Disclosed is a reconciliation processing system for reconciling different name values used to refer to the same entity across a set of data sources. The data sources including structured data sources having labelled data values each describing a property of one of the entities, the entity being identifiable by a key value, and the property being identifiable by the label of the data value. The system has a data source categoriser allocates each of the data sources into a first category in which for each key value there is only one name value or a second category in which the data source has key values each having more than one different name value. The system also has a first reconciliation processor that, when there is more than one data source in the first category, uses one of the data sources as an index data source, for each name value finds any name values from the remaining data sources that satisfy a similarity criterion and stores the found name values and the name value from the index data source as aliases for one another in a first results list. A second reconciliation processor for each data source in the second category, for each key value in the data source, identifies any labelled data values which are name values attributing a name to the entity identified by the key value, and if more than one different name value is identified, to store the different name values as aliases for one another in a second results list. Finally there is a results list combiner, that generates a combined results list by appending the second results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list as aliases for one another in the combined results list.

FIGURE 1

```
Data source 20

First reconciliation processor 11
Second reconciliation processor 12
Third reconciliation processor 13

Results list combiner 16

Data source categoriser 14

Reconciliation processing system 10
```
FIGURE 1
FIGURE 2
Scan data in the Data Storage S401

Duplicated Keys? S402

YES

NO → Mapping Value = 1:1 S406

Duplicated Name Values? S403

YES

NO → Mapping Value = 1:1 S407

Name Value is String? S404

YES

NO → Mapping Value = 1:m-u S408

Mapping Value = 1:m-s S405
FIGURE 5

"Fujitsu Ltd. (富士通株式会社, Fujitsu Kabushiki-Kaisha) is a Japanese multinational information technology equipment and services company headquartered in Tokyo, Japan. It is the world's second-largest services provider measured by revenues (after IBM). Fujitsu chiefly makes computing products, but the company and its subsidiaries also offer a diversity of products and services in the areas of personal computing, telecommunications, and advanced microelectronics. It has approximately 172,000 employees and its products and services are available in over 50 countries. Fujitsu is listed on the Tokyo Stock Exchange and is a constituent of the Nikkei 225 and TOPIX indices."

FIGURE 6

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>My Company Ltd.</td>
</tr>
<tr>
<td>120</td>
<td>Your Company Corp.</td>
</tr>
<tr>
<td>130</td>
<td></td>
</tr>
</tbody>
</table>

Example of 1:1 mapping

<table>
<thead>
<tr>
<th>Key</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>110</td>
<td>My Company Ltd.</td>
</tr>
<tr>
<td>110</td>
<td>My Company Limited</td>
</tr>
<tr>
<td>120</td>
<td>Your Company Corp.</td>
</tr>
</tbody>
</table>

Example of 1:m-s mapping

Example of 1:m-u mapping

Fujitsu Ltd. (富士通株式会社, Fujitsu Kabushiki-Kaisha) is a Japanese multinational information technology equipment and services company headquartered in Tokyo, Japan. It is the world's third-largest IT services provider measured by revenues (after IBM and HP).

FIGURE 7
### Example of the Reconciliation Dispatcher Registry

<table>
<thead>
<tr>
<th>Data Source</th>
<th>Mapping Value</th>
<th>Data Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>file://data/file1.csv</td>
<td>1:1</td>
<td>CSV</td>
</tr>
<tr>
<td>file://data/file2.csv</td>
<td>1:1</td>
<td>CSV/structured</td>
</tr>
<tr>
<td><a href="http://data/file5.csv">http://data/file5.csv</a></td>
<td>1:m-u</td>
<td>CSV/text</td>
</tr>
<tr>
<td>file://data/file3.ttl</td>
<td>1:m-s</td>
<td>turtle</td>
</tr>
<tr>
<td>file://data/file4.xml</td>
<td>1:m-s</td>
<td>XML</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**FIGURE 8**

```
<data source 1> reconcile <data source 2> => result 1
<data source 1> reconcile <data source 3> => result 2
<data source 1> reconcile <data source 4> => result 3
<data source 1> reconcile <data source 5> => result 4
...
```

Append result 2, result 3, and result 4 to result 1 once each of them finished processing.

**FIGURE 9**
Fujitsu Ltd. (富士通株式会社 Fujitsu Kabushiki-Kaisha) is a Japanese multinational information technology equipment and services company headquartered in Tokyo, Japan. It is the world's third-largest IT services provider measured by revenues (after IBM and HP).

Pattern = "\(^{(.*)(.*))\) is a \(.*\) company/corporation"

Fujitsu Ltd. (富士通株式会社 Fujitsu Kabushiki-Kaisha)
Reconciliation Processing System, Method, and Program

This invention is in the field of big data and open data analysis, and in particular relates to processing techniques for reconciling heterogeneous naming conventions across a plurality of data sources.

Entity Identifiers are crucial in order to form connections between data. This has become more and more important especially with the recent Open Data initiatives, which are encouraged by many governments in countries including U.S., U.K, and Japan. However, entity identification is subject to local variations.

Inherited from Big Data’s 3Vs nature (high volume, high velocity, high variety), the Open Data present similar challenges, especially in variety, e.g., the entity identifiers can appear in structured, semi-structured, or even unstructured data sources. This presents challenges in recognising different identifiers for the same entity.

To reconcile entity identifiers from heterogeneous data sources, a very common method is to use entity name as the mediator. However, the ambiguity of the names can severely affect the accuracy of the reconciliation result. This is because the names are composed in strings, which, when affected by social factors, a definitive or legally registered entity name could have multiple versions. This issue is exacerbated when data is multilingual.

Natural language processing is a well-established technology that can be applied to solve text based analysis and processing. However, the variety of tasks and techniques offered by NLP are all isolated, and narrow-focused to solve specific problems.

Embodiments include a reconciliation processing system for reconciling different name values used to refer to the same entity across a plurality of data sources, the plurality of data sources including structured data sources being composed of labelled data values each describing a property of one of a plurality of entities, the entity being identifiable by a key value, and the property being identifiable by the label of the data value; the
system comprising: a data source categoriser, being configured to identify, for each of the plurality of key values, one or more name values, a name value being a labelled data value for which the label indicates that the property attributes a name to the entity identified by the key value, and to allocate each of the structured data sources into one of two categories: a first category in which for each key value in the structured data source, there is only one name value; and a second category in which the structured data source includes one or more key values each having more than one different name value; a first reconciliation processor being configured to, in a case in which there is more than one structured data source in the first category, using one of the structured data sources in the first category as an index structured data source, for each name value in the index structured data source, find any name values from the remaining structured data sources that satisfy a first similarity criterion when compared to the name value from the index structured data source, and to store the found name values and the name value from the index structured data source as aliases for one another in a first results list; a second reconciliation processor being configured to, for each structured data source in the second category, for each key value in the structured data source, identify any labelled data values which are name values attributing a name to the entity identified by the key value, and if more than one different name value is identified, to store the more than one different name values as aliases for one another in a second results list; and a results list combiner, being configured to generate a combined results list by appending the second results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list as aliases for one another in the combined results list.

