

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第5331203号
(P5331203)

(45) 発行日 平成25年10月30日(2013.10.30)

(24) 登録日 平成25年8月2日(2013.8.2)

(51) Int.Cl.

F I

G 0 6 F 11/34 (2006.01)

G 0 6 F 11/34

B

請求項の数 20 (全 35 頁)

(21) 出願番号 特願2011-516825 (P2011-516825)
 (86) (22) 出願日 平成21年6月30日(2009.6.30)
 (65) 公表番号 特表2011-527052 (P2011-527052A)
 (43) 公表日 平成23年10月20日(2011.10.20)
 (86) 国際出願番号 PCT/US2009/049299
 (87) 国際公開番号 W02010/002919
 (87) 国際公開日 平成22年1月7日(2010.1.7)
 審査請求日 平成24年4月23日(2012.4.23)
 (31) 優先権主張番号 61/077,022
 (32) 優先日 平成20年6月30日(2008.6.30)
 (33) 優先権主張国 米国(US)

(73) 特許権者 509123208
 アビニシオ テクノロジー エルエルシー
 アメリカ合衆国 02421 マサチュー
 セッツ州 レキシントン スプリング ス
 トリート 201
 (74) 代理人 100079108
 弁理士 稲葉 良幸
 (74) 代理人 100109346
 弁理士 大貫 敏史
 (72) 発明者 スチュダー スコット
 アメリカ合衆国 01833 マサチュー
 セッツ州 ジョージタウン ピルスベリー
 レーン 27

最終頁に続く

(54) 【発明の名称】 グラフベース計算におけるデータ記録

(57) 【特許請求の範囲】

【請求項1】

データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様であって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義するルール仕様、を受け取るステップ；

前記受け取られたルール仕様に基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式を生成するステップであって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスを提供することを含むステップ；および、

前記生成された変換式を用いて少なくとも一つのデータフローを変換するステップであって、ランタイムにて前記グラフベース計算における前記データ処理成分の実行をトレースすること、前記設定されたログ特性に準じて前記トレースされた実行に基づいてログ情報を生成すること、および、前記生成されたログ情報を格納または出力すること、を含むステップ、

を含む方法。

【請求項2】

請求項1に記載の方法において、

前記ログの特性を設定することは、ログ情報が生成される少なくとも一つのイベントを選択することを含む、方法。

【請求項 3】

請求項 2 に記載の方法において、

前記少なくとも一つのイベントは、前記受け取られるルール仕様に関連付けられたルールに準じて前記データフローに係るレコードを変換することに関連付けられる、方法。

【請求項 4】

請求項 3 に記載の方法において、

前記少なくとも一つのイベントは、変換されるレコードに係るエラーを特定することを含む、方法。

【請求項 5】

請求項 3 に記載の方法において、

前記少なくとも一つのイベントは、与えられたレコードについてのルールケースの条件を満たすことを含む、方法。

10

【請求項 6】

請求項 5 に記載の方法において、

与えられたレコードについてのルールケースの条件を満たすことは、前記与えられたレコードに係る値に基づく論理式を満たすことを含む、方法。

【請求項 7】

請求項 5 に記載の方法において、

与えられたレコードについてのルールケースの条件を満たすことは、前記レコードに係る値と前記ルールケースに関連付けられた値とを比較することを含む、方法。

20

【請求項 8】

請求項 1 に記載の方法において、

ログ情報を生成することは：

前記受け取られたルール仕様によって定義される前記ルールを含むルールセットの詳細を各々含む一または複数の凡例ログメッセージを生成すること、および、

複数のトレースログメッセージであって、各トレースログメッセージは凡例レコードと関連付けられるとともに、前記受け取られたルール仕様に関連付けられたルールに準じて前記データフローに係るレコードを変換することに関連付けられた少なくとも一つのイベントを記述するトレースログメッセージ、を生成すること、

を含む、方法。

30

【請求項 9】

請求項 8 に記載の方法において、

前記イベントを記述するトレースログメッセージは、前記凡例レコードに係るインデックスを用いてデータ処理成分の少なくとも一つの入力または出力を記述する、方法。

【請求項 10】

請求項 8 に記載の方法において、

一または複数の凡例メッセージを生成することは、グラフベース計算の一回の実行ごとに一つの凡例メッセージを生成することを含む、方法。

【請求項 11】

請求項 1 に記載の方法において、

前記生成された変換式を用いる前記データ処理成分は、前記受け取られたルール仕様に関連付けられた第 1 のルールに準じて前記データフローに係る入力レコードを変換する、方法。

40

【請求項 12】

請求項 11 に記載の方法において、

前記トレースされた実行に基づいてログ情報を生成することは、前記規定された判断基準が満たされる前記第 1 のルールの各ルールケースに対してログメッセージを生成することを含む、方法。

【請求項 13】

請求項 11 に記載の方法において、

50

前記トレースされた実行に基づいてログ情報を生成することは、前記生成された変換式を用いて前記データ処理成分によって生成された変換されたレコードにおける各値に対してログメッセージを生成することを含む、方法。

【請求項 14】

請求項 11 に記載の方法において、

前記生成されたログ情報を格納または出力することは、前記データ処理成分の一または複数のログポートからログメッセージを出力することを含む、方法。

【請求項 15】

請求項 14 に記載の方法において、

前記生成されたログ情報を格納または出力することは、データ処理成分における前記ログポートからのログメッセージに係るデータフローを受け取ること、および、前記ログメッセージの少なくとも一部から派生するログ情報を格納すること、をさらに含む、方法。

10

【請求項 16】

請求項 15 に記載の方法であって、

ログメッセージに係る前記受け取ったデータフローをフィルタリングすること、および、前記ログメッセージのサブセットから派生するログ情報を格納すること、をさらに含む、方法。

【請求項 17】

請求項 14 に記載の方法において、

前記生成されたログ情報を格納または出力することは：

20

前記規定された判断基準が満たされるトリガされたルールケースを指し示すデータ処理成分における前記ログポートからのログメッセージに係るデータフローを受け取ること、

前記入力レコードの全てによってトリガされる前記ルール仕様によって定義される前記ルールの各々に係る各ルールケースに対する少なくとも一つのログメッセージを提供する入力レコードの縮小セットを決定するために前記ログメッセージを検査すること、および、

前記入力レコードの縮小セットを格納すること、

をさらに含む、方法。

【請求項 18】

データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様を格納する格納システムであって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義する、格納システム；

30

前記受け取られたルール仕様に基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式を生成するように構成された生成器であって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスを提供する生成器；および、

前記生成された変換式を用いて少なくとも一つのデータフローを変換するように構成されたグラフベース計算システムであって、ランタイムにて前記グラフベース計算における前記データ処理成分の実行をトレースすること、前記設定されたログ特性に準じて前記トレースされた実行に基づいてログ情報を生成すること、および、前記生成されたログ情報を格納または出力すること、を含むグラフベース計算システム、

40

を含むコンピュータシステム。

【請求項 19】

コンピュータ読み取り可能な媒体に格納されたコンピュータプログラムであって、該コンピュータプログラムは：

データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様であって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義するルール仕様、の受け取り；

50

前記受け取られたルール仕様にに基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式の生成であって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスの提供を含む、該変換式の生成；および、

前記生成された変換式を用いた少なくとも一つのデータフローの変換であって、ランタイムにおける前記グラフベース計算における前記データ処理成分の実行のトレース、前記設定されたログ特性に準じて前記トレースされた実行に基づくログ情報の生成、および、前記生成されたログ情報の格納または出力、を含む該データフローの変換、

をコンピュータシステムに実行させる命令を含むコンピュータプログラム。

【請求項 20】

データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様を受け取る手段であって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義する、該ルール仕様を受け取る手段；

10

前記受け取られたルール仕様にに基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式を生成する手段であって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスを提供することを含む手段；および、

前記生成された変換式を用いて少なくとも一つのデータフローを変換する手段であって、ランタイムにて前記グラフベース計算における前記データ処理成分の実行をトレースすること、前記設定されたログ特性に準じて前記トレースされた実行に基づいてログ情報を生成すること、および、前記生成されたログ情報を格納または出力すること、を含む手段、

20

を含むコンピュータシステム。

【発明の詳細な説明】

【関連する出願の相互参照】

【0001】

本出願は、2008年6月30日に提出された米国出願 No. 61/077,022 に基づく優先権を主張するものであり、その内容はここに参照として取り込まれる。

【技術分野】

【0002】

本記載は、一般に、グラフベース計算におけるデータ記録に関する。

30

【背景技術】

【0003】

複雑な計算は、有向グラフによるデータフロー（該計算の成分がグラフの頂点に関係付けられ、各成分間のデータフローがグラフのリンク（弧、辺）に対応する。）として表現し得ることが多い。このようなグラフベース計算を採用するシステムは、米国特許第5,966,072号、「グラフとして表現される計算の実行」に記載されている。頂点と関係付けられる上記計算は、人が読み取れる形式にて記述される場合もあり、「ビジネスルール」とも称呼される

【発明の概要】

40

【0004】

一の態様において、一般に、方法は、データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様であって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義するルール仕様、を受け取るステップ；前記受け取られたルール仕様にに基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式を生成するステップであって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスを提供することを含むステップ；および、前記生成された変換式を用いて少なくとも一つのデータフローを変換するステップであって、ランタイムにて前記グラフベース計算における前記データ

50

処理成分の実行をトレースすること、前記設定されたログ特性に準じて前記トレースされた実行に基づいてログ情報を生成すること、および、前記生成されたログ情報を格納または出力すること、を含むステップ、を含む。

【0005】

いくつかの態様には、後述する特徴の一または複数が含まれ得る。

【0006】

前記ログの特性を設定することは、ログ情報が生成される少なくとも一つのイベントを選択することを含む。

【0007】

前記少なくとも一つのイベントは、前記受け取られるルール仕様に関連付けられたルールに準じて前記データフローに係るレコードを変換することに関連付けられる。

10

【0008】

前記少なくとも一つのイベントは、変換されるレコードに係るエラーを特定することを含む。

【0009】

前記少なくとも一つのイベントは、与えられたレコードについてのルールケースの条件を満たすことを含む。

【0010】

与えられたレコードについてのルールケースの条件を満たすことは、前記与えられたレコードに係る値に基づく論理式を満たすことを含む。

20

【0011】

与えられたレコードについてのルールケースの条件を満たすことは、前記レコードに係る値と前記ルールケースに関連付けられた値とを比較することを含む。

【0012】

ログ情報を生成することは：

前記受け取られたルール仕様によって定義される前記ルールを含むルールセットの詳細を各々含む一または複数の凡例ログメッセージを生成すること、および、

複数のトレースログメッセージであって、各トレースログメッセージは凡例レコードと関連付けられるとともに、前記受け取られたルール仕様に関連付けられたルールに準じて前記データフローに係るレコードを変換することに関連付けられた少なくとも一つのイベントを記述するトレースログメッセージ、を生成すること、を含む。

30

【0013】

前記イベントを記述するトレースログメッセージは、前記凡例レコードに係るインデックスを用いてデータ処理成分の少なくとも一つの入力または出力を記述する。

【0014】

一または複数の凡例メッセージを生成することは、グラフベース計算の一回の実行ごとに一つの凡例メッセージを生成することを含む。

【0015】

前記生成された変換式を用いる前記データ処理成分は、前記受け取られたルール仕様に関連付けられた第1のルールに準じて前記データフローに係る入力レコードを変換する。

40

【0016】

前記トレースされた実行に基づいてログ情報を生成することは、前記規定された判断基準が満たされる前記第1のルールの各ルールケースに対してログメッセージを生成することを含む。

【0017】

前記トレースされた実行に基づいてログ情報を生成することは、前記生成された変換式を用いて前記データ処理成分によって生成された変換されたレコードにおける各値に対してログメッセージを生成することを含む。

【0018】

前記生成されたログ情報を格納または出力することは、前記データ処理成分の一または

50

複数のログポートからログメッセージを出力することを含む。

【 0 0 1 9 】

前記生成されたログ情報を格納または出力することは、データ処理成分における前記ログポートからのログメッセージに係るデータフローを受け取ること、および、前記ログメッセージの少なくとも一部から派生するログ情報を格納すること、をさらに含む。

【 0 0 2 0 】

前記方法は、ログメッセージに係る前記受け取ったデータフローをフィルタリングすること、および、前記ログメッセージのサブセットから派生するログ情報を格納すること、をさらに含む。

【 0 0 2 1 】

前記生成されたログ情報を格納または出力することは：

前記規定された判断基準が満たされるトリガされたルールケースを指し示すデータ処理成分における前記ログポートからのログメッセージに係るデータフローを受け取ること、

前記入力レコードの全てによってトリガされる前記ルール仕様によって定義される前記ルールの各々に係る各ルールケースに対する少なくとも一つのログメッセージを提供する入力レコードの縮小セットを決定するために前記ログメッセージを検査すること、および、

前記入力レコードの縮小セットを格納すること、をさらに含む。

【 0 0 2 2 】

他の態様において、一般に、コンピュータシステムは、データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様を格納する格納システムであって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義する、格納システム；前記受け取られたルール仕様に基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式を生成するように構成された生成器であって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスを提供する生成器；および、前記生成された変換式を用いて少なくとも一つのデータフローを変換するように構成されたグラフベース計算システムであって、ランタイムにて前記グラフベース計算における前記データ処理成分の実行をトレースすること、前記設定されたログ特性に準じて前記トレースされた実行に基づいてログ情報を生成すること、および、前記生成されたログ情報を格納または出力すること、を含むグラフベース計算システム、含む。

