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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 April 2008 (17.04.2008)

PCT

(10) International Publication Number
WO 2008/045634 A1

(51) International Patent Classification:
G06Q 10/00 (2006.01) *G06F 17/30* (2006.01)
G06Q 99/00 (2006.01)

(21) International Application Number:
PCT/US2007/077272

(22) International Filing Date: 30 August 2007 (30.08.2007)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/539,971 10 October 2006 (10.10.2006) US

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

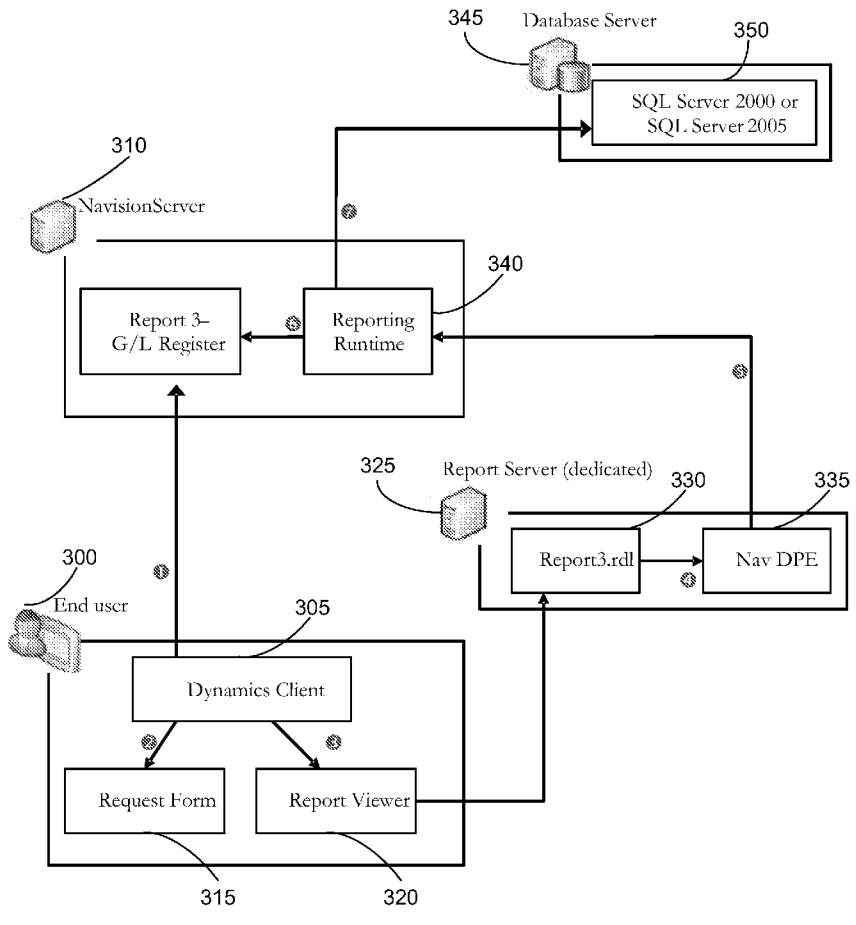
(72) Inventors: **ANDERSEN, Claus Busk**; c/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **CHRISTIANSEN, Lars-Bo**; c/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **LISOVAYA, Julia**; c/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US). **KIZILTUNC, Mehmet Kerem**; c/o Microsoft Corporation International Patents, One Microsoft Way, Redmond, Washington 98052-6399 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, 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, 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,

[Continued on next page]

(54) Title: INTEGRATION OF DATABASE REPORTING WITH ERP SYSTEMS



(57) Abstract: Increased reporting capabilities from a database system may be available in a customer relationship system while functionality of the customer relationship management system is maintained.

WO 2008/045634 A1

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, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, 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*

INTEGRATION OF DATABASE REPORTING WITH ERP SYSTEMS

Background

[0001] Enterprise resource planning ("ERP") systems have made tracking resources and planning resource use easier. These interactive systems provide immediate information about resources which may be used to better service customers. In addition, the ERP system may store data in a different manner than many systems, such as storing actual total amounts rather than storing formulas to calculate total amounts. ERP systems have report generating abilities that make it easy to visualize the data in the ERP system.