Advantageously, embodiments provide a comprehensive reconciliation processing system that can effectively reconcile different names used for the same entity in data sources having different arrangements. The same entity can be attributed more than one different name value across multiple data sources, or can be attributed different name values within the same data source. In order to gain a complete view of data stored about a particular entity, it is therefore desirable to obtain a list of different name values that may be attributed to the entity. Differences may exist due to the data
sources having different native languages, and due to variations in formality of language used, among other reasons.

Embodiments provide a reconciliation processing system that is configured to divide a plurality of data sources into groups in dependence upon characteristics of the structure of data within the data source, and to perform reconciliation processing on each group with a different reconciliation processor, each reconciliation processing performing a different reconciliation process to identify aliases for name values used to name entities described in the data sources in the respective group. The system further comprises a results combiner configured to identify when aliases identified by one reconciliation processor are aliases for name values identified by another reconciliation processor, and to combine the results (the lists of aliases output by each reconciliation processor) accordingly.

A data source is a file (such as a csv file, excel file) or file repository (such as multiple csv files), such as a database, table (such as an RDBMS table), data graph, website, or document storage facility. Data sources store data describing, or representing knowledge about, real world entities. The thing that is being described by data in a data source is referred to as an entity. A data source is identified by its unique location, which is defined by an address. A key value identifies, or uniquely identifies, an entity within a data source. A name value is a value of a property (in terms of a table that is a column entry) within the data source.

Data within data sources are arranged in different ways, and embodiments utilise different reconciliation processors to recognise when different name values are aliases (refer to the same entity) in dependence upon a characterisation of the arrangement of data within the data source. A results list combiner then performs a process for recognising when mutual aliases from one of the reconciliation processors are aliases of mutual aliases from another of the reconciliation processors, and combines the aliases accordingly. In this way, heterogeneity between data sources is utilised as a means to reveal more aliases for a particular entity.
Structured data sources in the context of embodiments is taken to mean those whose data values are labelled, tagged, or otherwise semantically enriched so that the meaning or semantic significance of the data value is identifiable. For example, such identification may be the data source categoriser having a predetermined list of words or substrings or data labels that identify the property described by the data value as a name value.

A key value is taken to be the value of the column, property, address, or other form of data that identifies an entity among the entities described in the data source. A key value is a value of a key, which in certain data sources may be a primary key.

The data source categoriser assesses how key values are mapped to name values within each data source, and uses that assessment to allocate each structured data source into either the first category or the second category. A name may be, for example, a legal title, and/or a linguistic identifier used to refer to the entity.

Different reconciliation processors perform reconciliation processing on each of the first category and the second category. This enables the reconciliation processing to be tailored to the characterisation of data arrangement in the individual data sources, and also allows for some parallelisation.

The first reconciliation processor is configured to cross reconcile between an index structured data source and each of the other structured data sources in the first category. The cross reconciliation is a string comparison between the each name value in the index structured data source and each name value in each of the other structured data sources. Any string comparison yielding a result exceeding a predetermined threshold (and not being identical to the name value from the index structured data source) is recorded as an alias of the name value from the index structured data source in a first results list.

The second reconciliation processor is configured to, for each data source in the second category, extract the different name values attributed to the entity referenced by the same key value, and if there are more than one different name value is
extracted, to add the different name values as aliases for one another in a second results list.

The results list combiner appends the first and second results list to each other. This list is an intermediate results list. Further processing of the intermediate results list recognises when aliases for name values appearing in the first results list also appear in the second results list. The results list combiner identifies name values that appear in both the first results list and second results list, and records the aliases for the name value that appear in the first results list and the aliases for the name value that appear in the second results list as aliases for the name value in the combined results list.

The output of the results list combiner is a list of aliases, where aliases exist in the plurality of data sources, for entities named in the index structured data source in the first category and for entities named more than once in the second category.

The utility of embodiments can be extended to include a routine for extracting name values from unstructured data sources. For example: if the plurality of data sources includes one or more unstructured data sources, each unstructured data source comprising a plurality of passages of text each describing an entity identifiable by a key value stored as part of or associated with the passage of text, then the data source categoriser is configured to identify the unstructured data sources from among the plurality of data sources and to allocate the unstructured data sources into a third category. Such a system further comprises a third reconciliation processor, being configured to use pattern matching to identify, from within each of the passages of text, substrings arranged in a pattern indicating that the substring includes a name value attributing a name to the entity identifiable by the key value stored as part of or associated with the passage of text, to extract the indicated name value from the identified substrings, and when more than one different name value is extracted from a single passage of text, to store the more than one different name values as aliases for one another in a third results list; wherein the results list combiner is configured to generate a combined results list by appending the second results list and the third results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list and/or the third results
list, storing the aliases for the name value from the first results list and the aliases for
the identical name value from the second results list and/or the third results list as
aliases for one another in the combined results list.

5 Embodiments utilise natural language processing techniques, but in a system which
facilitates a comprehensive, rather than data-source or data-arrangement specific,
analysis and extraction of name values from plural data sources.

A data source may be allocated to either the first or second category and the third
category. For example, a structured data source may include unstructured data and
thus it may be desirable to subject such a data source to processing by multiple
reconciliation processors in order to identify more aliases than could be identified by a
single reconciliation processor.

15 The third reconciliation processor is configured to analyse unstructured data fields or
passages of text from the unstructured data source in order to extract candidate name
values for the entity that is the subject of the data field or passage of text. A passage
of text or unstructured data field may be identifiable by the data source categoriser by
the number of text characters contained therein. For example, 100 characters or
above may be characterised as unstructured.

The results list combiner identifies name values that appear in both the first results list
and third results list, and records the aliases for the name value that appear in the first
results list and the aliases for the name value that appear in the second results list as
aliases for the name value in the combined results list.

As a particular example of a third reconciliation processor: the third reconciliation
processor is configured to use pattern matching by using each of a predetermined set
of regular expressions, processed into nondeterministic finite automaton, to identify
substrings that match the regular expression in the passage of text, the presence of the
regular expression in the passage of text being an indication that a name value is
present at a particular position in the passage of text relative to the regular expression.
The predetermined set of regular expressions may be input for a plurality of languages to enhance the utility of the third reconciliation processor. Regular expression identification is supported in programming languages such as Java, Python, and C++, so the third reconciliation processor may be coded using one of these languages. The regular expression is an expression that includes one or more name values in conjunction with "regular" words, phrases, or substrings. Therefore, the presence of the "regular" words, phrases, or substrings is an indication that a name value can be extracted from, for example, the word or other section of text proceeding or preceding the regular words, phrases, or substrings, or the substring between one regular word, phrase, or substring and another such word, phrase or substring.

As a particular example of the functionality of the first reconciliation processor: the first reconciliation processor is configured to, in a case in which there is more than one structured data source in the first category, using one of the structured data sources in the first category as an index structured data source, for each name value in the index structured data source, to compare the name value with each of the name values from each of the remaining structured data sources in the first category as target structured data sources, and, for any comparison satisfying a first similarity criterion, to add the name value from the target structured data source to a list of aliases for the name value from the index structured data source.

