(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
10 June 2010 (10.06.2010)

PCT

(10) International Publication Number
WO 2010/065511 A1

(51) International Patent Classification:
G06F 7/00 (2006.01)

(21) International Application Number:
PCT/US2009/066210

(22) International Filing Date:
1 December 2009 (01.12.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/119,164 2 December 2008 (02.12.2008) US

(71) Applicant *(for all designated States except US)*: **AB INITIO SOFTWARE LLC** [US/US]; 201 Spring Street, Lexington, Massachusetts 02421 (US).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **WAKELING, Tim** [GB/US]; 11 Abbot Street, Andover, Massachusetts 01810 (US). **WEISS, Adam** [US/US]; 15 Rawson Avenue, Lexington, Massachusetts 02420 (US).

(74) Agents: **HENNESSEY, Gilbert, H.** et al.; Fish & Richardson P.C., P.O. Box 1022, Minneapolis, Minnesota 55440-1022 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— *with international search report (Art. 21(3))*

— *with amended claims (Art. 19(1))*

(54) Title: MAPPING INSTANCES OF A DATASET WITHIN A DATA MANAGEMENT SYSTEM

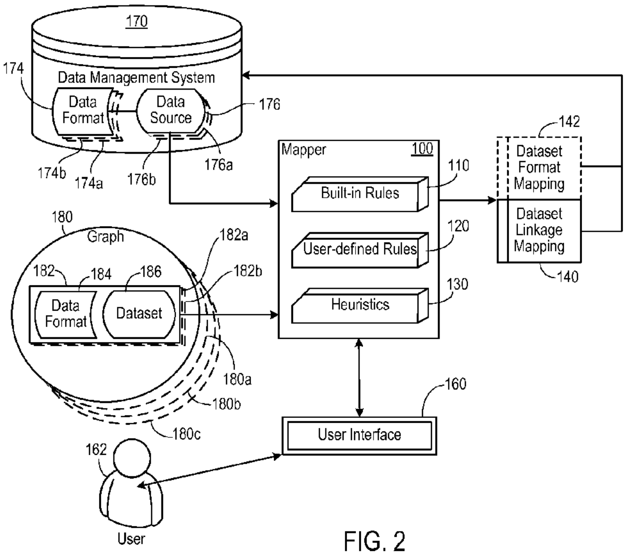

FIG. 2

(57) Abstract: Mapping data stored in a data storage system (170) for use by a computer system includes processing specifications of dataflow graphs (180) that include nodes representing computations interconnected by links representing flows of data. At least one of the dataflow graphs receives a flow of data from at least one input dataset and at least one of the dataflow graphs provides a flow of data to at least one output dataset. A mapper (100) identifies one or more sets of datasets. Each dataset in a given set matches one or more criteria for identifying different versions of a single dataset. A user interface (160) is provided to receive a mapping between at least two datasets in a given set. The mapping received over the user interface is stored in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

MAPPING INSTANCES OF A DATASET WITHIN A DATA MANAGEMENT SYSTEM

CROSS-REFERENCE TO RELATED APPLICATIONS

This application claims priority to U.S. Application Serial No. 61/119,164, filed on December 2, 2008, incorporated herein by reference.

BACKGROUND

This description relates to mapping instances of a dataset within a data management system.

A modern data management system may include a multitude of elements representing different aspects of the system. Systems of lesser complexity often allow data to be viewed directly without additional processing for the purpose of accurate visualization. Systems of greater complexity may require additional mechanisms for the data to be meaningfully viewed. A complex data management system made up of many elements may store data in many different forms and process data in many different ways. These forms of storage and processing many relate to each other in ways that are not apparent without a way to analyze the relationships.

SUMMARY

In a general aspect, a method for mapping data stored in a data storage system for use by a computer system includes processing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset; identifying one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset; providing a user interface to receive a mapping between at least two datasets in a given set; and storing the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

In another general aspect, a system for mapping data stored in a data storage system includes a data storage system storing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset; a mapper that identifies one or more sets of datasets associated with the dataflow graphs, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset; a user interface that receives a mapping between at least two datasets in a given set, and stores the mapping in the data storage system in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

In another general aspect, a system for mapping data stored in a data storage system includes: means for processing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset; means for identifying one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset; means for providing a user interface to receive a mapping between at least two datasets in a given set; and means for storing the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

In another general aspect, a computer program for mapping data stored in a data storage system is stored on a computer-readable medium, and includes instructions for causing a computer to process specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset; identify one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset; provide a user interface to receive a mapping between at least two datasets in a given set; and store the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

Aspects can include one or more of the following features.

The set is presented over the user interface.

A list of possible mappings ordered according to a quantification of a match to the one or more criteria is presented over the user interface.

The list of possible mappings includes candidates that are more likely to be an instance of a given dataset ordered higher in the list.

One of the criteria is built into a mapper that identifes the one or more sets of datasets.

One of the criteria is received from the user interface.

At least one of the possible mappings indicates a component of a dataflow graph that represents a dataset, and at least one of the possible mappings indicates a component of a dataflow graph that does not represent a dataset.

A sub-graph of a dataflow graph including multiple components represents a dataset.

The sub-graph includes a data component.

The sub-graph includes an executable component.

Identifying one or more sets of datasets includes using heuristics for determining if a dataset in a given set has one or more characteristics in common with another dataset.

The characteristics include the quantity of bytes and records in a representation of the dataset.

The characteristics include the name of a representation of the dataset.

The characteristics include the date of creation of a representation of the dataset.

The characteristics include the data format of a representation of the dataset.

At least one of the datasets of the mapping belongs to a group of datasets known to a data management system.

A format mapping is provided between datasets in a given set.

The mapping includes an identifier that points to a record in the data management system that keeps track of the dataset.

The mapping is updated based on a change in a dataset.

Aspects of the invention can include one or more of the following advantages.

By identifying sets of datasets according to version identification criteria, a match between two instances of a dataset can be made more efficiently than purely manual operation. Further, by providing a user interface to receive a mapping between at least two datasets, the mapping will be more accurate than if the system was purely automatic.

Other features and advantages of the invention will become apparent from the following description, and from the claims.

DESCRIPTION OF DRAWINGS

FIG. 1 is a dataflow graph.

FIG. 2 is an overview of a dataset mapper and associated components.

FIGS. 3A-3E are diagrams of different scenarios handled by a dataset mapper.

FIG. 4 is a flowchart of dataset mapper operation.

FIG. 5 is a dataset linkage mapping.

FIG. 6 is a dataset format mapping.

DESCRIPTION

1 Overview

A data processing element may be in the form of a graph. A graph-based computation is implemented using a "dataflow graph" that is represented by a directed graph, with vertices in the graph representing components (either data storage components corresponding to stored data or computation components corresponding to executable processes), and the directed links or "edges" in the graph representing flows of data between components. A dataflow graph (also called simply a "graph") is a modular entity. Each graph can be made up of one or more other graphs, and a particular graph can be a component in a larger graph. A graphic development environment (GDE) provides a user interface for specifying executable graphs and defining parameters for the graph components.

Referring to FIG. 1, an example of a dataflow graph 101 includes an input component 102 providing a collection of data to be processed by the executable components 104a – 104j of the dataflow graph 101. For example, the dataset 102 can include data records associated with a database system or transactions associated with a transaction processing system. Each executable component is associated with a portion of the computation defined by the overall dataflow graph 101. Work elements (e.g., individual data records from the data collection) enter one or more input ports of a component, and output work elements (which are in some cases the input work elements, or processed versions of the input work elements) typically leave one or more output ports of the component. In graph 101, output work elements from components 104e, 104g, and 104j are stored in output data components 102a – 102c.

A dataset is an object (e.g., stored in an object oriented database) that represents a particular collection of data. In the context of a system of dataflow graphs, a component is capable of representing a dataset. In these cases, a graph may interact with the component representing a dataset (or simply "dataset component") in one or more ways. A dataset component includes instructions for accessing the physical data represented by a given dataset, so a graph can accept input from a dataset using a dataset component, provide output to a dataset using a dataset component, and process data of a dataset using a dataset component at an intermediate step. A dataset component can include various kinds information associated with a given dataset object including an instance of the dataset object. Such a system could have many dozens, hundreds, or thousands of graphs and associated dataset components. As the complexity of such a system increases, the relationships between different graphs and dataset components become more difficult to manage. More than one dataset component in the system can represent the same data source and each such dataset component can be associated with a different graph, graph subset, or executable component.

