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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 December 2009 (17.12.2009)

PCT

(10) International Publication Number
WO 2009/151873 A1

(51) International Patent Classification:
B65G 47/08 (2006.01)

(21) International Application Number:
PCT/US2009/044258

(22) International Filing Date:
15 May 2009 (15.05.2009)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
12/136,616 10 June 2008 (10.06.2008) US

(71) Applicant *(for all designated States except US)*: **MICROSOFT CORPORATION** [US/US]; One Microsoft Way, Redmond, WA 98052-6399 (US).

(72) Inventors: **HUGHES, Gregory, L.**; One Microsoft Way, Redmond, WA 98052-6399 (US). **KERER, Clemens**; One Microsoft Way, Redmond, WA 98052-6399 (US). **OLENICK, Brad M.**; One Microsoft Way, Redmond, WA 98052-6399 (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, 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, 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, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

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

(54) **Title**: VIRTUALIZING OBJECTS WITHIN QUERIES

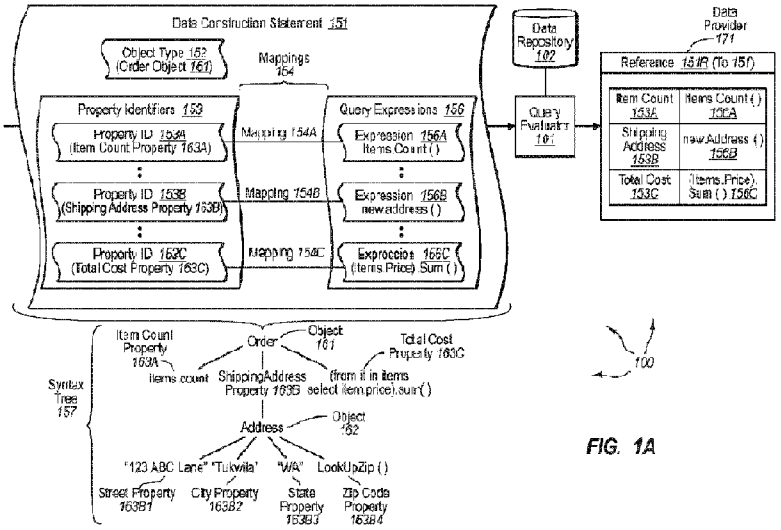


FIG. 1A

(57) **Abstract**: The present invention extends to methods, systems, and computer program products for virtualizing objects within queries. Embodiments of the invention virtualize data access for use with queries. Virtualization can be implemented within any portion of a syntax tree. For example, data can be virtualized for a property of an object that is itself another object. Data virtualization facilitates lazy evaluation of query expressions. That is, actual property values for properties within a data construction statement are virtualized until a query specifically requests the actual property values. Further, data virtualization also conserves resources and results in more efficient query evaluations.

WO 2009/151873 A1

VIRTUALIZING OBJECTS WITHIN QUERIES

BACKGROUND

1. Background and Relevant Art

[0001] Computer systems and related technology affect many aspects of society. Indeed, the computer system's ability to process information has transformed the way we live and work. Computer systems now commonly perform a host of tasks (e.g., word processing, scheduling, accounting, etc.) that prior to the advent of the computer system were performed manually. More recently, computer systems have been coupled to one another and to other electronic devices to form both wired and wireless computer networks over which the computer systems and other electronic devices can transfer electronic data. Accordingly, the performance of many computing tasks are distributed across a number of different computer systems and/or a number of different computing environments.

[0002] Most, if not all, computing tasks typically include software (e.g., an application or operating system) performing operations on, such as, for example, creating, modifying, deleting, etc., data to obtain a desired result. As such, for software to obtain a desired result, the software must have access to appropriate data. For example, to make edits to a word processing document, a user must locate the word processing document (e.g., within a file system) and enter commands into to the word processor to instruct the word processor to open the word processing document.

[0003] If the user is unaware of the documents location, the user can use various search tools to locate the document. The user can enter various search terms into searching module (e.g., file finder). The searching module can search the file system for files that include characteristics identified in the search terms and return the location of any identified files to the user. Thus, to some extent the user submits a query (the search terms) to the searching module, the searching module processes the query (identifying any matches), and to the searching module returns results (any identified matches).

[0004] In many environments, more complex data operations are used to create and access data subsets of a larger portion of data. For example within a database system, data creation statements can be used to create views on data in a database. Queries can be issued against views (or other objects) on (or even the entire) database to retrieve desired subsets of data. Based on the arrangement and volume of data in a database, data creation statements and queries can be relatively complex. Further, a series of data creation statements and queries can be used in combination with one another to access data more granularly, resulting in additional complexities.

[0005] Additionally, it is often the case that less than all of a set (or subset) of grouped data is requested to perform a desired computed computing operation. For example, a database may contain an address object containing a business address for a number of customers. Each address object may contain street, city, state, and zip code fields. However, to perform a desired operation (e.g., determining number of customers per zip code) only values in the zip code field are relevant. Thus, a query can be issued to obtain a value form the zip code field of each address object.

[0006] Unfortunately, in most, if not all, database systems, when a query references a portion of (e.g., a field within) an object, the entire object is instantiated, requiring eager evaluation of the entire object. That is, the entire object is evaluated prior to processing the query that references only a portion of the object. For example, the zip code query would require instantiation of the entire address object prior to processing the zip code query. Instantiating and evaluating data that is not relevant to a query unnecessarily consumes computer system resources and leads to query processing inefficiencies.

[0007] Further, as object size increases, unnecessarily consumption of computer system resources and query processing inefficiencies also increases. For example, consider the following objet query that describes the creation of an object representing an order:

```
Order order =
        new Order()
        {
                ItemCount = items.Count(),
```

```
                ShippingAddress =
                        new Address()
                        {
                                Street = "123 Abc Lane",
                                City = "Tukwila",
                                State = "WA",
                                ZipCode = LookupZipCode(…)
                        },
                TotalCost =
                        (from it in items
                        select item.Price).Sum()
        }
```

And the query over the object:

```
Address address = order.Address
```

[0008] Typical evaluation of this query would require the Order object to first be instantiated, resulting in the eager evaluation of the expressions for the ItemCount, ShippingAddres, and TotalCost properties at that time. Then, the value of the ShippingAddress property would be retrieved from the Order object instance. The creation of the Order object and evaluation of the expression for the ItemCount and TotalCost properties are not used by the second query but computer system resources and time were used in their creation and evaluation anyway.

BRIEF SUMMARY

[0009] The present invention extends to methods, systems, and computer program products for virtualizing objects within queries. In some embodiments, an object is created in such a way that portions of the object are accessible without constructing the entire object. A query evaluator receives a data construction statement (e.g., an object query fragment). The data construction statement includes an indication of the type of object that the data construction statement represents the construction of and one or more property identifiers. Each property identifier identifies a property of the object. The data construction statement also includes one or more corresponding mappings. Each corresponding mapping maps a property identifier to a query expression. The query expression indicates how to compute the value for the property of the object identified by the property identifier.

[0010] The query evaluator creates a data provider in response to receiving the data construction statement. The data provider virtualizes data access so that

properties of the object described by the data construction statement can be accessed without having to evaluate the entire data construction statement or fully construct the entire object. In some embodiments, the data provider includes a reference to the data construction statement. The reference transitively includes references back to the type, property identifiers, and corresponding mappings of the data construction statement.

[0011] In other embodiments, a query evaluator responds to a query. The query evaluator receives a property value access query. The property value access query queries for a property value of a property identified in a previously received data construction statement. The query evaluator identifies a data provider created from the previously received data construction statement based on data contained in the property value access query. The data provider virtualizes data access so that properties of an object described by the data construction statement can be accessed without having to evaluate the entire data construction statement or fully construct the entire object.

[0012] The query evaluator places the identified data provider in a known location associated with the query evaluator (e.g., a stack used by the query evaluator). The query evaluator creates a data consumer for the property value access query. The data consumer accesses the identified data provider from the known location.