The list of aliases for the name value from the index structured data source is an entry in the first results list, and is the output of the first reconciliation processor.

The first reconciliation processor may be configured to perform pre-processing to format the name values into a form suitable for comparison, the particular form being dependent upon the comparison process being used. An exemplary comparison process is an algorithm to compare the distance between two strings. An example of such an algorithm is the Jaro-Winkler distance. Thus, the pre-processing may be to ensure that all name values are stored as strings at a location accessible to the first reconciliation processor, or the conversion to string (where required) may be performed on the fly as name values are read from data sources for comparison.
The output of the comparison between two name values performed by the first reconciliation processor may be a numerical value representing the distance between the two strings (the strings being the form of name value required for comparison). In a particular example: the first similarity criterion is satisfied if a string similarity comparison of two name values generates a result exceeding a predetermined threshold value.

As a further example, the similarity criterion may only be satisfied by two name values that are non-matching and for which the string similarity comparison generates a result exceeding the predetermined threshold value. The predetermined threshold value may be configurable at the implementation of the system or at runtime for each overall reconciliation process. In more complex examples, the comparison of two name values may involve using more than one different string comparison algorithm to compare the two name values, with the similarity criterion being a predetermined threshold aggregate value.

As an extension of the functionality of the second reconciliation processor:

6. A reconciliation processing system according to any of the preceding claims, wherein the second reconciliation processor is further configured to compare the labels of data values attributed to the entity identified by the key value with a predefined list of labels indicating equivalence between entities, and to extract the data values labelled with labels included in the predefined list as aliases for the name value of the entity in the second results list.

Advantageously, this functionality enhances the ability of the second reconciliation processor to identify aliases for a particular name value, by using properties other than names to recognise when different key values are used to refer to indicate equivalence. This leverages other labelled data values describing properties of the entity that may reveal that the entities being described in each case are equivalent. Some knowledge of the data sources and their syntax is required in forming the predefined list of labels. This functionality provides a mechanism for that knowledge of the data source syntax
to generate content-specific results. An example of such a label is the "dbpedia-owl:wikiPageRedirects" property.

Advantageously, the system can reconcile data from data sources storing data in various forms. In particular: the plurality of data sources includes data sources storing relational data and data sources storing graph data.

Both are examples of structured data sources. In the relational data, labelled data values are arranged in tables, with rows being identified by key values, and labels being the column headers. In the case of graph data, which may be encoded as triples, the key values are the subjects of triples, which may be given as URLs, and the labelled data values are the objects of triples, with the label being provided by the predicate of the triple.

In a particular implementation: the system further comprises a data extractor configured to provide data from the plurality of data sources to the reconciliation processors for reconciliation processing.

Advantageously, a dedicated data extractor may reduce bottlenecks associated with accessing data sources at system runtime. The data extractor may write data extracted from the plurality of data sources to a storage location local to the system, and in particular, accessible to the reconciliation processors. The data extractor may be configured to filter data, so that only particular data values are extracted from the data sources.

The system may further comprise: a combined results propagator configured, for at least one of the plurality of data sources, for each name value in the data source that matches a name value in the combined results list, adding each of the aliases for the name value stored in the combined results list to the data source with an indication that the alias is an alternative to the name value.

Advantageously, such a results propagator enhances the content of the data sources to which aliases are added. As an additional or alternative functionality, and in
embodiments in which each alias appearing in the results lists is stored along with information identifying the data source in which it appears (such as an address), the combined results propagator may be configured to add, to descriptions of entities in a data source that have a name value appearing in the combined results list, a link or some other form of content access means to the entry or entries in the other data sources that have name values that are aliases.

The selection of the index structured data source for the first reconciliation processor may be automated, for example, by selecting the structured data source with the most name values. Alternatively: the first reconciliation processor may be further configured, in the case in which there is more than one structured data source in the first category, to prompt a user for a selection of an index structured data source from among the more than one structured data source in the first category, and to use the selection as the index structured data source.

A user in this case may be a human user, or may be an application accessing the system.

Embodiments of another aspect include a reconciliation processing method for reconciling different name values used to refer to the same entity across a plurality of data sources, the plurality of data sources including structured data sources being composed of labelled data values each describing a property of one of a plurality of entities, the entity being identifiable by a key value, and the property being identifiable by the label of the data value; the method comprising: a categorising step, comprising identifying, for each of the plurality of key values, one or more name values, a name value being a labelled data value for which the label indicates that the property attributes a name to the entity identified by the key value, and allocating each of the structured data sources into one of two categories: a first category in which for each key value in the structured data source, there is only one name value; and a second category in which the structured data source includes one or more key values each having more than one different name value. The method further comprises: a first reconciliation process comprising, in a case in which there is more than one structured data source in the first category, using one of the structured data sources in the first
category as an index structured data source, for each name value in the index structured data source, finding any name values from the remaining structured data sources that satisfy a first similarity criterion when compared to the name value from the index structured data source, and storing the found name values and the name value from the index structured data source as aliases for one another in a first results list; a second reconciliation process comprising, for each structured data source in the second category, for each key value in the structured data source, identifying any labelled data values which are name values attributing a name to the entity identified by the key value, and if more than one different name value is identified, storing the more than one different name values as aliases for one another in a second results list; and a combining step, comprising generating a combined results list by appending the second results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list as aliases for one another in the combined results list.

Optionally, the plurality of data sources includes one or more unstructured data sources, each unstructured data source comprising a plurality of passages of text each describing an entity identifiable by a key value stored as part of or associated with the passage of text; the categorising step in that case also comprises identifying the unstructured data sources from among the plurality of data sources and to allocate the unstructured data sources into a third category; and the method further comprises: a third reconciliation process comprising using pattern matching to identify, from within each of the passages of text, substrings arranged in a pattern indicating that the substring includes a name value attributing a name to the entity identifiable by the key value stored as part of or associated with the passage of text, extracting the indicated name value from the identified substrings, and when more than one different name value is extracted from a single passage of text, storing the more than one different name values as aliases for one another in a third results list; wherein the combining step comprising generating a combined results list by appending the second results list and the third results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list and/or
the third results list, storing the aliases for the name value from the first results list and
the aliases for the identical name value from the second results list and/or the third
results list as aliases for one another in the combined results list.

Embodiments of another aspect include a computer program which, when executed by
a computing apparatus, causes the computing apparatus to function as a computing
apparatus defined above as an invention embodiment.

Embodiments of another aspect include a computer program which, when executed by
a computing apparatus, causes the computing apparatus to perform a method defined
above or elsewhere in this document as an invention embodiment.

Furthermore, embodiments of the present invention include a computer program or
suite of computer programs, which, when executed by a plurality of interconnected
computing devices, cause the plurality of interconnected computing devices to perform
a method embodying the present invention.

Embodiments of the present invention also include a computer program or suite of
computer programs, which, when executed by a plurality of interconnected computing
devices, cause the plurality of interconnected computing devices to function as a
computing apparatus defined above or elsewhere in this document as an invention
embodiment.