[0002] Database systems have been around even longer than ERP systems. Database systems store large amounts of data in a way that is easy to search and manipulate. Data is usually stored as individual entries and totals usually do not have their own entry but are calculated as needed by adding up the individual elements from the database. As database systems have been around for a significant period of time, the ability to create reports is refined.

Summary

[0003] A enterprise resource planning system may be used in conjunction with a database system to create reports. A report object may be used that contains layout information, a report subclass and metadata. The metadata may be used to determine what data should be accessed from a database, how it should be formatted and how it should be displayed. As a result, increased report capabilities from the database system may be available in the enterprise resource planning system while functionality of the enterprise resource planning system is maintained.

Figures

[0004] Fig. 1 illustrates a method of using a ERP system to access the report capabilities of a database system;

[0005] Fig. 2 is an illustration of one embodiment of the report object;

[0006] Fig. 3 is a high level review of the creation of a report; and

[0007] Fig. 4 is a specific illustration of the construction of a report.

Description

[0008] An enterprise resource planning system ("ERP") may be able to create some reports, but additional report creating abilities may be created by allowing the ERP system to use report functions from a database system. One method to allow a ERP system to access a database system is described herein. A sample ERP system may be Dynamics Nav by Microsoft Corporation and a sample database program may be SQL Reporting Services by Microsoft Corporation, but the principles of the method and apparatus may be application to virtually any ERP system and any database system. Fig. 1 may illustrate a method of using a ERP system to access the report capabilities of a database system.

[0009] Referring to Fig. 1, at block 100, the method may select to run a selected report. A user or other program may define desired data items for a report. Data items may be a table with a key, sort order, filter criteria and a join/link condition with another data item such that multiple tables may be used in a report. The user may then pick the fields from the data item that he wants to see on the report. Next, the layout of the report may be selected. A layout designer may be to select that layout of the fields from the data items selected. The report may be designed using an addition program module, such as Visual Studio by Microsoft Corporation. The user may then save the report and continue with other tasks like defining a request form, and calling the report from a menu item or from other code.

[0010] At block 110, the method may create a report object based on selected report. Creating a report may include creating a layout description file using a layout generator, creating a subclass that contains report specific variables and triggers using a code generator and creating a report metadata file using a metadata generator.

[0011] Fig. 2 may be an illustration of one manner in which the report object 200 is created. A layout generator 210 may be used to create the layout which is report definition language client ("RDLC") file 220 or any other layout file.

[0012] Also, a code generator 230 such as a C# Code Generator may be used to create a subclass for the report, such as a NavReport subclass. For each report, code for a subclass may be auto-generated. The code may be in a language such as C#. The code generation may basically follow the auto-code generation rules of other

objects like tables, forms, and XML ports, etc. The subclass may contain report specific variables and implementations of the report and data item triggers.

[0013] Sample C# code may be as follows:

```
[0014] public class Report3 : NavReport
[0015]  {
[0016]   private Table45 GLRegister;
[0017]   private Table17 GLEntry;
[0018]   private Table15 GLAcc;
[0019]   private NavText GLRegFilter;
[0020]   private void InitializeComponent()
[0021]   {
[0022]   }
[0023]   protected override void OnInitReport()
[0024]   {
[0025]    GLRegFilter = GLRegister.GetFilters();
[0026]   }
[0027]  }
```

[0028] The report object 200 also may use a metadata generator 250 to create a metadata report 250. The metadata report 250 may be in XML, for example, or in any other suitable language. The metadata file may simply be an XML document that describes the basics of a report. It may contain:

[0029] • The report name, captions, permissions, transaction mode, etc.

[0030] • The data items used in the report, with table views and links between the data items; and

[0031] • The fields of each data item.

[0032] A sample metadata file may be as follows:

[0033]

```
<?xml version="1.0" encoding="utf-8" ?>
```

[0034] <Report>

[0035] <Id>108</Id>

[0036] <Name>Customer - Order Detail</Name>

[0037] <DataItems>

[0038] <DataItem>

[0039] <DataItemIndent>0</DataItemIndent>

[0040] <DataItemTable>Customer</DataItemTable>

[0041] <DataItemVariableName></DataItemVariableName>

[0042] <Fields>

[0043] <FieldName>No.</FieldName>

[0044] <FieldType>Code</FieldType>

[0045] </Fields>

[0046] <ReqFilterFields>No.,Search Name,Priority</ReqFilterFields>

[0047] <PrintOnlyIfDetail>Yes</PrintOnlyIfDetail>

[0048] </DataItem>

[0049] Report metadata can be seen as an XML version of the report properties. This dataset and layout are then presented to SQL Reporting Services for rendering the report for preview or printing purposes.

