ以上の他のコンポーネントと相互作用することもできる。

【００４０】

[0057] 更に、以上で開示したシステムの種々の部分、および以下で開示する方法は、人工知能、機械学習、あるいは知識または規則ベース・コンポーネント、サブコンポーネント、プロセス、手段、方法、またはメカニズムを含む、あるいは採用することができる（例えば、ベクトル・マシン、ニューラル・ネットワーク、エキスパート・システム、ベイジアン信念ネットワーク(Bayesian belief networks)、ファジー理論、データ融合エンジン、クラシファイア．．．をサポートする）。このようなコンポーネントは、とりわけ、それらによって実行される一定のメカニズムまたはプロセスを自動化し、これらのシステムおよび方法の一部を更に効率的およびインテリジェントにするだけでなく、適応的にすることもできる。一例として、そして限定ではなく、スケジューラ・コンポーネント１３０およびユーザ・インタフェース１１０はこのようなメカニズムを採用して、以前の相互作用およびその他のコンテキスト情報に基づいて、それぞれ、ジョブ・スケジュールおよびデータ提示を決定または推論することができる。

【００４１】

[0058] 以上で説明した例証的なシステムを基にして、開示する主題にしたがって実装することができる方法は、図１２から図１７のフロー・チャートを参照すると一層良く認められよう。説明の簡素さのために、方法を一連のブロックとして示し説明するが、特許請求する主題は、ブロックの順序によって限定されないことは理解されそして認められよう。これは、ブロックの中には異なる順序で現れてもよいもの、および／または本明細書において図示および説明するブロック以外の他のブロックと同時に現れてもよいものもあるからである。更に、図示するブロック全てが、この後に説明する方法を実行するために必要という訳ではない。

【００４２】

[0059] 図１２を参照すると、ジョブ・スケジューリングおよび監視方法１２００が示されている。符号１２１０において、１組のジョブを受け取る。これらのジョブは、ジョブ毎に依存性を含み、少なくともジョブが依存するデータの識別と、そのジョブによって生

成されたデータとを含む。言い換えると、各ジョブに関して指定されたデータ依存性を含む、１組のジョブを受け取る。符号１２２０において、依存性、ならびに各ジョブによって定められる実行時間および頻度に関する他の情報に基づいて、１組のジョブを処理するためのスケジュールを生成する。符号１２３０において、生成したスケジュールを表示する。１つの態様によれば、このスケジュールをガン・チャートで表示することができ、ジョブの実行開始および終了時刻、ならびにジョブによって生成されたデータを含む。符号１２４０において、スケジュールにしたがってジョブの処理、即ち、ジョブ・ランの実行(execution of a job run)を開始することができる。例えば、処理のためにジョブを実行コンポーネントに供給することができる。１２５０において、実行が開始されたジョブに関して、プロセスの成功または失敗を判定する。このような判定は、単に、実行コンポーネントからの実行ステータスに関する通知を受け入れることによって、あるいはメモリまたはディスクのような位置から実行ステータスを引き出すことによって行うことができる。符号１２６０において、ジョブ・ランの成功または失敗をスケジュールに関して表示する。例えば、ジョブ・ランの成功または失敗に関する明らかな指示を与えるために、実行に成功したジョブ・ランを緑で着色することができ、ジョブ・ランの失敗を赤で着色することができる。更に、ジョブの依存性に基づいて、このジョブが成功するまたは失敗すると予測し、それに応じて着色することもできる。

【００４３】

[0060] 図１３は、関連ジョブを区別する方法１３００を示す。符号１３１０において、スケジュールから１つ以上のジョブ・ランを含むジョブを選択する信号を受け取る。例えば、ジョブ上で左クリックするまたはタッチすることによって、例えば、「重複除去」というようなジョブを、実行のためにスケジューリングされた１つ以上のジョブおよびジョブ・ランを示すタイムライン・ビューから選択することができる。符号１３２０において、選択されたジョブに依存するジョブを識別する。符号１３３０において、選択されたジョブが依存するジョブを識別する。ジョブの識別は、依存性の分析に基づくことができる。一実施形態によれば、依存性を有向グラフとして表現することができる。有向グラフは、ジョブを頂点として、そしてデータ集合を、ジョブ同士を接続する有向エッジとして有する。この実例では、選択されたジョブをこのグラフ内において識別することができ、選択されたジョブに依存するジョブを識別するために、選択されたジョブから順方向に分析を進めることができ、選択されたジョブが依存するジョブを識別するために逆方向に分析を進めることができる。符号１３４０において、選択されたジョブおよび識別されたジョブを強調する。言い換えると、スケジュール上で、例えば、選択されたジョブおよび識別されたジョブを、表示されている他のジョブから視覚的に区別する。このように、ユーザは素早くジョブ間の関係を理解することができる。

【００４４】

[0061] 図１４は、成功および失敗ジョブ実行を区別する方法１４００を示す。符号１４１０において、ジョブ・ランの実行を開始する。ここで、ジョブ・ラン(job run)とは特定の時刻に実行する(run)ようにスケジューリングされている、ジョブのインスタンスのことである。符号１４２０において、実行が開始されたジョブ・ランの実行に依存するジョブ・ランを識別する。従属ジョブ・ランを識別するためには、依存性情報を利用することができる。例えば、依存性情報を有向グラフに記録することができる。有向グラフは、ジョブが頂点であり、データ集合が、ジョブ同士を接続する有向エッジとして接続される。実行が開始されたジョブは、グラフにおいて識別することができ、従属ジョブは、そのジョブからグラフを下に横断することによって識別することができる。続いて、従属ジョブのスケジューリングされたランを、スケジュールから、またはスケジューリングされたランがジョブ毎に記録されている場合はグラフから判定することができる。符号１４３０において、ジョブ・ランの実行に関して失敗があったか否かに関して判定を行う。失敗がなかった場合（「ＮＯ」）、本方法は１４４０に進み、成功した実行をそのジョブ・ランに対して記録する。１つの態様によれば、動作依存性情報を記録する。この動作依存性情報は、依存性情報に加えて、ジョブ・ステータスを含む。成功した実行は、これに関して保存

することができるジョブ・ステータスである。次に、１４５０において、成功した実行を、実行したジョブに関して表示し、更に必要に応じて、従属ジョブに対して実行成功の予測として表示する。例えば、スケジュール上に表示されるジョブ・ランは緑で着色することができる。１４３０において、実行失敗が判定された場合（「ＹＥＳ」）、本方法は１４６０に進み、ここで実行失敗を記録する。失敗とは、先に注記したように、動作依存性情報を保存する(preserve)ことに関してセーブすることができるジョブ・ステータスである。次いで、本方法は１４７０に進み、例えば、スケジュール上に、実行したジョブに関する失敗の表示を示す。更に、実行したジョブに依存するジョブも、実行したジョブが失敗した場合、失敗する可能性が高い。したがって、従属ジョブ・ランは、失敗が予測されたことを示すように、表示することができる。例えば、実行したジョブ・ランおよび従属ジョブ・ランを赤で着色し、それぞれ、失敗および失敗の予測を示すことができる。