Although the aspects (software/methods/apparatuses) are discussed separately, it
should be understood that features and consequences thereof discussed in relation to
one aspect are equally applicable to the other aspects. Therefore, where a method
feature is discussed, it is taken for granted that the apparatus embodiments include a
unit or apparatus configured to perform that feature or provide appropriate functionality,
and that programs are configured to cause a computing apparatus on which they are
being executed to perform said method feature.
In any of the above aspects, the various features may be implemented in hardware, or as software modules running on one or more processors. Features of one aspect may be applied to any of the other aspects.

The invention also provides a computer program or a computer program product for carrying out any of the methods described herein, and a computer readable medium having stored thereon a program for carrying out any of the methods described herein. A computer program embodying the invention may be stored on a computer-readable medium, or it could, for example, be in the form of a signal such as a downloadable data signal provided from an Internet website, or it could be in any other form.

Preferred features of the present invention will now be described, purely by way of example, with reference to the accompanying drawings, in which:

Figure 1 illustrates a reconciliation processing system of an embodiment;
Figure 2 illustrates a reconciliation processing system of another embodiment;
Figure 3 illustrates a reconciliation processing system of another embodiment annotated with method steps;
Figure 4 illustrates a logic process followed by a data source categoriser of an embodiment;
Figure 5 illustrates data from a structured data source;
Figure 6 illustrates data from an unstructured data source;
Figure 7 illustrates further exemplary data;
Figure 8 illustrates an exemplary form of data maintained by the categoriser registry of an embodiment;
Figure 9 illustrates an overview of a reconciliation procedure performed by a first reconciliation processor of an embodiment;
Figures 10A to 10C illustrate RDF statements for processing by a second reconciliation processor of an embodiment;
Figures 11A to 11C illustrate a passage of text from an unstructured data source, a regular expression used to process the text by a third reconciliation processor of an embodiment, and a result of said processing; and
Figure 12 illustrates a combination process performed by a results list combiner of an embodiment.

Figure 1 illustrates a reconciliation processing system 10. The reconciliation processing system 10 comprises a data source categoriser 14, a first reconciliation processor 11, a second reconciliation processor 12, a third reconciliation processor 13, and a results list combiner 16. The plurality of data sources 20 are illustrated as being external to the system. However, data from the plurality of data sources 20 may be extracted and stored locally (i.e. within the system) by a data extractor 18. The plurality of data sources 20 are accessible to the system 10. The third reconciliation processor 13 is illustrated in Figure 1, however embodiments may include the first and second reconciliation processors without the third.

The components of the system 10 illustrated in Figure 1 are functional components, each representing a distinct function performed by the system 10. In terms of hardware components, the system 10 may be considered to be a computing device comprising a data storage apparatus, a processor, and a memory. However, the system 10 may also be realised by a plurality of such devices operating in cooperation with one another. Each functional component may be realised by a distinct computing device. Alternatively, the system 10 may be a program running as a virtual server and configured to employ physical hardware resources as and when they are required in carrying out the functions of the functional components. For example, when a reconciliation process is initiated by a user or otherwise, and a plurality of data sources 20 identified to be the subject of said reconciliation, the system 10 may request hardware resources from a management server.

Lines connecting components in Figure 1 represent flows of data/information, but are by no means the only exchanges of information between components. For example, the reconciliation processors are each configured to access the data sources 20 themselves in order to extract data values for comparison, or to access a local data storage unit 17 that has been populated by a data extractor 18. The link from the data source categorises to each of the reconciliation processors illustrates the notification by
the categoriser 14 of the membership of the respective categories of data source to the processors.

The plurality of data sources 20 includes structured data sources 20 being composed of labelled data values each describing a property of one of a plurality of entities, the entity being identifiable by a key value, and the property being identifiable by the label of the data value. The plurality of data sources 20 may further include one or more unstructured data sources, each unstructured data source comprising a plurality of passages of text each describing an entity identifiable by a key value stored as part of or associated with the passage of text. The data sources 20 may be open data sources or may be closed/private data sources which the system 10 is authorised to access. The data sources 20 may be internet sites, such as Wikipedia or DBpedia, may be databases including open databases (such as are maintained in the UK by the Land Registry service or by Royal Mail), and may be document repositories. In terms of file formats, data sources 20 may be in the form of text, csv files, Microsoft excel files, RDF stores, database tables, and in any other format in which entities are identified by key values.

The data source categoriser 14 is configured to identify, among data values describing a property of an entity identified by a particular key value, the name values. Such identification is based on a label (or other form of tag or semantic enrichment mechanism) attributed to the data value. The data sources 20 are categorised into the first or second category in dependence upon whether the data source includes more than one different name value for each key value. If there is more than one different name value for any key value, then the data source is allocated to the second category. Otherwise it is included in the first category. Furthermore, the data source categoriser 14 is configured to add data sources that include unstructured data to the third category, in order that the unstructured data may be analysed to determine whether any potential name values are included. Passages of text exceeding a certain length, for example 100 characters, may qualify as unstructured data, depending on the implementation. Otherwise, unstructured data may be strings or text fields which exceed a particular length and are either not labelled or are labelled only once for the whole string or text field.
The first reconciliation processor 11 cross-reconciles name values across data source boundaries. An index data source defines the name values for which aliases are sought in the remaining data sources. The first reconciliation processor 11 reconciles the index data source with each of the remaining data sources in the first category by performing a string comparison of the name values. Different algorithms exist for comparing string values to identify similarities, and a system designer may select one or more to be performed by the first reconciliation processor 11 in comparing name values. A predetermined threshold value may be used to determine whether or not the result of the string comparison is deemed to be sufficiently similar for the compared strings to be considered aliases. The value of the threshold is configurable, for example by a system designer or at runtime as an argument when instructing the system 10.

The second reconciliation processor 12 performs reconciliation within data source boundaries, so each data source allocated to the second category is internally reconciled to extract the different name values used for the same entity within that data source. In a basic example, this may be by simply identifying the labels (such as headers, predicate values, or other indicators of semantic significance of data values) of data values describing properties of the entity referenced by a particular key value that indicate that the data value is a name value and extracting the name value. However, semantic referencing and other techniques may be used to identify, based on a number of labelled data values other than name values that have similar or the same data values and similar or the same labels for entities referenced by different key values. More than a threshold similarity (such as a minimum number/proportion of matching/similar label/data value pairs, or a weighted metric based on semantic significance of the labels/data values (i.e. data values that appear less frequently in the data source are more semantically significant) may be taken as an indication that the different key values actually refer to the same entity.

The third reconciliation processor 13 is configured to use pattern matching by using each of a predetermined set of regular expressions, processed into nondeterministic finite automaton, to identify substrings that match the regular expression in the
passage of text, the presence of the regular expression in the passage of text being an
indication that a name value is present at a particular position in the passage of text
relative to the regular expression. The third reconciliation processor 13 is configured to
use language processing techniques to extract substrings which, based on their
position within an expression or pattern of text, are candidates for name values. When
two or more such substrings are extracted from the same passage of text (that is, the
passage of text referenced by a single key value), the extracted substrings are stored as
aliases for one another in a third results list. The third reconciliation processor 13 uses
structured queries to extract unknown (at the time of query design) substrings
positioned between known/specified characters/substrings, or positioned at either side
of known/specified characters/substrings.