[0050] The reporting runtime may be a .NET implementation that is basically built around an object such as NavReport, and its subtypes. This runtime may be responsible for the following:

[0051] • Integrating with the rest of the runtime environment;

[0052] • Based on data items, accessing data, sorting, filtering records to be accessed;

[0053] • Executing application code (which may be transformed from application language into C#);

[0054] • Implementing specific application security and indirect security concepts from the ERP system;

[0055] • Supporting transaction model in the report properties; and

[0056] • Binding report parameters and request forms together.

[0057] At block 120, the method may review metadata of the report object to determine whether a request form for a database system is needed. If the report needs to display a request form, the report metadata may note this. If a request form is needed, an event may be raised which may be caught. Then, communication may begin to start a communication with the client to display the request form. At this point, the report object may be instantiated. The method is designed such that the code-behind functionality between the request form and the report object instance may be supported, so when the user makes changes on a request form, the report variables get updated properly.

[0058] A sample schema for the metadata may be as follows:

[0059] <?xml version="1.0" encoding="utf-8"?>

[0060] <xs:schema targetNamespace="http://www.microsoft.com/Dynamics/Nav/Metadata/Report.xsd" elementFormDefault="qualified" xmlns="http://www.microsoft.com/Dynamics/Nav/Metadata/Report.xsd" xmlns:mstns="http://www.microsoft.com/Dynamics/Nav/Metadata/Report.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema">

[0061] <xs:element name="Report">

[0062] <xs:complexType>

[0063] <xs:sequence>

[0064] <xs:element name="Id" type="xs:string" minOccurs="1" maxOccurs="1"/>

[0065] <xs:element name="Name" type="xs:string" minOccurs="1" maxOccurs="1"/>

[0066] <xs:element name="Caption" type="xs:string" minOccurs="1" maxOccurs="1"/>

[0067] <xs:element name="CaptionML" type="xs:string" minOccurs="1" maxOccurs="1"/>

[0068] <xs:element name="ShowPrintStatus" type="xs:boolean" minOccurs="1" maxOccurs="1"/>

[0069] <xs:element name="UseRequestForm" type="xs:boolean" minOccurs="1" maxOccurs="1"/>

[0070] <xs:element name="ProcessingOnly" type="xs:boolean" minOccurs="1" maxOccurs="1"/>

[0071] <xs:element name="UseSystemPrinter" type="xs:boolean" minOccurs="1" maxOccurs="1"/>

[0072] <xs:element name="TransactionType" minOccurs="1" maxOccurs="1">

[0073] <xs:simpleType>

[0074] <xs:restriction base="xs:string">

[0075] <xs:enumeration value="UpdateNoLocks" />

[0076] <xs:enumeration value="Update" />

[0077] <xs:enumeration value="Snapshot" />

[0078] <xs:enumeration value="Browse" />

[0079] <xs:enumeration value="Report" />

[0080] </xs:restriction>

[0081] </xs:simpleType>

[0082] </xs:element>

[0083] <xs:element name="Description" type="xs:string" minOccurs="1" maxOccurs="1"/>

[0084] <xs:element name="Permissions" type="xs:string" minOccurs="1" maxOccurs="1"/>

[0085] <xs:sequence>

[0086] <xs:element ref="DataItem" minOccurs="1" maxOccurs="unbounded"/>

[0087] </xs:sequence>

[0088] </xs:sequence>

[0089] </xs:complexType>

[0090] </xs:element>

[0091] <xs:element name="DataItem">

[0092] <xs:complexType>

[0093] <xs:sequence>

[0094] <xs:element name ="DataItemIndent" type ="xs:integer" minOccurs="1" maxOccurs="1"/>

[0095] <xs:element name ="DataItemTable" type ="xs:integer" minOccurs="1" maxOccurs="1"/>

[0096] <xs:element name ="DataItemTableView" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[0097] <xs:element name ="DataItemLinkReference" type ="xs:integer" minOccurs="1" maxOccurs="1"/>