【００４５】

[0062] 図１５は、スケジューリング・チェーンのトラブルシューティングを容易にする方法１５００を示す。符号１５１０において、失敗したジョブ・ランまたは失敗すると予測されたジョブ・ランを選択する信号を受け取る。更に具体的には、ここでは、例えば、スケジュールに関して失敗が示される。例えば、スケジュールは複数のジョブを含むことができ、ジョブ毎に、ある時点での実行のためにジョブ・ランをスケジューリングする。更に、ジョブ・ラン毎に、既に実行されている場合には、ジョブ・ランが成功したまたは失敗したこと、あるいは未だ実行されていない場合には、成功または失敗すると予測されたことを記す視覚指示を与えることができる。この選択信号には、失敗を示すジョブ・ランの内の１つと関連付けることができ、例えば、ユーザがこのジョブ・ランの視覚表現をクリックまたはタッチしたときに生成することができる。符号１５２０において、失敗したまたは失敗すると予測された関連ジョブ・ランを識別する。関連ジョブ・ランは、選択されたジョブ・ランに依存するジョブ・ラン、および選択されたジョブ・ランが依存するジョブ・ランを含む。ジョブに関する依存性情報を分析することによって、関連ジョブを識別することができる。一例として、選択されたジョブを有向グラフにおいて識別することができる。有向グラフは、ジョブを頂点として含み、データ集合は、ジョブ同士を接続する有向エッジとして接続される。続いて、選択されたジョブからこのグラフを上流側に横断し、更に選択されたジョブから下流側に横断して、関連ジョブを識別することができる。一実施形態によれば、ジョブ・ランおよびステータスを、依存性情報と共に記録することができる。したがって、関連ジョブの識別時に、ジョブ・ランを識別することができ、失敗ステータスを含むものを、実際のまたは予測されたもののいずれも識別することができる。勿論、ジョブ・ランを含む何らかの情報を、スケジュールに関してセーブし、関連ジョブ・ランを判定するために利用することができる。符号１５３０において、選択されたジョブ・ランおよび関連ジョブ・ランを強調する。別の言い方をすると、選択されたジョブ・ランおよび関連するジョブ・ランを、他のジョブ・ランから視覚的に区別することができる。これは、ユーザに、トラブルシューティングのために、失敗および連鎖的な(cascading)失敗に集中させることができる。

【００４６】

[0063] 図１６は、関連ジョブおよびデータ集合を区別する方法１６００のフロー・チャート図である。符号１６１０において、ジョブ（またはデータ集合）を選択する信号を受け取る。例えば、ジョブおよびデータ集合のスケジュール・ビューまたは図解ビュー(diagrammatic view)においてユーザが１つ以上の所定のジェスチャによってジョブを選択したときに、信号を受け取ることができる。符号１６２０において、関連ジョブおよびデータ集合を識別する。実施形態によれば、ジョブおよびデータ集合に関する依存性情報をセーブして、１つまたは複数の選択されたジョブに依存するジョブおよびデータ集合（例えば、下流側）、ならびに選択されたジョブが依存するデータ集合（例えば、上流側）を含む、関連ジョブおよびデータ集合を識別するときに参照することができる。１つの実例では、有向グラフは、ジョブを頂点として、そしてデータ集合を、ジョブ同士を接続する有向エッジとして含むことができる。このグラフにおいて、選択されたジョブ（またはデー

タ集合）を識別することができ、このグラフを横断して、選択されたジョブに依存するジョブおよびデータ集合、ならびに選択されたジョブ（またはデータ集合）が依存するジョブおよびデータ集合を識別することができる。符号１６３０において、選択されたジョブ（またはデータ集合）ならびに関連ジョブおよびデータ集合を、視覚表示上で強調する。言い換えると、選択されたジョブおよび関連ジョブ、ならびにデータ集合を、他のジョブおよびデータ集合から視覚的に区別する。例えば、指定ジョブおよびデータ集合の図解ビューにおいて、関連ジョブおよびデータ集合の部分集合を視覚的に区別することができる。これは、多数のジョブおよびデータ集合ならびに複雑なパイプラインまたはチェーンが存在するときに、関係を理解するのに特に役立つ。

【００４７】

[0064] 図１７は、ジョブのデータ先導処理(data driven processing)方法１７００を示す。符号１７１０において、生成される出力データ・スライスを決定する。ここで、データ・スライスとは、これらが関連付けられる時間期間毎のデータ集合スライスにおけるデータ片(pieces of data)である。特定の一実施態様では、データ・スライスのステータスを分析することによって、出力データ・スライスを識別することができる。ステータスが「実行保留」等である場合、データ・スライスは、生成される出力データ・スライスであると判定することができる。符号１７２０において、出力データ・スライスを生成のために選択する。ここで、複数の生成される出力データ・スライス間からの選択は、ポリシーに基づくことができる。例えば、ジョブ実行についてのポリシーが、ステータスがいつ「実行保留」等に設定されたかに基づいて、最も古い出力データ・スライスを最初にまたは最も新しい出力データ・スライスを最初に生成のために選択することを指定することができる。符号１７３０において、依存性期間を判定する。依存性期間とは、出力データ・スライスを生成することを要求される入力データの時間範囲である。これは、出力データ・スライスを生成するジョブに関して定められた依存性情報に基づいて、選択出力データ・スライス(select output data slice)に対して決定することができる。例えば、ジョブが第１ソースからの３時間分のデータ、および第２ソースからの１時間分のデータにわたって処理する(operate)ことを指定することができる。したがって、全てのデータ集合が時間毎のスケジュールを有する（例えば、スライスが１時間である）場合、第１ソースからの３時間分のデータ、および第２ソースからの１時間分のデータは、１時間分の出力スライス・データを生成することが要求される。符号１７４０において、依存性期間内における全ての要求された入力データ・スライスの準備ができているか否か判定を行う。これは、入力データ・スライスのステータスについて問い合わせることによって判定することができる。入力データ・スライスの各々が「準備完了」のステータスを有し、データが消費の準備ができていることを意味する場合、本方法は符号１７５０に進むことができる。それ以外の場合、本方法はループして、入力データ・スライスが準備完了になるのを待ち続けることができる。符号１７５０において、選択された出力スライスを生成するジョブの実行をトリガまたは開始する符号１７６０において、出力スライスのステータスを設定することができる。データが生成されている間、ステータスを「進行中」に設定することができ、データ・スライスがジョブのために生成されていることを意味する。一旦実行が完了したなら（または実行に失敗したなら）、出力スライスを生成する試みが失敗した場合には、ステータスを「失敗」に変更することができ、生成する試みが成功し、データ・スライスが消費のための準備ができた場合には「準備完了」に変更することができる。符号１７７０において、実行に関する情報を記録する。例えば、出力スライスのステータスを、実行の開始および終了時間と共に記録することができる。このようなデータは、ジョブ実行の理解および制御を容易にするために、情報生成、およびユーザへの提示のための可視化に関して利用することができる。

【００４８】

[0065] 本開示は、ジョブ・スケジューリングおよび監視に関する種々のアクションを実行する、または実行するように構成された種々の製品およびプロセスをサポートする。以下に説明するのは、１つ以上の例証的な方法およびシステムである。