Any of the first results list, second results list, or third results list, may be presented to
a human user via an interface to allow for false positives to be removed from the
results lists prior to combination by the results list combiner 16. Alternatively, it may be
that only the combined results list is presented to the human user to allow for false
positives to be removed from lists of aliases.

The first processing unit, second processing unit, and third processing unit are input a
notification of the data sources belonging to the respective categories by the data
source categorised. They are each then configured to obtain data from the data
sources in their respective categories, either directly from the data sources, via a data
extractor 18, or from a locally stored version of the data sources. The obtained data is
then reconciled using the reconciliation process of the respective reconciliation
processor. The output from each is a table or list, for example a comma separated list,
of name values that are considered by the reconciliation processor in question to be
aliases for one another. A name value in a list may be accompanied by an identifier of
the data source from which it originates and/or the key value identifying the entity to
which the name value attributes a name.

The results list combiner 16 is configured to receive the results list generated by each
of the reconciliation processors and combine them. The combination may be simply
appending the individual results lists to one another. The combination may further
include, for any name value included in the first results list which also appears in the second and/or third results list, storing the aliases for that name value in each list that it appears as aliases for one another in the combined results list.

Figure 2 illustrates an architecture of a reconciliation processing system 10. Components having the same reference numeral as a component in Figure 1 are assumed to have the features and functionality of their counterpart in the explanation of Figure 1 above.

Figure 3 illustrates the architecture of Figure 2 annotated with some method steps, which will be explained in relation to the architecture of Figure 2.

The system 10 of Figure 2 comprises a first reconciliation processor 11, a second reconciliation processor 12, and a third reconciliation processor 13. These three processors may be collectively referred to as the reconciliator. The system 10 further comprises a results list combiner 16, a data source categoriser 14, a registry 141, a data storage unit 17, and a data extractor 18. A plurality of data sources 20 are illustrated as external to the system 10.

The first reconciliation processor 11 may also be referred to as a 1:1 processor. The second reconciliation processor 12 may also be referred to as a 1:m-s processor. The third reconciliation processor 13 may also be referred to as a 1:m-u processor. The results list combiner 16 may also be referred to as a final combiner or merely a combiner. The data source categoriser 14 may also be referred to as a categoriser 14 or as a reconciliation dispatcher 14.

The data sources 20 can be any Open Data source, e.g. Linking Open Data (http://lod-cloud.net/), DBPedia (dbpedia.org), etc., or data that is publicly available on the Internet. The data sources can also be private (that is, restricted access) databases which the system 10 is authorised to access. The data sources may be selected by a user at system runtime, for example as arguments to an instruction provided to the system.
The data extractor 18 is configured to retrieve relevant information from Open Data sources. This extraction is represented by step S301 in Figure 3. That is, information for analysis/processing by the reconciliator. Due to the varied nature of the data sources, functionalities provided by the data extractor 18 may include file downloading, web crawling, and structured/semi-structured/unstructured data parsing and extraction.

For data sources that provide structured/semi-structured data downloading, the downloaded data is saved to the data storage unit 17 in the system 10, optionally in its original format. The step of saving data or file systems extracted by the data extractor 18 to the data storage unit 17 is represented by step S302 in Figure 3. Data sources providing such data are categorised in the first or second category by the categoriser 14 and as such the data is for processing by the first or second reconciliation processor 12. The data storage unit 17 in the system 10 is readily accessible by the reconciliator and may also be configured to enable parallel read accesses. For unstructured data, e.g. a passage of text such as an “Abstract” or “Description” crawled from DBPedia or WikiPedia, some pre-processes need to be carried out, which uses the text’s web URL (which is unique to each web resource) as its key value, and the block of text as the description of the entity, to generate a CSV file. All the same type of information but describing other entities can be appended to the same file.

An exemplary use case of the data extractor 18 will now be described. An example of the block text is the abstract provided by the DBPedia. The following url present information about the BBC on DBPedia in RDF format: http://dbpedia.org/page/BBC, there is a property called: dbpedia-owl:abstract, and the object (in the sense of a triple of the form s-p-o) is a block of text which contains the company name information: British Broadcasting Corporation. However, DBPedia does not offer a download link to download the above link into a CSV file, hence a SPARQL query needs to be issued against their open endpoint: http://dbpedia.org/sparql. If we submit some query like “select ?s ?abstract where {?s a Company . ?s dbpedia-owl:abstract ?abstract}”, the query result will be a CSV file with all the company urls and their abstract that satisfy the query. “appended” refers the case when multiple query results returned with the same query, e.g. sometimes dbpedia only return 10000 record at a time, so in the sparql query we need to say limit 10000 offset 10000.
The data extractor 18 is provided as an alternative to having the reconciliator access the data sources directly itself. The data extractor 18 removes a potential performance bottleneck for the reconciliator by placing all of the data that the reconciliator needs to access in a data storage unit 17.

The data storage unit 17 may be a single data storage unit 17, or may be a data storage apparatus comprising multiple units each storing a portion or subset of the information/data retrieved by the data extractor 18.

The data storage unit 17 is configured to store data/information that are downloaded or extracted from the data sources. Although the reconciliator can extract data at run-time, in some implementations, for performance reasons, it may be desirable to store at least some of the data in a local network, thus avoiding the network access overhead.

The data source categoriser 14 may also be referred to as a reconciliation dispatcher because, once a data source has been categorised, the relevant reconciliation processor can be dispatched to begin reconciliation processing of the data source. The exception to this statement is the case of the first reconciliation processor 11, which cannot perform any reconciliation processing until an index data source has been selected and at least one other data source has been allocated to the first category. However, once these pre-requisites have been fulfilled, an addition of a new data source to the first category can be considered to be a dispatch of the first reconciliation processor 11.

The process performed by the data source categoriser 14, that is, assessing each of the plurality of data sources and allocating them to categories, may be performed by allocating a mapping value to each data source, with the mapping value determining the reconciliation processor that will process the data source. A mapping value is so-called because it represents the mapping of key values to name values within a data source.
Once the data have been extracted and stored in the data storage unit 17, the categoriser 14 is initialised, represented by step S303 in Figure 3. The categoriser 14 is configured to inspect each data source to produce a mapping value representing the mapping or relationship between the key and name for each data source. The inspection or examination of data from the data sources stored in the data storage unit 17 by the categoriser 14 is represented by step S304 in Figure 3. In this context, key and name are the label, header, or class definitions, with a key value or name value being a particular instance of either. The mapping value may be derivable by inspecting a schema or ontology definition for a data source and comparing labels (wherein labels is taken to encompass column headers, predicates, and other mechanisms for naming properties) with a predetermined list of labels known to refer to key values or name values. Alternatively or additionally, it may be that the data instantiating the schema or ontology definition is inspected in order to determine the mapping value.