[0013] The data consumer requests the property value of the identified property from the data provider. The data consumer receives the property value of the identified property from the data provider. The data consumer places the received property value in the known location for further processing.

[0014] This summary is provided to introduce a selection of concepts in a simplified form that are further described below in the Detailed Description. This Summary is not intended to identify key features or essential features of the claimed subject matter, nor is it intended to be used as an aid in determining the scope of the claimed subject matter.

[0015] Additional features and advantages of the invention will be set forth in the description which follows, and in part will be obvious from the description, or may be learned by the practice of the invention. The features and advantages of the

invention may be realized and obtained by means of the instruments and combinations particularly pointed out in the appended claims. These and other features of the present invention will become more fully apparent from the following description and appended claims, or may be learned by the practice of the invention as set forth hereinafter.

BRIEF DESCRIPTION OF THE DRAWINGS

[0016] In order to describe the manner in which the above-recited and other advantages and features of the invention can be obtained, a more particular description of the invention briefly described above will be rendered by reference to specific embodiments thereof which are illustrated in the appended drawings. Understanding that these drawings depict only typical embodiments of the invention and are not therefore to be considered to be limiting of its scope, the invention will be described and explained with additional specificity and detail through the use of the accompanying drawings in which:

[0017] Figure 1A illustrates an example computer architecture that facilitates creating an object that virtualizes data access.

[0018] Figure 1B illustrates the example computer architecture of Figure 1A including further components that facilitate responding to a query.

[0019] Figure 1C illustrates the example computer architecture of Figure 1B including further components that facilitate responding to a query.

[0020] Figure 2 illustrates a flow chart of an example method for creating an object hat virtualizes data access.

[0021] Figure 3 illustrates a flow of an example method for responding to a query

DETAILED DESCRIPTION

[0022] The present invention extends to methods, systems, and computer program products for virtualizing objects within queries. In some embodiments, an object is created in such a way that portions of the object are accessible without constructing the entire object. A query evaluator receives a data construction statement (e.g., an object query fragment). The data construction statement includes an indication of the type of object that the data construction statement represents the construction of and one or more property identifiers. Each property identifier identifies a property

of the object. The data construction statement also includes one or more corresponding mappings. Each corresponding mapping maps a property identifier to a query expression. The query expression indicating how to compute the value for the property of the object identified by the property identifier.

[0023] The query evaluator creates a data provider in response to receiving the data construction statement. The data provider virtualizes data access so that properties of the object described by the data construction statement can be accessed without having to evaluate the entire data construction statement or fully construct the entire object. In some embodiments, the data provider includes a reference to the data construction statement. The reference transitively includes references back to the type, property identifiers, and corresponding mappings of the data construction statement.

[0024] In other embodiments, a query evaluator responds to a query. The query evaluator receives a property value access query. The property value access query queries for a property value of a property identified in a previously received data construction statement. The query evaluator identifies a data provider created from the previously received data construction statement based on data contained in the property value access query. The data provider virtualizes data access so that properties of an object described by the data construction statement can be accessed without having to evaluate the entire data construction statement or fully construct the entire object.

[0025] The query evaluator places the identified data provider in a known location associated with the query evaluator (e.g., a stack used by the query evaluator). The query evaluator creates a data consumer for the property value access query. The data consumer accesses the identified data provider from the known location.

[0026] The data consumer requests the property value of the identified property from the data provider. The data consumer receives the property value of the identified property from the data provider. The data consumer places the received property value in the known location for further processing.

[0027] Embodiments of the present invention may comprise or utilize a special purpose or general-purpose computer including computer hardware, as discussed in

greater detail below. Embodiments within the scope of the present invention also include physical and other computer-readable media for carrying or storing computer-executable instructions and/or data structures. Such computer-readable media can be any available media that can be accessed by a general purpose or special purpose computer system. Computer-readable media that store computer-executable instructions are physical storage media. Computer-readable media that carry computer-executable instructions are transmission media. Thus, by way of example, and not limitation, embodiments of the invention can comprise at least two distinctly different kinds of computer-readable media: physical storage media and transmission media.

[0028] Physical storage media includes RAM, ROM, EEPROM, CD-ROM or other optical disk storage, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store desired program code means in the form of computer-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer.

[0029] A "network" is defined as one or more data links that enable the transport of electronic data between computer systems and/or modules and/or other electronic devices. When information is transferred or provided over a network or another communications connection (either hardwired, wireless, or a combination of hardwired or wireless) to a computer, the computer properly views the connection as a transmission medium. Transmissions media can include a network and/or data links which can be used to carry or desired program code means in the form of computer-executable instructions or data structures and which can be accessed by a general purpose or special purpose computer. Combinations of the above should also be included within the scope of computer-readable media.

[0030] Further, upon reaching various computer system components, program code means in the form of computer-executable instructions or data structures can be transferred automatically from transmission media to physical storage media (or vice versa). For example, computer-executable instructions or data structures received over a network or data link can be buffered in RAM within a network interface module (e.g., a "NIC"), and then eventually transferred to computer

system RAM and/or to less volatile physical storage media at a computer system. Thus, it should be understood that physical storage media can be included in computer system components that also (or even primarily) utilize transmission media.

[0031] Computer-executable instructions comprise, for example, instructions and data which cause a general purpose computer, special purpose computer, or special purpose processing device to perform a certain function or group of functions. The computer executable instructions may be, for example, binaries, intermediate format instructions such as assembly language, or even source code. Although the subject matter has been described in language specific to structural features and/or methodological acts, it is to be understood that the subject matter defined in the appended claims is not necessarily limited to the described features or acts described above. Rather, the described features and acts are disclosed as example forms of implementing the claims.

[0032] Those skilled in the art will appreciate that the invention may be practiced in network computing environments with many types of computer system configurations, including, personal computers, desktop computers, laptop computers, message processors, hand-held devices, multi-processor systems, microprocessor-based or programmable consumer electronics, network PCs, minicomputers, mainframe computers, mobile telephones, PDAs, pagers, routers, switches, and the like. The invention may also be practiced in distributed system environments where local and remote computer systems, which are linked (either by hardwired data links, wireless data links, or by a combination of hardwired and wireless data links) through a network, both perform tasks. In a distributed system environment, program modules may be located in both local and remote memory storage devices.

[0033] Figure 1A illustrates an example computer architecture 100 that facilitates creating an object that virtualizes data access. Figure 1B illustrates computer architecture 100 including further components that facilitate responding to a query. Figure 1C illustrates computer architecture 100 including further components that facilitate responding to a query. Thus, computer architecture 100 includes query

evaluator 101, data repository 102, and cache 103. At different times during object creation and/or query evaluation, computer architecture 100 can also include on or more data consumers, such as, for example, data consumers 106 (Figure 1B), 107 (Figure 1C) and 108 (Figure 1C) and one or more data providers, such as, for example, data provider 171 (Figures 1A and 1B) and data provider 173B (Figure 1C).

[0034] Each of the depicted components can be connected to one another over (or be part of) a network, such as, for example, a Local Area Network ("LAN"), a Wide Area Network ("WAN"), and even the Internet. Accordingly, each of the depicted components as well as any other connected components, can create message related data and exchange message related data (e.g., Internet Protocol ("IP") datagrams and other higher layer protocols that utilize IP datagrams, such as, Transmission Control Protocol ("TCP"), Hypertext Transfer Protocol ("HTTP"), Simple Mail Transfer Protocol ("SMTP"), etc.) over the network.

[0035] Embodiments of the invention also generally implement a protocol that includes request / response pairs of messages. The request is sent from a data consumer to a data provider and then a response is returned from the data provider to the data consumer. The request contains an identifier for the property to be accessed and possible other information for the maintenance of secondary protocols between the consumer and provider. The response contains the value of the property that was accessed (which may be another virtualized representation of a data object / protocol provider) and other information for the maintenance of secondary protocols between the provider and consumer.