【００４９】
[0066] 方法は、ディスプレイ上のインタフェースにおいて、１つ以上のデータ変換ジョブのビューを提示するアクトと、インタフェースを介して、１つ以上のデータ変換ジョブから１つのジョブを選択する、ユーザからの第１信号を受け取るアクトと、ジョブ間におけるデータ依存性に基づいて、選択されたジョブに対する１つ以上の関連ジョブを自動的に判定するアクトと、選択されたジョブ、および１つ以上の関連ジョブを、１つ以上の他のジョブから視覚的に区別して提示するアクトとを含む。更に、この方法は、ディスプレイ上のインタフェースにおいて、実行のためにスケジューリングされた１つ以上のジョブを提示するアクトを含む。更に、この方法は、ディスプレイ上のインタフェースにおいて、１つ以上の既に実行されたジョブを提示するアクトを含む。更に、この方法は、ディスプレイ上のインタフェースにおいて、既に実行されたジョブを、スケジューリングされたジョブとは視覚的に区別して提示するアクトを含む。更に、この方法は、１つ以上のジョブの実行中に計算負荷を判定するアクトと、ディスプレイ上のインタフェースにおいて、１つ以上のジョブと整列させて計算負荷を提示するアクトを含む。更に、この方法は、ディスプレイ上のインタフェースにおいて、少なくとも１つのジョブが正常な実行に失敗したことの指示を提示するアクトを含む。更に、この方法は、正常な実行に失敗した少なくとも１つのジョブ・ランから１つを選択する第２信号を受け取るアクトと、選択されたジョブ・ランに関連付けられた関連ジョブ・ランの内、正常な実行に失敗した１つ以上、または正常な実行に失敗したジョブ・ランに対する依存性に基づいて失敗すると予測された１つ以上を自動的に判定するアクトと、ディスプレイ上のインタフェースにおいて、選択されたジョブ・ランと、１つ以上の関連ジョブ・ランとを強調するアクトとを含む。更に、この方法は、正常な実行に失敗した少なくとも１つのジョブから１つを選択する第２信号を受け取るアクトと、第２信号に応答して、少なくとも１つのジョブの実行を再スケジューリングするアクトとを含む。更に、この方法は、選択されたジョブおよび１つ以上の関連ジョブを、ジョブおよびデータ集合、ならびにこれらのジョブおよびデータ集合間の接続の表現を含む図(diagram)において提示するアクトを含む。
【００５０】
[0067] 方法は、以下のアクトを実行するために、メモリに格納されたコンピュータ実行可能命令を実行するように構成された少なくとも１つのプロセッサを使用するステップを含む。アクトは、グラフィカル・ユーザ・インタフェースを介してデータ変換ジョブの選択を検出するアクトと、ジョブの選択を検出したことに応答して、ジョブ間におけるデータ依存性に基づいて１つ以上の関連ジョブを自動的に判定するアクトと、グラフィカル・ユーザ・インタフェース上に表示された他のジョブとは視覚的に区別して、選択されたジョブおよび１つ以上の関連ジョブを提示するアクトとを含む。更に、この方法は、ジョブ実行と併せてコンピュータ・リソース利用度を監視するアクトと、１つ以上のそれぞれのジョブと整列させてリソース利用度の可視化を提示するアクトとを含む。更に、この方法は、ジョブ・ランの実行の成功または失敗を判定するアクトと、失敗した実行ランを成功した実行ランと区別して提示する(different from)アクトとを含む。更に、この方法は、正常な実行に失敗したジョブ・ランに関して受け取った信号に基づいて、ジョブ・ランの実行を再スケジューリングするアクトを含む。更に、この方法は、選択失敗ジョブ・ラン(select failed job run) および１つ以上の関連失敗ジョブ・ラン、または選択失敗ジョブ・ランを識別する信号を受け取った後に、失敗ジョブ・ランに対する依存性に基づいて、失敗すると予測されたランを強調するアクトを含む。更に、この方法は、ジョブおよびデータ集合、ならびにこれらのジョブおよびデータ集合間の接続の表現を含む図において、選択されたジョブおよび１つ以上の関連ジョブを提示するアクトを含む。
【００５１】
[0068] システムは、メモリに結合されたプロセッサであって、このメモリに格納された以下のコンピュータ実行可能コンポーネントを実行するように構成される、プロセッサを含む。ジョブによって指定された要件に基づいて実行のためにデータ変換ジョブをスケジューリングするように構成された第１コンポーネント。スケジュールにしたがってタイム

ライン図上にこのジョブを提示するように構成された第２コンポーネント。選択ジョブを識別する信号に応答して、選択ジョブに関連する１つ以上のジョブを、タイムライン図上においてデータ依存性に基づいて強調するように構成された第３コンポーネント。更に、第３コンポーネントは、選択ジョブに依存する１つ以上のジョブを強調するように構成される。更に、第３コンポーネントは、選択ジョブが依存する１つ以上のジョブを強調するように構成される。更に、このシステムは、ジョブ実行が成功かまたは失敗か検出するように構成された第４コンポーネントを含む。更に、このシステムは、タイムライン上における失敗ジョブ実行の表現の選択に応答して、失敗ジョブ実行に対する依存性に基づいて、関連失敗ジョブ実行または予測された失敗ジョブ実行を強調するように構成された第５コンポーネントを含む。

【００５２】

[0069] システムは、１組のデータ変換ジョブのビューを生成する手段と、１組のデータ変換ジョブからのジョブの選択、即ち、選択されたジョブを受け取る手段と、依存性情報に基づいて、選択されたジョブに対する１つ以上の関連ジョブを自動的に判定する手段と、ビューにおいて、選択されたジョブおよび１つ以上の関連ジョブを強調する手段とを含む。更に、このシステムは、１組のデータ変換ジョブの実行のためにスケジュールを生成する手段を含む。ビューを生成する手段は、スケジュールのビューを生成する。更に、このシステムは、スケジュールにしたがって１組のデータ変換ジョブの内１つを実行する手段を含む。更に、このシステムは、１組のデータ変換ジョブの内の１つのランの成功または失敗を検出する手段を含む。ビューを生成する手段は、ジョブ・ランの成功または失敗の識別を含むスケジュールのビューを生成する。選択を受け取る手段は、失敗ジョブ・ランの選択を受け取り、強調する手段は、失敗した、または失敗ランに対する依存性に基づいて失敗すると予測された１つ以上の関連ジョブ・ランを強調する。

【００５３】

[0070] 本開示の態様は、データ変換を対象とする、または、言い換えると、分析ツールによる今後の公開または消費のために使用可能な形態にデータを変える(place)ことを対象とする。更に特定すると、態様は、データ変換ジョブをスケジューリングおよび監視する技術的な問題を対象とする。この問題に取り組むために採用される技術的手段は、ユーザが選択したジョブに関連するジョブを判定することを含む。これらのジョブには、ジョブ依存性に基づいて、選択されたジョブに依存するジョブおよび／または選択されたジョブが依存するジョブを含む。ジョブの選択、ならびに選択されたジョブおよび判定された関連ジョブの強調は、グラフィカル・ユーザ・インタフェースと併せて実行され、グラフィカル・ユーザ・インタフェースは、ユーザがジョブ間の関係を理解しトラブルシューティングするときに補助する。その結果、技術的な効果は、ジョブ・スケジューリングおよびエラー低減に関してユーザ効率を高めることを含むが、これに限定されるのではない。

【００５４】

[0071] 「例証的な」(exemplary)という単語またはその種々の形態は、本明細書においては、例、実例、または例示として役割を果たすという意味で使用される。「例証的」であるとして本明細書において記載される態様または設計はいずれも、必ずしも他の態様や設計に対して好ましいとも有利であるとも解釈されない。更に、例は明確化および理解のために提示されるに過ぎず、特許請求する主題や本開示の関連部分をいかなる方法によっても限定または制限することは意図していない。尚、様々な範囲の無数の追加または代わりの例を提示することもできたが、簡潔さのために省略したことは認められよう。