[0098] <xs:element name ="DataItemLink" type ="xs:integer" minOccurs="1" maxOccurs="1"/>

[0099] <xs:element name ="RequestFilterHeading" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[00100] <xs:element name ="RequestFilterHeadingML" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[00101] <xs:element name ="RequestFilterFields" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[00102] <xs:element name ="MaxIteration" type ="xs:integer" minOccurs="1" maxOccurs="1"/>

[00103] <xs:element name ="DataItemVariableName" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[00104] <xs:element name ="PrintOnlyIfDetail" type ="xs:boolean" minOccurs="1" maxOccurs="1"/>

[00105] <xs:element ref="DataItemField" minOccurs="1" maxOccurs="unbounded"/>

[00106] </xs:sequence>

[00107] </xs:complexType>

[00108] </xs:element>

[00109] <xs:element name="DataItemField">

[00110] <xs:complexType>

[00111] <xs:sequence>

[00112] <xs:element name ="Name" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[00113] <xs:element name ="DataType" type="xs:boolean" minOccurs="1" maxOccurs="1"/>

[00114] <xs:element name ="SourceExpression" type ="xs:string" minOccurs="1" maxOccurs="1"/>

[00115] <xs:element name ="AutoCalcField" type="xs:boolean" minOccurs="1" maxOccurs="1"/>

[00116] </xs:sequence>

[00117] </xs:complexType>

[00118] At block 130, if the request form is not needed, the report is created as usual.

[00119] At block 140, if the request form is needed, the request form is created using the metadata of the report object for the database system. At the time of compilation or during import, two artifacts may be generated: Report metadata, which is an XML document that may be saved into an object metadata table as with any other metadata document, and a NavReport subclass, specific to the report being compiled. If the report has a request form, those also may be transformed into the corresponding artifacts as well.

[00120] Report metadata can be seen as an XML version of the report properties. It contains enough information for the reporting runtime to generate a report dataset if the report does not contain any code. In other words, reporting runtime should be able to take a Report Metadata and a NavReport subclass that has no event handlers or local variables of its own, and then generate a report dataset.

[00121] At block 150, a filter may be selected for the selected report. The filter may be based on data items and may be used to filter query results. Filters are well known and any logical filter may be used.

[00122] At block 160, the method may determine tables in the database system needed using the metadata of the report object. The metadata may contain virtually any information. Depending on the database used, the metadata may vary. Sample metadata has been disclosed previously.

[00123] At block 170, the method may execute the database query on the database system to obtain report results. Database queries are well known. In short, criteria are used to obtain the desired data out of a large quantity of data in the database.

[00124] At block 180, the method may create a report using the report results from the database query. By using the ERP system, advanced report features may be available such as increased colors, option to create reports in PDF, etc.

[00125] Fig. 3 may be a high level review of the creation of a report. An end user 300 may select a menu item in a ERP system 305 such as Navision to run a report. The ERP system 305 may make a call to a ERP server such as the Navision Service Tier 310 to initialize a NavReport object and check its metadata to see if it requires a request form. If the report requires a request form 315, the ERP system 305 may build a request form based on the parameters on the report metadata, typically using request form 315 specific parameters on the data items. The user may enter a filter. The ERP client may communicate the changes back to the ERP server 310. The user may select on Preview such as a report viewer 320. The request may be communicated to a report server 325. The ERP system 305 may initiate the execution of the report query through a reporting runtime by communicating a file such as a data processing extension file 335 to the report runtime 340. Reporting runtime 340 may use the report metadata for detecting which tables in the database 345 to use, the report object to get the parameters set and to execute the code on triggers and

retrieves data from SQL 350. The user may retrieve the resulting data (DataSet) and the format file (RDLC) from the ERP service tier 310 and passes them to the Report Viewer 320, which renders and displays the report.

[00126] Fig. 4 may be a specific illustration of the construction of a report. In this example, Dynamics Nav is the ERP system used. At block 400, an end user may select to run a report 405 using the Dynamics client 410. A run report application may begin 415. In Fig. 5, the application may be RunReport(reportID). The application may call to a server to complete the request 420. In the example in Fig. 5, the method may call to a Navision server service 420. The Navision server service 420 may call a report run function 425 such as Run(reportID). This request may be forwarded to a specific report running function 425 such as NavReport 430. The report running function 425 may call for a new report 435 using the specific request data 440, such as Report 3, Instance42 in Fig. 4. The report 435 may initialize 445 the needed components and the BeginInitialization function 450 may begin. A request for a new data item may be made 455. Metadata for the requested report 460 may be stored in a table and the metadata may be used to determine the data to be retrieved 465 from the database 470.