[0036] Referring to Figure 1, query evaluator 101 is generally configured to execute a set of instructions that have been compiled from an expression. Query evaluator can maintain a set of internal state that may be read and updated by each instruction as it is executed. Generally, an instruction reads results of previous instructions from the current state of query evaluator 101 and writes results back to the current state of query evaluator 101 so that they are available to further instructions.

[0037] Whenever the query evaluator 101 encounters an object creation instruction (e.g., new Order() { … }), it creates a data provider and populates the data provider with information about the type of the object that the data provider is virtualizing (e.g. a list of property identifiers). The data provider is then placed in the current state of the evaluator, making it available for use by further instructions. Whenever query evaluator 101 encounters a property access instruction (e.g. order.ItemCount), it creates a data consumer, passing the identifier of the property to the data consumer along with the provider retrieved from the current state of query evaluator 101. The result of the evaluation of the data consumer is placed into the current state of query evaluator 101 where it can be used by further instructions.

[0038] Accordingly, query evaluator 101 is generally configured to receive data construction statements and create data providers from the data construction statements.

[0039] In some embodiments, data construction statements represent objects defined in accordance with syntax trees. For example, data construction statement 151 represents order object 161 defined in accordance with syntax tree 157. Generally, a syntax tree (also sometimes referred to as an abstract syntax tree ("AST") is a directed tree where each interior node represents a construct and the children of that node represent meaningful components of the construct. For example, referring to syntax tree 157, the child of ShippingAddress property 163B is an Address construct. Street property 163B1, City property 163B2, State property 163B3, and ZipCode property 163B4 represent meaningful components of the Address construct child of ShippingAddress property 163B.

[0040] Syntax tree 157 can represent a query fragment similar to:

```
Order order =
        new Order()
        {
                ItemCount = items.Count(),
                ShippingAddress =
                        new Address()
                        {
                                Street = "123 Abc Lane",
```

```
                    City = "Tukwila",
                    State = "WA",
                    ZipCode = LookupZipCode(...)
                },
            TotalCost =
                (from it in items
                select item.Price).Sum()
        }
```

[0041] Accordingly, query evaluator 101 can receive data construction statement 151 and create data provider 171 from data construction statement 151. Data provider 171 virtualizes data access for object 161. Visualizing data access facilitates providing access to properties of object 161 without having to entirely evaluate data construction statement 151 (i.e., obtain actual data values for each property in syntax tree 157). Vitalizing data access also includes providing access to properties of object 161 without having to fully construct object 161.

[0042] Data repository 102 represents a database or other data storage location that stores data. Data construction statements can be configured to add data to, provide views on data from, or otherwise modify data stored in data repository 101.

[0043] Generally, a data provider is created (e.g., by query evaluator 101) each time a query expression that describes the construction of data is evaluated, such as, for example, Order order = new Order() { ItemCount = items.Count(), ... }. Each data provider contains information about the type of the object it represents. Part of this information is a list of all of the identifiers of the properties contained on the object (e.g. ItemCount). Each data provider also contains an association map between each property identifier in the list and the query expression that will be used to compute the value of the property for that virtualized instance (e.g. items.Count()). A data provider can also maintain further association maps for the property identifiers (such as a cache of results for each property).

[0044] When a request message is received at a data provider, the data provider extracts the property identifier from the message and looks up the corresponding query expression in its association map. The data provider can also retrieve any information from its other association maps for the property (e.g. a previously-cached value for the property). The data provide can also use any additional

information contained in the request message (e.g. a time-to-live specification for the cached value).

[0045] A data provider can then invoke query evaluator 101 to process the query expression for the property, possibly passing along any secondary information from its internal maps and/or the request message. The result of query evaluation (including any secondary information) is used to update the association maps within the data provider (e.g. caching the result of the query). The result of the query (and any secondary information) is packaged up into a response message. The response message is then sent to the client that initiated the request against the data provider.

[0046] Referring now to Figure 2, Figure 2 illustrates a flow chart of an example method 200 for creating an object that virtualizes data access. Method 200 will be descried with respect to the components and data of computer architecture 100 depicted in Figure 1A.

[0047] Method 200 includes an act of receiving a data construction statement (act 201). For example, query evaluator 101 can receive data consecution statement 151. Data construction statement 151 can be received from some other networked computer system in computer architecture 100 (not shown). Alternately, data construction statement 151 can be received as the result of query evaluator 101 evaluating a query expression.

[0048] The data construction statement includes an indication of the type of object that the data construction statement represents the construction of. For example, data construction statement 151 includes object type 152 indicating that data construction statement 151 represents construction of an order object 161. Data construction statement 151 can define a view on data in data repository 102.

[0049] The data construction statement also includes one or more property identifiers. For example, data constructions statement 151 includes property identifies 153. Each property identifier identifying a property of the object. For example, property ID 153A identifies ItemCount property 163A, property ID 153B identifies ShippingAddress property163B, and property ID 153C identifies TotalCost property 163C.

[0050] The data construction statement also includes one or more corresponding mappings. For example, data construction statement 151 includes mappings 154. Each corresponding mapping maps a property identifier to a query expression. For example, mapping 154A maps property ID 153A to query expression 156A, mapping 154B maps property ID 153B to query expression 156B, and mapping 154C maps property ID 153C to query expression 156C. Each query expression indicates how to compute the value for the property of the object identified by the property identifier. For example, query expression 156A indicates how to compute the value for ItemCount property 163A, query expression 156B indicates how to compute the value for ShippingAddress property 163B, and query expression 156C indicates how to compute the value for TotalCost property 163C.

[0051] Method 200 includes an act of creating a data provider in response to receiving the data construction statement (act 202). For example, query evaluator 101 can create data provider 171 in response to receiving data construction statement 151. Data provider 171 virtualizing data access so that properties of order object 161 (described by the data construction statement 151) can be accessed without having to entirely evaluate data construction statement 151 or fully construct order object 161.

[0052] For example, instead of query evaluator 101 evaluating all the properties of ShippingAddress property 163B, data provider 171 instead indicates how values for those properties (e.g., 163B1, 163B2, 163B3, and 163B4) can be evaluated in the event they are subsequently requested (e.g., evaluating a further data construction statement "new Address()" in response to a subsequent property value access query). Further, data provider 171 represents partial construction of order object 161. Accordingly, resources are conserved by minimizing potentially unnecessary access to data repository 102 and minimizing potentially unnecessary allocation of memory to store portions of object 161. Minimizing data access and memory allocation also facilitates more efficient query processing.

[0053] In some embodiments, a data provider (e.g., an initially created data provider for a top level object) includes a reference to the data construction statement. For example, data provider 171 includes reference 151R to data

construction statement 151. The reference transitively includes references to the type, property identifiers, and corresponding mappings of the data construction statement. For example, reference 151R transitively includes references to object type 152 property identifiers 153, and mappings 154.

[0054] Referring now to Figure 1B, Figure 1B illustrates computer architecture 100 including further components that facilitate responding to a query. Thus generally, query evaluator 101 is also configured to response to queries, such as, for example, property value access queries. In response to a property value access query, query evaluator 101 can identify an appropriate data provider to respond to the property value access query. Query evaluator 101 can place the identified data provider on stack 104.

[0055] Query evaluator 101 can create a data consumer to interact with the identified data provider. The data consumer can access the identified data provider from stack 104 and send a property identifier to the identified data provider. The data provider can evaluate the query expression corresponding to the property identifier and return the result to the data consumer. The data consumer can place the result on stack 104. Evaluation of a query expression can include referring to database 102 to access actual data values.

[0056] In some embodiments, a previously created data provider is accessed from cache 103.

[0057] Figure 3 illustrates a flow chart of an example method 300 for responding to a query. Method 300 will be descried with respect to the components and data of computer architecture 100 depicted in Figure 1B.