【００５５】

[0072] 本明細書において使用する場合、「コンポーネント」および「システム」という用語、ならびにその種々の形態（例えば、複数のコンポーネント、複数のシステム、サブシステム．．．）は、コンピュータ関連エンティティを指すことを意図しており、ハードウェア、ハードウェアおよびソフトウェアの組み合わせ、ソフトウェア、または実行中のソフトウェアのいずれかである。例えば、コンポーネントは、プロセッサ上で実行するプロセス、プロセッサ、オブジェクト、インスタンス、実行可能ファイル、実行のスレッド

、プログラム、および／またはコンピュータであってもよいが、そうであることに限定されない。例示として、コンピュータ上で実行するアプリケーションおよびコンピュータの双方がコンポーネントであることが可能である。１つ以上のコンポーネントがプロセスおよび／または実行のスレッド内に常駐する(reside)ことができ、コンポーネントが１つのコンピュータ上に集中配置される(localize)こと、および／または２つ以上のコンピュータ間で分散されることが可能である。

【００５６】

[0073] 「または」(or)という接続詞は、本説明および添付する特許請求の範囲において使用する場合、別段指定されていないまたは文脈から明らかでないならば、排他的な「または」ではなく、内包的な「または」を意味することを意図している。言い換えると、「’Ｘ’または’Ｙ’」は、「Ｘ」および「Ｙ」のあらゆる内包的組み合わせ(permutation)を意味することを意図している。例えば、「’Ａ’が’Ｘ’を採用する」、「’Ａ’が’Ｙ’を採用する」、または「’Ａ’が’Ｘ’および’Ｙ’双方を採用する」場合、「’Ａ’が’Ｘ’または’Ｙ’を採用する」が以上の実例のいずれの下においても満たされる。

【００５７】

[0074] 「含む」(includes)、「収容する」(contains)、「有する」(has)、「有している」(having)という用語、またはこれらの形態における異体が詳細な説明または特許請求の範囲において使用される限りにおいて、このような用語は、「含む」(comprising)という用語が請求項において移行性単語として使用されるときに「含む」が解釈されるときと同様に、包含的であることを意図している。

【００５８】

[0075] 特許請求する主題に対するコンテキストを与えるために、図１８および以下の論述は、主題の種々の態様を実現することができる、適した環境の端的で全体的な説明を行うことを意図している。しかしながら、この適した環境は例に過ぎず、使用範囲や機能に関して限定を示唆することは全く意図していない。

【００５９】

[0076] 以上で開示したシステムおよび方法は、１つ以上のコンピュータ上で実行するプログラムのコンピュータ実行可能命令という一般的なコンテキストで説明することができるが、他のプログラム・モジュール等との組み合わせでも態様を実現できることは当業者には認められよう。一般に、プログラム・モジュールは、とりわけ、ルーチン、プログラム、コンポーネント、データ構造を含み、特定のタスクを実行する、および／または特定の抽象データ型を実装する。更に、以上のシステムおよび方法は、種々のコンピュータ・システム構成でも実施することができ、１つのプロセッサ、マルチプロセッサまたはマルチコア・プロセッサ・コンピュータ・システム、ミニコンピューティング・デバイス、メインフレーム・コンピュータ、更にはパーソナル・コンピュータ、ハンドヘルド・コンピューティング・デバイス（例えば、パーソナル・ディジタル・アシスタント（ＰＤＡ）、電話機、腕時計．．．）、マイクロプロセッサ・ベースのまたはプログラマブル消費者用または産業用電子機器等が含まれることは、当業者には認められよう。また、分散型コンピューティング環境においても態様を実施することができ、タスクは、通信ネットワークを通じてリンクされたリモート処理デバイスによって実行される。しかしながら、特許請求する主題の全ての態様ではないにしても、その一部は、単体コンピュータ（stand-alone computer)上で実施することができる。分散型コンピューティング環境では、プログラム・モジュールは、ローカルまたはリモート・メモリ・デバイスの一方または双方に位置することができる。

【００６０】

[0077] 図１８を参照すると、汎用コンピュータまたはコンピューティング・デバイス１８０２の例（例えば、デスクトップ、ラップトップ、タブレット、腕時計、サーバ、ハンドヘルド、プログラマブル消費者用または産業要電子機器、セットトップ・ボックス、ゲーム・システム、コンピュータ・ノード(compute node)．．．）が示されている。コンピュータ１８０２は、１つ以上のプロセッサ１８２０、メモリ１８３０、システム・バス

１８４０、大容量記憶デバイス（１つまたは複数）１８５０、および１つ以上のインタフェース・コンポーネント１８７０を含む。システム・バス１８４０は、通信可能に少なくとも以上のシステム構成要素を結合する。しかしながら、コンピュータ１８０２は、その最も簡単な形態では、メモリ１８３０に結合された１つ以上のプロセッサ１８２０を含むことができ、プロセッサ１８２０が、メモリ１８３０に格納されている種々のコンピュータ実行可能アクション、命令、および／またはコンポーネントを実行することは認められよう。

【００６１】

[0078] プロセッサ（１つまたは複数）１８２０は、汎用プロセッサ、ディジタル信号プロセッサ（ＤＳＰ）、特定用途集積回路（ＡＳＩＣ）、フィールド・プログラマブル・ゲート・アレイ（ＦＰＧＡ）またはその他のプログラマブル・ロジック・デバイス、ディスクリート・ゲートまたはトランジスタ・ロジック、ディスクリート・ハードウェア・コンポーネント、あるいは本明細書において説明した機能を実行するように設計された、これらの任意の組み合わせによって実現することができる。汎用プロセッサとは、マイクロプロセッサでもよいが、代わりに、プロセッサは任意のプロセッサ、コントローラー、マイクロコントローラー、または状態機械でもよい。また、プロセッサ（１つまたは複数）１８２０は、コンピューティング・デバイスの組み合わせ、例えば、ＤＳＰおよびマイクロプロセッサの組み合わせ、複数のマイクロプロセッサ、マルチコア・プロセッサ、ＤＳＰコアと併せた１つ以上のマイクロプロセッサ、あるいは任意の他のこのような構成として実現することもできる。一実施形態では、プロセッサ（１つまたは複数）はグラフィクス・プロセッサであることが可能である。

【００６２】

[0079] コンピュータ１８０２は、コンピュータ１８０２の制御を容易にして特許請求する主題の１つ以上の態様を実現するために、種々のコンピュータ読み取り可能媒体を含む、またそうでなければこれと相互作用することができる。コンピュータ読み取り可能媒体は、コンピュータ１８０２によってアクセスすることができる入手可能な媒体であればいずれでも可能であり、揮発性および不揮発性媒体、ならびにリムーバブルおよび非リムーバブル媒体を含む。コンピュータ読み取り可能媒体は、２つの全く異なる、そして相互に排他的な種類、即ち、コンピュータ記憶媒体および通信媒体を含むことができる。