For example, in one possible scenario, a single dataset may be stored in more than one location associated with the data management system. In this scenario, two or more data sources contain similar or identical versions of the same data. Two graphs in the system might handle this single dataset, but each graph reads from and writes to a different data file, a different database table, or another type of dataset component.

In a similar scenario, the data (e.g., data files) represented by a given dataset may be not only stored in more than one location, but also interpreted using different data storage formats. As with the above example, two graphs may operate on two separate data files containing the same data, differing only in format. Each data file may have a different arrangement of data types, despite containing instances of the same data.

In an alternative scenario, one graph may operate on a data file containing an instance of the dataset, and another graph may operate on a database table also containing an instance of the dataset. In such a case, a data file and a database table will generally have two different data formats.

In another scenario, the data management system may access different versions of the same dataset each in different ways. One graph may access an instance of the dataset directly, such as by reading in a data file through a standard file input/output mechanism. Another graph may retrieve a file by querying an external source, such as a data repository available via a network. A graph may also access a database table retrieved through a similar external query, such as a query to a networked database.

The data management system may also make reference to different instances of the same dataset each in different ways. For example, a graph may be capable of accessing different data locations according to a parameter. Such a parameter could point to any number of data locations over time. A graph that operates multiple times may access different locations on different occasions if the parameter varies between executions of the graph.

In some scenarios, the representation of a dataset within a graph may not be a single component, but rather a collection of components, such as a "sub-graph" component within a graph that is itself implemented as a graph with multiple components. The collection may include one or more dataset components, and could also include one or more executable components.

All of these scenarios can potentially pose a problem for visualizing and analyzing the data handled by the data management system. If a user requires a consolidated view of the components that interact with a given dataset, various approaches can be used to reconcile the different instances of the dataset that may exist.

One approach is an automatic mechanism that identifies multiple instances of the same dataset and creates linkages between them. However, some automatic mechanisms have drawbacks, such as the following three drawbacks. First, the mechanism may require that each instance of a dataset be stored in particular manner, such as under a unified naming scheme and directory structure. This provides the mechanism with a way to identify and locate each one in the storage system associated with the data management system. However, this arrangement limits the flexibility of the data management system and may be too restrictive for some uses of the system.

Second, under several scenarios of operation, the mechanism may not properly identify instances of the same dataset and form the correct linkages. For example, this is likely if a dataset is accessed using an externally-referenced entity, and the automatic mechanism does not have access to that entity. Similarly, this is likely if a component accesses a dataset according to an independent parameter in a parameter list, and the mechanism does not have a way to access or interpret the parameter list. Further, this is likely if a dataset is represented by a complex entity made up of one or more dataset components and executable components, such as a sub-graph. An automatic mechanism may be unable to discern what particular combination of components represents a particular dataset.

Third, the mechanism may form redundant or unnecessary linkages between dataset instances. For example, some of the datasets handled by the data management system may represent extraneous data, such as the contents of error logs. Any linkages between instances of these datasets are unnecessary. Further, some of the instances of a dataset handled by the data management system may be redundant instances, such as cached data or other temporary copies of data. A linkage that connects to this type of data quickly becomes obsolete and would be confusing to a user examining the data management system.

An alternative approach is a system in which a user manually consolidates instances of the same dataset via a user interface. A user is less likely to miss essential linkages between instances of a dataset, and is also less likely to create redundant or unnecessary linkages between instances of a dataset. However, if the data management system has hundreds or thousands of components, the amount of time needed for the user to manually create the necessary linkages is prohibitively large.

In a partially-automated approach, a dataset mapper is used to provide some automatic analysis, and to enable some interaction with a user in a way that is not prohibitive for a user of a large and/or complex system.

FIG. 2 is a block diagram of one embodiment of an exemplary dataset mapper 100 showing the interrelationship between associated principal elements. A dataset mapper 100 is capable of analyzing a set of one or more graphs 180, 180a, 180b, 180c. Each graph is associated with one or more dataset components 182, 182a, 182b, where each dataset component could correspond to a data file, a database table, a sub-graph, or another kind of component representing a dataset. The mapper 100 analyzes the graphs for the purpose of forming linkages between dataset components that contain instances of the same dataset 186. The mapper 100 processes each dataset component according to a combination of built-in rules 110, user-defined rules 120, and heuristics 130, to determine if a dataset component 182 may contain an instance of one of several datasets representing data sources 176, 176a, 176b known to a data management system 170. The mapper 100 passes this information to a user interface 160, which allows a user 162 to select the proper dataset, if any, that corresponds to the dataset component 182. For example, the user interface 160 presents a list of possible candidate mappings based on a match to one or more criteria for identifying different versions or instances of a single dataset. Examples of such criteria, including criteria based on built-in rules, user-defined rules, and heuristics, are described in more detail below. The list can be ordered according to quantification of the match to the one or more criteria (e.g., candidates that are more likely to be an instance of a given dataset are ordered higher in the list). The mapper 100 then generates a dataset linkage mapping 140 that indicates that the dataset component 182 contains an instance of the dataset representing a data source 176.

Further, the dataset component 182 can have a data format 184 that differs from the format 174 of a corresponding linked data source 176. Depending on the requirements of the data management system 170, the user may choose to establish a single data format for all instances of the dataset. The system stores a format 174, 174a, 174b for each data source 176, 176a, 176b. Alternatively, the user can choose to create an optional mapping 142 between the format 184 of the dataset component 182 and the established format 174 of the corresponding data source 176. The optional data format mapping 142 allows the system 170 to retain information about the data types for each instance of the dataset.

The mapper 100 also enables a user to indicate a linkage between an executable component and a single dataset component, which may have no other linkages to it. For example, a dataset component may correspond to a source dataset with only one reader or a target dataset with only one writer. If the dataset object already exists in the system and has other relevant metadata, such as the correct record format, documentation, data profiles, etc., the linkage enables the dataset component to be mapped to the correct dataset.

2 Mapping Process

The mapper 100 is capable of handling common scenarios that arise in complex data management system. In a first scenario, shown in FIG. 3A, one graph 210 provides a dataset component 212 as output, and another graph 220 accepts a different dataset component 222 as input. Each dataset component contains an instance of the same dataset 216. This dataset may be the same as a dataset representing a data source 176 known to the data management system. Further, the first dataset component 212 has a data format 214 that may be the same as the format belonging to the second dataset component 222, or, alternatively, the second component may have a different format 224. The mapper 100 is capable of identifying the second dataset component 222 as being an instance of the dataset 216 represented by the first dataset component 212 and creating an appropriate linkage mapping 140.

In a second scenario, shown in FIG. 3B, a graph 230 is associated with an external dataset component 232 using an external reference 238 to an external source 239. The external dataset component 232 has a data format 234 and is an instance of a dataset 236. As in the first scenario, the dataset 236 represented by the external dataset component may be a dataset representing a data source 176 known to the data management system 170. The mapper 100 is capable of identifying this external dataset component 232 as being an instance of another dataset and creating an appropriate linkage mapping 140.

In a third scenario, shown in FIG. 3C, a graph 240 is associated with a dataset component 242 using a parameter 248 in a parameter list 247. The referenced dataset component 242 has a data format 244 and is an instance of a dataset 246. As in the first and second scenarios, the dataset 246 represented by the referenced dataset component may be a dataset representing a data source 176 known to the data management system 170. The mapper 100 is capable of identifying this referenced dataset component 242 as being an instance of another dataset and creating an appropriate linkage mapping 140.

In a fourth scenario, shown in FIG. 3D, a graph 250 is associated with an external component 251 using an external reference 258 to an external source 259. The external component 251 is not a dataset component, but rather another kind of component, such as an executable component. The mapper 100 is capable of identifying this external component 251 as inapplicable to the dataset linkage mapping process.

In a fifth scenario, shown in FIG. 3E, a graph 260 is associated with a sub-graph component 263, itself made up of several components. These components include at least one dataset component 262, and, in this example, one or more executable components 261a, 261b, 261c. Under this scenario, the sub-graph 263 as a single entity represents at least one dataset. Other exemplary sub-graphs may include multiple dataset components, and any number of executable components, including zero. Further, this sub-graph 263 has multiple outputs 265a, 265b. Each output is capable of providing a different instance of a dataset to the component that receives the output. Another exemplary sub-graph could also have any number of inputs. A further exemplary sub-graph may have no inputs or outputs that correspond to a respective dataset. For cases where the sub-graph does represent at least one dataset, the mapper 100 is capable of identifying the sub-graph 263 as being an instance of at least one dataset and creating at least one appropriate linkage mapping 140.