【 0 0 2 3 】

他の態様において、一般に、コンピュータプログラムは、コンピュータ読み取り可能な媒体に格納され、該コンピュータプログラムは：データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様であって、該ルール仕様は入力データに応じた一または複数の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義するルール仕様、の受け取り；前記受け取られたルール仕様に基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式の生成であって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスの提供を含む、該変換式の生成；および、前記生成された変換式を用いた少なくとも一つのデータフローの変換であって、ランタイムにおける前記グラフベース計算における前記データ処理成分の実行のトレース、前記設定されたログ特性に準じて前記トレースされた実行に基づくログ情報の生成、および、前記生成されたログ情報の格納または出力、を含む該データフローの変換、をコンピュータシステムに実行させる命令を含む。

【 0 0 2 4 】

他の態様において、一般に、コンピュータシステムは、データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算についての少なくとも一つのルール仕様を受け取る手段であって、該ルール仕様は入力データに応じた一または複数

10

20

30

40

50

の出力値を定めるための判断基準を規定する一または複数のルールケースと各々関連付けられたルールを定義する、該ルール仕様を受け取る手段；前記受け取られたルール仕様に基づくグラフベース計算における少なくとも一つのデータ処理成分に対する変換式を生成する手段であって、前記生成された変換式に関連付けられたログの特性を設定するインターフェイスを提供することを含む手段；および、前記生成された変換式を用いて少なくとも一つのデータフローを変換する手段であって、ランタイムにて前記グラフベース計算における前記データ処理成分の実行をトレースすること、前記設定されたログ特性に準じて前記トレースされた実行に基づいてログ情報を生成すること、および、前記生成されたログ情報を格納または出力すること、を含む手段、を含む。

【0025】

10

以下、本発明の一または複数の実施形態の詳細について、図面および記述とともに説明する。本発明の他の特徴、目的および利点は、同記述、同図面および請求の範囲から理解され得る。

【図面の簡単な説明】

【0026】

【図1A】グラフベース計算環境における入力データから出力データへの変換を表す。

【図1B】グラフベース計算におけるグラフの一例を表す。

【図1C】情報の記録とともに変換式を生成することについてのブロック図を表す。

【図2A】スプレッドシート型のルール項目の一例を表す。

【図2B】個別型のルールの一例を表す。

20

【図3】グラフベース計算環境においてデータ記録を実行することについてのフローチャートを表す。

【図4】グラフベース計算環境における記録方式の設定についてのグラフィカルユーザインターフェイスの一例を表す。

【発明を実施するための形態】

【0027】

データ記録および監査のメカニズムの一例として、グラフベース計算システムにおけるグラフベース計算に関連付けられるメタデータを保持したグラフベース計算構造が挙げられる。本例において、各々のコンピュータプログラムは、計算グラフ（データフローグラフまたは単にグラフとも称呼される。）を用いて実行される。グラフは、データ処理成分を表す一または複数の節点または頂点であって、成分間のデータの流れを表す有向辺によって接続された節点または頂点を含む。グラフは、並行処理環境において実行され得る。システムは、グラフの作成における変化を記録し、統計的依存性の分析を行い、グラフの作成に関するメタデータを管理する。グラフに関連するメタデータを保管することにより、生じ得るデータインパクトの分析（データがグラフにおいて実行されているときにどのように変化するか、および、他のグラフにおいてその変化が与える影響を、ユーザに対して視覚的に表現すること）が可能となる。さらに、コードが変更されても複数のバージョンのグラフを保管することを可能とする設定/変更マネジメントをシステムが提供することにより、最新のコードおよびデータが確実に入手され得る。

30

【0028】

40

メタデータのサブセットとしてのビジネスルールは、システムに格納される。ビジネスルールの種々の態様は、例えば米国出願No. 11/733,434に記載されており、その内容はここに参照として取り込まれる。各々のビジネスルールは、個々のオブジェクトに格納され得る。ビジネスルールは、データを一のフォーマットから他のフォーマットへ変換するための判断基準、データに関する決定をするための判断基準、または、入力データのセットに基づいて新たなデータを生成するための判断基準、のセットとして表現され得る。例えば、図1Aにおいて、フライト予約システムにおけるレコード102は、各フィールドに格納される値として、乗客の名前104、乗客の今年度の飛行マイル106、乗客のチケットのクラス108、および、乗客の座席の行110、を含む。ビジネスルールは、この乗客が搭乗グループ1に属すべきことを指示する。ビジネスルールは、一般

50

に人間には容易に理解されるものの（例えば、「ファーストクラスの乗客はグループ 1 に属する」）、データを操作するために同ビジネスルールを用いる際にはコンピュータが理解可能な形態に翻訳される必要がある場合がある。

【 0 0 2 9 】

グラフベース計算環境においてビジネスルールを実行するために、変換式 1 1 2 が生成される。同変換式 1 1 2 は、一または複数のデータソース（例えば、入力データセット 1 0 0）からレコード 1 0 2 のような入力レコードを受け取り、出力レコード（例えば、出力データセット 1 2 0 に用いられる乗客名 1 0 4 および乗客の属するグループ 1 1 8 を示すレコード 1 1 4）を提供する。本例において、上記データセットはレコードの一例を示すものであり、一般にデータセットに含まれるレコードの数は制限されない。入力および出力データセットは、データストリームとして処理され得る（例えば、データセットを構成するデータがグラフに流入または流出するように）。

10

【 0 0 3 0 】

そこで、変換式は、データフローを表すリンク要素によって接続されるデータ処理成分を有するグラフベース計算に実装され得る。例えば、図 1 B における単純な計算グラフ 1 3 0 においては、2 つのデータセット 1 3 2 および 1 3 4（例えば、利用頻度の高い乗客のデータおよびフライト予約データ）が入力され、各データセットのデータが一体として使用可能となるように同データがフォーマット成分 1 3 6 および 1 3 8 において個別にフォーマットされ、そして、結合部分 1 4 0 においてそれらが結合されて出力データセット 1 4 2 が生成される。変換式は、計算グラフ 3 0 内にあるようなグラフベース計算そのものであってもよく、グラフの成分（例えば、グラフ 1 3 0 を構成する個々の成分 1 3 6，1 3 8 および 1 4 0）内に実装されているものよい。

20

【 0 0 3 1 】

技術分野に詳しくないユーザであっても容易に変換式を生成することができるように、ルールセットとも称呼されるビジネスルールのセットをそれらユーザが理解し易い形式にて入力するツールが提供される。同ツールは、ユーザが変換式に実行させたい内容をコンピュータシステムに伝えるものである。一の変換式を生成するルールのセットが一のルールセットである。また、ルールセットは、入力に応じてルールの異なる出力値を決定する一または複数のルールケースによって構成され得る。また、ルールセットは、他のルールを含んでもよい。ルールセットに含まれる他のルールは、追加の出力値または代替の出力値を決定してもよい。また、ルールセットは、「被包含」ルールセットとも称呼される他のルールセットを含んでもよい。

30

【 0 0 3 2 】

図 1 C は、情報を記録しながら変換式を生成するシステムの一般的なモデルを示す。ビジネスルール環境（BRE）は生成器 1 5 0 を含む。同生成器 1 5 0 は、エディタ 1 5 4 からルールセット 1 5 2 を入力として受け取って変換式 1 5 6 を生成する。変換式生成のオプションの一つとして、それに続いて、グラフィカルユーザインターフェイスにおいて種々の記録イベントおよび情報のカスタマイズによるロギングが行われ得る。ログは、所定のシステムおよびネットワーク内にて生じるイベントについての記録である。ログはエントリーによって構成され、各エントリーはシステムおよびネットワークにて生じる特定のイベントに関連する情報を含む。ログは、トラブルシューティング、および、種々の機能（システムやネットワークのパフォーマンスの最適化、ユーザの操作の記録、ならびに、異常処理の調査に有用な情報の提供など）に供するように用いられ得る。また、ログは、種々の異なる種類のイベントに関連する情報を含み得る。また、生成された変換式 1 5 6 は、システムのアーキテクチャならびに変換式およびビジネスルールの目的に応じて、一のグラフまたはグラフ全体そのもので使用されるべき成分としてグラフベース計算システム 1 5 8 で提供され得る。生成器 1 5 0 として、例えば、コンパイラ、特注のプログラム、または、ルールセット 1 5 2 を受け取って変換式 1 5 6 を出力するための標準的なツールを用いて構成される他のグラフベース計算、が挙げられる。

40

【 0 0 3 3 】

50

生成器 150 は、ルールセット 152 が編集されたときに変換式 156 をアップデートし得る。また、エディタ 154 は、ルールセット 152 が編集されたときにそのルールセット全体を提供してもよく、新しいまたは変更されたルールまたは新しいまたは変更されたルールケース 152 a だけを提供してもよい。また、生成器 150 は、変換式を用いるシステムの性能および要求に応じて、元の変換式 156 の全体を置き換えるための新しい変換式を生成してもよく、変換式を含む成分 156 a を提供してもよい。

【0034】

グラフベース計算 158 の際にロギングを行うための個別・専用の実行エンジンは、必要ではない。ロギングは、その記録が行われる際にグラフ成分によって呼び出される機能を用いて行われるように構成され得る。記録の異なる態様として、ルールの実行について
10
の監査報告が提供され得る。例えば、図 1C において（実線矢印によって示される実際のデータフローとは異なる）破線矢印に示されるように、ルールセット 152 内の特定の入力記録にログ 160 a が戻されてもよいし、事前に実行された特定のルールケース 152 a にログ 160 b が反映されてもよい。

【0035】

図 2A に示すように、いくつかの例において、スプレッドシートのフォーマットにルールが入力され得る。スプレッドシート 200 のトリガ列 202, 204, 206, 208
20
が入力可能なデータ値に対応し、行 210 a ~ h がルールケース（すなわち、入力可能なデータ値に関連する判断基準のセット）に対応する。ルールケースが判断基準を備えているトリガ列それぞれに関し、所定のレコード（例えば、図 1A における 102）内のデータ値がトリガ判断基準を満たしていれば、そのレコードにルールケース 210 n が適用される。ルールケース 210 n が適用されると、一または複数の出力列 212 に基づいて出力が生成される。ここで、全てのトリガ判断基準が満たされたルールケースは、「トリガされた」とも称呼される。各出力列 212 は潜在的な出力変数に対応し、その出力変数に関し、適用可能な行 210 n の対応するセル内の値が（もしあれば）出力を決定する。後述するように、セルは、上記変数に割り当てられる値を含むことができるし、出力値を生成するために計算されなければならない数式を含むこともできる。図 2A においては出力列は一のみ示されているが、一よりも多い出力列が設けられていてもよい。

【0036】

変数に対応する列、数式ではあるが一旦計算された後に変数のように扱われる数式を格納した列、および、数式のみを格納した列といった様々な幾つかの形式のトリガ列が挙げられる。他の形式の列の例として、データのみを格納した列、および、データのみを各ノズルした列に基づいて列毎に計算する数式を特定する列が挙げられる。数式のみを格納した列は、変数に対応する列または変数として扱われる列よりも簡便である。
30

【0037】

図 2A の例においては、第 1 の行 210 a は列 202 のみに判断基準を有し、これは乗客の合計飛行マイルが 1,000,000 以上であれば他の列の値が如何なる値であるかに係わらずルールケースが適用されることを表している。この場合、このユーザに対する出力変数「搭乗グループ」はグループ 1 に設定される。同様に、第 2 のルールケース 210 b は、ファーストクラスの乗客であれば常にグループ 1 に設定されることを表している
40
。いくつかの例においては、ルールが順に評価されれば、1,000,000 マイル以上を有し且つファーストクラスのチケットを有する乗客はグループ 1 に設定されるが、第 1 のルールケース 210 a のみがトリガされる。一のルールケースがトリガされれば、そのルールの他のルールケースは評価される必要はない。

【0038】

次のルールケース 210 c は 2 つの入力値 202 および 204 に基づくものであり、合計飛行マイルおよび今年度のマイルの双方の判断基準が満たされれば、乗客はグループ 2 に設定される。第 4 のルールケース 210 d においては、ビジネスクラスの乗客であれば常にグループ 2 に設定される。残りのルールケース 210 e ~ h は、他のルールケースに
50
関係する判断基準（すなわち、「他」および「同上」）を有する。「他」は、その行より

も上の行であってその左側において同じ判断基準を有する行（すなわち、ルール 2 1 0 b および 2 1 0 d）において、その列におけるいずれの判断基準も満たされないことを表している。一方、「同上」は、その列におけるその上のルールケースが適用された場合にそのルールケースが適用されることを表している。よって、第 5 のルールケース 2 1 0 e は、2 つの列 2 0 2 または 2 0 4 における判断基準がいずれも満たされず（満たされていれば、ルールケース 2 1 0 a または 2 1 0 c によって処理される）、かつ、「座席クラス」の列が「ファースト」または「ビジネス」ではなく（列 2 0 6 が「他」であるから）、かつ、「座席の行」の値 2 0 8 が 1 0 以下である記録に適用される。残りのルールケース 2 1 0 f ~ h は、列 2 0 2 および 2 0 4 に関する上のいずれのルールケースにも適合せず、且つ、「座席クラス」の列が「ファースト」または「ビジネス」ではなく、かつ、「座席の行」に適切な値がある記録に適用される。