An exemplary process performed by the categoriser 14 in categorising a data source can be described as follows, with reference to Figure 4:

S401: Identify the data source in the data storage unit 17 and scan the data.
S402: Check the key value to see if there are multiple entries in the data with the same key value.
S406. If there are no duplicated entries at step S102, then the mapping value is 1:1. This corresponds to the first category and the data source will be processed by the first reconciliation processor 11. An example of a data source having a 1:1 mapping value is presented in Figure 7.
S403. If there are duplicated entries (two entries with the same key value), check the name values to see whether they match.
S407. If at S403 the name values are the same for each duplicated entry, then the mapping is still 1:1, and the data source is allocated to the first category and will be processed by the first reconciliation processor 11.
S404. Otherwise, if there are different name values for the same key value, the mapping value is 1:m. An example of this 1:m duplication is shown in Figure 5. Figure 5 illustrates a segment of a CSV file as a result of the SPARQL query mentioned above. CSV file is a comma separated file, and similar to a spreadsheet but with a
simpler format. The first column is the URL on the DBPedia, (e.g., each resource has a unique URL assigned by DBPedia), the second is the name, e.g. in DBPedia sense, dbpprop:name. The dbpprop:name predicate has two object values, one is in English, one is in Japanese; this RDF format ends up in two entries in a CSV file with the same URL (the key value).

S404. In the case of a data source given a 1:m mapping value, further check the name value type, if it is one simple string object, for which the length is less than a predetermined threshold, for example 100 (which will vary depending on implementation, but in this example we are considering company names, and hence 100 characters seems sufficient), then proceed to S405 and the mapping value is 1:m-s (with the m indicating multiple name values per key value and the s indicating structured or semi-structured). This corresponds to the second category and hence the data source will be processed by the second reconciliation processor 12. However, if the length is longer than 100, this is an indication that a passage of text is present that must be parsed and name values extracted, therefore, the flow proceeds to S408, mapping value is 1:m-u (the u indicating unstructured). This corresponds to the third category and the data source will be processed by the third reconciliation processor 13.

An example of unstructured data is illustrated in Figure 6. Figures 5 and 6 provide examples distinguishing between 1:m:s and 1:m:u mapping, in which 1:m:s stands for one to many structured case (e.g. a column in a table), and corresponds to the first category, and 1:m:u stands for one to many unstructured case, (e.g. a block of text) and corresponds to the third category. The data source is dbpedia in both cases, and the url is still the key, but if the column we are trying to analyse is an unstructured text instead of a simple name in string, e.g. IBM, then the pattern based analysis of the third reconciliation processor 13 is applied. E.g., we will be looking at characters like "(" or "same as", or "well known" to determine the string on the two side of the characters are referring to the same entity.

Figure 7 presents a further example of a data source with a 1:1 mapping value (first category), a 1:m-s mapping value (second category), and a 1:m-u mapping value (third category).
The mapping value for each data source, or the category to which each data source is allocated, is stored in a registry 141 in the data source categoriser 14. Once the plurality of data sources have been categorised, the reconciliation processors are initialised, as represented by step S305 in Figure 3. The physical location of the registry 141 may be in memory or in a distributed setting, depending on the implementation details. Figure 8 illustrates an example of values stored in the registry 141.

There may be more than three mapping values and hence more than three categories of data source, depending on the implementation details.

The reconciliation processors each refer to the data source categoriser registry 141 for a notification of which data sources to process. The first reconciliation processor 11 processes data sources allocated to the first category, that is, with a 1:1 mapping value. The second reconciliation processor 12 processes data sources allocated to the second category, that is, with a 1:m-s mapping value. The third reconciliation processor 13 processes data sources allocated to the third category, that is, with a 1:m-u mapping value.

The reconciliator comprises three reconciliation processors, each responsible for performing reconciliation processing on data sources allocated to a particular category by the categoriser 14, or having a particular mapping value. The reconciliation processors may operate in parallel.

The first reconciliation processor 11 performs reconciliation processing on data sources allocated to the first category; those having a 1:1 mapping value. A data source is selected to serve as the index (also referred to as the index structured data source). The index data source may be defined manually depending on users' preference), or the selection may be automated, for example, selecting the data source with the most name values. The index data source is then subjected to reconciliation processing by the first reconciliation processor 11 with each of the other data sources in the first category in turn. This process is illustrated in Figure 9. Reconciling one data source with another is a process of comparing each name value in one data source with each
name value in another data source, and outputting a result to a list when the comparison satisfies a criterion.

For example, the reconciliation method may use a string similarity comparison as the mechanism for comparing two name values, e.g. Jaro-Winkler distance (Winkler, 1990) is an algorithm to measure the similarity between the two name strings. The result of the comparison is a score whose value is between 0 and 1, which 0 equates to no similarity and 1 is an exact match. The system 10 can adopt any value as the criterion for two name values being sufficiently similar to be considered aliases and hence entered into the first results list. As an example, 0.98 can be defined as the minimum score to confirm that the two variations of the names are pointing to the same entity. E.g.,

\[
\text{if } f(\text{name1}, \text{name2}) \geq 0.98, \text{ then } \text{name1} \approx \text{name2}
\]

Although the selection of this minimum score is flexible. In order to produce more accurate result, the standard can be raised. Lowering the standard will increase the number of false positives, but this may be desirable from a point of view of including all potential aliases, the trade-off being an increased burden at a results filtering stage. As an additional element to the criterion, it may be that exact matches are excluded from being entered into the results list.

The first results list is the appended results from the comparison of the index data source with each of the other first category data sources. The results are consolidated so that each name value in the index data source appears once, associated (i.e on the same row as) any aliases found by the first reconciliation processor 11. An indication of the data source from which the alias was found may also be included.

The second reconciliation processor 12 performs reconciliation processing on data sources allocated to the second category; those having a 1:m-s mapping value. The second reconciliation processor 12 is configured to reconcile names for data sources having mapping value of 1:m-s. The second reconciliation processor 12 may be configured to employ a semantic referencing technique to build the relationship between the plural name values for the same key value. The procedure carried out by the second reconciliation processor 12 is applicable to different types of data files, e.g.
CSV, JSON, or Turtle, but for illustration purpose, here we choose RDF Turtle to explain the theory, which is also because RDF is a well-known semantic rich language.

RDF data consists of statements. In which, each statement is produced in the form of subject-predicate-object expressions. The data types for the subject, predicate, object are typically URLs. For example, Figure 10a illustrates an exemplary RDF statement having a value for each of subject (the key value), predicate (the label), and object (the name value).

In Figure 10a, the object is of the URL type. However, the object data value can be a literal type, e.g. a string value.

For the objects that have URL type and multiple values, the inference result between the different values can only be that they belongs to the same type of class. For example, Figure 10B shows two RDF statements in which the objects (dbpedia:Thomas_J._Watson and dbpedia:Charles_Renlett_Flint) are both are type of person and are of the URL type. Since these two URLs cannot be further de-referenced, this means that they must contain different information, and they are not naming the same entity, even though the key values match and the data values are labelled in a manner which suggests that a name value is present.