[0058] Method 300 includes an act of receiving a property value access query, the property value access query querying for a property value of a property identified in a previously received data construction statement (act 301). For example, query evaluator 101 can receive property value access query 121. Property value access query 121 is a query for a property value of ItemCount property 163A from data construction statement 151. Query evaluator 101 can parse property value access query 121 to determine that the "order" portion of property value access query 121 identifies order object 161 and the "ItemCount" portion of property value access

query 121 identifies ItemCount property 163A. Thus, from this information query evaluator 101 can also determine that the property ID 153A corresponds to the property being queried.

[0059] Method 300 includes an act of identifying a data provider created from the previously received data construction statement based on data contained in the property value access query (act 302). For example, query evaluator 101 can identify data provider 171 (created from data construction statement 151) based the term "order" included in property value access query 121.

[0060] The accessed data provider virtualizes data access so that properties of an object described by the data construction statement can be accessed without having to evaluate the entire data construction statement or fully construct the entire object. For example, as previously described data provider 171 virtualizes data access to the properties of order object 161.

[0061] Method 300 includes an act of placing the identified data provider in a known location associated with the query evaluator (act 303). For example, query evaluator 101 can place data provider 171 on stack 104.

[0062] Generally, a protocol consumer is evaluated (e.g. by query evaluator 101) each time a query expression that describes an access of a property value is evaluated, such as, for example, order.ItemCount. A protocol consumer is given (e.g., by the query evaluator 101) the identifier of the property that it is to access from a data provider and the instance of the data provider that it is to communicate with.

[0063] The data consumer constructs a request message containing the property identifier and possibly any additional out-of-band information available from the query evaluator (e.g., TTL of cached values). The consumer then sends the request to the provider and waits for a response. When a response is received from the data provider, the consumer pulls out the result value of the property and returns it to query evaluator 101, query evaluator 101 can then place the result value on stack 104 for consumption by further queries. The data consumer also updates any additional state / out-of-band information of query evaluator 101 based on the additional information contained within the response message.

[0064] Accordingly, method 300 includes an act of creating a data consumer for the property value access query (act 304). For example, query evaluator 101 can create data consumer 106. A created data consumer can be passed the property identifier corresponding to the property for which a property value is being queried. For example, query evaluator 101 can pass property ID 153A to data consumer 106 (or can construct data consumer 106 to include property ID 153A).

[0065] Method 300 includes an act of the data consumer accessing the identified data provider from the known location (act 305). For example, data consumer 106 can access data provider 171 from stack 104. Method 300 includes an act of the data consumer requesting the property value of the identified property from the data provider (act 306). For example, data consumer 106 can submit property ID 153A to data provider 171 to request a property value for ItemCount property 163A.

[0066] Data provider 171 can match property ID 153A to query expression 156A. Query evaluator 101 can then evaluate query expression 156A to create a value for ItemCount property 163A. For example, query evaluator 101 can refer to data repository 102 and access data 197 to calculate value 108 (e.g., the total number of items in an order). Data provider 171 can return value 108 to data consumer 106.

[0067] Method 300 includes an act of the data consumer receiving the property value of the identified property from the data provider (act 307). For example, data consumer 106 can receive value 108 from data provider 171. Method 300 includes an act of the data consumer placing the received property value in the known location for further processing (act 308). For example, data consumer 106 can place value 108 on stack 104 (e.g., replacing data provider 171). Accordingly, a property value for ItemCount property 163A can be obtained without full creation of object 161 and without having to evaluate actual values for properties 163C, 162B1, 162B2, 162B3, and 162B4.

[0068] Figure 1C illustrates computer architecture 300 and includes further components that facilitate responding to a query. Generally, query evaluator 101 is also configured to create further data providers from further data construction statements that are evaluated during the processing of a received property value access query. Referring now to Figure 1C, in some embodiments, a property ID

can map to a query expression that is a further data construction statement. For example, in data provider 171, property ID 153B is mapped to query expression 156B, which is a data construction statement "new Address()" for creating an address object. Evaluation of the further data construction statement can result in creation of a further data provider. For example, evaluation of query expression 156B can result in creation of data provider 173B.

[0069] Thus, properties of objects that are themselves objects can also be virtualized to conserve resources and facilitate efficient query processing. For example, still referring to Figure 1C, query evaluator can receive property value access query 121. Property value access query 121 is a query for a property value of Street property 163B1 from data construction statement 151. Query evaluator 101 can parse property value access query 121 to determine that the "order" portion of property value access query 121 identifies order object 161, the "ShippingAddress" portion of property value access query 121 identifies Shipping Address property 163B, and the "Street portion of property value access query 121 identifies Street property 163B1. Thus, from this information query evaluator 101 can also determine that a property ID 153B1 corresponds to the property being queried.

[0070] Query evaluator 101 can identify data provider 171 (created from data construction statement 151) based the term "order.ShippingAddress.Street" included in property value access query 121. Query evaluator 101 can place data provider 171 on stack 104.

[0071] Query evaluator 101 can create data consumer 107 and provide it with property ID 153B. From property 153B, data consumer 107 is able to request a value for ShippingAddress property 163B. To do this, data consumer 107 can access data provider 171 from stack 104. Data consumer 107 can then submit property ID 153B to data provider 171 to request a property value for ShippingAddress property 163B.

[0072] Data provider 171 can match property ID 153B to query expression 156B. Query evaluator 101 can then evaluate query expression 156B. Evaluation of query expression 156B (a further data construction statement) results in creation of data

provider 173B. Data provider 173B virtualizes data access to the object representing the Address object 162 that is the value of ShippingAddress property 163B, similar to how Data provider 151 virtualizes data access to order object 161. Data provider 171 can return data provider 173B to data consumer 107. Data consumer 107 can receive data provider 173B and place data provider 173B on stack 104 (e.g., replacing data provider 171).

[0073] Query evaluator 101 can create data consumer 108 and passed property ID 153B1. From property 153B1, data consumer 108 is able to request a value for street property 163B1. Data consumer 108 can then access data provider 173B from stack 104. Data consumer 108 can submit property ID 153B1 to data provider 173B to request a property value for Street property 163B1.

[0074] Data provider 173B can match property ID 153B1 to query expression 156B1. Query evaluator 101 can then evaluate query expression 156B1. Query expression 156B1 evaluates to the literal data value "123 ABC Lane". Data provider 173B returns literal data value "123 ABC Lane" to data consumer 108. Data consumer 108 can receive literal data value "123 ABC Lane" and place literal data value "123 ABC Lane"on stack 104 (e.g., replacing data provider 173B).

[0075] In some embodiments, out-of-band data is transmitted through composed queries. The out-of-band data can be used to establish and maintain internal protocols among queries. For example, time-to-live ("TTL") values can be used to determine how long a data provider is to be cached. For example, a data consumer can include a TTL value, for example, 250ms, in a request to indicate to a data provider how long it is to be remain cached.

[0076] Accordingly, embodiments of the invention virtualize data access for use with queries. Virtualization can be implemented within any portion of a syntax tree. For example, data can be virtualized for a property of an object that is itself another object. Data virtualization facilitates lazy evaluation of query expressions. That is, actual property values for properties within a data construction statement are virtualized until a query specifically requests the actual property values. Further, and as previously described, data virtualization also conserves resources and results in more efficient query evaluations.

[0077] The present invention may be embodied in other specific forms without departing from its spirit or essential characteristics. The described embodiments are to be considered in all respects only as illustrative and not restrictive. The scope of the invention is, therefore, indicated by the appended claims rather than by the foregoing description. All changes which come within the meaning and range of equivalency of the claims are to be embraced within their scope.