[00127] For each data item 475 retrieved from the database 470, a new record 480 may be created and the data item 485 may be registered for an event 490. The data item 485 may then be added to the report 495. At block 497, the initialization may end and the report 435 may be ready.

[00128] The following may be psuedocode implementing the above example.

[00129] public class Report3 : NavReport

[00130] {

[00131] public Report3()

[00132] : base(3)

[00133] {

[00134] InitializeComponent();

[00135] }

[00136] public Report3(NavConnection connection)

[00137] : base(connection, 3)

[00138] {

[00139] InitializeComponent();

[00140] }

[00141] private void InitializeComponent()

[00142] {

[00143] BeginInitialization();

[00144] glEntry = new Table17();

[00145] glRegister = new Table45();

[00146] glAcc = new Table15();

[00147] glRegFilter = new NavText();

[00148] /* Instantiate DataItems and register their event handlers. */

[00149] glEntryDataItem = new NavDataItem(glEntry);

[00150] Add(glEntryDataItem);

[00151] glEntryDataItem.OnAfterGetRecord = glEntryDataItem_OnAfterGetRecord;

[00152] glEntryDataItem.OnPreDataItem = glEntryDataItem_OnPreDataItem;

[00153] glRegisterDataItem = new NavDataItem(glRegister);

[00154] Add(glRegisterDataItem);

[00155] EndInitialization();

[00156] }

[00157] void glEntryDataItem_OnPreDataItem()

[00158] {

[00159] glEntry.SetRange("Entry No.", glRegister.FromEntryNo, glRegister.ToEntryNo);

[00160] }

[00161] void glEntryDataItem_OnAfterGetRecord()

```
[00162]  {
[00163]   if (!glAcc.Get(DataError.ThrowError, "G/L Account No."))
[00164]   {
[00165]    glAcc.Init();
[00166]   }
[00167]  }
[00168]  protected override void OnInitReport()
[00169]  {
[00170]   glRegFilter = glRegister.GetFilters();
[00171]  }
[00172]  private Table17 glEntry;
[00173]  private Table45 glRegister;
[00174]  private NavDataItem glEntryDataItem;
[00175]  private NavDataItem glRegisterDataItem;
[00176]  private Table15 glAcc;
[00177]  private NavText glRegFilter;
[00178] }
```

[00179] Other advantages to the method may be available. By keeping the ERP system functional and not just running reports from the SQL system, items that are part of the ERP system may be available as part of the report. For example, ERP systems often have "flowfields" or fields that are actually summations of a plurality of other fields. In a traditional SQL reporting system, each of the individual fields may have to be selected and added before being added to a report. Using the flowfield concept from ERP, only one entry needs to be selected.

[00180] In addition, by keeping the ERP system as the interface into the database system, an extra layer of protection is added to the database system. As the data in the database is likely to be sensitive and unauthorized editing of the data could be

devastating, allowing access to the database only through the ERP system may keep users further removed from the underlying data.

[00181] Additionally, ERP systems allow users to design a single report to work with multiple companies. For example, when a user designs a report, it may be designed generically against a table, and it may be run against any company. By accessing a plurality of different databases such as databases specific to each company, the generic report will be company specific.

[00182] Another novel aspect is that it is possible to execute additional business logic by calling other application objects like tables, or codeunits to make calculations while generating the report. For example, a tax amount may be calculated on the fly and this tax amount would not be stored in a database alone. As the business logic resides in the service tier and not in the database, a database reporting system may be unable to access a calculated tax amount (assuming it is not stored in the database as a separate entry).

[00183] Although the forgoing text sets forth a detailed description of numerous different embodiments, it should be understood that the scope of the patent is defined by the words of the claims set forth at the end of this patent. The detailed description is to be construed as exemplary only and does not describe every possible embodiment because describing every possible embodiment would be impractical, if not impossible. Numerous alternative embodiments could be implemented, using either current technology or technology developed after the filing date of this patent, which would still fall within the scope of the claims.