An example of a sequence of operation of the mapper is shown in FIG. 4. In step 302, the mapper first identifies, of the elements associated with a graph, which elements represent datasets. Generally, a graph will have one or more inputs and outputs, and each input and each output could be an instance of a dataset. Each graph may also handle an instance of a dataset at some intermediate step. As a result, each graph can be connected to multiple components that are capable of being dataset candidates. In some cases, the data management system has information about the characteristics of some of the components, including information about whether or not the component represents a dataset. In those cases, the mapper adds the potential dataset components to a table of dataset candidates in step 304. In some cases, a component could be a sub-graph made up of multiple components, including dataset components and executable components. A sub-graph could represent at least one instance of a dataset. Accordingly, the mapper compiles a list of all such sub-graphs and adds them to the table of dataset candidates as part of step 304. In other cases, the nature of the component may not be available to the data management system. The component could be accessed through a reference to an external entity, where the reference may be a query to a database table, a Uniform Resource Locator pointing to an Internet server, a parameter in a parameter list, or another type of reference. In these cases, the mapper generally has no means by which it can independently access the entity pointed to by the reference. Accordingly, the mapper compiles a list of all such references and adds them to the table of dataset candidates as part of step 304.

Next, in step 306, for a given dataset candidate, the mapper generates a list of known datasets that the dataset candidate could map to. The mapper uses a combination of user-defined rules, built-in rules, and heuristics to evaluate which known datasets could map to a dataset candidate.

Next, in step 308, the user then selects the known dataset that corresponds to the dataset candidate. The user may also access a full list of all known datasets, if none of the suggested known datasets is the correct match. In addition, the user can indicate that the dataset candidate is not a dataset. For example, a reference to a remote server could be a call to a remote executable procedure, which is not a data entity. As another example, the dataset candidate may represent data, but it may be data of a kind not pertinent to the data management system, such as an error log. In this case, the user may indicate to the user interface that this data is to be ignored in the mapping process.

Next, in step 310, the user identifies the data format of the newly-mapped dataset. The system may have a set of data format templates, one of which can be selected. Alternatively, the user can create a new data format in the user interface.

Next, in step 312 the mapper uses this information to generate a linkage mapping for the dataset candidate, and, optionally, a format mapping.

Next, the mapper offers the next dataset candidate to the user for linkage generation in another iteration of steps 308, 310, and 312, unless the mapper has processed all dataset candidates.

Next, in step 314, the user views the components associated with the data management system, to ensure that a visualization of the associations between graphs and dataset components is accurate based on the new linkages between components. In step 316, the user has the option of making any adjustments to the linkage and format mapping.

Finally, in step 318, the mapper delivers the linkage and format mapping to the data management system. The mappings can be stored alongside one or more graphs, or in a separate storage entity associated with the data management system, or by another means.

3 Dataset Mapping Maintenance

The mapper 100 is capable of handling multiple scenarios that may arise that affect the integrity of the dataset linkages.

The first scenario includes identifying new dataset candidates when new components are added to the data management system 170. Under this scenario, the mapper 100 analyzes each component and presents possible linkages to the user. The mapper 100 is capable of operating on any new components to generate the appropriate linkages as needed.

The second scenario includes maintaining the existing linkages as the data management system 170 changes over time. For example, new instances of a dataset may have come into existence over the course of the normal operation of the graphs associated with the system. As another example, a dataset may have changed its identity, such as its name or location in the system. As a further example, a dataset may have been deleted entirely. As another further example, a dataset candidate may have been overlooked in a previous round of linkage creation, and so the collection of linkages is incomplete. The user interface 160 of the mapping system allows a user 162 to modify the existing linkages to remedy any mappings that are incomplete or outdated.

The third scenario includes automatically updating linkages for dataset references that invariably follow a known pattern. For example, a graph may handle a dataset that is referenced in a parameter list 247. Such a parameter list may change over time. If the parameter list follows a standard format known to the data management system, the mapper can identify changes in the parameter list and update the existing linkages accordingly.

4 Dataset Linkage Mapping

As shown in FIG. 5, a dataset linkage mapping 140 contains a component name 402, a dataset name 404, a dataset type 406, a format 408, a master dataset location 410, and a flag 412. The component name 402 is the dataset component or sub-graph that represents this instance of the dataset. The dataset name 404 is an identifier that points to the dataset represented by this component. The dataset type 406 indicates the category that this instance of the dataset falls under, for example, a data file, or a database table, or another type. The format 408 is the format or arrangement that this instance of the dataset uses to represent its data. The master dataset location 410 is an identifier that points to the record in the data management system that keeps track of this dataset. Finally, the flag 412 indicates whether or not this instance of the dataset should be ignored, for example, if the user has identified this instance of the dataset as not applicable to the data management system and should be excluded from the set of linkages.

5 Built-in Rules

The mapper 100 has a set of built-in rules 110 that operate according to standard conventions of the data management system. The mapper can identify datasets corresponding to a dataset component with the highest degree of accuracy if the dataset component follows the built-in rules 110. In one exemplary implementation of a rule, externally-referenced database tables containing dataset candidates must be placed in persistent storage under a standardized directory structure used by the data management system. Further, a graph that accesses an externally-referenced dataset component according to a parameter must use a parameter that the data management system is also capable of accessing and resolving. Further, the format of a dataset component must be available in persistent storage and accessible by the data management system. Other built-in rules are also possible, depending on the data management system.

6 User-defined Rules

In addition to the built-in rules that the mapper uses to identify dataset candidates, the mapper 100 also has a collection of optional user-defined rules 120. These rules 120 may be enabled or disabled by a user, depending on which are applicable to the user's particular data management system. In one exemplary implementation, the mapper has six user-defined optional rules. The mapper can ignore some of the information in the name of a database table, if some of the information in the name obscures the identity of the table, such as information about the a user who defined the table. Further, the mapper can eliminate this information from the name of a database table. Further, the mapper can ignore a particular category of data files that are known to contain data that is not pertinent to the datasets associated with the data management system. Such a category could be a data file type or data file extension. Further, the mapper can resolve references to a particular parameter in a parameter list and replace the reference with the name of the parameter itself. Further, the mapper can eliminate references to a parameter entirely. The user can also create other rules for the mapper to follow.

7 Heuristics

In addition to following the built-in and user-defined rules to evaluate dataset candidates, the mapper 100 also uses a set of heuristics 130. The heuristics 130 allow the mapper to analyze the characteristics of a given dataset component and compare those characteristics to known datasets. A dataset component with similar characteristics to a known dataset is likely to be an instance of that dataset. In one exemplary implementation, the mapper uses two heuristics. One heuristic is the characteristics of the data of a given dataset component. For example, if the data associated with a dataset component has the same quantity of bytes and records as does the data associated with a known dataset, then that dataset component is likely to be an instance of that dataset. Further, if the dataset component has a name or date of creation similar to that of a known dataset, then the dataset component is likely to be an instance of that dataset. A second heuristic is the data format of a dataset component. If a dataset component shares a data format with a known dataset, then the dataset component is likely to be an instance of the dataset. This heuristic is less reliable in situations where multiple distinct datasets use the same data format.

8 Dataset Formats and Mapping

Each dataset representing a data source has an associated data format that indicates, for each element in the dataset, what type of data the element represents. For example, the data format of a database table indicates the data types of each field within a given record. The data management system 170 retains a single data format 174, 174a, 174b for each dataset representing a data source 176, 176a, 176b.

If the mapper 100 has encountered a dataset component 182 that represents a new dataset 186, then the mapper 100 creates a corresponding data format to be stored by the data management system, based on the data format 184 of the dataset component 182.

In some cases where a dataset component 182 represents a known dataset representing a data source 176, the dataset component 182 has a different data format 184 than the data format 174 of the known dataset representing a data source 176. The data management system 170 handles the dataset representing a data source 176 as a single entity, independent of the number of instances of that dataset that may exist. Consequently, the data management system 170 relies on the mapper 100 to consolidate the different formats 174, 184 when these situations arise. In one implementation, the mapper is capable of addressing each situation in one of four different ways depending on the requirements of the user and the data management system. The user 162 can choose any one of the four methods of consolidation for each situation.