CLAIMS

What is claimed:

1. At a computer system, a method for creating an object (161) so that portions of the object (161) are accessible without construction of the entire object (161), the method comprising:

an act of receiving a data construction statement (151), the data construction statement (151) including:

an indication of the type of object (152) that the data construction statement represents the construction of;

one or more property identifiers (153), each property identifier identifying a property of the object; and

one or more corresponding mappings (154), each corresponding mapping mapping a property identifier to a query expression (156), the query expression (156) indicating how to compute the value for the property of the object identified by the property identifier; and

an act of creating a data provider (171) in response to receiving the data construction statement (151), the data provider (171) virtualizing data access so that properties of the object (161) described by the data construction statement (151) can be accessed without having to evaluate the entire data construction statement (151) or fully construct the entire object (161), the data provider including:

a reference (151R) to the data construction statement, the reference (151R) transitively including references to the type, property identifiers, and corresponding mappings of the data construction statement.

2. The method as recited in claim 1, wherein the act of receiving a data construction statement comprises an act of receiving user input submitting the data construction statement for evaluation prior to the submission of any queries against properties of the virtualized object described by the data construction statement.

3. The method as recited in claim 1, wherein the act of receiving a data construction statement comprises an act of receiving the data construction statement from an existing data provider, the data construction statement being

contained within a query expression from the property mappings of the existing data provider.

4. The method as recited in claim 1, further comprising:

an act of receiving a request from a data consumer for the value of a property within the data provider;

an act of referring to the corresponding one or more mappings to locate a query expression corresponding to the identified property;

an act of evaluating the query expression to obtain a property value for the property; and

an act of the returning the property value to data consumer.

5. The method as recited in claim 4, wherein the act of evaluating the query expression to obtain a property value for the property comprises an act of evaluating the query expression to obtain the actual property value for the property.

6. The method as recited in claim 5, wherein the act of evaluating the query expression to obtain the actual property value for the property comprises an act of referring to a data repository.

7. The method as recited in claim 6, wherein the act of referring to a data repository comprises an act of referring to the data repository to access data for calculating the actual property value without accessing data for calculating the values of any other properties described by the data construction statement.

8. The method as recited in claim 4, wherein the an act of evaluating the query expression to obtain a property value for the property comprises an act of evaluating the query expression to obtain a further data provider, the further data provider virtualizing the property value from the data provider.

9. The method as recited in claim 8, wherein the act of evaluating the query expression to obtain a further data provider comprises an act of evaluating a further data construction statement contained in the data provider to create the further data provider.

10. At a computer system including a query evaluator (101), a method for responding to a query (121), the method comprising:

an act of receiving a property value access query (121), the property value access query (121) querying for a property value of a property identified in a previously received data construction statement;

an act of identifying a data provider (171) created from the previously received data construction statement based on data contained in the property value access query (121), the data provider (171) virtualizing data access so that properties of an object described by the data construction statement can be accessed without having to evaluate the entire data construction statement or fully construct the entire object;

an act of placing the identified data provider (171) in a known location associated with the query evaluator (101);

an act of creating a data consumer (106) for the property value access query (121);

an act of the data consumer (106) accessing the identified data provider (171) from the known location (104);

an act of the data consumer (106) requesting (153A) the property value of the identified property from the data provider (171);

an act of the data consumer (106) receiving the property value (108) of the identified property from the data provider (151) ;

an act of the data consumer (106) placing the received property value (108) in the known location (104) for further processing.

11. The method as recited in claim 10, wherein the act of the data consumer receiving the property value of the identified property from the data provider comprises an act of the data consumer receiving the actual property value of the property from the data provider.

12. The method as recited in claim 10, wherein the act of receiving the actual property value of the property from the data provider comprises an act of receiving the actual property value of the property without having to evaluate query expressions for other properties described by the previously received data construction statement.

13. The method as recited in claim 10, wherein the act of the data consumer receiving the property value of the identified property from the data provider comprises an act of the data consumer receiving a further data provider from the data provider, the further data provider virtualizing the property value from the data provider.

14. The method as recited in claim 10, wherein the act of the data consumer placing the received property value in the known location for further processing comprises an act of the data consumer placing the actual property value of the property in the known location.

15. The method as recited in claim 10, wherein the act of the data consumer placing the obtained property value in the known location for further processing comprises an act of placing a further data provider in the known location, the further data provider virtualizing the property value from the data provider.

16. The method as recited in claim 15, further comprising:

an act of creating a further data consumer for the property value access query;

an act of the further data consumer accessing the further data provider from the known location;

an act of the further data consumer requesting the property value of the identified property from the further data provider;

an act of the data consumer receiving the property value of the identified property from the data further provider;

an act of the data consumer placing the received property value in the known location for further processing.

17. The method as recited in claim 16, wherein the act of receiving the property value of the property comprises an act of receiving the actual property value of the property without having refer to a data repository to evaluate query expressions for properties described by the previously received data construction statement and without having to evaluate query expressions for other properties described by the further data construction statement.

18. At a computer system including a query evaluator (101), a method for virtualizing data to facilitate more efficient queries, the method comprising:

an act of receiving a data construction statement (151), the data construction statement including:

an indication of the type of object (152) that the data construction statement (151) represents the construction of;

one or more property identifiers (153), each property identifier identifying a property of the object (161); and

one or more corresponding mappings (154), each corresponding mapping mapping a property identifier to a query expression (156), the query expression indicating how to compute the value for the property of the object identified by the property identifier; and

an act of creating an initial data provider (171) in response to receiving the data construction statement (151), the data provider (171) virtualizing data access so that properties of the object (161) described by the data construction statement (151) can be accessed without having to evaluate the entire data construction statement (151) or fully construct the entire object (161), the data provider (171)including:

a reference to the data construction statement (151R), the reference transitively including references to the type, property identifiers, and corresponding mappings of the data construction statement.

an act of receiving a property value access query (121), the property value access query (121) including a property identifier from the data construction statement;

an act of identifying the initial data provider (171) for responding to the property value access query (121) based on the property identifier;

an act of placing the initial data provider (171) in a known location (104) associated with the query evaluator (101);

an act of creating a data consumer (106) for the property value access query (121);

an act of the data consumer (106) accessing the initial data provider (171) from the known location (104);

an act of the data consumer requesting (153A) the property value of the identified property from the initial data provider (171);

an act of the initial data provider (171) looking up a query expression (156A) corresponding to the identified property from the previously received data construction statement (151);

an act of evaluating the query expression (156A) to obtain a property value (108) for the property;

an act of the initial data provider (171) returning the obtained property value (108) to the data consumer (106);

an act of the data consumer (106) receiving the obtained property value (108); and

an act of the data consumer (106) placing the received property value (108) in the known location (104) for further processing.

19. The method as recited in claim 18, wherein the act of evaluating the query expression to obtain a property value for the property comprises an act of accessing data from a database.

20. The method as recited in claim 18, wherein the act of evaluating the query expression to obtain a property value for the property comprises an act of constructing a further data provider.