However, for RDF statements in which the objects have literal type and multiple values, the inference result can be translated as similar to. Any object values which are deemed to be names and fulfil the “similar to” criteria will still be added to the second results list as aliases for one another. For at least at semantic level, there is no obvious difference in-between two string objects having different values but labelled in the same way and being properties of the same subject resource (i.e. having the same key value), for they are not further de-reference-able. So, in the example of Figure 10C, because the object values are of the string type and are different, even though the strings are apparently not similar (i.e. different length etc), the second reconciliation processor 12 will determine them to be aliases for one another and add them to the second results list. This is especially true when the object value we are examining is explicitly labelled as name by the predicate, which is purely identification, not
necessarily uniquely, for an entity. When the 1:m-s processor processes the data in Figure 10C, for example, it looks for the predicate value, if it contains "name" or variations of name, it will then look at the type of its object, if the type is literal, and one subject and predicate have multiple object literal values, we can reason the inference as:

"International Business Machines Corporation" similar to "IBM Corp."

And the two name values are added to the second results list as aliases for one another.

With this method, the reconciliation result can be further enlarged comparing to just by using the first reconciliation processor 11, because it covers the scenario when the similarity score is less than 0.98, but the names are actually representing the same entity.

The functionality of the second reconciliation processor 12 may be extended to include comparing the labelled data values, other than name values, describing properties of an entity that also provide aliases for the name value. For example, the second reconciliation processor may be further configured to compare the labels of data values attributed to the entity identified by the key value with a predefined list of labels indicating equivalence between entities, and to extract the data values labelled with labels included in the predefined list as aliases for the name value of the entity in the second results list. This functionality provides a mechanism for that knowledge of the data source syntax to generate content-specific results. An example of such a label is the "dbpedia-owl:wikiPageRedirects" property. For example, if the "dbpedia-owl:wikiPageRedirects" is included in a predefined list of labels by a user or administrator, or at system design-time, then objects linked to by that particular predicate are known to contain links to other articles/pages that are created, with slightly different name labels.

The third reconciliation processor 13 is configured to process the unstructured data, e.g. a block/passage of text, which contains one or more values of name information. It is particularly effective in a multi-lingua text scenario where name values are written/translated into different languages. The primary language content that this
processor can process is configurable either by a system designer, administrator, or by a user at run time as an argument when instructing the system 10. In the example below, English is used as the core language, that is, the language used to identify the presence of regular expressions and hence the location of name values. The technique executed by the third reconciliation processor 13 is also applicable to other languages.

The core technique to find two or more names matching is pattern matching. That is to say, the third reconciliation processor 13 is configured to use regular expressions to describe sequence patterns (a text string), which is a sequence of characters that forms a search pattern to search in the target text (the block or passage of text being processed). A regular expression processor (a sub-component of the third reconciliation processor 13) is then configured to process the or each regular expression into a nondeterministic finite automaton (NFA), for comparison with the text passage to recognise substrings that match the regular expression. Many programming languages provide regular expression capabilities, e.g. processors to extract the searching substring out of a block of text. Therefore, the third reconciliation processor 13 may be implemented as a program written using a programming language such as Java, Python, or C++, since in those languages there is an inherent function for parsing a regular expression statement.

Figure 11A illustrates an exemplary text passage extracted from an unstructured data source, and Figure 11B illustrates an exemplary form of regular expression stored by the third reconciliation processor 13 and used to recognise substrings matching the regular expression and hence the location of name values. In the example of Figures 11A-C, the entities for which name values are sought are companies, and hence company and corporation are used as part of the regular expression.

Figure 11C illustrates the result returned when the third reconciliation processor 13 uses the regular expression of Figure 11B when processing the block of text of Figure 11A.
The third reconciliation processor 13 is configured to conduct further analysis to extract the individual name values (i.e. Fujitsu Ltd. and 富士通株式会社) again to refine the pattern matching result, achieved by sub-string manipulation. The extracted name values are stored as aliases for one another in the third results list.

The results list combiner 16 is configured to process a union method the results list produced by the individual reconciliation processors. In a particular example, rather than a simple merge all operation, a cross-reference type of reconciliation across the results lists generated by the individual reconciliation processors is performed. The exemplary processing logic is as follows:

Given resultsSet1 (the first results list, produced by the first reconciliation processor 11), resultsSet2 (the second results list, produced the second reconciliation processor 12), and resultsSet3 (the third results list, produced by the third reconciliation processor 13), if any of the name values in the set of resultsSet1, can find at least one matching name in the set of resultSet2, then the results list combiner 16 concludes that the matching name value and its aliases from resultsSet1 can be reconciled with the matching name value and its aliases from resultSet2. The same logic is applied by the results list combiner 16 when combining resultsSet1 and resultSet3 to produce the final combined results list. Figure 12 illustrates the process flow, in which in resultsSet1 a, b, and c are all name values stored as aliases for one another, in resultsSet2 d, a, and e are all name values stored as aliases for one another, and in resultsSet3 f, g, and b are all name values stored as aliases for one another. The arrows in Figure 12 indicate matching name values. Name values a to g are stored as aliases for one another in the combined results list.

Further information can be incorporated into the combined results list by the results list combiner 16. Either prior to or after producing the combined results list, the results list combiner 16 may query the data sources (or the data extracted by the data extractor 18) to obtain, for each name value in the combined results list, or for each name value in the first, second, and/or third results lists, information identifying the data source (for example a location, title, or address) from which each name value was extracted, along with the key value identifying the entity named by the name value.
With the accurate combined results, entity identities can be accurately extracted from different data sources. For example, in the field of finance, the identities issued by different financial regulators can be extracted, and based on these identities, relevant financial reports can be downloaded thus allowing decision makers to build a comprehensive portfolio of a particular entity.
CLAIMS

1. A reconciliation processing system for reconciling different name values used to refer to the same entity across a plurality of data sources, the plurality of data sources including structured data sources being composed of labelled data values each describing a property of one of a plurality of entities, the entity being identifiable by a key value, and the property being identifiable by the label of the data value; the system comprising:

   a data source categoriser, being configured to identify, for each of the plurality of key values, one or more name values, a name value being a labelled data value for which the label indicates that the property attributes a name to the entity identified by the key value, and to allocate each of the structured data sources into one of two categories:

   a first category in which for each key value in the structured data source, there is only one name value; and

   a second category in which the structured data source includes one or more key values each having more than one different name value;

   a first reconciliation processor being configured to, in a case in which there is more than one structured data source in the first category, using one of the structured data sources in the first category as an index structured data source, for each name value in the index structured data source, find any name values from the remaining structured data sources of the first category that satisfy a first similarity criterion when compared to the name value from the index structured data source, and to store the found name values and the name value from the index structured data source as aliases for one another in a first results list;

   a second reconciliation processor being configured to, for each structured data source in the second category, for each key value in the structured data source, identify any labelled data values in the same structured data source which are name values attributing a name to the entity identified by the key value, and if more than one different name value is identified, to store the more than one different name values as aliases for one another in a second results list; and

   a results list combiner, being configured to generate a combined results list by appending the second results list to the first results list and, for each name value
appearing in the first results list, if an identical name value appears in the second results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list as aliases for one another in the combined results list.