10

【 0 0 3 9 】

図 2 A の例におけるルールケース 2 1 0 a ~ h は、図 2 B に示すように各々がスプレッドシートに格納された個別の単純なルールとしても表され得る。ルール 2 2 0 a ~ d は図 2 A の行 2 1 0 a ~ d にそれぞれ対応し、ルール 2 2 0 e は行 2 1 0 e ~ h に対応する 4 つのルールケースを有する。ユーザは、図 2 A に示すテーブル全体を生成するのではなく、これら個々のルールを個別に生成することができる。各ルールケースは、全てのトリガ列に値を格納しているとともに全ての出力列に値を格納している（値は空欄であってもよい。すなわち、「いずれも可」に設定されてもよい。）。複数のルールが同じ出力を生成する場合、それらルールは順序付けられ、一のルール内のルールケースが入力によってトリガされて出力を生成するまで順に検討される。一のルール内のいずれのルールケースもトリガされなければ、同じ出力を生成する次のルールが処理される。全てのルール内のいずれのルールケースもトリガされなければ、初期設定値が使用される。

20

【 0 0 4 0 】

いくつかの例において、エディタのインターフェイス 1 5 0 は、数式を格納したセルをグラフィカルに特定してもよい。これにより、真か偽かが評価される数式と、行の変数と比較される値を返す数式と、の違いをユーザが理解し易くなる。入力の際、ユーザは、例えばアスタリスクを最初に入力することによって特定のセルが数式セルであることを指定し得る。

【 0 0 4 1 】

出力変数に対応する列について、以下のうちの 하나가セルに格納され得る。

- ・ 値

出力変数に割り当てられる値。

- ・ 数式

この数式の値が出力変数に割り当てられる。数式によって NULL が求められた場合、出力フィールドが NULL 禁止でなければ、フィールドに NULL が設定される。出力フィールドが NULL 禁止であれば、エラーが生成される。

- ・ 「NULL」の文字

出力フィールドが NULL 容認であれば、フィールドに NULL が割り当てられる。そうでない場合、エラーが生成される。

- ・ 空の文字列

出力フィールドが初期設定値であれば、初期設定値が割り当てられる。そうでない場合、セルは NULL が格納されているように扱われる。

- ・ 「同上」の文字

出力フィールドに上のセルにて計算された値と同じ値が割り当てられる。

【 0 0 4 2 】

可能であれば、エラーが検出されたとき（すなわち、NULL 禁止フィールドである出力列に NULL が設定されたとき）、エラーが報告される。しかし、テストタイムまたはランタイムまで報告されないエラーもあり得る。

【 0 0 4 3 】

50

テーブルの行として生成された場合でも個別のルールとして生成された場合でも、各ルールは所定の属性のセットを有する。ルールセットは、同ルールセットに含まれるルールに適用されるこれら属性を定めてもよい。これら属性の例として、名前、ルールタイプ、説明やコメントフィールド、出力変数のリスト、入力変数のリスト、引数のリスト、トリガ列のリスト、更新履歴、テストデータセット、および、エラー処理動作、が挙げられる。名前は読んで字の如くであり、ルールをルールセット内に並べるために用いられる。いくつかの例において、ルールタイプはルールセットのプロパティである。出力変数のリストは、ルールによって生成または割り当てられた変数のセットである。これはルールセットから引き継いでもよく、出力が一または複数となってもよい。入力変数のリストは、ルールがレコードを評価するために必要とする全ての変数を特定するものであり、列の最上段の変数および数式内にて用いられる変数（例えば、図 2 A におけるルール 2 1 0 c にて用いられる「前年度の飛行マイル」値は、数式内にて用いられるものの、それ自体の列を有さない。）が含まれる。

10

【 0 0 4 4 】

ルールは、単独発動型であっても複数発動型であってもよい。例えば、複数のルールケースが一または複数の出力のための複数の値を生成するために用いられてもよい。複数のルールケースをトリガすることができるルールは、複数発動型ルールとも称呼される。複数発動型ルールは、そのルールによって計算される出力のタイプに基づいてのみ特定される。ルールによって計算される出力がリスト（各レコードについて複数の値を有し得る出力）であれば、ルールは複数発動型ルールである。複数発動型ルールにおいて、ルールケースが一旦トリガされると、対応する出力値は出力のための値のリストに付け加えられる。しかし、単独発動型ルールと異なり、複数発動型ルールにおいては、一のルールケースがトリガされた後においても評価は継続される。各々のルールケースが順に評価され、トリガされる全てのルールケースの出力値が出力のための値のリストに付け加えられる。

20

【 0 0 4 5 】

いくつかの例において、上述した例とは逆に、ルールは、行におけるルールケースを A N D 形式にて評価するとともに列を O R 形式にて評価してもよい。つまり、一つのセルが真でありさえすれば各々の行はトリガされ（列は O R 形式）、全ての行がトリガされた場合にのみルールが出力を生成する（行は A N D 形式）。

【 0 0 4 6 】

引数のリストは、関数ルールに対してのみ存在する。そのリストは、ルールに入力されるパラメータの名前および形式を特定するものであり、ルールセットのプロパティであってもよい。トリガ列のリストは、どの列がルールのアプリケーションをトリガし得るかを特定する。図 2 A および図 2 B に示される入力変数に限られず、トリガ列には、パラメータ、ルックアップ変数、先に実行されるルールからの出力変数、被包含ルールセットの出力変数、ルールセットのパラメータ、または、数式、が対応され得る。トリガ列には、関数ルール（すなわち、引数）からの出力変数が含まれ得る。

30

【 0 0 4 7 】

エラー処理は、ルールを評価する際に生じるエラーをルールセットから生成された変換式がどのように扱うかを決定する。トリガ数式におけるエラーの処理の選択肢として、エラーを許可すること（この場合、変換式はエラーを引き起こすレコードを拒否する。）、または、エラーを無視すること（トリガ数式が偽であるとみなすことと同等であり、次の数式に移動する。）、が挙げられる。出力数式に対しては、エラーを許可してレコードを拒否すること、エラーを無視して出力に N U L L を設定すること、または、その行を無視して次の行に進むこと、によってエラーが処理され得る。

40

【 0 0 4 8 】

上述したように、変換式はルールセットから生成される。ルールセットは、後述する属性を有し得る。

【 0 0 4 9 】

< 名前、記述およびコメント >

50

これらはルールセットを特定する。バックエンド実行に応じて、ルールセットは、システム内の自らの位置についての識別情報を含み得る。いくつかの例において、ルールセットの位置は、プロジェクト内のパスである。いくつかの例において、ルールセットは、リレーショナルデータベース内にて体系化され、名前によって位置を特定される。変更履歴は、変更名、日付、および、チェックインコメントを含み得る。

【 0 0 5 0 】

< 変換式のタイプ >

これは、ルールセットから生成される変換式のタイプを決定する。後述するように、可能な値の例として、リフォーマット、ジョイン、ロールアップおよびフィルタが挙げられる。

10

【 0 0 5 1 】

< 入力データセット >

これらは、編集のためのフィールドのリストおよび名前付き定数を提供する。いくつかの例において、初期設定により、変換式が生成されるとき、入力データセットのうちの一つのレコードフォーマットが採用されることになる。異なる環境に対してルールセットが変換式を生成することが許容されるように、複数の入力データセットがあってもよい。これにより、論理対物理マップの複数のセット（すなわち、物理名のセットが異なるもの）も許容される。いくつかの例においては、一または複数のデータセットを備えた入力マッピングテーブルがあってもよい。いくつかの例においては、ジョイン成分は、複数の入力マッピングテーブルを有してもよく、その各々が複数のデータセットを有してもよい。

20

【 0 0 5 2 】

< 出力データセット >

これらは、出力フィールド名のリストを提供する。初期設定により、変換式が生成されるとき、出力データセットうちの一つのレコードフォーマットが採用されることになる。出力データセットは入力データセットと同一であってもよい。被包含ルールセットは、出力データセットを有さない。いくつかの例において、入力データセットと同様に、異なる環境に対してルールセットが変換式を生成することが許容されるように、複数の出力データセットがあってもよい。

【 0 0 5 3 】

< 被包含ルールセットのリスト >

一のルールセットは他のルールセットによって計算された出力フィールド（出力レコードフォーマットのフィールドではなく、明示的にリストにされた出力フィールド）を使用し得る。包含ルールセット内に視認される被包含ルールセットの出力変数のセットを定める被包含ルールセットマッピングテーブルに基づき、被包含ルールセットにおける出力変数は、包含ルールセットにおける変数として用いられ得る。

30

【 0 0 5 4 】

< 被包含変換式ファイルのリスト >

ルールセットが処理されるときに用いられる変換式を特定する一または複数のファイルが、オプションとして含まれ得る。

【 0 0 5 5 】

< 変数と定数とをリストにする一連のマッピングテーブル >

これらテーブルは、入力データセットおよび出力データセットに関連付けられる。それらは、変数のリストをエディタに知らせ、ビジネス名と技術名との間のマップを文書化する。各変数は、ビジネス名、（数式を用いて計算され得る）技術名、および、ベースタイプ（文字列、数字、日付または日時）を有する。ビジネス名と技術名との間のマップを文書化する定数のリストが、オプションとして各変数に関連付けられる。変数テーブルの詳細は、後述される。

40

【 0 0 5 6 】

< 外部テストデータファイルへの参照 >

上述した組み込みテストデータセットと同様、テストファイルはルールをテストするた

50

めに用いられる。

【 0 0 5 7 】

< 非拒否フラグ >

このフラグが設定されているとき、ルールセットによって生成された変換式はレコードを拒否しない（エラーを生じる）。このフラグは、エラーを生じるルールがトリガされないかのように、そのルールを無視するように用いられる。

【 0 0 5 8 】

< デプロイメントテーブル >

これは、どのルールが各ビルドに含まれるべきかを（間接的に）示す—または複数のデプロイメントをリストにする。デプロイメントテーブルの詳細は、後述される。

10

【 0 0 5 9 】

< オプションキー >

これは、ジョイン型またはロールアップ型のルールセットのためのキーを表す特別な入力フィールドのビジネス名をユーザが特定できるようにする。いくつかの例において、キーは入力値のテーブルの項目として実際に採用される。

【 0 0 6 0 】

< ルックアップファイルのオプションリスト >

これは、ビジネス名、キー情報、および、入力変数および定数の完全なテーブル、ルックアップファイル毎に1つのテーブル、を提供する。ルックアップファイルのサポートの詳細は、後述される。

20

【 0 0 6 1 】

< パラメータのテーブル >

これは、環境またはランタイムにおけるパラメータに由来する変数をリストする。

【 0 0 6 2 】

ルールセットが関連付けられるいくつかの異なるテーブルとして：

1．入力変数および定数のテーブル

変換型ルールセットとして、このテーブルは、ルール内で参照される入力レコードフォーマットのフィールドを含む。レコードフォーマットの全てのフィールドがリストされる必要はないが、通常はそうにされる。ジョイン型ルールセットとして、複数の入力テーブルがあり、各テーブルはジョイン操作に対する1つの入力データセットを示す。

30

2．全ての被包含ルールセットに対する入力変数および定数のテーブル

被包含ルールセットを用いる場合、各被包含ルールセットは、それ自体の入力変数および定数のテーブルを有する。変換式が生成されたとき、被包含ルールセットによって用いられる入力変数は、包含を行うルールセットに即して実際の入力箇所に振り分けられる。そのため、このリストは、包含ルールセットへ引き継がれる。複数の被包含ルールセットが包含される場合、各入力変数のテーブルが引き継がれる（被包含ルールセット自体がルールセットを包含する場合、二次的変数は引き継がれない）。被包含ルールセットから引き継がれる入力変数および定数は、包含ルールセットでの使用には利用できない。このテーブルが含まれることにより、被包含ルールセットへの入力と包含ルールセットへの入力との間のマッピングが確立される。詳細については後述される。

40

3．全ての被包含ルールセットに対する出力変数および定数のテーブル

ルールセットが包含されている場合、被包含ルールセットの出力は包含ルールセットへの入力となる。このテーブルは、それら全ての変数をリストにする。このテーブルは、初期状態では、全ての被包含ルールセット内の出力の変数および定数のテーブルから直接読み込まれる。ただし、名前の衝突を避けるためにビジネス名を変更し得る。このテーブルにおいて、技術名は被包含ルールセット内のビジネス名のままである。

4．出力変数および定数のテーブル

変換型ルールセットでは、このテーブルは、ルールセットによって計算される出力レコードフォーマットのフィールドを含む。計算されない出力変数も包含され得るが、ルールセットにより無視されることになる。（生成される変換式は、入力を出力へコピーするワ

50

イルドカードのルールを有する。さらに、出力は、包含された初期設定値を有し得る。)

出力変数は、中間変数としても用いられ得る。これは、一のルールから生成された出力の値が後のルールにおいて参照され得ること、を意味する。出力がこの用途に用いられるとともに、変換式からの出力レコードに出力が直接含まれない場合もある。

5. パラメータのテーブル

ルールは、パラメータへの参照を含んでもよい。パラメータは、グラフのパラメーターセットに即してランタイム時に用いられる。他の変数と同様、ルールセット内にて、パラメータは、ビジネス名、技術名(例えば、\$ R U N D A T E)およびタイプを有する。