Under the first method of consolidation, the mapper 100 uses the data format 184 of the dataset component 182 as the master data format of the dataset and updates the data management system 170 accordingly.

Under the second method of consolidation, the mapper 100 uses the data format 174 of the existing dataset as the master data format of the dataset and updates the data management system 170 accordingly.

Under the third method of consolidation, the mapper 100 retains both data formats, and generates a mapping 142 between the fields of each data format. As shown in FIG. 6, the dataset format mapping 142 indicates which fields 512a, 512b, 512c of the dataset format 510 correspond to which fields 522a, 522b, 522c of the format of the dataset instance, e.g. the dataset component.

Under the fourth method of consolidation, the mapper generates a new union data format capable of acting as either data format.

9 General Computer Implementation

The dataset mapping approach described above can be implemented using software for execution on a computer. For instance, the software forms procedures in one or more computer programs that execute on one or more programmed or programmable computer systems (which may be of various architectures such as distributed, client/server, or grid) each including at least one processor, at least one data storage system (including volatile and non-volatile memory and/or storage elements), at least one input device or port, and at least one output device or port. The software may form one or more modules of a larger program, for example, that provides other services related to the design and configuration of dataflow graphs. The nodes and elements of the graph can be implemented as data structures stored in a computer readable medium or other organized data conforming to a data model stored in a data repository.

The software may be provided on a storage medium, such as a CD-ROM, readable by a general or special purpose programmable computer or delivered (encoded in a propagated signal) over a communication medium of a network to the computer where it is executed. All of the functions may be performed on a special purpose computer, or using special-purpose hardware, such as coprocessors. The software may be implemented in a distributed manner in which different parts of the computation specified by the software are performed by different computers. Each such computer program is preferably stored on or downloaded to a storage media or device (e.g., solid state memory or media, or magnetic or optical media) readable by a general or special purpose programmable computer, for configuring and operating the computer when the storage media or device is read by the computer system to perform the procedures described herein. The inventive system may also be considered to be implemented as a computer-readable storage medium, configured with a computer program, where the storage medium so configured causes a computer system to operate in a specific and predefined manner to perform the functions described herein.

A number of embodiments of the invention have been described. Nevertheless, it will be understood that various modifications may be made without departing from the spirit and scope of the invention. For example, some of the steps described above may be order independent, and thus can be performed in an order different from that described.

It is to be understood that the foregoing description is intended to illustrate and not to limit the scope of the invention, which is defined by the scope of the appended claims. For example, a number of the function steps described above may be performed in a different order without substantially affecting overall processing. Other embodiments are within the scope of the following claims.

What is claimed is:

1. A method for mapping data stored in a data storage system for use by a computer system, the method including:

processing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

identifying one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset;

providing a user interface to receive a mapping between at least two datasets in a given set; and

storing the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

2. The method of claim 1, including presenting the set over the user interface.

3. The method of claim 1, including presenting over the user interface a list of possible mappings ordered according to a quantification of a match to the one or more criteria.

4. The method of claim 3, wherein the list of possible mappings includes candidates that are more likely to be an instance of a given dataset ordered higher in the list.

5. The method of claim 3, wherein one of the criteria is built into a mapper that identifes the one or more sets of datasets.

6. The method of claim 3, wherein one of the criteria is received from the user interface.

7. The method of claim 3, wherein at least one of the possible mappings indicates a component of a dataflow graph that represents a dataset, and at least one of the possible mappings indicates a component of a dataflow graph that does not represent a dataset.

8. The method of claim 1, wherein a sub-graph of a dataflow graph including multiple components represents a dataset.

9. The method of claim 8, wherein the sub-graph includes a data component.

10. The method of claim 8, wherein the sub-graph includes an executable component.

11. The method of claim 1, wherein identifying one or more sets of datasets includes using heuristics for determining if a dataset in a given set has one or more characteristics in common with another dataset.

12. The method of claim 11, wherein the characteristics include the quantity of bytes and records in a representation of the dataset.

13. The method of claim 11, wherein the characteristics include the name of a representation of the dataset.

14. The method of claim 11, wherein the characteristics include the date of creation of a representation of the dataset.

15. The method of claim 11, wherein the characteristics include the data format of a representation of the dataset.

16. The method of claim 1, wherein at least one of the datasets of the mapping belongs to a group of datasets known to a data management system.

17. The method of claim 1, further including providing a format mapping between datasets in a given set.

18. The method of claim 1, wherein the mapping includes an identifier that points to a record in the data management system that keeps track of the dataset.

19. The method of claim 1, further including updating the mapping based on a change in a dataset.

20. A system for mapping data stored in a data storage system, the system including

a data storage system storing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

a mapper that identifies one or more sets of datasets associated with the dataflow graphs, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset;

a user interface that receives a mapping between at least two datasets in a given set, and stores the mapping in the data storage system in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

21. The system of claim 20, wherein the user interface presents the set.

22. The system of claim 20, wherein the user interface presents a list of possible mappings ordered according to a quantification of a match to the one or more criteria.

23. The system of claim 22, wherein the list of possible mappings includes candidates that are more likely to be an instance of a given dataset ordered higher in the list.

24. The system of claim 22, wherein one of the criteria is built into the mapper.

25. The system of claim 22, wherein one of the criteria is received by the user interface.

26. The system of claim 22, wherein at least one of the possible mappings indicates a component of a dataflow graph that represents a dataset, and at least one of the possible mappings indicates a component of a dataflow graph that does not represent a dataset.

27. The system of claim 20, wherein a sub-graph of a dataflow graph including multiple components represents a dataset.

28. The system of claim 27, wherein the sub-graph includes a data component.

29. The system of claim 27, wherein the sub-graph includes an executable component.

30. The system of claim 20, wherein the mapper uses heuristics for determining if a dataset in a given set has one or more characteristics in common with another dataset.

31. The system of claim 30, wherein the characteristics include the quantity of bytes and records in a representation of the dataset.

32. The system of claim 30, wherein the characteristics include the name of a representation of the dataset.

33. The system of claim 30, wherein the characteristics include the date of creation of a representation of the dataset.

34. The system of claim 30, wherein the characteristics include the data format of a representation of the dataset.

35. The system of claim 20, wherein at least one of the datasets of the mapping belongs to a group of datasets known to a data management system.

36. The system of claim 20, wherein the mapper generates a format mapping between datasets in a given set.

37. The system of claim 20, wherein the mapping includes an identifier that points to a record in the data management system that keeps track of the dataset.

38. The system of claim 20, wherein the mapper updates the mapping based on a change in a dataset.

39. A system for mapping data stored in a data storage system, the system including:

means for processing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

means for identifying one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset;

means for providing a user interface to receive a mapping between at least two datasets in a given set; and

means for storing the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

40. A computer-readable medium storing a computer program for mapping data stored in a data storage system, the computer program including instructions for causing a computer to:

process specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

identify one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset;

provide a user interface to receive a mapping between at least two datasets in a given set; and store the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

AMENDED CLAIMS
received by the International Bureau on 11 March 2010

1. A method for mapping data stored in a data storage system for use by a computer system, the method including:

processing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

identifying one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset, each version of the single dataset representing data received or provided by a different one of the dataflow graphs;

providing a user interface to receive a mapping between at least two datasets in a given set; and

storing the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

2. The method of claim 1, including presenting the set over the user interface.

3. The method of claim 1, including presenting over the user interface a list of possible mappings ordered according to a quantification of a match to the one or more criteria.

4. The method of claim 3, wherein the list of possible mappings includes candidates that are more likely to be an instance of a given dataset ordered higher in the list.

5. The method of claim 3, wherein one of the criteria is built into a mapper that identifies the one or more sets of datasets.

6. The method of claim 3, wherein one of the criteria is received from the user interface.

7. The method of claim 3, wherein at least one of the possible mappings indicates a component of a dataflow graph that represents a dataset, and at least one of the possible mappings indicates a component of a dataflow graph that does not represent a dataset.

8. The method of claim 1, wherein a sub-graph of a dataflow graph including multiple components represents a dataset.

9. The method of claim 8, wherein the sub-graph includes a data component.

10. The method of claim 8, wherein the sub-graph includes an executable component.

11. The method of claim 1, wherein identifying one or more sets of datasets includes using heuristics for determining if a dataset in a given set has one or more characteristics in common with another dataset.