2. A reconciliation processing system according to claim 1, wherein the plurality of data sources includes one or more unstructured data sources, each unstructured data source comprising a plurality of passages of text each describing an entity identifiable by a key value stored as part of or associated with the passage of text;

the data source categoriser being configured to identify the unstructured data sources from among the plurality of data sources and to allocate the unstructured data sources into a third category;

the system further comprising:

a third reconciliation processor, being configured to use pattern matching to identify, from within each of the passages of text, substrings arranged in a pattern indicating that the substring includes a name value attributing a name to the entity identifiable by the key value stored as part of or associated with the passage of text, to extract the indicated name value from the identified substrings, and when more than one different name value is extracted from a single passage of text, to store the more than one different name values as aliases for one another in a third results list; wherein the results list combiner being configured to generate a combined results list by appending the second results list and the third results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list and/or the third results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list and/or the third results list as aliases for one another in the combined results list.

3. A reconciliation processing system according to claim 2, wherein the third reconciliation processor is configured to use pattern matching by using each of a predetermined set of regular expressions, processed into nondeterministic finite automaton, to identify substrings that match the regular expression in the passage of text, the presence of the regular expression in the passage of text being an
indication that a name value is present at a particular position in the passage of text relative to the regular expression.

4. A reconciliation processing system according to any of claims 1 to 3, wherein
   the first reconciliation processor is configured to, in a case in which there is more
   than one structured data source in the first category, using one of the structured data
   sources in the first category as an index structured data source, for each name value in
   the index structured data source, to compare the name value with each of the name
   values from each of the remaining structured data sources in the first category as
   target structured data sources, and, for any comparison satisfying a first similarity
   criterion, to add the name value from the target structured data source to a list of
   aliases for the name value from the index structured data source.

5. A reconciliation processing system according to any of the preceding claims,
   wherein
   the first similarity criterion is satisfied if a string similarity comparison of two name
   values exceeds a predetermined threshold value.

6. A reconciliation processing system according to any of the preceding claims,
   wherein
   the second reconciliation processor is further configured to compare the labels of
   data values attributed to the entity identified by the key value with a predefined list of
   labels indicating equivalence between entities, and to extract the data values labelled
   with labels included in the predefined list as aliases for the name value of the entity in
   the second results list.

7. A reconciliation processing system according to any of the preceding claims,
   wherein the plurality of data sources includes data sources storing relational data and
   data sources storing graph data.

8. A reconciliation processing system according to any of the preceding claims,
   wherein the system further comprises:
a data extractor configured to provide data from the plurality of data sources to the reconciliation processors for reconciliation processing.

9. A reconciliation processing system according to any of the preceding claims, wherein the system further comprises:

a combined results propagator configured, for at least one of the plurality of data sources, for each name value in the data source that matches a name value in the combined results list, adding each of the aliases for the name value stored in the combined results list to the data source with an indication that the alias is an alternative to the name value.

10. A reconciliation processing system according to any of the preceding claims, wherein the first reconciliation processor is further configured, in the case in which there is more than one structured data source in the first category, to prompt a user for a selection of an index structured data source from among the more than one structured data source in the first category, and to use the selection as the index structured data source.

11. A reconciliation processing method for reconciling different name values used to refer to the same entity across a plurality of data sources, the plurality of data sources including structured data sources being composed of labelled data values each describing a property of one of a plurality of entities, the entity being identifiable by a key value, and the property being identifiable by the label of the data value; the method comprising:

a categorising step, comprising identifying, for each of the plurality of key values, one or more name values, a name value being a labelled data value for which the label indicates that the property attributes a name to the entity identified by the key value, and allocating each of the structured data sources into one of two categories:

a first category in which for each key value in the structured data source, there is only one name value; and

a second category in which the structured data source includes one or more key values each having more than one different name value;
a first reconciliation process comprising, in a case in which there is more than one structured data source in the first category, using one of the structured data sources in the first category as an index structured data source, for each name value in the index structured data source, finding any name values from the remaining structured data sources that satisfy a first similarity criterion when compared to the name value from the index structured data source, and storing the found name values and the name value from the index structured data source as aliases for one another in a first results list;

a second reconciliation process comprising, for each structured data source in the second category, for each key value in the structured data source, identifying any labelled data values which are name values attributing a name to the entity identified by the key value, and if more than one different name value is identified, storing the more than one different name values as aliases for one another in a second results list; and

a combining step, comprising generating a combined results list by appending the second results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list as aliases for one another in the combined results list.

12. A reconciliation processing method according to claim 11, wherein the plurality of data sources includes one or more unstructured data sources, each unstructured data source comprising a plurality of passages of text each describing an entity identifiable by a key value stored as part of or associated with the passage of text;

the categorising step also comprising identifying the unstructured data sources from among the plurality of data sources and to allocate the unstructured data sources into a third category;

the method further comprising:

a third reconciliation process comprising using pattern matching to identify, from within each of the passages of text, substrings arranged in a pattern indicating that the substring includes a name value attributing a name to the entity identifiable by the key value stored as part of or associated with the passage of text, extracting the indicated
name value from the identified substrings, and when more than one different name value is extracted from a single passage of text, storing the more than one different name values as aliases for one another in a third results list; wherein the combining step comprising generating a combined results list by appending the second results list and the third results list to the first results list and, for each name value appearing in the first results list, if an identical name value appears in the second results list and/or the third results list, storing the aliases for the name value from the first results list and the aliases for the identical name value from the second results list and/or the third results list as aliases for one another in the combined results list.

13. A computer program which, when executed by a computing device, causes the computing device to execute the method of claim 11 or 12.

14. A suite of computer programs which, when executed by one or more computing devices, causes the one or more computing devices to function as the reconciliation processing system of any of claims 1 to 10.
Patents Act 1977: Search Report under Section 17

Documents considered to be relevant:

<table>
<thead>
<tr>
<th>Category</th>
<th>Relevant to claims</th>
<th>Identity of document and passage or figure of particular relevance</th>
</tr>
</thead>
</table>

Categories:

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>Document indicating lack of novelty or inventive step</td>
<td>A</td>
</tr>
<tr>
<td>Y</td>
<td>Document indicating lack of inventive step if combined with other documents</td>
<td>P</td>
</tr>
<tr>
<td>&amp;</td>
<td>Member of the same patent family</td>
<td>E</td>
</tr>
</tbody>
</table>

Field of Search:
Search of GB, EP, WO & US patent documents classified in the following areas of the UKCN:

Worldwide search of patent documents classified in the following areas of the IPC

G06F

The following online and other databases have been used in the preparation of this search report

EPODOC, WIPO, TXTE.

International Classification:

<table>
<thead>
<tr>
<th>Subclass</th>
<th>Subgroup</th>
<th>Valid From</th>
</tr>
</thead>
<tbody>
<tr>
<td>G06F</td>
<td>0017/30</td>
<td>01/01/2006</td>
</tr>
</tbody>
</table>