6. ルックアップファイルごとの変数マッピングのテーブル

これらは、入力テーブルと同様であるが、ルックアップファイル用のレコードフォーマットでフィールドへ振り分けられる。

【0063】

非共有ルールセット(変換式を生成するよう設計されている。)は、通常、入力データセットおよび出力データセットの両方に結びつけられている。入力データセットは入力変数のソースである。出力データセットは出力変数のソースである。ルールセットは、複数の入力データセットおよび/または複数の出力データセットを有する場合もある。その場合、各々の入力データセットおよび出力データセットが、変換式の入力または出力となり得る。入力変数のセットは1つのみ(ジョイン操作は除く。)であっても、異なるデータセットに対してはビジネス名と技術名との間のマッピングが異なってもよい。いくつかの例において、入力変数はルールセットによって使用されるとともに、入力変数が第1の入力データセット内には存在する一方で第2の入力データセット内には存在しなくてもよい。その場合、第2の入力データセットにおいて、関数が不足している変数の技術名として特定される。ルールセットが入力変数を使用しない場合、全ての入力データセットに対して技術名を提供する必要はない。

【0064】

被包含ルールセット取扱いはいくらか異なる。被包含ルールセットは、関係する入力および出力のデータセットを有さなくてもよい。それらに代えて、被包含ルールセットは、入力変数および出力変数を有する。被包含ルールセットを包含するルールセットの役割は、入力と出力をマッピングすることである。

【0065】

< 変数 >

変数は以下のプロパティを有し、テーブル形式にてユーザに提供される：

1. ビジネス名(論理名)

ビジネス名は、ルールにて使用される名前である。いくつかの例において、2つの入力変数が同一の名前を有さず、2つの出力変数が同一の名前を有さず、被包含ルールセットからの2つの出力が同一の名前を有さず、さらに、同一ルックアップファイル内の2つのルックアップ変数が同一の名前を有さない、ような制限が設けられる。入力変数は、出力変数と同一の名前を有し得る。その場合、ユーザインターフェースは、コンテキストに基づき、または、出力変数の名前の前に「out」のような接頭語を用いることにより、入力と出力とを区別させてもよい。異なるルックアップファイル内のルックアップ変数は、同一の名前を有し得る。その場合、ルックアップファイルの名前自体のような接頭語を用いることにより、区別され得る。

2. 単純な型

いくつかの例において、4つの基本型(文字列、数字、日付および日時)がサポートされ得る。これらは、文字列型(int)、10進数型(20)、日付型(「YYYY-MM-DD」)および日時型(「YYYY-MM-DD HH24:MI:SS.nnnnnn」)に対応する。基本型と実際型との間の変換は、ビジネスルールの編集とは別に(例えば、生成される変換式成分によって)扱われる。

3. 初期設定値

初期設定値は、出力変数に対してのみ必要である。この値は、(1)その出力に対する

10

20

30

40

50

ルールの出力列に空のセルがある場合、または(2)いずれのルールもその出力の値を計算するためにトリガされない場合、に用いられる。初期設定値は、出力変数がNULL容認な限り、NULLと(および空のセルをNULLと解釈するように)し得る。

初期設定値は、ルール数式テーブルの出力列で使用される式と全く同じ式である。これは、初期設定値が入力変数または出力定数を参照でき、または数式を含み得ることを意味する。初期設定値は、循環参照とならない限り、他の出力を参照し得る。

4. 技術名(物理名)または数式

これは変数を特定する数式である。入力および被包含変数のフィールド名の代わりに数式を使用できる(いくつかの例において、出力変数に数式を使用することは許容されない。)。ベクトルの場合、数式は完全修飾されるべきである。

入力を促されている変数および被包含ルールセットからの入力変数および出力変数を処理する場合、変数に関連付けられる技術名は、共有ルールセット内で使用されるビジネス名そのものである。内部でのみ使用される出力変数を処理する場合(一のルールにおいて計算されると共にそれに続くルールにおいて使用される中間変数)、技術名をブランクとし得る。

5. オプションとしての説明およびコメント

【0066】

< 定数 >

変数の種々のテーブルは、変数だけでなく定数のマッピングも含む。定数はC++のenumに対応する。システムは、有効値および無効値に由来する定数値、ならびに有効範囲および無効範囲に由来する定数範囲をサポートし得る。さらに、異なる値および/または範囲のセットを表す定数を生成することも可能である。

【0067】

定数は変数と関係付けられる。これは、定数のビジネス名はルールセット全体を通じて固有名である必要がないことを意味する。エディタは、通常、その定数がルール内のどの列に現れるかに基づいて任意の定数に対するコンテキストを知る。しかし、ユーザは、数式中の異なる変数に属する定数を選択することも可能である。その場合、定数は、変数名(例えば、「飛行クラス・ビジネス」によって修飾される。

【0068】

出力変数を計算する場合、単一の値の定数値のみが使用される(出力フィールドに範囲を割り当てるのは無意味である。))。

【0069】

定数は、以下のプロパティを有し、テーブル形式でユーザに提供される(変数および定数は、テーブルを別のテーブルに組み込む場合と同様に、混ざり合ってもよい)。

1. 変数名

定数は1つの変数に当てはまる。変数名は、実際には関連付けられる変数自体の一部である。

2. ビジネス名

ビジネス名はルールで使用される名前である。名前は、値を特定するものでなくてもよい(具体的には、内部スペースおよび句読点が許容される。)。定数に対するビジネス名は、適用される変数内にて固有のものである場合がある。

3. 定数型

値、範囲またはセットのうちの1つ。上記したように、範囲およびセットは、比較(入力)にて用いられる場合には適正であるが、割り当て(出力)にて用いられる場合は適正ではない。

4. 値に関して: 実際値

本実施例において、文字列は引用符で囲まれ、数字は囲まれな。日付および日時は、初期設定形式にて引用符で囲まれる(例えば、「YYYY-MM-DD」)。変数型へ自動変換される単純な型を数式が返す限り、数式を用いてもよい。

定数が、被包含ルールセットの入力または出力のテーブルの一部である場合、値は存在

10

20

30

40

50

しない。その代わり、値は、対応する入力変数または出力変数に対して関連付けられた定数のビジネス名である。

5．範囲に関して：最小値および最大値

上述した実際値と同様、両者は定数または数式である。範囲は、ルール内比較の簡易表現として使用される。等式による比較だけが範囲に許され、本システムは、範囲を「変数 $> =$ 最小値および変数 $< =$ 最大値」に翻訳する。最小値が特定されない場合、その部分の比較はスキップされる。最大値についても同様である。範囲は、コンマで最小値と最大値を区切る形式にて格納される。

6．セットに関して：コンマで区切られた値のリスト

リストの各要素は、上述した実際値と同様、定数または数式である。等式による比較だけがセットに許され、本システムは、セットを「変数の集合 [値のベクトルリスト]」の形式にて数式に翻訳する。

7．オプションとしての説明およびコメント

【 0 0 7 0 】

共有ルールセットから引き継がれた変数进行处理する場合、定数も引き継がれる。共有ルールセットに対する入力変数および出力変数を示すテーブルにおいて、これら変数に関連付けられる定数も示される。これら定数に対する初期設定のマッピングは、引き継がれる情報の一部であるが、ユーザは定数値を書き換える得る。

【 0 0 7 1 】

本システムは、変数の使用時に定数の不一致に起因するコンフリクトの可能性がある場合を検出する。具体的には、(1) ある変数の値が別の変数へコピーされる場合、(2) 両方の変数が定義された定数を有す場合、および(3) 定数のセットが名前および値の両方で同一でない場合、ユーザがーの変数の値を他の変数の値に翻訳する必要があるエラーが発生する。ソース変数の例として、入力変数、ルックアップ変数、被包含ルールセットからの出力、および、入力として使用される出力変数、が挙げられる。ターゲット変数の例として、出力変数、および、被包含ルールセットへの入力、が挙げられる。割り当ては、ルール数式または変数テーブルにおいて行われ得る。

【 0 0 7 2 】

< 変数の順序付け >

循環ロジックを防ぐために、本システムは、変数およびルールを厳格に順序付けする。全般的な順序付けの例は、以下の通りである。

入力変数およびパラメータ。

第 1 の被包含ルールセットの入力マッピング。

第 1 の被包含ルールセットの出力値。

...

N 番目の被包含ルールセットの入力マッピング。

N 番目の被包含ルールセットの出力値。

第 1 のルックアップファイルの初期設定キー値。

第 1 のルックアップファイルの出力フィールド。

...

N 番目のルックアップファイルの初期設定キー値。

N 番目のルックアップファイルの出力フィールド。

全ての出力変数の初期設定値。

【 0 0 7 3 】

各項目の計算では前回のステップで計算された値が用いられる。これは、例えば、第 1 の被包含ルールがそのマッピングテーブルの入力変数およびパラメータを参照し得ることを意味する。しかし、第 2 の被包含ルールは、その入力を第 1 の被包含ルールから計算された出力へ振り分け得る。同様に、各出力変数に対する初期設定値はいずれのルールよりも前に計算されるので、それら初期設定値は、入力変数、パラメータ、ルックアップファイル、または、いずれかの被包含ルールからの出力の値、に基づく。ルールの出力を実際

10

20

30

40

50

に計算する際、後のルールが先のルールで計算される値を用いることができるように、ルールは順に評価される。

【 0 0 7 4 】

＜変数へのデータセットのリンク＞

いくつかの例において、入力変数のテーブルは入力データセットレコードフォーマットから直接得られ、ビジネス名は入力データセットのメタデータから得られる。しかし、いくつかの例において、ルールセット内にこのマッピングのコピーを有することには利点がある。第 1 に、ルールセット内に変数マッピングテーブルのコピーを有することにより、生成環境の状況から外れたルールセットの編集が可能となる。ルールセットおよび関連付けられるルールは、サンドボックス内でシリアル化されるとともに、サンドボックスプロジェクトの一部として編集され得る。第 2 に、入力変数マッピングテーブルのコピーを有することにより、ユーザによるコンフリクトの解決または既存のメタデータの書き換えが可能となる。例えば、入力データセット内の 2 つのフィールドが同じビジネス名に振り分けられた場合、これらビジネス名の 1 つは入力変数テーブルの中で変更され得る。

【 0 0 7 5 】

ルールセットが始めて生成されたとき、入力変数テーブルは空である。ユーザが入力データセットを特定すると直ぐに、入力変数テーブルが入力データセットのメタデータから自動的に記入される（同じロジックが出力変数および出力データセットにも適用されるが、これ以降の説明は簡便のために入力データセットだけに絞られる。）。

【 0 0 7 6 】

この説明において、簡便のために「入力データセット」との用語が用いられる。入力変数にリンクし得るゼロ以上の入力データセットと、出力データセットにリンクできる別のゼロ以上の入力データセットと、がある。具体的に述べると、入力変数テーブルは、ビジネス名に対して 1 つの列、タイプなどに対して 1 つの列、および技術名に対して多数の列（入力データセットあたり 1 つ）を有する。一の入力データセットが指定されると、同様な技法を用いて 2 番目が追加され得る。ただし、2 番目または後続のデータセットの場合には、技術名とビジネス名との間のマッピングが不完全なことがある。これは、特に 2 番目および後続の各フィールドがいずれの変数に振り分けられるかを、システムが算定し得ない場合があるからである。そのような実施例では、ユーザは、いずれの不足情報をも手動で訂正し得る。

【 0 0 7 7 】

最初に入力データセットから入力テーブルを生成するとき、入力データセット内の各々フィールド毎に 1 つの入力変数が生成される。入力変数の技術名はフィールド名とする。タイプはフィールドのタイプに基づいて割り当てる。ボイドは文字列のように処理され、実数は数字のように処理される。サブレコード内のフィールドは対応する入力変数を有するが、サブレコードは対応する入力変数を有さない。ユニオンはユニオンの各ブランチに対する入力変数をもたらず。要素がベクトルの場合、対応する入力変数の技術名は、ベクトルの第 1 の要素（`in . vect [0]`）を利用する。ユーザはこれを上書きし得る。例えば、複数出力変換式では、ユーザは技術名を `in . vect [index]` とするように変更し得る。または、ベクトルが固定長の場合、ユーザは、ベクトルの他の要素に対応する追加の入力変数を生成してもよい。ユニオンおよびベクトルは、出力データセットではサポートされなくてもよい（それらのために出力変数が生成されなくてもよい）。いくつかの例において、複数出力成分のバリエーションは複数の出力レコードの代わりに出力ベクトルを出力してもよい。

【 0 0 7 8 】

いくつかの実施例では、ビジネス名はメタデータから計算される。フィールドのビジネス名を決定するためのロジックの例は、以下の通りである。

フィールド（物理要素）が表示名を有する場合、フィールドの表示名がビジネス名として用いられる。

そうでない場合、フィールドが論理要素を有するとともに論理要素が表示名を有すれば

10

20

30

40

50

、論理要素の表示名がビジネス名として用いられる。

そうでない場合、フィールドが論理要素を有すれば、論理要素名がビジネス名として用いられる。

そうでない場合、ビジネス名は技術名から算出される。

【 0 0 7 9 】

コンフリクト（二重の名前）が生じる場合、一のビジネス名のみが割り当てられる。他のフィールドにはいずれのビジネス名も割り当てられない。

【 0 0 8 0 】

いくつかの例において、ルールセットとデータセットのメタデータとの間の動的リンクはない。ユーザがメタデータのデータを変更する場合（例えば、論理要素の改名）、その変更は本システムによって自動的にピックアップされない。いくつかの例において、データ間の双方向関係性を用いてその変更を検出できるようにし得る。