【００６３】

[0080] コンピュータ記憶媒体は、揮発性および不揮発性、リムーバブルおよび非リムーバブル媒体を含み、コンピュータ読み取り可能命令、データ構造、プログラム・モジュール、またはその他のデータというような情報の格納のための任意の方法または技術で実現される。コンピュータ記憶媒体は、メモリ・デバイス（例えば、ランダム・アクセス・メモリ（ＲＡＭ）、リード・オンリー・メモリ（ＲＯＭ）、電子的消去可能プログラマブル・リード・オンリ・メモリ（ＥＥＰＲＯＭ）．．．）、磁気記憶デバイス（例えば、ハード・ディスク、フロッピー・ディスク、カセット、テープ．．．）、光ディスク（例えば、コンパクト・ディスク（ＣＤ）、ディジタル・バーサタイル・ディスク（ＤＶＤ）．．．）、およびソリッド・ステート・デバイス（例えば、ソリッド・ステート・ドライブ（ＳＳＤ）、フラッシュ・メモリ・デバイス（例えば、カード、スティック、キー・ドライブ．．．）．．．）のような記憶デバイス、あるいは、送信または伝達するのではなく、コンピュータ１８０２によってアクセス可能な所望の情報を格納する任意のその他の同様の媒体を含む。したがって、コンピュータ記憶媒体は、変調データ信号、および通信媒体を含むものを除外する。

【００６４】

[0081] 通信媒体は、コンピュータ読み取り可能命令、データ構造、プログラム・モジュール、またはその他のデータを、搬送波のような変調データ信号、またはその他の移送メカニズムに具体化し、任意の情報配信媒体を含む。「変調データ信号」という用語は、信号内に情報をエンコードするように、その特性の１つ以上が設定されたまたは変更された信号を意味する。一例として、そして限定ではなく、通信媒体は、有線ネットワークまた

は直接有線接続のような有線媒体、ならびに音響、ＲＦ、赤外線、およびその他のワイヤレス媒体のようなワイヤレス媒体を含む。

【００６５】

[0082] メモリ１８３０および大容量記憶デバイス（１つまたは複数）１８５０は、コンピュータ読み取り可能記憶媒体の例である。コンピューティング・デバイスの正確な構成および種類に応じて、メモリ１８３０は揮発性（例えば、ＲＡＭ）、不揮発性（例えば、ＲＯＭ、フラッシュ・メモリ．．．）、またはこれら２つの何らかの組み合わせであってもよい。一例として、起動中のように、コンピュータ１８０２内部にあるエレメント間で情報を転送するための基本ルーチンを含む基本入力／出力システム（ＢＩＯＳ）は、不揮発性メモリに格納することができ、一方揮発性メモリは、とりわけ、プロセッサ（１つまたは複数）１８２０による処理を促進するために外部キャッシュ・メモリとして作用することができる。

【００６６】

[0083] 大容量記憶デバイス（１つまたは複数）１８５０は、メモリ１８３０と比較すると大容量のデータの格納のためのリムーバブル／非リムーバブル、揮発性／不揮発性コンピュータ記憶媒体を含む。例えば、大容量記憶デバイス（１つまたは複数）１８５０は、磁気または光ディスク・ドライブ、フロッピ・ディスク・ドライブ、フラッシュ・メモリ、ソリッド・ステート・ドライブ、またはメモリ・スティックのような１つ以上のデバイスを含むが、これらに限定されるのではない。

【００６７】

[0084] メモリ１８３０および大容量記憶デバイス（１つまたは複数）１８５０は、オペレーティング・システム１８６０、１つ以上のアプリケーション１８６２、１つ以上のプログラム・モジュール１８６４、およびデータ１８６６を含む、またはその中に格納しておくことができる。オペレーティング・システム１８６０は、コンピュータ１８０２のリソースを制御し割り当てるように動作する。アプリケーション１８６２は、システムおよびアプリケーション・ソフトウェアの一方または双方を含み、メモリ１８３０および／または大容量記憶デバイス（１つまたは複数）１８５０に格納されているプログラム・モジュール１８６４およびデータ１８６６を使用して(through)、オペレーティング・システム１８６０によるリソースの管理を利用し、１つ以上のアクションを実行することができる。したがって、アプリケーション１８６２は汎用コンピュータ１８０２を、それによって設けられるロジックにしたがって特殊機械に変換することができる。

【００６８】

[0085] 特許請求する主題の全部または一部は、開示した機能を実現するためにコンピュータを制御するソフトウェア、ファームウェア、ハードウェア、またはこれらの任意の組み合わせを生成する標準的なプログラミングおよび／または設計技法を使用して、実現することができる。一例としてそして限定ではなく、ジョブ・システム１００またはその一部は、アプリケーション１８６２であること、またはその一部を形成することができ、メモリおよび／または大容量記憶デバイス（１つまたは複数）１８５０に格納されている１つ以上のモジュール１８６４およびデータ１８６６を含むことができ、その機能は、１つ以上のプロセッサ（１つまたは複数）１８２０によって実行されるときに実現することができる。

【００６９】

[0086] 特定の一実施形態によれば、プロセッサ（１つまたは複数）１８２０は、１つの集積回路基板上にハードウェアおよびソフトウェアの双方を含む、または言い換えると、集積する、システム・オン・チップ（ＳＯＣ）または同様のアーキテクチャに対応することができる。ここで、プロセッサ（１つまたは複数）１８２０は、とりわけ、プロセッサ（１つまたは複数）１８２０およびメモリ１８３０に少なくとも類似する、１つ以上のプロセッサおよびメモリを含むことができる。従来のプロセッサは、最小量のハードウェアおよびソフトウェアを含み、外部ハードウェアおよびソフトウェアに大きく頼る。対照的に、プロセッサのＳＯＣ実施態様は、内部にハードウェアおよびソフトウェアを埋め込み

、外部ハードウェアおよびソフトウェアには最低限頼ってまたは全く頼らずに、特定の機能を可能にするので、一層強力である。例えば、ジョブ・システム１００および／または関連する機能を、ＳＯＣアーキテクチャのハードウェア内に埋め込むことができる。

【００７０】

[0087] また、コンピュータ１８０２は、１つ以上のインタフェース・コンポーネント１８７０も含む。インタフェース・コンポーネント１８７０は、通信可能にシステム・バス１８４０に結合され、コンピュータ１８０２との相互作用を容易にする。一例として、インタフェース・コンポーネント１８７０は、ポート（例えば、シリアル、パラレル、ＰＣＭＣＩＡ、ＵＳＢ、Ｆｉｒｅ Ｗｉｒｅ．．．）、またはインタフェース・カード（例えば、サウンド、ビデオ．．．）等とすることができる。一実施態様例では、インタフェース・コンポーネント１８７０は、例えば、１つ以上のジェスチャまたは音声入力によって、１つ以上の入力デバイス（例えば、マウスのようなポインティング・デバイス、トラックボール、スタイラス、タッチ・パッド、キーボード、マイクロフォン、ジョイスティック、ゲーム・パッド、衛星ディッシュ、スキャナー、カメラ、他のコンピュータ．．．）を介して、ユーザがコマンドおよび情報をコンピュータ１８０２に入力することを可能にするユーザ入力／出力インタフェースとして具体化することができる。他の実施態様例では、インタフェース・コンポーネント１８７０は、とりわけ、ディスプレイ（例えば、ＬＣＤ、ＬＥＤ、プラズマ．．．）、スピーカ、プリンタ、および／または他のコンピュータにデータを供給する出力周辺インタフェースとして具体化することができる。更にまたその上、インタフェース・コンポーネント１８７０は、有線またはワイヤレス通信リンクを介してというようにして、他のコンピューティング・デバイス（図示せず）との通信を可能にするネットワーク・インタフェースとして具体化することができる。

【００７１】

[0088] 以上で説明したものには、特許請求する主題の態様の例が含まれる。勿論、特許請求する主題を説明する目的のためにコンポーネントまたは方法の想起し得るあらゆる組み合わせを記載することは不可能であるが、開示した主題の多くの更に他の組み合わせや置換が可能であることは、当業者であれば認めることができよう。したがって、開示した主題は、添付した請求項の主旨および範囲に該当するあらゆるこのような変更、修正、および変形を包含することを意図している。