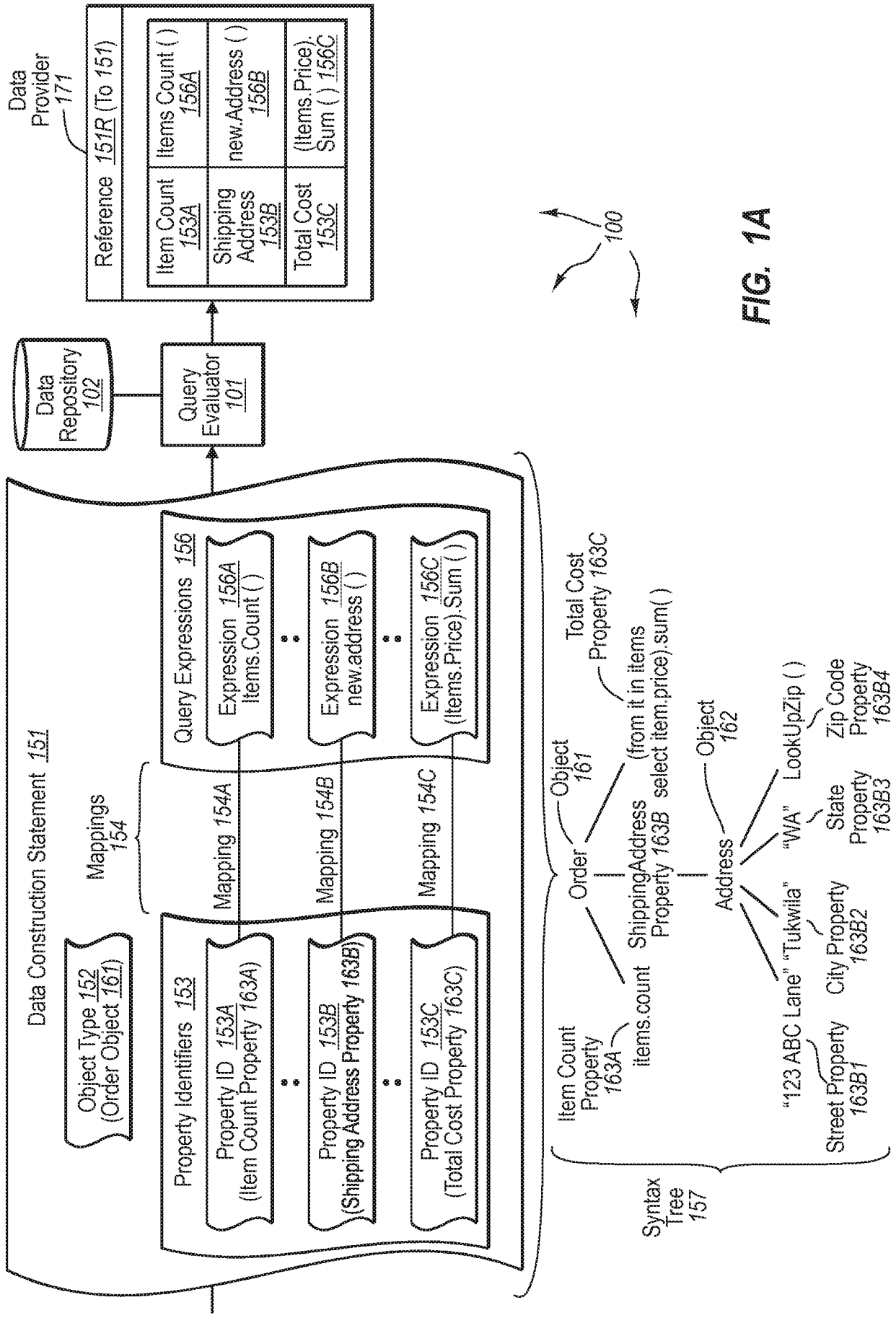

Data Provider 171
Reference 151R (To 151)
Item Count 153A
Shipping Address 153B
Total Cost 153C
Items Count ( ) 156A
new.Address ( ) 156B
(Items.Price). Sum ( ) 156C
Data Repository 102
Query Evaluator 101
100
Data Construction Statement 151
Object Type 152 (Order Object 161)
Mappings 154
Property Identifiers 153
Property ID 153A (Item Count Property 163A)
Property ID 153B (Shipping Address Property 163B)
Property ID 153C (Total Cost Property 163C)
Mapping 154A
Mapping 154B
Mapping 154C
Query Expressions 156
Expression 156A Items.Count ( )
Expression 156B new.address ( )
Expression 156C (Items.Price).Sum ( )
Syntax Tree 157
Object 161
Order
Item Count Property 163A
items.count
ShippingAddress Property 163B
Total Cost Property 163C
(from it in items select item.price).sum( )
Object 162
Address
"123 ABC Lane"
Street Property 163B1
"Tukwila"
City Property 163B2
"WA"
State Property 163B3
LookUpZip ( )
Zip Code Property 163B4

FIG. 1A

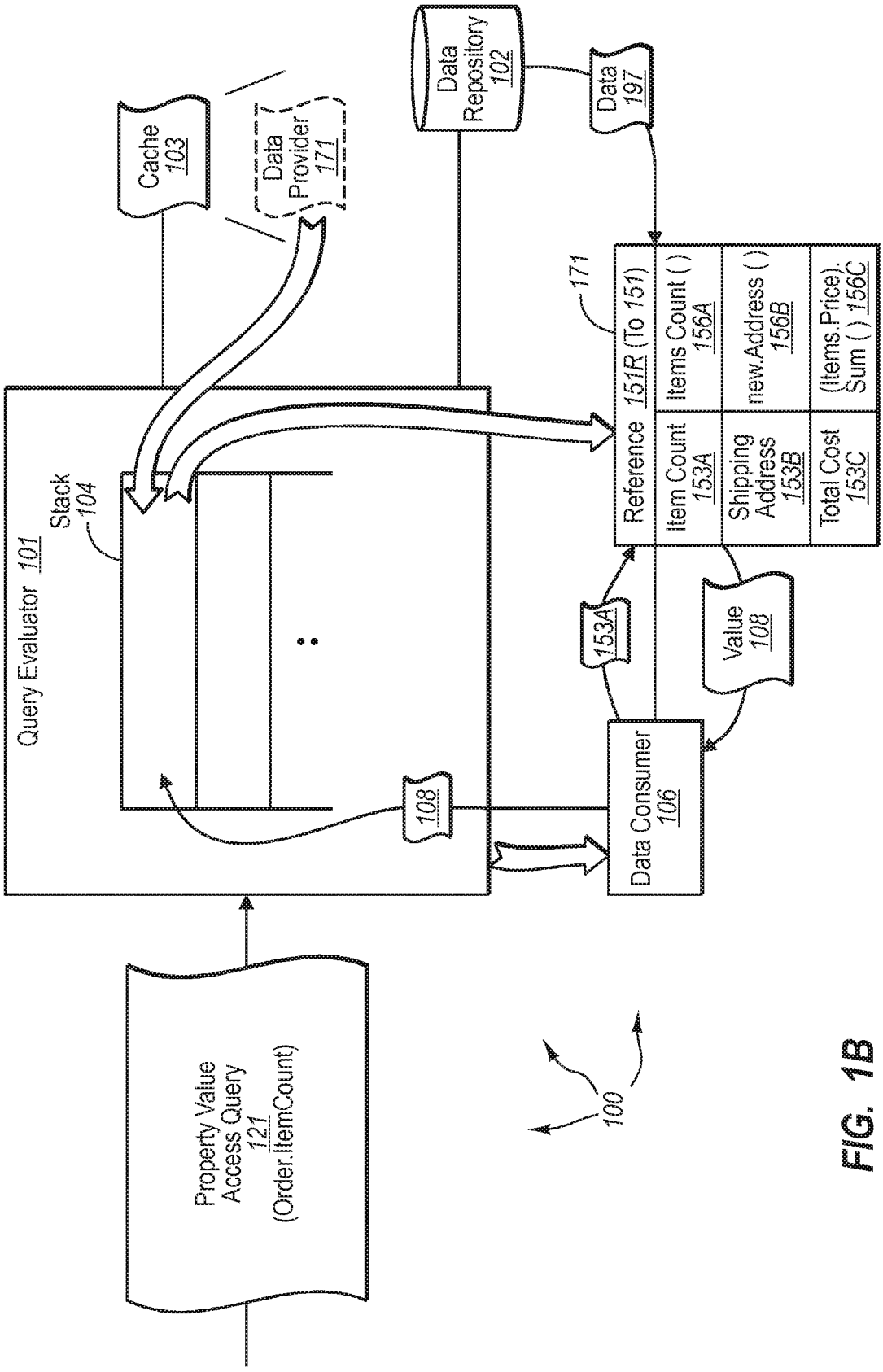

Property Value Access Query 121 (Order.ItemCount)
Query Evaluator 101
Stack 104
Cache 103
Data Provider 171
Data Repository 102
Data 197
171
Reference 151R (To 151)
Item Count 153A
Shipping Address 153B
Total Cost 153C
Items Count ( ) 156A
new.Address ( ) 156B
(Items.Price). Sum ( ) 156C
153A
Value 108
108
Data Consumer 106
100