【 0 0 8 1 】

ユーザが第2のデータセットをルールセットへ追加する場合、本システムは、上記にてリストしたものと同様の物理・論理マッピングルールを用いて、各々のビジネス名に対するフィールドを満たそうとする。変数がマッピングされない場合、その変数に対する技術語は追加データセットのために空のままとされ、ユーザはフィールド名または数式を手動にて記入しなければならない。利用可能なフィールドは、ユーザインターフェースにおいてプルダウンリスト形式にて表示される。

【 0 0 8 2 】

入力変数テーブルがデータセットのメタデータから生成されるとき、定数もデータセットのメタデータから入力変数テーブルに追加される。本システムは、論理または物理要素のそれぞれに関係する有効性確認スベックに関連する全ての有効値および無効値ならびに全ての有効範囲および無効範囲に対して定数を生成する。

【 0 0 8 3 】

定数に対するビジネス名を決定するための論理の例は、以下の通りである。

有効値（有効範囲等）が表示名を有する場合、表示名をビジネス名として用いられる。

そうでない場合、有効値（有効範囲等）が記述を有すれば、その記述がビジネス名として用いられる。

そうでない場合、定数は、ビジネス名なしで変数テーブルに含まれる。

【 0 0 8 4 】

データセットで始まる変数を生成することは必ずしも必要ではない。入力変数のリストを生成する第2の方法は、基礎システム内の論理入力进行特定することである。論理入力を選択すると、本システムは、論理入力内の論理要素毎に1つの変数を有する変数テーブルを生成する。変数のビジネス名は論理要素の表示名とされる。論理要素が有効性確認スベックを有する場合、以前の文書ルールを用いて定数も生成される。

【 0 0 8 5 】

最後に、入力変数および出力変数は、変数テーブルにそれらを順次追加するか、またはルールを編集しながらそれらを生成するかのいずれかにより、手動にて追加され得る。例えば、ユーザが列をルールに追加するとき、ユーザはどの入力変数をその列に使用すべきかを選択する。ただし、ユーザは「new...」を選択するとともにオンザフライで入力変数を生成することもできる。本システムは、ユーザにデータ型およびオプションのコメントを入力するように促す。技術名は後で入力されてもよい。

【 0 0 8 6 】

本システムには、ルールの編集を可能にするための変数リストが必要である。しかし、ビジネス名と技術名との間のマッピングは、後で完成されてもよい。マッピングは、ユーザが外部テストファイルに対して全ルールセットをテストするか、または、ルールセットから変換式を実際に生成するか、の準備ができている場合のみ必要とされてもよい。

【 0 0 8 7 】

< 被包含ルールセット >

10

20

30

40

50

いくつかの例において、ルールセットを共有できる。具体的に述べると、被包含ルールセットは、別のルールセット内部に包含されるように設計されるので、その論理は包含ルールセットの生成変換式の一部となる。

【 0 0 8 8 】

被包含ルールセットは、専ら共有されるように設計されるのが通常であるが、被包含ルールセットは変換式を生成するために独立して使用され得る。例えば、ユーザは、フィルタ型変換式のブーリアン出力を計算するルールセットを生成できる。しかし、同時に、そのルールセットを別の変換式内に包含させて、ブーリアン出力（包含ルールセットで利用可能な、共有ルールセットの出力変数）を用いてもっと複雑な出力を計算し得る。

【 0 0 8 9 】

被包含ルールセットは、他の形式のルールセットと同様である。被包含ルールセットは入力変数および出力変数を有する。被包含ルールセットは、それ自体、他の被包含ルールセットを包含できる。しかし、被包含ルールセットの入力および出力の変数の取扱いは、変換型ルールセットを有するものと異なる。変換型ルールセットでは、入力および出力の変数は技術名に割り当てられるので、変換式が生成され得る。しかし、被包含ルールセットでは、入力および出力の変数を技術名にマッピングする必要がない。（あるルールセットが変換式を生成するために共有され且つ使用される場合、変換式を生成するデプロイメントのために入力および出力変数は技術名にマッピングされる。）

【 0 0 9 0 】

ユーザが被包含ルールセットを別のルールセットに包含させる場合、包含ルールセットは、被包含ルールセットの入力および出力をマッピングするための変数マッピングテーブルを有する必要がある。包含ルールセットに関連して、共有されるルールセットの入力変数および出力変数だけが視認される。共有ルールセットに包含されるいずれのルールセットのいずれの変数も、包含ルールセットには現れない。

【 0 0 9 1 】

包含ルールセットに関連して、共有ルールセットの入力変数は、包含ルールセットの変数、または、これらの変数を用いる数式に割り当てられる必要がある。共有ルールセットのビジネス名は変数マッピングテーブルにリストされるが、それら名前は包含ルールセット内のルールで使用するには利用されない。それに代えて、包含ルールセットは、共有ルールセットの各入力変数（ビジネス名による）を包含ルールセット内の数式に一致させることだけを必要としてもよい。

【 0 0 9 2 】

被包含ルールセットの評価は、変数、パラメータおよびルックアップを入力する前に行われるように考えられているので、被包含ルールセットの出力はルックアップのキーとして使用され得る。いくつかの例において、評価の順序はよりフレキシブルであり、ルックアップの順序付けと被包含ルールセットの評価とは、依存関係に基づいて自動的に決定される。いずれかの出力変数が計算されてから被包含ルールセットが評価されるので、包含ルールセット内の出力変数は被包含ルールセット内の入力にマッピングされない。被包含ルールセット入力へのマッピングを単純な入力変数に行うことができない場合、代わりに数式が用いられ得る。

【 0 0 9 3 】

被包含ルールセット入力変数へのマッピングは、被包含ルールセット内の入力変数がNULL容認な限り、NULLとし得る。マッピングは空のままにし得る。マッピングが空のままである場合、その入力変数が包含ルールセットの出力の計算で必要とされる場合であり且つその場合に限り、変換式生成時にエラーが報告される。いくつかの例において、全てがNULL容認とされることにより、ユーザインターフェースが簡略化される。

【 0 0 9 4 】

包含ルールセットに関連して、共有ルールセットの出力変数も、包含ルールセット内のビジネス名にマッピングされる必要がある。このマッピングテーブルは上記のマッピングテーブルとは逆である。共有ルールセットの入力変数をマッピングするとき、そのテーブ

10

20

30

40

50

ルは、共有ルールセットの入力変数のビジネス名を包含ルールセット内の既存の変数にマッピングする。しかし、共有ルールセットの出力変数をマッピングするとき、包含ルールセットは、共有ルールセット出力のビジネス名を規定する表（包含ルールセット内の名前を共有ルールセット内の対応する名前にマッピングする表）を有する。

【 0 0 9 5 】

出力変数マッピングは、潜在的な名前コンフリクトを解決するために必要である。初期設定のマッピングは、包含ルールセットおよび共有ルールセットの双方にて同一のビジネス名を単純に用いている。しかし、共有ルールセット内の出力変数の名前は包含ルールセットで既に定義されている変数のビジネス名とコンフリクトする可能性があるため、包含ルールセット内のマッピングされた名前は変更され得る。

10

【 0 0 9 6 】

共有ルールセットからの出力の全てがマッピングされる必要はない。ある出力がマッピングされないままの場合、その出力を包含ルールセットにて用いることはできず、共有ルールセットからの対応するロジックは無視される。他方、共有ルールセットからの入力全てはマッピングされてもよい。ただし、ルールセット設計者が必要ないと確信している場合、それらは関心のない変数へマッピングされ得る。いくつかの例において、本システム自体は、どの入力を実際にマッピングされる必要があるかを決定してもよい。

【 0 0 9 7 】

いくつかの例において、マッピングテーブルは、参照によって行われずビジネス名で行われる。共有ルールセットが別のルールセットに包含される場合、包含ルールセットは、共有ルールセットから入力および出力のコピーを得る。これらの名前は、マッピング情報とともに包含ルールセットに格納される。共有ルールセットの変更は可能なため、いくつかの入力または出力を追加、削除、または改名させることができる。

20

【 0 0 9 8 】

包含ルールセットと被包含ルールセットの間の参照整合性問題は、ルールセットがシステムからロードされるときに包含ルールセットによって取り扱われ得る。共有ルールセットから消える入力変数は、包含ルールセットから削除される。共有ルールセットに追加される入力変数は、包含ルールセット内のマッピングテーブルに追加されるが、マッピングされないままとされる。同様に、共有ルールセットに追加される出力変数は、包含ルールセット内のマッピングテーブルに追加されるが、マッピングされないままとされる。出力変数が共有ルールセットから削除されて包含ルールセット内で使用されない場合にはマッピングテーブルから削除されるだけであるが、出力変数が包含ルールセット内で使用される場合には変数がもはや利用できない旨のエラーをユーザは通知される。

30

【 0 0 9 9 】

包含ルールセットは、共有ルールセットからの冗長な情報を実際に維持する。具体的に述べると、入力および出力の変数マッピングテーブルにおいて、包含ルールセットは、共有ルールセット内のビジネス名のリストを包含ルールセット内の対応する名前と併せて維持することだけが必要とされもよい。効率向上のため、包含ルールセットは、共有ルールセットからコピーされたタイプ、初期設定値、説明およびコメントも維持する。これら値は、包含ルールセットを編集するときに読み取られるに過ぎないが、報告の生成および他の分析の効率向上のために包含される。

40

【 0 1 0 0 】

共有ルールセットマッピングテーブルは、包含ルールセット内に1つの追加項目（追加コメント）も有する。これにより、ユーザは、マッピングされた値に別のコメントを追加することが許される。

【 0 1 0 1 】

< ルックアップファイル >

ルールセットは、オプションとして1つ以上のルックアップファイルを有し得る。ルールセット内の各ルックアップファイルは以下の情報を含む。

1 . ルックアップファイルのビジネス名

50

2. オプションの記述およびコメント

3. キーを作成するフィールドのビジネス名のリスト

これらの名前は、ルックアップファイルが数式に追加される場合に用いられ、ユーザは、`lookup(MyLookupFile,`
`<顧客名キー>`、`<アカウントタイプキー>`)のような表示を見る。

4. 各々のキーの初期設定数式のリスト

5. ルックアップファイルの技術名

いくつかの例において、これはデプロイメントにおいて上書きされ得る。

6. 1つ以上のルックアップデータセット

各ルックアップファイルは、ルールセットが入力データセットと関連付けられているように、システム内にてデータセットと緩く関連付けされている。初期設定により、ルールセット内の各ルックアップファイルと関連付けられる1つのルックアップデータセットがあるが、他のデプロイメントで使用するためのより多くのルックアップデータセットがあり得る。

7. 入力変数および定数のテーブル

これは、ルックアップファイル毎に1つのテーブルがあることを除き、ルールセットの入力変数および定数のテーブルと同様である。入力変数と同様、ルックアップファイルの入力変数および定数のテーブルは、関連付けられるルックアップデータセットの各々に対応する複数の技術名を有し得る。

【0102】

1よりも多いルックアップファイルが存在してもよいことを除き、ルックアップファイルは入力変数と同様に取り扱われる。各ルックアップファイルは、1つのページ上で編集され、ビジネス名と技術名との間のマッピングテーブルを有し、複数のデータセットと関連付けられ得る。それらは、各フィールドと関係付けられる定数も有する。入力変数のメタデータが入力データセットからロードされるように、ルックアップファイルのマッピングは、ルックアップデータセットのメタデータを読み込むことによって初期化され得る。

【0103】

ユーザがルックアップフィールド変数を用い且つそのキーがルックアップ内に見つからない場合、フィールドの値はNULとされる。フィールドがNULの場合にルールケースが特にトリガしない限り、ルールケースは偽と評価されてスキップされる。そのような場合、エラーは生じない。ユーザがルックアップファイル変数（フィールドではなく、ルックアップファイル自体）を用いる場合、関数ルックアップの一致が推測され、ルックアップファイル変数が真または偽であるか評価される。双方の場合が入力または出力列に対するルール数式へ適用される。ユーザが出力変数の初期設定としてルックアップフィールド変数を用いる場合、ルックアップの発見の失敗はNULに読み替えられる。

【0104】

ルックアップ処理（参照ファイルから一または複数のデータレコードを検索するためにキーが用いられる。）は、通常の処理に比べて処理が遅い。BREには、ルックアップ結果をキャッシュすることによって高負荷のルックアップ処理の数を制限するように設計されたコードが、各レコードに対して含まれている。ルールがルックアップ変数（ルックアップ処理から返される値のうちの一つ）を参照するときにはいつでも、変換式生成プロセスにおいてルックアップ処理はサブルーチン呼び出しへ変換される。サブルーチンは、サブルーチンが現在のレコードに対してすでに呼び出されているか否かを示すグローバルブーリアン（各レコードが開始される際に偽に初期設定される。）を含む。ルックアップサブルーチンが最初に呼び出されるとき、ブーリアンは偽である。この場合、ブーリアンは真に設定される。そして、実際のルックアップ処理が実行され、ルックアップコールによって返されるレコードは変数内にキャッシュされる。最後に、テストが可能となったとき、ルックアップ処理の結果がイベントログに追加される。