【図面】

【図１】

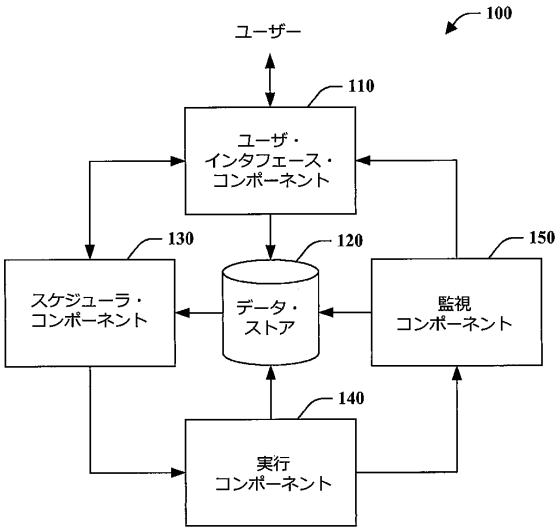

FIG. 1

【図２】

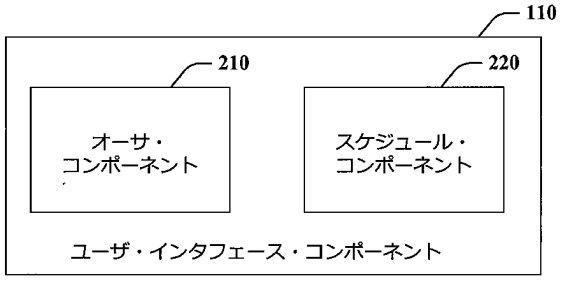

FIG. 2

【図３】

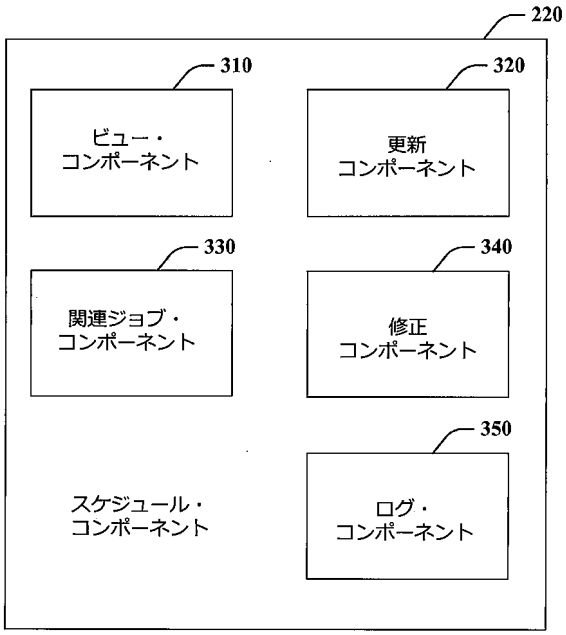

FIG. 3

【図４】

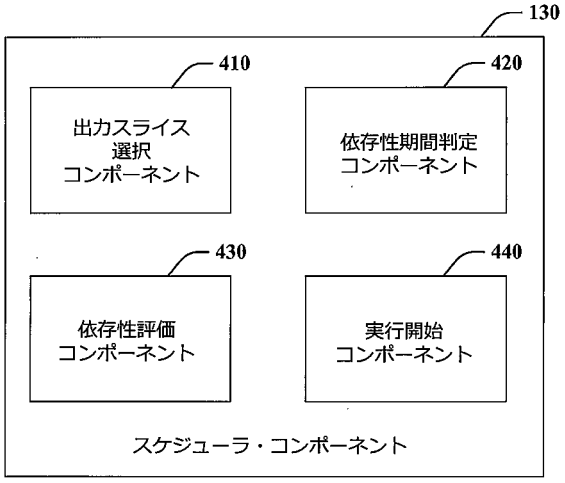

FIG. 4

【図５】

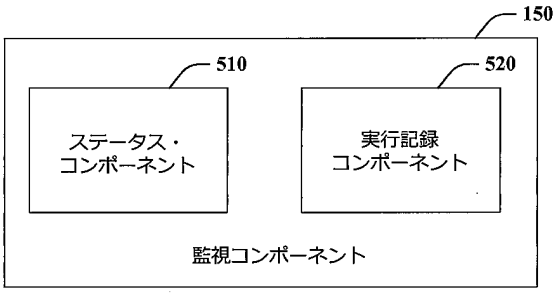

FIG. 5

【図６】

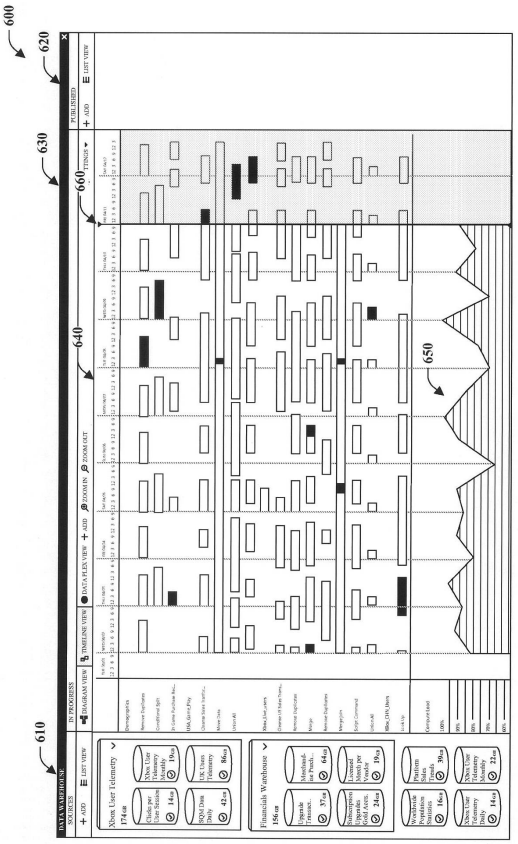

FIG. 6

【図７】

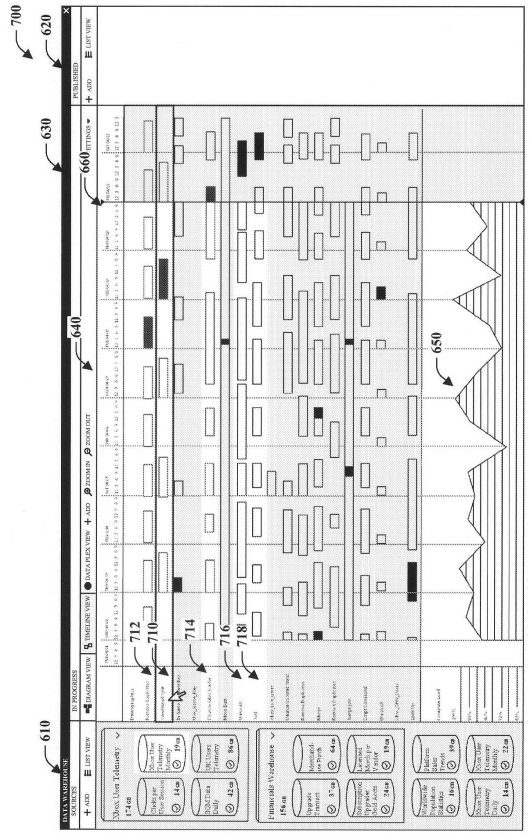

FIG. 7

【図８】

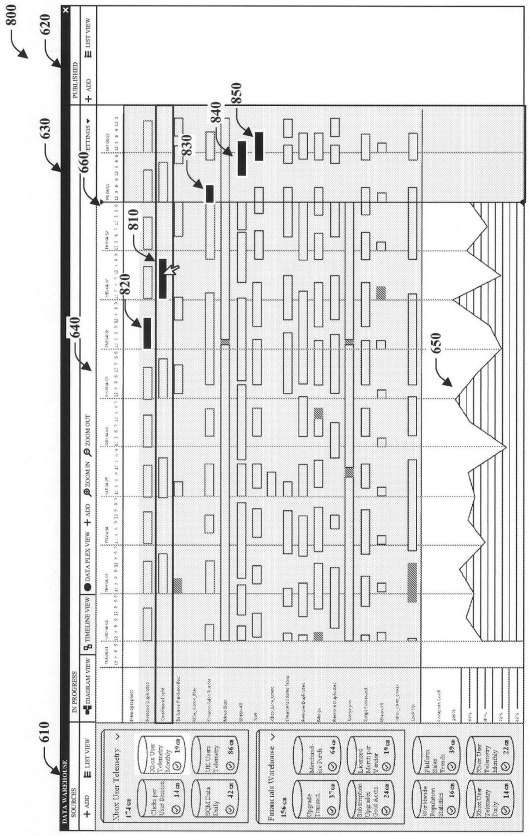

FIG. 8

【図９】

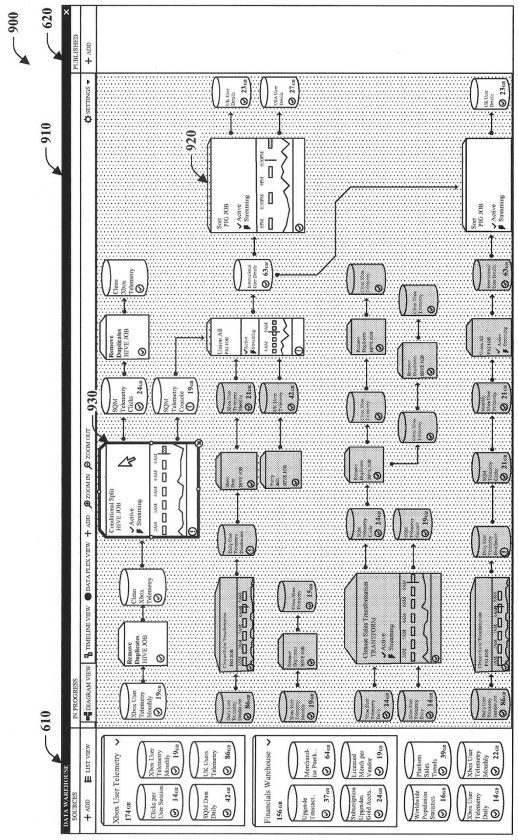

FIG. 9

【図１０】

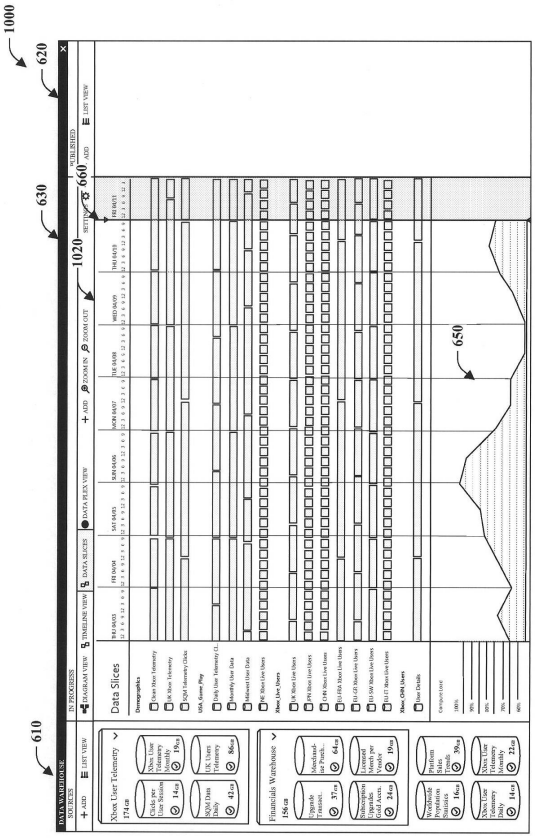

FIG. 10

【図１１】

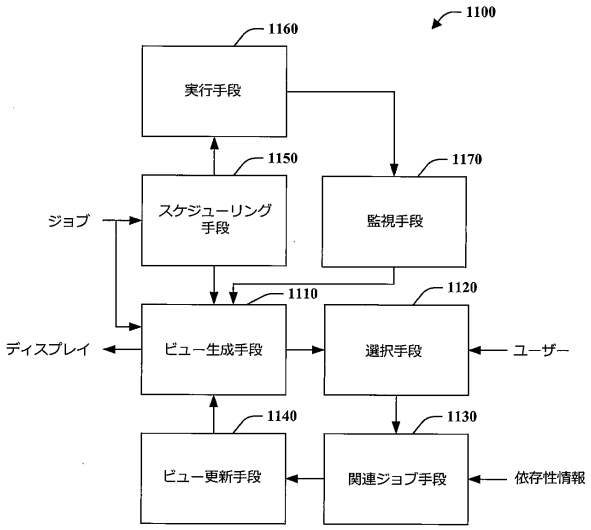

FIG. 11

【図１２】

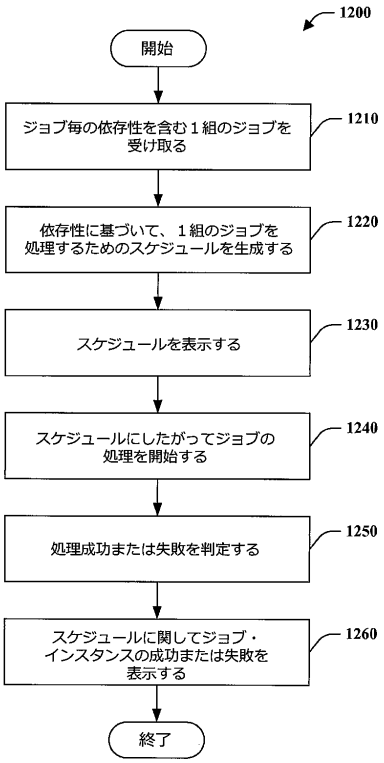

FIG. 12

【図１３】

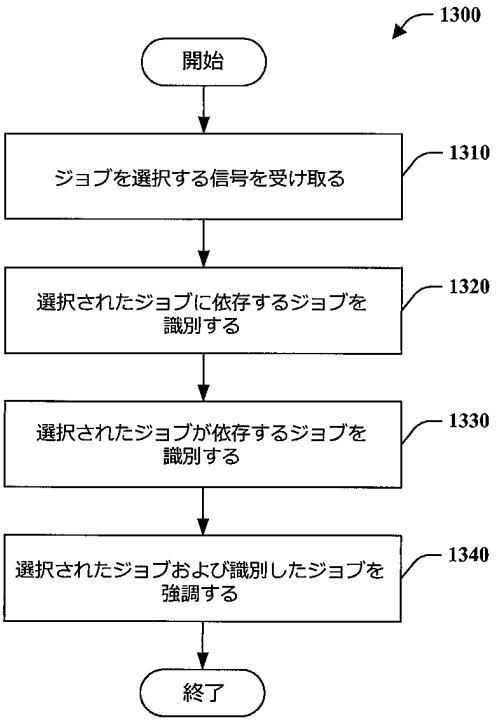