12. The method of claim 11, wherein the characteristics include the quantity of bytes and records in a representation of the dataset.

13. The method of claim 11, wherein the characteristics include the name of a representation of the dataset.

14. The method of claim 11, wherein the characteristics include the date of creation of a representation of the dataset.

15. The method of claim 11, wherein the characteristics include the data format of a representation of the dataset.

16. The method of claim 1, wherein at least one of the datasets of the mapping belongs to a group of datasets known to a data management system.

17. The method of claim 1, further including providing a format mapping between datasets in a given set.

18. The method of claim 1, wherein the mapping includes an identifier that points to a record in the data management system that keeps track of the dataset.

19. The method of claim 1, further including updating the mapping based on a change in a dataset.

20. A system for mapping data stored in a data storage system, the system including

a data storage system storing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

a mapper that identifies one or more sets of datasets associated with the dataflow graphs, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset, each version of the single dataset representing data received or provided by a different one of the dataflow graphs;

a user interface that receives a mapping between at least two datasets in a given set, and stores the mapping in the data storage system in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

21. The system of claim 20, wherein the user interface presents the set.

22. The system of claim 20, wherein the user interface presents a list of possible mappings ordered according to a quantification of a match to the one or more criteria.

23. The system of claim 22, wherein the list of possible mappings includes candidates that are more likely to be an instance of a given dataset ordered higher in the list.

24. The system of claim 22, wherein one of the criteria is built into the mapper.

25. The system of claim 22, wherein one of the criteria is received by the user interface.

26. The system of claim 22, wherein at least one of the possible mappings indicates a component of a dataflow graph that represents a dataset, and at least one of the possible mappings indicates a component of a dataflow graph that does not represent a dataset.

27. The system of claim 20, wherein a sub-graph of a dataflow graph including multiple components represents a dataset.

28. The system of claim 27, wherein the sub-graph includes a data component.

29. The system of claim 27, wherein the sub-graph includes an executable component.

30. The system of claim 20, wherein the mapper uses heuristics for determining if a dataset in a given set has one or more characteristics in common with another dataset.

31. The system of claim 30, wherein the characteristics include the quantity of bytes and records in a representation of the dataset.

32. The system of claim 30, wherein the characteristics include the name of a representation of the dataset.

33. The system of claim 30, wherein the characteristics include the date of creation of a representation of the dataset.

34. The system of claim 30, wherein the characteristics include the data format of a representation of the dataset.

35. The system of claim 20, wherein at least one of the datasets of the mapping belongs to a group of datasets known to a data management system.

36. The system of claim 20, wherein the mapper generates a format mapping between datasets in a given set.

37. The system of claim 20, wherein the mapping includes an identifier that points to a record in the data management system that keeps track of the dataset.

38. The system of claim 20, wherein the mapper updates the mapping based on a change in a dataset.

39. A system for mapping data stored in a data storage system, the system including:

means for processing specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

means for identifying one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset, each version of the single dataset representing data received or provided by a different one of the dataflow graphs;

means for providing a user interface to receive a mapping between at least two datasets in a given set; and

means for storing the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

40. A computer-readable medium storing a computer program for mapping data stored in a data storage system, the computer program including instructions for causing a computer to:

process specifications of dataflow graphs that include nodes representing computations interconnected by links representing flows of data, with at least one of the dataflow graphs receiving a flow of data from at least one input dataset and at least one of the dataflow graphs providing a flow of data to at least one output dataset;

identify one or more sets of datasets, where each dataset in a given set matches one or more criteria for identifying different versions of a single dataset, each version of the single dataset representing data received or provided by a different one of the dataflow graphs;

provide a user interface to receive a mapping between at least two datasets in a given set; and

store the mapping received over the user interface in association with a dataflow graph that provides data to or receives data from the datasets of the mapping.

41. The method of claim 1, wherein each version of a single dataset is associated with a different graph, graph subset, or executable component.

42. The method of claim 1, wherein each version of a single dataset is stored in a different location associated with the data storage system.

43. The method of claim 1, wherein each version of a single dataset is interpreted using a different data storage format.

44. The method of claim 1, wherein each version of a single dataset is accessed using a parameter that varies between executions of the dataflow graph.

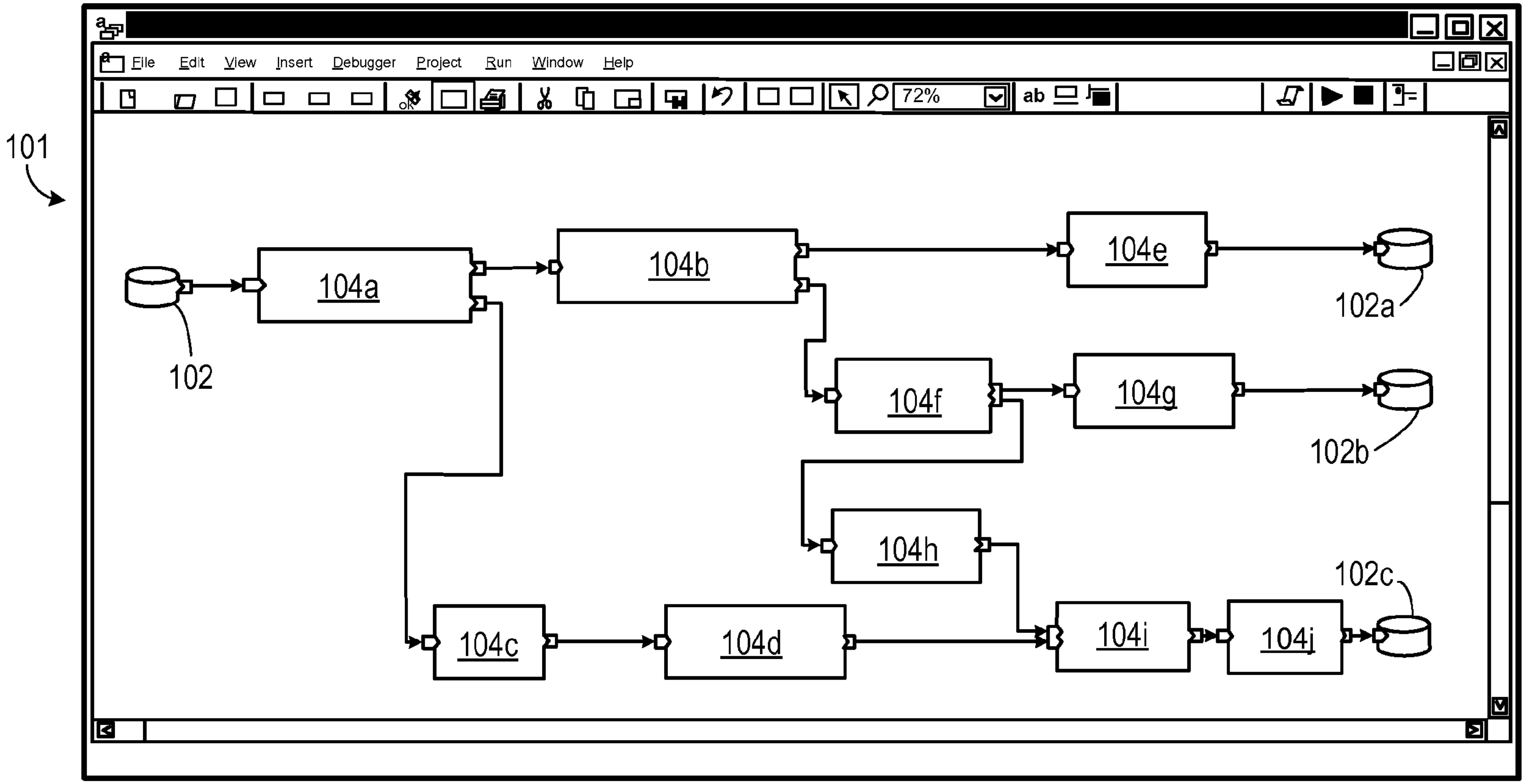
101
File
Edit
View
Insert
Debugger
Project
Run
Window
Help
72%
ab
102
104a
104b
104c
104d
104e
104f
104g
104h
104i
104j
102a
102b
102c