【0105】

同じレコードの処理における後続するルックアップ処理は、同じサブルーチンを呼び出

10

20

30

40

50

す。しかし、後続のサブルーチンが呼び出されるとき、ブーリアンは真である。これにより、冗長なルックアップ処理に代えて（および追加のログイベントを生成することを避けるために）すでに読み込まれてキャッシュされたデータを返し得るように、サブルーチンの処理が変更される。

【 0 1 0 6 】

簡便のため、キャッシュは1つのキー値に対してのみ行われる。異なるキー値を用いて同じルックアップファイルへの複数参照が（同じレコードにおいて）行われる場合、それらルックアップ結果のうちの1つのみがキャッシュされる。他のルックアップサブルーチン呼び出しの全ては、実際のルックアップ処理に変換される。これにより、熟練した技術者は、単なる変数に代えてキャッシュ結果に対するハッシュテーブルを用いることにより、異なるキーによる複数ルックアップをサポートするための拡張を如何に行うべきかを把握し得る。

10

【 0 1 0 7 】

< パラメータ >

ルールセットはパラメータを参照してもよい。いくつかの例において、各ルールセットは、オプションのパラメータテーブル（変数テーブルと同様、パラメータのビジネス名を技術名にマッピングする。）を有する。パラメータテーブルの各項目は、以下の属性を有する。

1 . ビジネス名

これは、ルール本体に現れるときのパラメータの名前である。一般に、パラメータは入力変数が使用される場所ではどこでも使用し得る。

20

2 . 技術名

これは、開発環境におけるパラメータの名前である。

3 . パラメータの型（文字列、10進数、日付または日時）

生成された変換式において、パラメータは必要に応じて他の型に変換され得る。

4 . オプションとしての説明およびコメント。

【 0 1 0 8 】

パラメータは、入力ファイル全体にわたってパラメータ値が定数であることを除いて変数と同様であり、その値は処理が開始されると外部から指定される。

【 0 1 0 9 】

30

< ルールのテストおよび記録 >

変換式の生成の一部には、変換式が対応することになるルールをテストすることが含まれる。ルールは検証もされる（すなわち、ルールのシンタックスおよびセマンティックの一貫性がチェックされる。）。検証とは逆に、ルールを実行して正当性をユーザが確認すること（例えば、期待される出力がもたらされるか、または、出力を期待値と手動で比較すること）によってもテストは行われる。

【 0 1 1 0 】

システムは、2つのレベルのテストをサポートする。上述したように、各ルールは、関連付けられたテストデータセット（値および期待される結果を組み込んだテーブル形式）を有してもよい。これはユニットテストとも称呼される。ルールを編集する場合、テストデータのライン毎にルールの出力を再評価することができる。実際の結果と期待される結果との間の不一致または有効な結果の生成の失敗は、解決すべくハイライトされる。

40

【 0 1 1 1 】

いくつかの例において、外部入力テストファイルは、標準メカニズムを用いてサーバープロセスにアクセス可能である。外部ファイルを用いるテストはファイルテストとも称呼される。テストファイルは、入力データセットをルールセットにマッチさせるレコードフォーマットを有する。いくつかの例において、代替レコードフォーマットが提供されてもよい。オプションとして、ユーザは、期待される結果を含むデータセットを特定し得る。システムは、テストデータセットに対してルールセットを実行し、どのような出力がなぜ生成されたかを表示する。期待された結果が含まれていた場合、システムは実際の結果と

50

期待された結果とを比較し、相違するあらゆるレコードをリストにする。いくつかの例において、ユーザが個々の値をインクリメントに再試行できるように、インターフェースが拡張され得る。

【 0 1 1 2 】

ユニットテストとファイルテストとの違いの例として、以下が挙げられる。

1. ルックアップファイルに関して：

ユニットテストモードにおいて、各テストケースについて、各ルックアップ変数の値がテストの一部として定義される。キーは規定されない；テストが実行されるとき、各テストケースについて、各ルックアップ変数に対して同一の値が採用される。テストデータセットは複数のテストケースを含み、各テストケースは各ルックアップ変数に対して異なる値を規定できる。ファイルテストモードでは、実際のルックアップファイルが用いられる。これは、異なるキーが異なる値を返すことを意味するとともに、特定のキーについて与えられるルックアップ変数に用いられる値をテスト中に変更することができないことも意味する。

10

2. 被包含ルールセットに関して：

ユニットテストモードにおいて、被包含ルールセットは、実行されず、完全である必要さえもない。それに代えて、各被包含ルールセットからの各出力に対し、テストデータセット内にて値が指定される。ファイルテストモードでは、被包含ルールセットが、稼働中に実行されるであろう方法にて実行される。これは、被包含ルールセットに必要とされるいずれのルックアップファイルまたはパラメータも、テストタイムに指定されなければならないということを暗示する。

20

3. パラメータに関して：

ユニットテストモードにおいて、各テストケースの各パラメータに対して異なる値が設定され得る。ファイルテストモードにおいて、各パラメータの値はテスト全体について定数である。

4. 現在の日付に関して：

テストにおいて、ルールが現在の日付または時間を参照する場合、ユーザは現在の日時として用いられるべき値を指定する。ユニットテストモードにおいて、日付および時間は各テストケースについて相違し得る。ファイルテストモードにおいて、テスト全体に対して単一の日付および時間の値が設定される（この値は、テスト実行時のマシンの日付および時間と異なってもよい。）。

30

5. レコードフォーマットおよびマッピングに関して：

ユニットテストにおいて、マッピングは指定される必要はない。テストは全体的に変数のビジネス名に基づいて実施される。ファイルテストについて、全ての変数は技術名にマッピングされ、入力、出力およびルックアップのレコードフォーマットが指定される。

【 0 1 1 3 】

図3におけるフローチャート300に示すように、ルールセットは、特別な特徴を有する記録を行いつつ、テストされて正当性を確認される。302において、一または複数のルールセットおよび対応するテストデータが入力として受け取られる。304において、生成器は、ルールセットに基づいて変換式を生成するとともに、テストデータ内の全てのレコードに対する出力値を算出するためにそれを用いる。オプションとして、ユーザは、ログ（生成された変換式を含むグラフベース計算のトレース実行308において生成される。）の特徴を設定することができる。グラフベース計算において、関連する成分の「ログ」ポートは出力をテストするために用いられる。「ログ」ポートは、成分のための帯域外通信チャンネルである。それは、実際の出力データのレコードフォーマットの変更を求めない変換式からの追加出力メタデータを取得する方法の1つである。310において、ストレージシステム内のログファイルへ集められたログメッセージに実行中においてアクセス可能であるとき、ログ情報はログポートから出力される。オプションとして、ログメッセージがフィルタされるとともに、それらのサブセットがログファイルに格納されてもよい。例えば、出力レコードが変更されない場合、生成された変換式を含む成分は、大部

40

50

分の入力レコードをパスしてもよい（例えば、与えられた入力レコードに対していずれのルールケースもトリガされない場合）。トリガされたルールケース（同ルールケースは入力レコードのフィールドにおける一または複数の値を変更する。）に対応するログメッセージのみを格納することが望ましい。グラフにおける一または複数の成分は、ログポートからのレコードを、それらログメッセージのみがログファイルに記載されるようにフィルタし得る。

【 0 1 1 4 】

テスト用に生成された変換式は、通常の実行用に生成される変換式とわずかに異なってもよい。例えば、通常の実行において、その変換式を用いる成分は、所定の出力ポートについて所定のタイプの出力レコードを生成してもよい。テストタイムにおいて、テスト関連出力は、出力レコードが不変であってもログポートに送られる。

10

【 0 1 1 5 】

第1のテストケースを始めとして、テストケースの入力は変換式に入力され、出力はどのルールがその出力を生成したかについての標識とともに出力配列に記載される。このプロセスは、最後の行が評価されるまで各行に対して繰り返される。出力配列は、上述したように、結果テーブルを生成するために用いられ得る。出力配列は、ルールセットが有効であるか否かを判定するために評価されてもよい。出力値は、生成された出力値についてのテストデータに含まれてもよく、一のテストにて生成された値がその前のテストにて生成された値と比較され得る。出力配列の第1の行を始めとして、生成された出力は、テストデータまたは事前のテスト結果から期待される出力と比較される。いずれかの出力が一致しない場合、不一致である旨が記録される。これが各行について繰り返される。いくつかの例において、評価ステップは出力生成ステップに統合される。上述したように、一のルールセットの出力は他のルールセットへの入力であってもよく、そのような場合には被包含ルールセットは包含ルールセットの一部として評価される。

20

【 0 1 1 6 】

ユーザは、出力フィールドまたは入力フィールドにて用いられる数式によってテストされるルールを制限し得る。いくつかの例において、ユーザは、テスト中にルールを実行不可とし得る。いくつかの例において、ユーザはテストファイル全体が処理されることを待つ必要はなく、テスト結果は始めのいくつかのレコードが実行されれば入手可能となる。

【 0 1 1 7 】

30

テストデータ自体に加え、ファイルテストおよび記録について以下の情報が308においてトレースされ得る。

1．入力データセットの物理位置

これは、各入力データセットに対する入力変数テーブルにおけるルールセットに格納される。ジョイン型データセットについて、全ての物理位置が必要とされる。物理位置が必要とされるときはいつでも、データベースのテーブル名が用いられ得る。

2．入力データセットのレコードフォーマット

初期設定として、これは、入力データセットに対するデータセット定義から取得される。サンドボックス内にチェックアウトされる異なるレコードフォーマットによってこれを上書きし得る領域が、入力変数内にある。ジョイン型変換式について、全てのレコードフォーマットが必要とされる。

40

3．どのデプロイメントが用いられるか。

4．全てのルックアップファイルについての物理位置

これはルールセットファイルテーブルに格納される。

5．各ルックアップファイルのレコードフォーマット

各ルックアップファイルに関連付けられるデータセット定義または上書きされたレコードフォーマットファイルから取得される。

6．各パラメータについての値

これはテストパラメータダイアログにて設定される。

7．出力ファイルについての物理位置

50

これは回帰（比較）テストが行われる場合にのみ必要とされる。これは出力変数テーブルに格納される。

8．出力ファイルについてのレコードフォーマット

上記同様、回帰テストが行われる場合にのみ必要とされ、出力データセット定義またはオプションとしての上書きレコードフォーマットファイルから取得される。

9．プロジェクトサンドボックスの位置

テストはホスト上のサンドボックスの外にて行われなければならない。サンドボックスは、ルールセットを含むプロジェクトのチェックアウトコピーであるべきである。全てのレコードフォーマットファイルは、サンドボックスから取得され得る。

10．ルールが「now」、「today」または同様の値を参照するときの日付と時間について用いられる値

【0118】

本例において、変換式は初期設定ではセル状態を記録しないが、この機能は図4に示すユーザインターフェイス400において有効化され得る。すなわち、ユーザは、種々の特定の判断基準の記録をオン・オフするように306にて記録の特徴を設定し得る。ユーザインターフェイス400は、いつログメッセージ（「ログレコード」とも称呼される。）が生成されるか（各入力レコードについて、いつエラーが生じるか、または、いつ特定の数式が真であるか、を含む。）を特定するセクション410、各ログメッセージに何が含まれているか（例えば、ルールケース実行、入力値、出力値、ルックアップ値、被包含ルールセットからの出力値、パラメータ値、および/または、各セルの評価の結果）を特定するセクション420、を含む。入力、出力などを記録することによって実行が遅くなるが、小さな程度に過ぎない。セル状態を記録することによって実行が相当程度遅くなり、一桁程度遅くなる可能性もある。テストが実行されると、検索された結果が所定の設定に従って表示される。

【0119】

テストおよび/または記録が行われない場合であっても、生成器は、入力、出力などを記録する変換式を生成してもよく、その変換式を稼働中に用いてもよい。拡張された変換式は、同様の出力を生成するとともに、どのルールが実行されたかを実行後に判定するために分析され得る一群のログメッセージを生成し得る。ユーザがログファイルにて生成されたログメッセージ310を保存した場合、テスト入力を用いられなくても、稼働中におけるルールセットの実行を再現するための事実の後にBREが用いられ得る。この実行はプレイバックと呼ばれ、監査において有用である。ログファイルは、上述した、入力、出力、ルックアップ、トリガされたルールケースなどの詳細を含む。それは、どのバージョン（名前およびバージョン）のルールケースがログファイルを生成した変換式を生成するために用いられたかを厳密に記述するヘッダも有する。稼働中において、顧客はログポートの出力をファイル（圧縮されてもよい。）に保存すべきである。監査に際してセッションをプレイバックするためには、顧客は、BREを起動してプレイバックセッションを開始する。顧客は、セッションログを含むファイルを特定する。BREは、ヘッダを読み取り、記載されているルールセットおよびバージョンを開き、ログファイルの残りをファイルまたはユニットテストモードにて実行されているように実行する。プレイバック中の表示は、以下の例外を除き、ユーザがファイルまたはユニットテストを実行した後に見る表示と同じ表示である。例外として、（1）回帰データとの比較がないこと、（2）セル状態のような情報がログから失われ表示されない可能性があること、および、（3）表示されるルールセットのバージョンが正しいバージョンではない場合があるので、ルールセットはプレイバック中に読み取り専用とされることが挙げられる。