FIG. 13

【図１４】

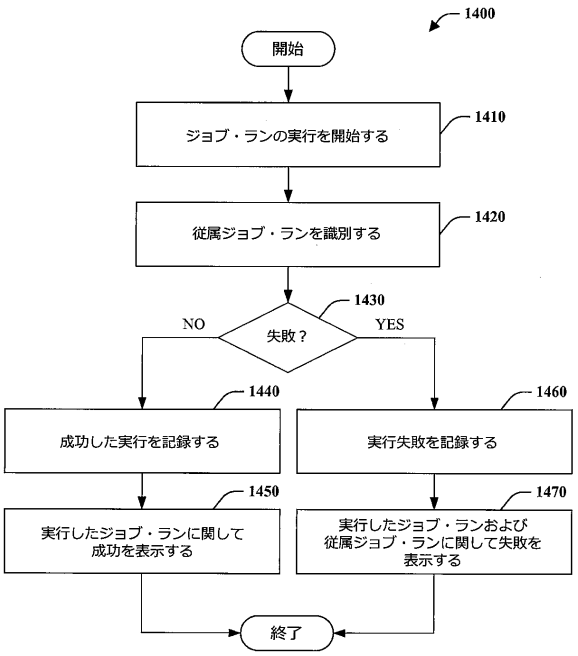

FIG. 14

【図１５】

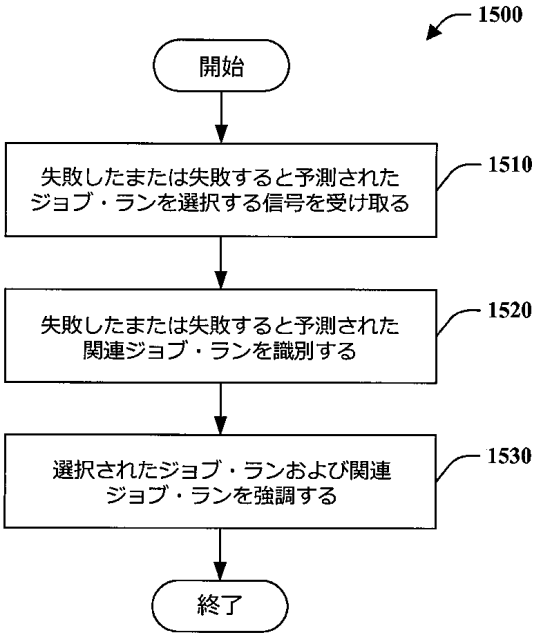

FIG. 15

【図１６】

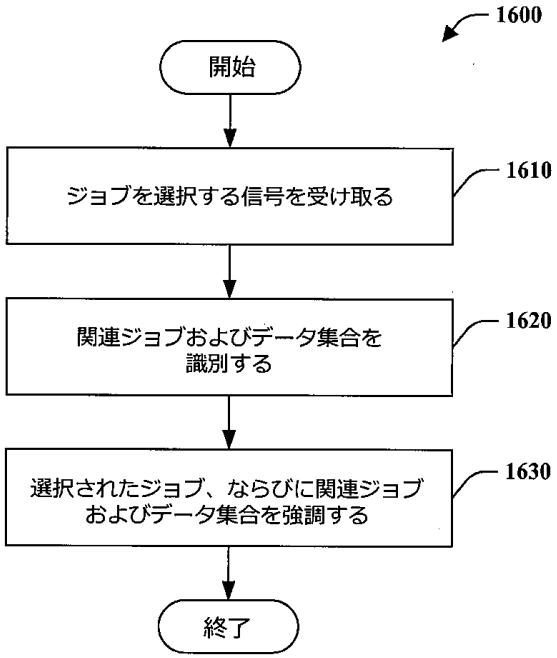

FIG. 16

【図１７】

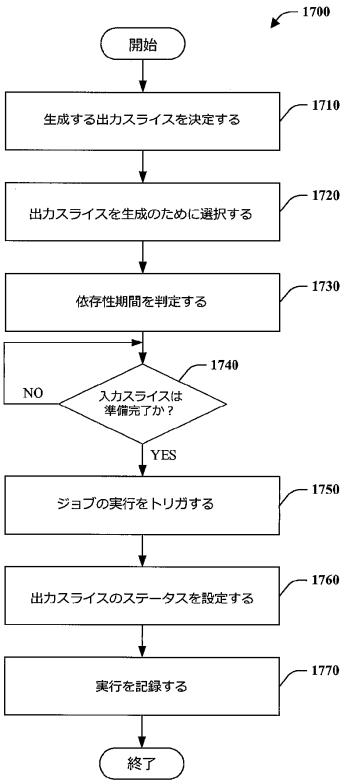

FIG. 17

【図１８】

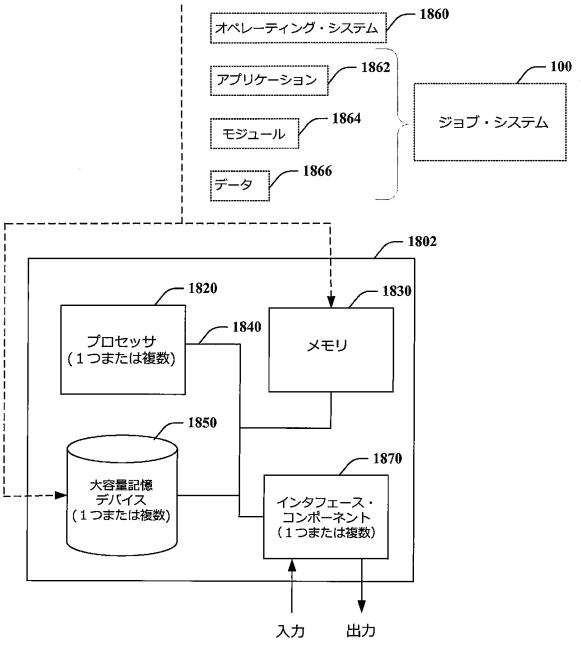

FIG. 18

フロントページの続き

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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 クーリス，シェリル
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 ストーム，クリスティナ
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 ネッツ，アミール
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 チュン，チウ・イーン
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 フラスコ，マイケル・ジェイ
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 グリーリッシュ，ケヴィン
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 デラ－リベラ，ジョヴァンニ・エム
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 カールソン，ソニア・ピー
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 ヘニンガー，マーク・ダブリュー
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 バック，ポーラ・エム
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）
(72)発明者 ネトルトン，デーヴィッド・ジェイ
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，マイクロソフト テクノロジー ライセンシング，エルエルシー，パテント・グループ・ドケッティング（ビルディング ８／１０００）

合議体
審判長 篠原 功一

審判官　　金子 秀彦
審判官　　山崎 慎一
(56)参考文献　　特開平１１－０９６０２４（ＪＰ，Ａ）
特開２００９－２３０５８４（ＪＰ，Ａ）
国際公開第２０１０／００１５５５（WO，Ａ１）
特開２００６－２４３９９６（ＪＰ，Ａ）
(58)調査した分野　(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　９／４５５　－　９／５４