FIG. 1B

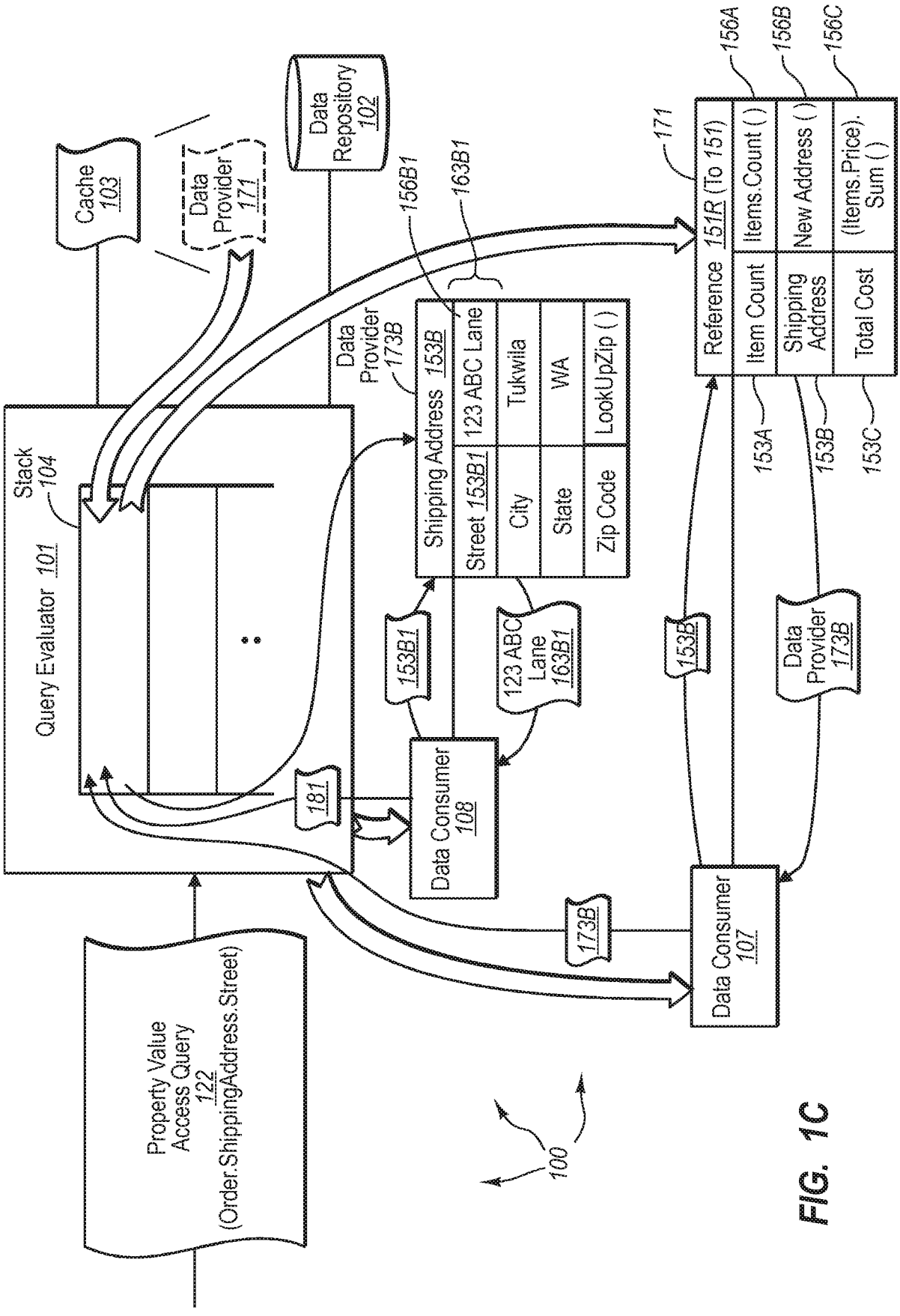

Property Value Access Query 122 (Order.ShippingAddress.Street)
Query Evaluator 101
Stack 104
Cache 103
Data Provider 171
Data Repository 102
Data Provider 173B
Shipping Address 153B
Street 153B1
City
State
Zip Code
123 ABC Lane
Tukwila
WA
LookUpZip ( )
156B1
163B1
171
Reference 151R (To 151)
Item Count
Shipping Address
Total Cost
Items.Count ( )
New Address ( )
(Items.Price). Sum ( )
156A
156B
156C
153A
153B
153C
153B1
123 ABC Lane 163B1
181
Data Consumer 108
Data Consumer 107
173B
153B
Data Provider 173B
100