FIG. 1

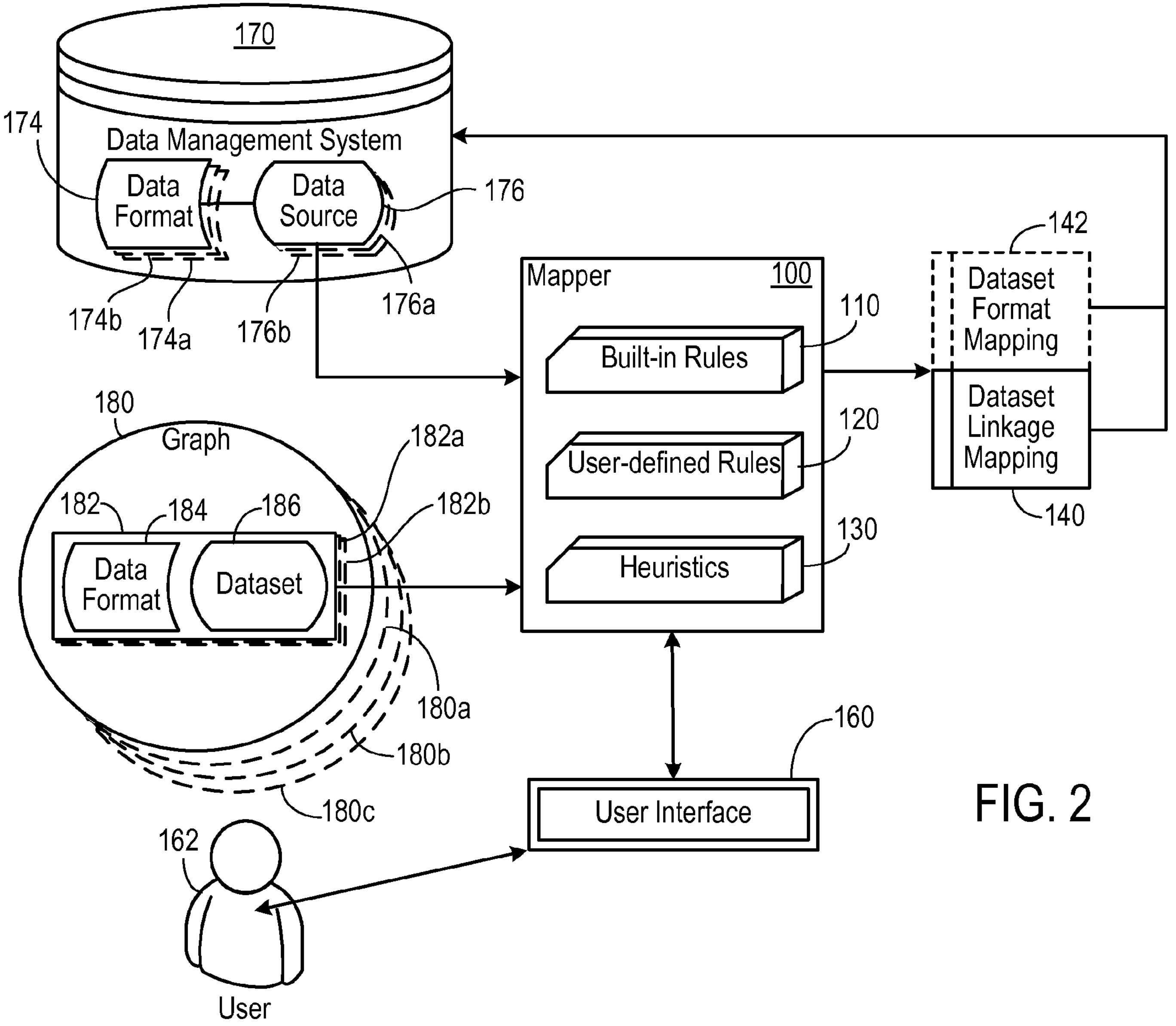
170
174
Data Management System
Data Format
Data Source
176
174b
174a
176b
176a
142
Dataset Format Mapping
Mapper
100
110
Built-in Rules
Dataset Linkage Mapping
120
User-defined Rules
140
130
Heuristics
180
Graph
182a
182
184
186
182b
Data Format
Dataset
180a
180b
180c
160
User Interface
162
User

FIG. 2

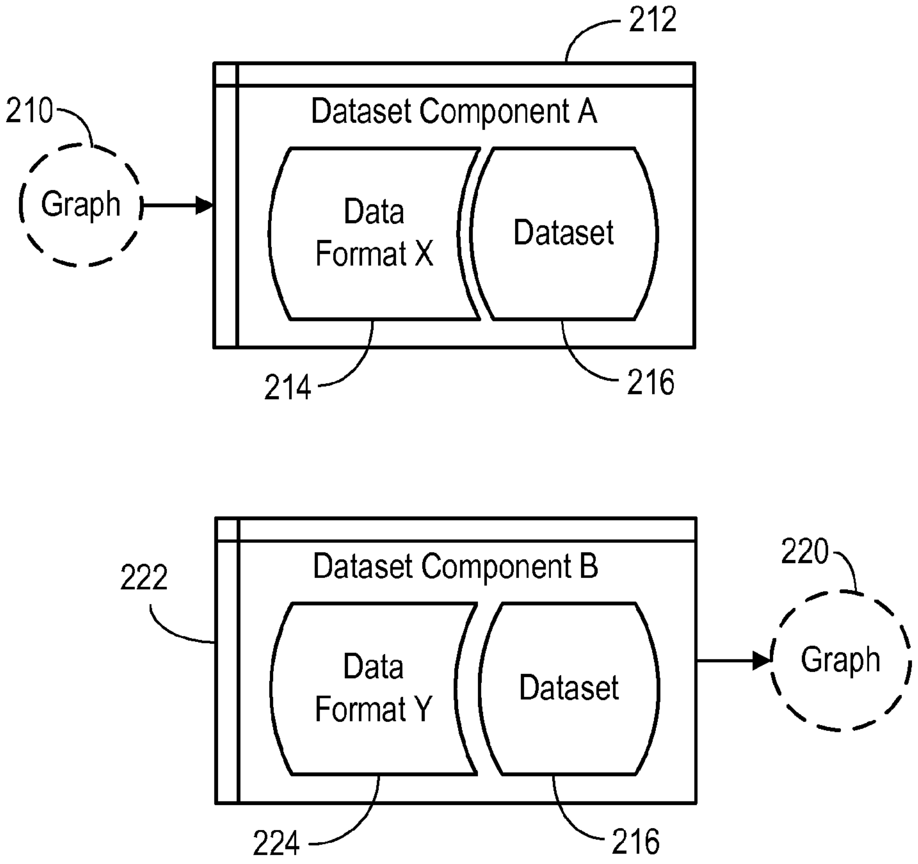
212
210
Graph
Dataset Component A
Data
Format X
Dataset
214
216
Dataset Component B
222
220
Data
Format Y
Dataset
Graph
224
216

FIG. 3A

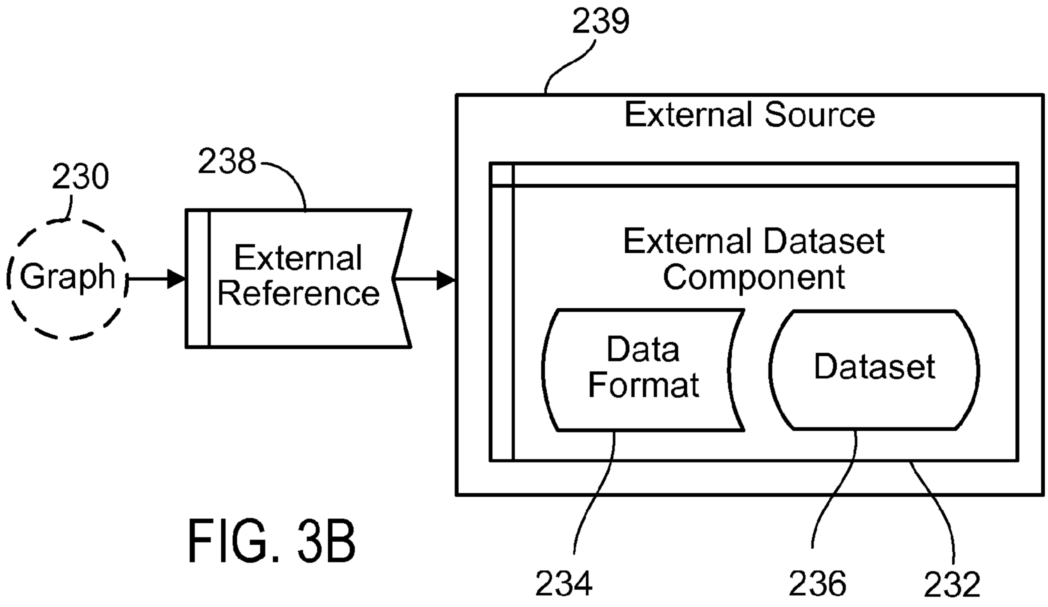
239
External Source
230
Graph
238
External Reference
External Dataset Component
Data Format
Dataset
234
236
232