【0120】

記録機能が可能とされているとき、生成された変換式はログステートメントを含むように修正される。ログメッセージは、任意文字列をログポートに出力する機能であるwrite__to__log（）をコールすることによって生成され得る。write__to__log（）が使用されるとき、特定のフォーマットのデータがログポートに書き出される。

例えば、ログポートフォーマットの例は、以下の通りである。

【 0 1 2 1 】

【 数 1 】

```
record
string("|") node;
string("|") timestamp;
string("|") component;
string("|") subcomponent;
string("|") event_type;
string("|\\n") event_text;
end;
```

10

【 0 1 2 2 】

ログ情報は全て `event_text` フィールド (`write__to__log()` のコールにおいて特定される文字列を含む。) に格納される。ログポートレコードの他のフィールドは、成分によって自動的に満たされ、BRE に無視される。

【 0 1 2 3 】

(ログメッセージに特有の) `event_text` のフォーマットの例は、以下の通りである。

【 0 1 2 4 】

【 数 2 】

```
record
decimal("|") count; // イベントの数
record
decimal("|") recNum; // レコード番号, 1 は第1レコード, ゼロは凡例
string(1) opCode; // 下記参照
decimal("|") rule set; // ルールセットを含む, ゼロはメインルールセット
decimal("|") group; // 入力グループ, ルール, その他
decimal("|") index; // 入力インデックス, 出力インデックス, ルールケース, その他
string(1) nullFlag; // 空欄またはNULLに対するアスタリスク ( " - " )
string("\\x01") value; // 入力値, 出力値, その他
end events [count];
end;
```

20

30

【 0 1 2 5 】

`opCodes` の例は、以下の通りである。

【 0 1 2 6 】

【 数 3 】

"I" は入力変数である。

"O" は出力変数である。

"K" はロックアップキーである。

"L" はロックアップ変数である。

40

"R" は、凡例でないレコードについてはトリガされたルールケースであり、凡例レコードについてはルールである。

"C" はルールセル (テストレベル追跡が可能である場合に使用される。常に使用される凡例を除く。) である。

"P" はパラメータである。

"H" は列見出しである。

"N" はルールセット名 (凡例レコードにおいてのみ使用される。) である。

"V" はバージョン番号 (凡例レコードにおいてのみ使用される。) である。

"D" は異なるレコード (テストの際のみに使用される。) である。

【 0 1 2 7 】

50

本例において、入力レコード毎に1つのログメッセージが存在する。ログメッセージは一連のイベントを含む。入力、出力、ルックアップフィールド、実行されるルールケース、などの1つ毎に1つのイベントが存在する。これにより、入力レコード毎に多量の一連のイベントが構成される。

【0128】

本例において、ログイベントにおける値フィールドは二進型ではない。それは、所定文字 `hex 01` または改行を含まない。これらはイベント間またはレコード間の境界を誤ってトリガする場合があるからである。代わりに、全ての値は印刷可能な文字列に変換される。印刷不能な文字は `hex` 文字列に（例えば、改行は「`\x0a`」に）変換される。二進型変数は十進型などに変換される。

10

【0129】

変換の最適化（内部ルックアップファイルを用いるような）は、記録の厳密さを確実にするために必要であれば、実行不可とされ得る。

【0130】

第1のレコードが確認されたとき、対応するルールセットの詳細を含むログメッセージが生成される。これはいわゆる「凡例」ログメッセージであり、与えられたルールセットに対するグラフ実行毎に生成される。凡例ログメッセージの第1の部分は、ルールセットの場所およびバージョン名（プレイバックに必要である。）に加え、ログフォーマットバージョンを含む。各々の入力、出力、ルックアップファイル、ルックアップフィールド、パラメータなどの名前を記述する情報がそれに続く。それら名前はインデックスと関連付けられる（1は第1の入力変数であり、2は第2の入力変数である、など）。これにより、イベントに関連付けられる後続する「トレース」ログメッセージが、（名前ではなく）インデックスを用いて変数およびルールを参照し、ログファイルに場所を保存することができるようになる。

20

【0131】

記録中のグラフによって処理されるデータフローの最初および全ての後続するレコードについて凡例ログメッセージがログファイルに記載された後、以下のログイベント（各々は対応するトレースログメッセージに関連付けられる。）が生じる。

【0132】

（1）入力レコードが記録されている場合、ログメッセージは、各入力レコードについての入力変数の値を記述する入力変数の全てに対して生成される。

30

（2）パラメータが記録されている場合、ログメッセージは、パラメータの値を記述する入力変数の全てについて生成される。

（3）被包含ルールセットが存在する場合、被包含ルールセットにおけるルールが実行される。それらルールはこのロジックに準じて再帰的にログメッセージを生成する。

（4）ケース状態が記録されている場合、全てのルールにおける全てのケースの値が計算されて記録される。

（5）連結された `if - then - else` ロジックまたは `switch` 分または内部ルックアップを用いて、実際のルールロジックは実行される。どのルールケースがトリガされるか確認された時点にて、そのトリガされるルールケースに対してログメッセージが生成される。

40

（6）どのルールケースがトリガされるか確認された時点にて、ルールの出力変数に値が書き込まれるとともに、割り当てられた値を記述するログメッセージ（例えば、変換されたレコードにおける値）が各出力に対して生成される。

（7）ルールが評価されている間に行われるルックアップ参照は、サブルーチンによって処理される。サブルーチンロジックは、ルックアップを行い、ルックアップファイルから読み込まれる全ての変数に対するログメッセージ（キーとして用いられる値および全ての見つけられたルックアップ変数の値を記述する。）を生成する。そして、サブルーチンは、ルックアップ値をメインロジックに返す。ルックアップサブルーチンは、それがすでに呼び出されたか否か、または、同じキーに対して二重のログイベントを生成することを

50

避けなかったか否かを示すブーリアンを維持する。

(8) いずれのルールもトリガされない場合、最後の `else` クローズ (または、 `switch` 文に対する初期設定クローズ) が実行されて初期設定出力値が割り当てられる。同時に、割り当てられた値を記述する各出力に対してログメッセージが生成される。

【 0 1 3 3 】

ログポートから提供されるログ情報は、種々の方法にて用いられ得る。例えば、いくつかの例において、テストデータセットとして用いられるデータセットと関連付けられる入力レコードの数は、グラフにおけるデータ処理成分のルールセットをテストするために用いられるテストデータセットのように、所定のシナリオに対して望ましい数よりも多くてもよい。ログポートからのログ情報は、グラフの初期処理中にグラフの成分によって処理され得る。それら成分は、ルールセットのルールにおける各ルールケースについての少なくとも一つのログメッセージを入力レコードの最小セットが提供するか、を決定するためにログ情報を調べることができる。例えば、成分は、各ルールにおける各ケースがトリガされる第 1 の入力レコードを特定する。そして、それら特定されたレコードは、縮小ルールセット (同ルールセットにおいては、全ての他の入力レコードが取り除かれる。) に関連付けられて格納される。限定テストデータセットは、ルールセットにおける同じルールおよびルールケースをテストすることができるが、テストを行う観点からより小さくより効率的であってもよい。

【 0 1 3 4 】

< オーバーラップ解析 >

いくつかの例において、上述したように、ユーザがルールセットに対してテストデータセットを実行したとき、トリガした全てのルール (すなわち、全ての入力条件が満たされたルールケースであって、優先度がより高いルールケースに関する全ての入力条件が満たされなかったならば出力を生成したルールケース) が調査され得る。テストデータが処理された後、システムはテスト出力データを後処理し、どのテストケースによってもトリガされなかった全てのルールまたはルールケースのリストを生成し得る。どのルールがトリガされたか又はされなかったかをユーザに直ちに示すため、この情報はエディタ内のルールの表示上にオーバーレイされ得る。この情報から、ユーザは、他のルールによって覆い隠されている可能性のあるルール (すなわち、オーバーラップしているルール) を探し得る。各ルールケースについてカウントが示され得る。カウントは、ルールケースがトリガされたか否かを知ることと同様に有用であり得る。特に、出力の望ましい分布を達成するように値を調整する際、および、性能を調整するのに最適なルールセットを特定する際、に有用である。

【 0 1 3 5 】

< 変換式生成 >

ビジネスルールは、各ルールセットを変換式へ変換することにより、アプリケーション (グラフ) 上にて評価される。変換式は、グラフ内の成分へアタッチされる。そのような成分は、変換式を実行するために特定の方法でリンクされる標準成分のセットを含むサブグラフであってもよい。これらサブグラフは、追加分 (例えば、ジョインおよびロールアップのためのキーを使用する。) とともに用いられ得る。

【 0 1 3 6 】

変換コードは、複数の方法にてビジネスルールから生成され得る。例えば、変換式の内部がユーザ編集されるように設計されていない場合、生成プロセスによって理解が困難な変換式が生成され得るが、ルールを 1 つずつ適用するよりもより効率良くルールが実施される。いくつかの例において、生成された変換式の性能を高めるべく、特殊なルックアップファイルまたは他の技術が用いられ得る。どのように変換式が生成されたかについてのいくつかの詳細は、デプロイメント内に格納され得る。複数のグラフで使用されるルールセットは、可能性がある別のユーザのための複数のデプロイメントを有してもよい。ルールセットは、ルールのスーパーセットを含んでもよく、その内のいくつかだけが各デプロイメント (そのデプロイメントは、変換式が生成されるときに使用されるルールを規定す

る。)にて要求される。ルールが `if then else` ロジックに代えて多数の定数値を有する場合(数式(もしあれば)は有さない。)、ルックアップテーブルが用いられ得る。この場合、ルックアップテーブルはルールの一部である(個別に保存されない。)。例として、以下のルールを検討する。

【 0 1 3 7 】

【表 1】

発都市	着都市	サービス クラス	数式	飛行マイル (出力)
BOS	LAX	ファースト		6000
BOS	LAX	ビジネス		3000
BOS	LAX	普通	マイル使用 = yes	0
BOS	LAX	普通	他	3000
BOS	CHI	ファースト		2000
BOS	CHI	普通		1000
BOS	NYC	ファースト		1500
BOS	NYC	ビジネス		1000
BOS	NYC	普通		500

10

【 0 1 3 8 】

このルールは、以下の情報とともにメモリ内ルックアップテーブルを構成することにより取り扱われる：

20

【表 2】

発	着	クラス	数式	マイル
BOS	LAX	1	0	6000
BOS	LAX	2	0	3000
BOS	LAX	3	1	3000
BOS	CHI	1	0	2000
BOS	CHI	2	0	1000
BOS	NYC	1	0	1500
BOS	NYC	2	0	1000
BOS	NYC	3	0	500

30

【 0 1 3 9 】

そして、変換式は以下になる。

【 0 1 4 0 】

【数 4】

```
int expr = lookup(from, dest, class).expr
int miles = lookup(from, dest, class).miles
```

```
if (expr == 1 and is_using_miles) miles = 0
return miles;
```

40

【 0 1 4 1 】

各ルールセットは、デプロイメントテーブル(名前をそのデプロイメントの構成についての詳細にマッピングするように適合されている。)を有する。図4に示すように、特定の実行情報を記録することを望むユーザは、各ルールセットのデプロイメントテーブルにおいて定義される属性に従い、グラフィカルユーザインターフェイス上の各項目を入力し得る。

1. デプロイメント名

ルールセット内部で一意的でなければならない任意の文字列。

2. 入力データセット名

入力変数テーブルに挙げられた複数の入力データセットがある場合、デプロイメントテ

50

ーブルの各項目は、そのデプロイメントに対してどの入力データセットが使用されるかを指定する。

3. 出力データセット名

出力変数テーブルに挙げられた複数の出力データセットがある場合、デプロイメントテーブルの各項目は、そのデプロイメントに対してどの出力データセットが使用されるかを指定する。

4. 各被包含ルールセットに対するデプロイメント名

各被包含ルールセットについて、包含ルールセットの対応するデプロイメントに対してどのデプロイメントが使用されるべきかが指示される必要がある。

5. 成分および生成されるべき変換式ファイルについてのターゲット場所。

10

【0142】

いくつかの例において、初期設定と称呼される少なくとも1つのデプロイメントが常に存在する。これは他のデプロイメントが特定されない場合に使用されるデプロイメントである。

【0143】

一の実施形態について、変換式生成の基本をここに示す。最初に、ルールセット内で計算される出力のためにルールが生成される。他の出力の全ては、変換式内のワイルドカードルールにて扱われる。一般に、内部のみにて使用される出力変数は、生成された変換式内にローカル変数を生成させる。すなわち、生成された変換式は、重複計算を避けるために、必要に応じて（例えば、最適化が領域確保よりも速度を目的とする場合）さらにローカルな変数を含んでもよい。

20

【0144】

変換型に応じて、いくつかの変換の違いがある。

・リフォーマット

入力は「in0」と呼ばれ、入力変数は「in.field」等の技術名を有してもよい。出力は「out」と呼ばれ、出力変数は「out.field」等の技術名を有してもよい。

・ジョイン

2つの入力は「in0」および「in1」と呼ばれる。出力は「out」と呼ばれ、ワイルドカードルールは「in0」を「out」にコピーすることとする。全てのパラメータは成分が生成されたときに設定される。ルールセットは、ジョインへの各入力毎に1つの複数の入力のセットを有する。ルールセットは、ジョイン型、入力が重複除外されるべきか否か、および、ジョインに対するキーとして用いられるフィールドのビジネス名、を指定する（いくつかの例において、これは各入力セットに存在しなければならない）。また、ユーザは、入力フィルタとして用いられる各入力についての数式を指定してもよい。