FIG. 1C

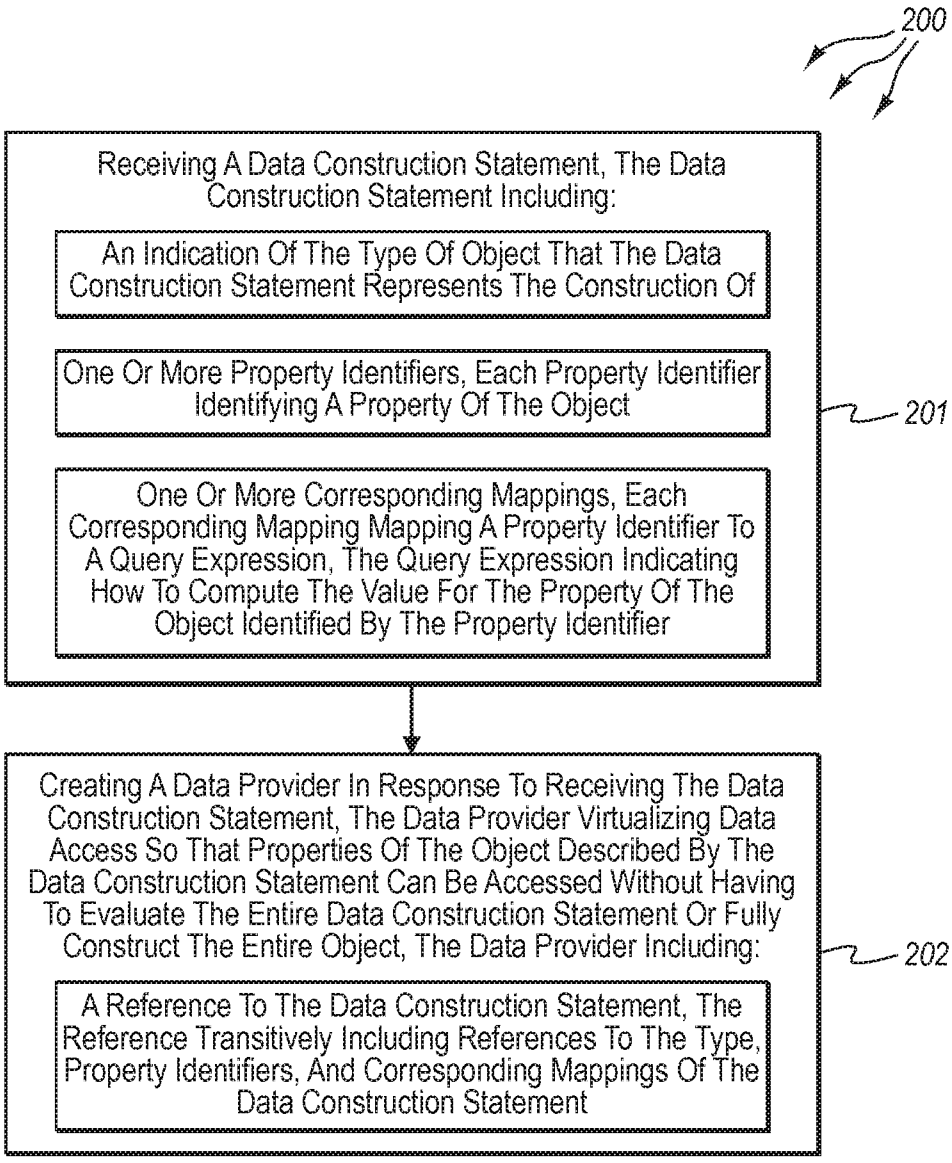

200
Receiving A Data Construction Statement, The Data Construction Statement Including:
An Indication Of The Type Of Object That The Data Construction Statement Represents The Construction Of
One Or More Property Identifiers, Each Property Identifier Identifying A Property Of The Object
One Or More Corresponding Mappings, Each Corresponding Mapping Mapping A Property Identifier To A Query Expression, The Query Expression Indicating How To Compute The Value For The Property Of The Object Identified By The Property Identifier
201
Creating A Data Provider In Response To Receiving The Data Construction Statement, The Data Provider Virtualizing Data Access So That Properties Of The Object Described By The Data Construction Statement Can Be Accessed Without Having To Evaluate The Entire Data Construction Statement Or Fully Construct The Entire Object, The Data Provider Including:
A Reference To The Data Construction Statement, The Reference Transitively Including References To The Type, Property Identifiers, And Corresponding Mappings Of The Data Construction Statement
202

FIG. 2

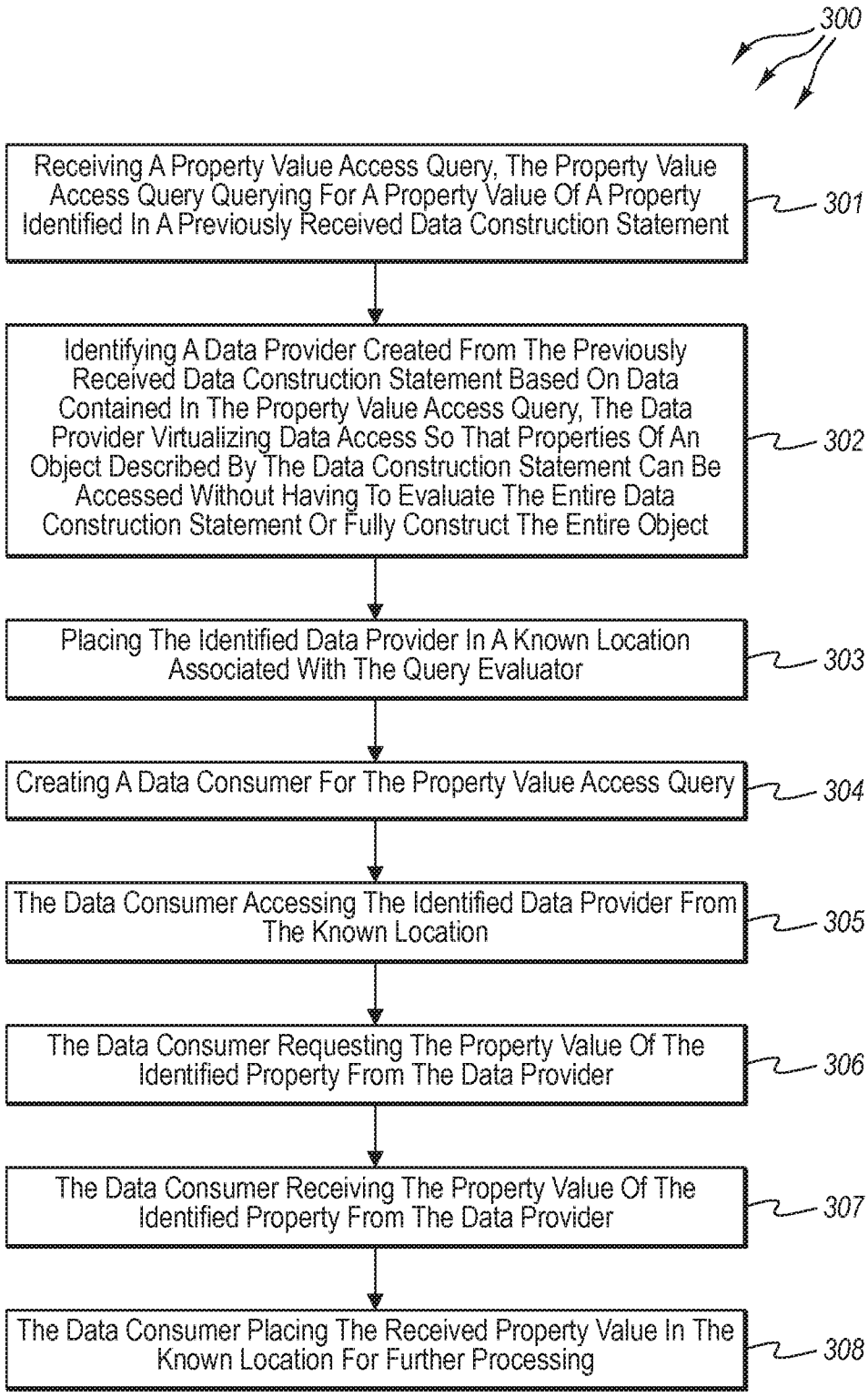

300
Receiving A Property Value Access Query, The Property Value Access Query Querying For A Property Value Of A Property Identified In A Previously Received Data Construction Statement
301
Identifying A Data Provider Created From The Previously Received Data Construction Statement Based On Data Contained In The Property Value Access Query, The Data Provider Virtualizing Data Access So That Properties Of An Object Described By The Data Construction Statement Can Be Accessed Without Having To Evaluate The Entire Data Construction Statement Or Fully Construct The Entire Object
302
Placing The Identified Data Provider In A Known Location Associated With The Query Evaluator
303
Creating A Data Consumer For The Property Value Access Query
304
The Data Consumer Accessing The Identified Data Provider From The Known Location
305
The Data Consumer Requesting The Property Value Of The Identified Property From The Data Provider
306
The Data Consumer Receiving The Property Value Of The Identified Property From The Data Provider
307
The Data Consumer Placing The Received Property Value In The Known Location For Further Processing
308

FIG. 3

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2009/044298

A. CLASSIFICATION OF SUBJECT MATTER
INV. B65G47/08

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)
B65G

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	EP 0 007 231 A (MITSUBISHI HEAVY IND LTD [JP]) 23 January 1980 (1980-01-23) page 7, paragraph 2 - page 9, paragraph 4; figure 6	1,12
A	US 6 843 360 B2 (PETERMAN THOMAS HAROLD [US] ET AL) 18 January 2005 (2005-01-18) cited in the application column 3, paragraph 3 - column 8, paragraph 4; figure 1	1,12

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

☒ See patent family annex.

* 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 document 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	Date of mailing of the international search report
29 July 2009	31/08/2009

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL – 2280 HV Rijswijk Tel. (+31–70) 340–2040, Fax: (+31–70) 340–3016	Schneider, Emmanuel

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2009/044298

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
EP 0007231 A	23-01-1980	AU 534221 B2	12-01-1984
		AU 4834379 A	24-01-1980
		DE 2962230 D1	01-04-1982
		JP 1329777 C	30-07-1986
		JP 55016818 A	05-02-1980
		JP 60054217 B	29-11-1985
		US 4238024 A	09-12-1980
US 6843360 B2	18-01-2005	AT 353842 T	15-03-2007
		AU 2003220542 A1	13-10-2003
		DE 60311778 T2	31-10-2007
		EP 1490280 A1	29-12-2004
		ES 2281637 T3	01-10-2007
		MX PA04009400 A	05-07-2005
		WO 03082712 A1	09-10-2003
		US 2005121289 A1	09-06-2005
		US 2003183484 A1	02-10-2003