FIG. 3B

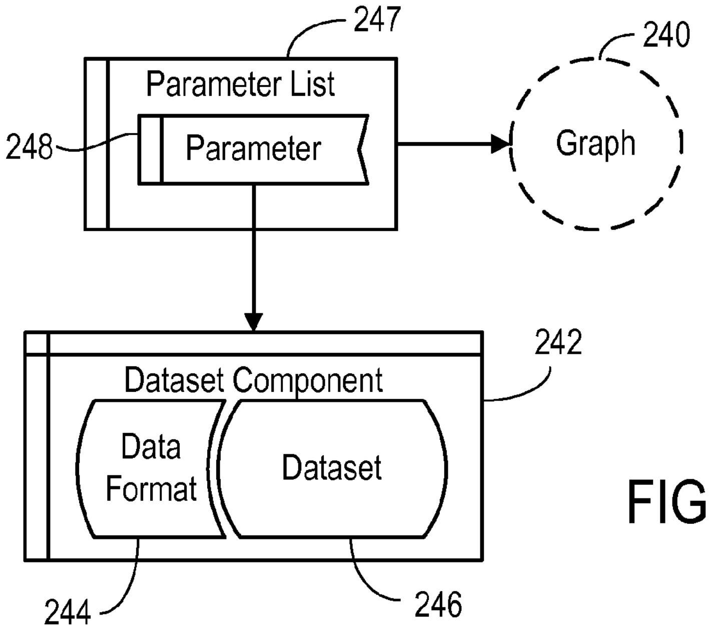
247
Parameter List
248
Parameter
240
Graph
242
Dataset Component
Data Format
Dataset
244
246

FIG. 3C

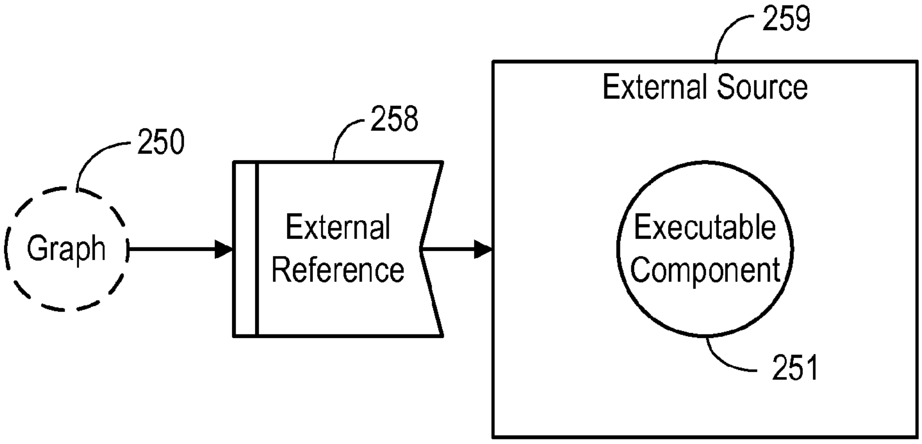
259
External Source
258
250
Graph
External Reference
Executable Component
251

FIG. 3D

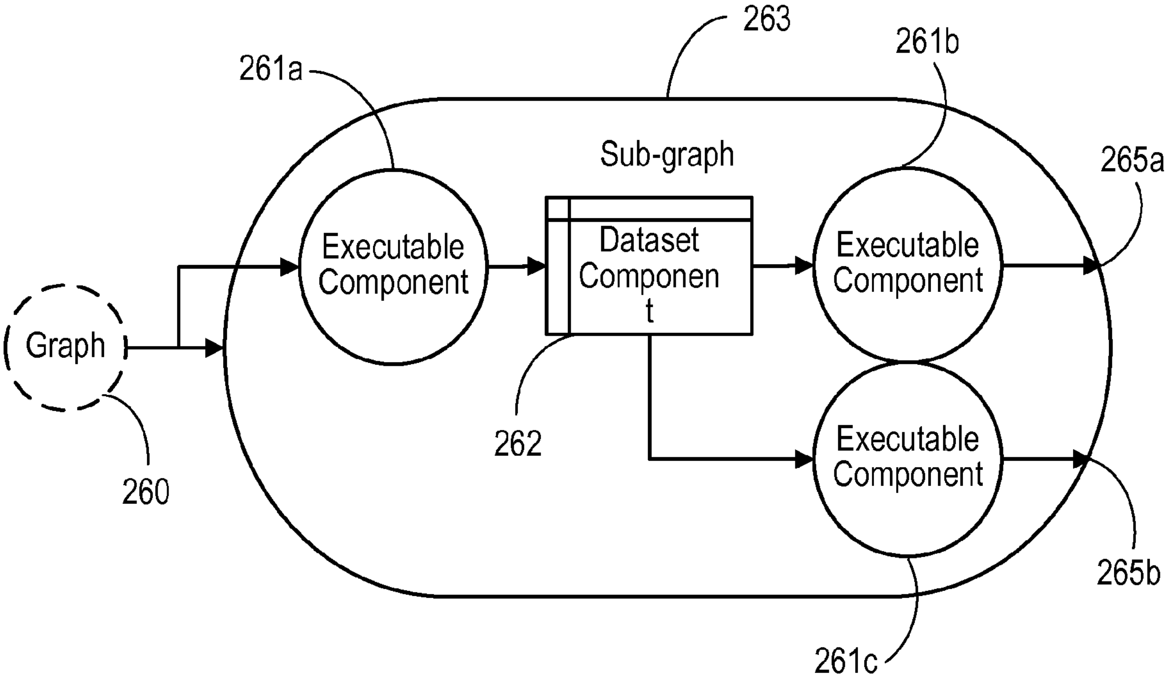
263
261a
261b
Sub-graph
265a
Executable Component
Dataset Componen t
Executable Component
Graph
262
Executable Component
260
265b
261c