30

・ロールアップ

入力は「in0」と呼ばれ、出力は「out」と呼ばれる。ロールアップ型のルールセットの場合、ユーザは集計関数を使用することが許される（他の変換型ではサポートされない）。技術名が「input__select」または「output__select」である出力変数をユーザが生成する場合、これら出力を計算するルールのロジックとともに「input__select」および/または「output__select」の関数が変換式に追加される。それら関数の双方の入力は「in0」と呼ばれる（「output__select」は、通常、そのパラメータを「out」と名付けるのであるが）。ジョイン型と同様、成分が生成されたときに全てのパラメータが設定され得る。

40

・フィルタ

2つの所定の定数のうち的一方が出力される。フィルタ型変換式の出力変数は、型成分「select」のみである。同出力変数が非ゼロかつ非NULLであれば出力が渡される。いくつかの例において、これはサブグラフ内のリフォーマット成分として採用される。

【0145】

50

追加の変換型も採用され得る。

・スキャン

スキャン型ルールセットについて、ユーザは、出力の一または複数の値であってレコード間にて記憶される値を特定し得る。これら出力の値は、通常、全てのレコードについて計算される。しかし、最後のレコードからの出力の値を含む出力の各々について、追加の組み込み入力生成されてもよい。これにより、ユーザは、例えば、次のレコードにおける入力として用いられる状態にある出力変数に部分和を格納することにより、複数のレコードのフィールドの合計を計算し得る。

さらに、スキャン型ルールセットについて、ユーザはオプションキーを特定し得る。このキーは、レコードをまとめるために用いられる一または複数のフィールドである。スキャン型ルールセットについてキーが特定される際、レコード間にて記憶される全ての出力の状態はキーの特定の値ごとに異なる。例えば、キーが顧客番号であり、かつ、各顧客についての全てのトランザクションの合計を計算するために一の出力が用いられる場合、部分和は全ての顧客番号について保存され、各顧客に対して異なる和が計算される。

・分類

ルールはN個の出力を有し、変換式は各レコードについてどの出力が用いられるべきかを決定する。この成分について、システムは、`out::classify(in)`関数を生成する。出力は1つの整数値であり、どの出力ポート(2つ以上あり得る。)を使用すべきかを示すものである。例えば、0の出力は0番のポートを意味し、1の出力は1番のポートを意味する。

分類型変換の出力変数は、型成分「`select`」のみである。同出力変数は、出力ポート(ゼロベース)のインデックスとされる。これは、出力として2個の代わりにN個の値が用いられることを除き、フィルタと同様である。

・関数

関数型ルールセットは、成分変換式としてではなく、変換式ファイルに変えられ得る。これに代えて、関数型ルールセットが変換式に変えられる場合、構築した変換ファイルは、他の変換式に含まれるように設計される。各出力変数は関数に変えられる。それら関数への入力は、ルールの型に依存する。関数型ルールにおいて、入力はテーブルにリスト化された順序のルールへの入力である。非関数型ルールにおいて、各出力関数は`in`と名付けられた単一の入力(入力変数に対応するフィールドの全てを有するレコード)を用いる。

【0146】

次にグラフの一部となる変換式を生成するためにルールセットが用いられる場合、グラフ成分にはルールセットの名前およびデプロイメントが含まれる。グラフ開発者は、成分内の生成された変換式に代えて、ルールセットを編集し得る。ルールセットを変更すると、変換式は再生成される。いくつかの例において、ユーザは、BREによって生成された成分上にてシフトを押しながらダブルクリックし得る。シフトを押しながらダブルクリックすることにより、グラフィカル開発環境(GDE)がEME名、ルールセット名およびデプロイメントをパスしてBREを起動する。一の例として、コマンドラインインターフェイスがBRE処理を起動するために用いられ得る。ただし、その他のプロセス間コミュニケーション機構も採用され得る。

【0147】

上述した記録手法は、コンピュータシステム上にて実行されるソフトウェアを用いて実現され得る。例えば、ソフトウェアは、一または複数のプログラムされた又はプログラム可能なコンピュータシステム(分散型、クライアント/サーバ型、または、グリッド型などの種々のアーキテクチャのものであってよい。)上にて実行される一または複数のコンピュータプログラム中のプロシージャを構成する。ここで、同コンピュータシステムは、少なくとも一つのプロセッサ、少なくとも一つのデータ格納システム(揮発性および不揮発性メモリおよび/または記憶要素を含む。)、少なくとも一つの入力装置またはポート、および、少なくとも一つの出力装置またはポート、を含む。ソフトウェアは、より

大きなプログラム（例えば、計算グラフの設計および構成に関連する他のサービスを提供する。）の一または複数のモジュールを構成し得る。グラフのノードおよび要素は、コンピュータ読み取り可能な媒体に格納されるデータ構造として、または、データリポジトリに格納されるデータモデルに準拠する他の体系化データとして、実現され得る。

【0148】

ソフトウェアは、汎用または専用のプログラマブルコンピュータによって読み取り可能な記憶媒体（CD-ROM等）上において提供されてもよく、または、ネットワークの通信媒体を介してソフトウェアが実行されるコンピュータに伝送されてもよい（伝搬信号にエンコードされてもよい。）。関数の全ては、専用コンピュータ上にて、または、コプロセッサなどの専用ハードウェアを用いて、実行され得る。ソフトウェアは、分散方式（ソフトウェアが規定する計算の異なる部分が、異なるコンピュータにより実行される。）にて実行され得る。そのようなコンピュータプログラムの各々は、汎用または専用のプログラマブルコンピュータによって読み取り可能な記録媒体または記憶装置（例えば、固体メモリまたは媒体、あるいは、磁気媒体または光媒体）に格納されるか又はダウンロードされるか、が好ましい。これにより、コンピュータシステムが記憶媒体または記憶装置を読み取ってここに記載されているプロシーダを実行する際、コンピュータが構成されて実行される。本発明のシステムは、コンピュータプログラムにて構成されたコンピュータ読み取り可能な記憶媒体として実施され得ることもできる。そのように構成された記憶媒体により、コンピュータシステムは、ここに記載した関数を実行する特定の及び所定の方法にて作動する。

【0149】

本発明のいくつかの実施形態を説明した。しかし、当然のことながら、本発明の精神および範囲を逸脱することなく多様な改変を実施し得る。例えば、上述したいくつかのステップの順序は自由であるので、上記と異なる順序にて実行され得る。

【0150】

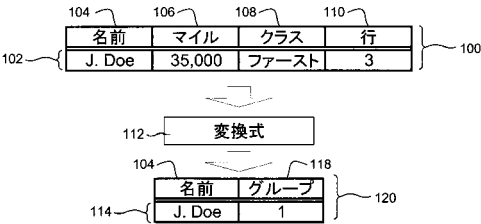
当然のことながら、上記記載は、本発明を説明することを意図したものであって本発明の範囲を限定することを意図するものではない。本発明は、添付される特許請求の範囲によって定義される。例えば、上述したいくつかの関数ステップは、全体のプロセスに実質的な影響を与えることなく異なる順序にて実行され得る。他の態様は以下の特許請求の範囲に含まれる。

10

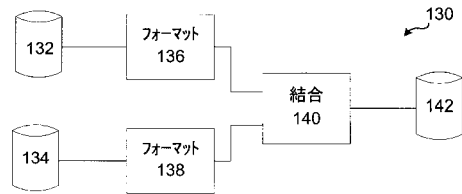
20

30

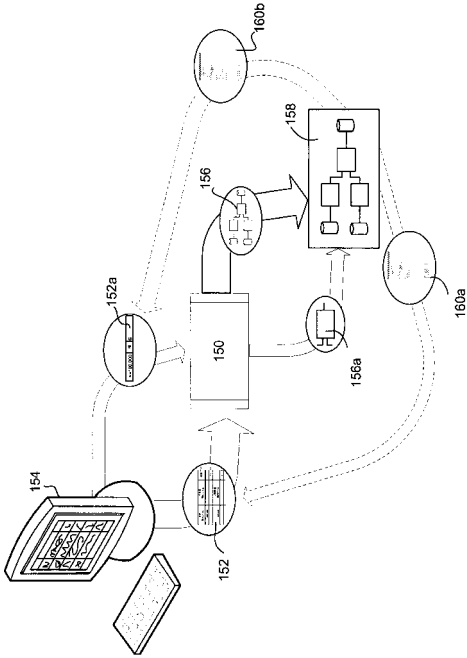
【図 1 A】



【図 1 B】



【図 1 C】



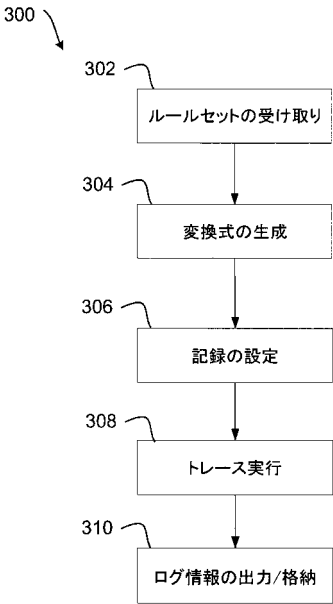
【図 2 A】

	202	204	206	208	212
	合計 飛行マイル	今年度 飛行マイル	座席クラス	座席の 行	搭乗 グループ
210a	>= 1,000,000				1
210b			ファースト		1
210c	>= 100,000	>=前年度 飛行マイル			2
210d			ビジネス		2
210e			他	<= 10	2
210f			同上	<= 40	3
210g			同上	<= 50	4
210h			同上	他	5

【図 2 B】

220a	220b														
<table><tr><th>数式</th><th>搭乗 グループ</th></tr><tr><td>合計 飛行マイル >=1,000,000</td><td>1</td></tr></table>	数式	搭乗 グループ	合計 飛行マイル >=1,000,000	1	<table><tr><th>数式</th><th>搭乗 グループ</th></tr><tr><td>座席クラス= ファースト</td><td>1</td></tr></table>	数式	搭乗 グループ	座席クラス= ファースト	1						
数式	搭乗 グループ														
合計 飛行マイル >=1,000,000	1														
数式	搭乗 グループ														
座席クラス= ファースト	1														
220c															
<table><tr><th>数式</th><th>搭乗 グループ</th></tr><tr><td>合計飛行マイル>=100,000 かつ 今年度飛行マイル>=前年度飛行マイル</td><td>2</td></tr></table>	数式	搭乗 グループ	合計飛行マイル>=100,000 かつ 今年度飛行マイル>=前年度飛行マイル	2											
数式	搭乗 グループ														
合計飛行マイル>=100,000 かつ 今年度飛行マイル>=前年度飛行マイル	2														
220d	220e														
<table><tr><th>数式</th><th>搭乗 グループ</th></tr><tr><td>座席クラス= ビジネス</td><td>2</td></tr></table>	数式	搭乗 グループ	座席クラス= ビジネス	2	<table><tr><th>席の行</th><th>搭乗 グループ</th></tr><tr><td><= 10</td><td>2</td></tr><tr><td><= 40</td><td>3</td></tr><tr><td><= 50</td><td>4</td></tr><tr><td>他</td><td>5</td></tr></table>	席の行	搭乗 グループ	<= 10	2	<= 40	3	<= 50	4	他	5
数式	搭乗 グループ														
座席クラス= ビジネス	2														
席の行	搭乗 グループ														
<= 10	2														
<= 40	3														
<= 50	4														
他	5														

【図 3】



【 図 4 】

400

Deployment: Default deployment

GENERATE COMPONENT SETTINGS

Name: Default deployment

Description:

Comment:

Transform EME Path: Projects\Dataset\InfoSample.xd

Component EME Path: Projects\Dataset\Components\Sample.rqp

Target Co-Operating System: <empty>

Dataset:

Type	Name	Dataset
Input	Input 1	Dataset 1
Output	Output	Dataset 1

LOGGING SETTINGS

Create a log record:

☒ Never

☐ For each input record

☐ When an error occurs

☐ When this expression is True

Include in each log record:

☐ Data node type (recommended)

☐ Input values

☐ Output values

☐ Error values

☐ Output values: non-multiple values

☐ Procedural values

☐ Result of processing in input (for output only)

INPUT DATA

Name	Kind	Location
1 Input	File	

BASLINE DATA

Name	Kind	Location
1 Outputs	File	

Sandbox:

410

420

フロントページの続き

(72)発明者 ゲールド ジョエル

アメリカ合衆国 01890 マサチューセッツ州 ウィンチェスター ソーンベリー ロード
50

(72)発明者 フィリモア ディヴィッド

アメリカ合衆国 01720 マサチューセッツ州 アクトン ジャクソン ドライブ 41

審査官 相崎 裕恒

(56)参考文献 特表2008-544423(JP,A)

米国特許出願公開第2007/0021995(US,A1)

特開平05-241890(JP,A)

特開平08-147168(JP,A)

特開平05-100856(JP,A)

(58)調査した分野(Int.Cl., DB名)

G06F	11/28	-	11/34
	17/30		
	19/00	-	19/28
G06N	5/00	-	7/06
G06Q	10/00	-	10/10
	30/00	-	30/08
	50/00	-	50/20
	50/26	-	99/00