FIG. 3E

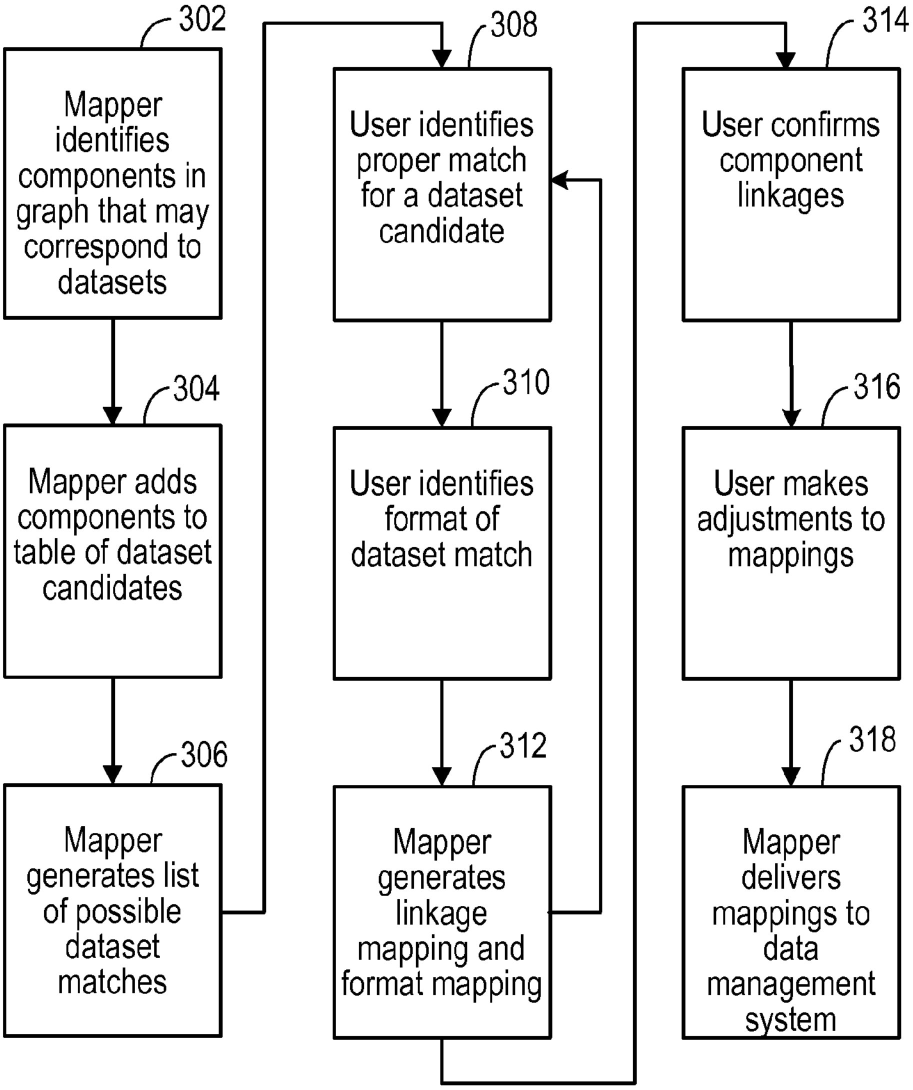
302
Mapper identifies components in graph that may correspond to datasets
304
Mapper adds components to table of dataset candidates
306
Mapper generates list of possible dataset matches
308
User identifies proper match for a dataset candidate
310
User identifies format of dataset match
312
Mapper generates linkage mapping and format mapping
314
User confirms component linkages
316
User makes adjustments to mappings
318
Mapper delivers mappings to data management system

FIG. 4

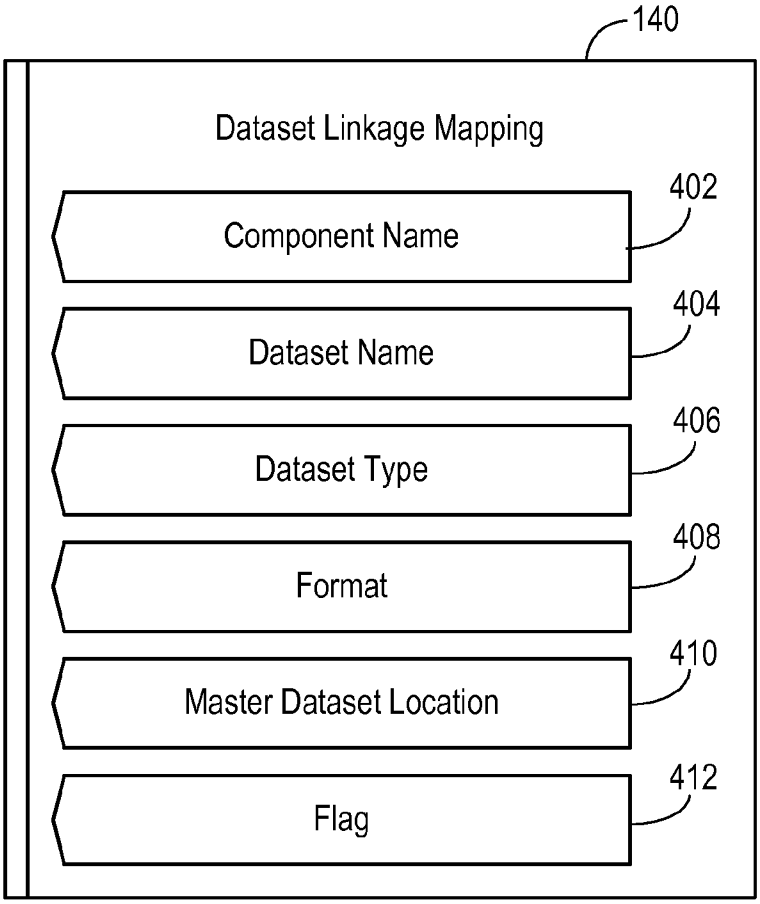
140
Dataset Linkage Mapping
Component Name
402
Dataset Name
404
Dataset Type
406
Format
408
Master Dataset Location
410
Flag
412

FIG. 5

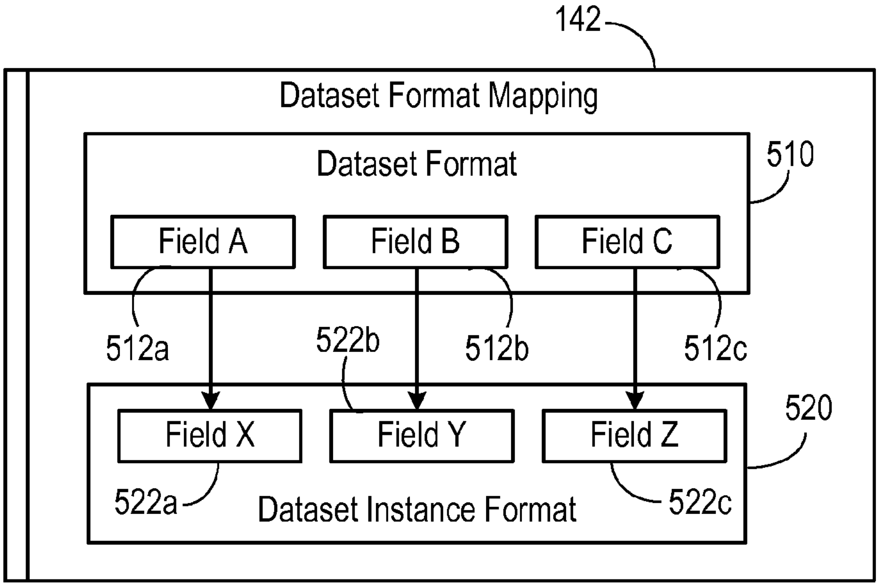
142
Dataset Format Mapping
Dataset Format
510
Field A
Field B
Field C
512a
522b
512b
512c
520
Field X
Field Y
Field Z
522a
Dataset Instance Format
522c

FIG. 6

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US 09/66210

A. CLASSIFICATION OF SUBJECT MATTER
IPC(8) - G06F 7/00 (2010.01)
USPC - 707/100
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC (8) - G06F 7/00 (2010.01)
USPC - 707/100

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
USPC - 706/934; 707/E17.011; 707/17.033; 707/E17.048; 707/E.091 (See Keywords Below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
Pub WEST (USPT, PGPB, JPAB, EPAB), Google Scholar
Search Terms Used: Process, analyze, interpret, graph, tree, data flow, map, link, associate, bind, data set, object, identifier, determine, estimate, find, select, display, output, provide, interface, connect, store, save, record, associate, criteria, qualifier, parameter, filter....

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 7,080,088 B1 (LAU), 18 July 2006 (18.07.2006), entire document, especially Abstract; col 5, ln 52 to col 6, ln 18; col 6, ln 57 to col 7, ln 25; col 8, ln 1-46; col 11, ln 40 to col 12 ln 15; col 13, ln 60 to col 14, ln 21; col 18, ln 43 to col 19, ln 8; col 19, ln 30-62	1-40
Y	US 2007/0011208 A (SMITH), 11 January 2007 (11.01.2007), entire document, especially para[0011]-[0013], [0044]-[0045], [0063]-[0064], [0075]-[0078], [0088]-[0089]	1-40
A	US 2004/0056908 A1 (BJORNSON et al.), 25 March 2004 (25.03.2004), entire document.	1-40
A	US 2005/0187984 A1 (CHEN), 25 August 2005 (25.08.2005), entire document.	1-40
A	US 2001/0014890 A1 (LIU et al.), 16 August 2001 (16.08.2001), entire document.	1-40

□ Further documents are listed in the continuation of Box C. □

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search
14 January 2010 (14.01.2010)

Date of mailing of the international search report
27 JAN 2010

Name and mailing address of the ISA/US
Mail Stop PCT, Attn: ISA/US, Commissioner for Patents
P.O. Box 1450, Alexandria, Virginia 22313-1450
Facsimile No. 571-273-3201

Authorized officer:
Lee W. Young
PCT Helpdesk: 571-272-4300
PCT OSP: 571-272-7774