[00184] Thus, many modifications and variations may be made in the techniques and structures described and illustrated herein without departing from the spirit and scope of the present claims. Accordingly, it should be understood that the methods and apparatus described herein are illustrative only and are not limiting upon the scope of the claims.

Claims:

1. A method of a customer relationship management system using a database system to create a report comprising:

selecting to run a selected report 100,

creating a report object based on selected report 110;

reviewing metadata of the report object to determine whether a request form for a database system 120;

if the request form is not needed, creating the selected report 130;

if the request form is needed, creating the request form using the metadata of the report object for the database system 140;

selecting a filter for the selected report 150;

determining tables in the database system needed using the metadata of the report object 160;

executing the database query on the database system to obtain report results 170;

creating the selected report using the report results from the database query 180.

2. The method of claim 1, wherein creating a report object further comprises:

creating a layout description file 220;

creating a subclass that contains report specific variables and triggers 240; and

creating a report metadata file 260.

3. The method of claim 2, wherein the layout description file is report definition language client ("RDLC") file 220.

4. The method of claim 2, wherein the metadata file 260 is an XML file.

5. The method of claim 4, wherein the metadata file 260 contains at least one selected from a group comprising:

a report name,

a report caption,

a report permission,

a transaction mode;

a data items used in the report; and

fields of each data item.

6. The method of claim 5, wherein the data items used in the report include table views and links between the data items.

7. The method of claim 2, wherein creating the subclass comprises using a code generator to change the application language to C# 230.

8. The method of claim 1, wherein data is accessed from the database using a request form 315.

9. The method of claim 1, further comprising using a reporting runtime program to take report metadata and a subclass to generate a report dataset 340.

10. A computer readable medium comprising computer executable code for creating reports in a customer relationship management system using a database system wherein the computer executable code comprises code for:

selecting to run a selected report 100,

creating a report object based on selected report 110;

reviewing metadata of the report object to determine whether a request form for a database system 120;

if the request form is not needed, creating the report 130;

if the request form is needed, creating the request form using the metadata of the report object for the database system 140;

selecting a filter for the selected report 150;

determining tables in the database system needed using the metadata of the report object 160;

executing the database query on the database system to obtain report results 170;

creating a report using the report results from the database query 180.

11. The computer readable medium of claim 10, wherein the computer executable code for creating a report object further comprises:

creating a layout description file 220;

creating a subclass that contains report specific variables and triggers 240; and

creating a report metadata file 260.

12 The computer readable medium of claim 11, wherein the layout description file is report definition language client ("RDLC") file 220.

13. The computer readable medium of claim 11, wherein the metadata file 260 is an XML file and the metadata file contains at least one selected from a group comprising:

a report name,

a report caption,

a report permission,

a transaction mode;

a data items used in the report; and

fields of each data item.

14. The computer readable medium of claim 11, wherein the data items used in the report include table views and links between the data items.

15. The computer readable medium of claim 11, wherein creating the subclass comprises using a code generator to change the application language in C# 230.

16. The computer readable medium of claim 11, further comprising using a reporting runtime program to take report metadata and a subclass to generate a report dataset 340.

17. A computer system comprising a processor for executing computer executable code, a memory for storing computer executable code and an input output circuit for communicating computer executable code, the computer executable code comprising code for:

selecting to run a selected report 100,

creating a report object based on selected report 110;

reviewing metadata of the report object to determine whether a request form for a database system 120;

if the request form is not needed, creating the report 130;

if the request form is needed, creating the request form using the metadata of the report object for the database system 140;

selecting a filter for the selected report 150;

determining tables in the database system needed using the metadata of the report object 160;

executing the database query on the database system to obtain report results 170;

creating a report using the report results from the database query 180.

18. The computer executable code of claim 17, wherein the computer executable code for creating a report object further comprises:

creating a layout description file 220;

creating a subclass that contains report specific variables and triggers 240; and

creating a report metadata file 260.

19 The computer executable code of claim 18, wherein the layout description file is report definition language client ("RDLC") file 220.

20. The computer executable code of claim 18, readable medium of claim 11, wherein the metadata 360 file is an XML file and the metadata file contains at least one selected from a group comprising:

a report name,

a report caption,

a report permission,

a transaction mode;

a data items used in the report; and

fields of each data item.

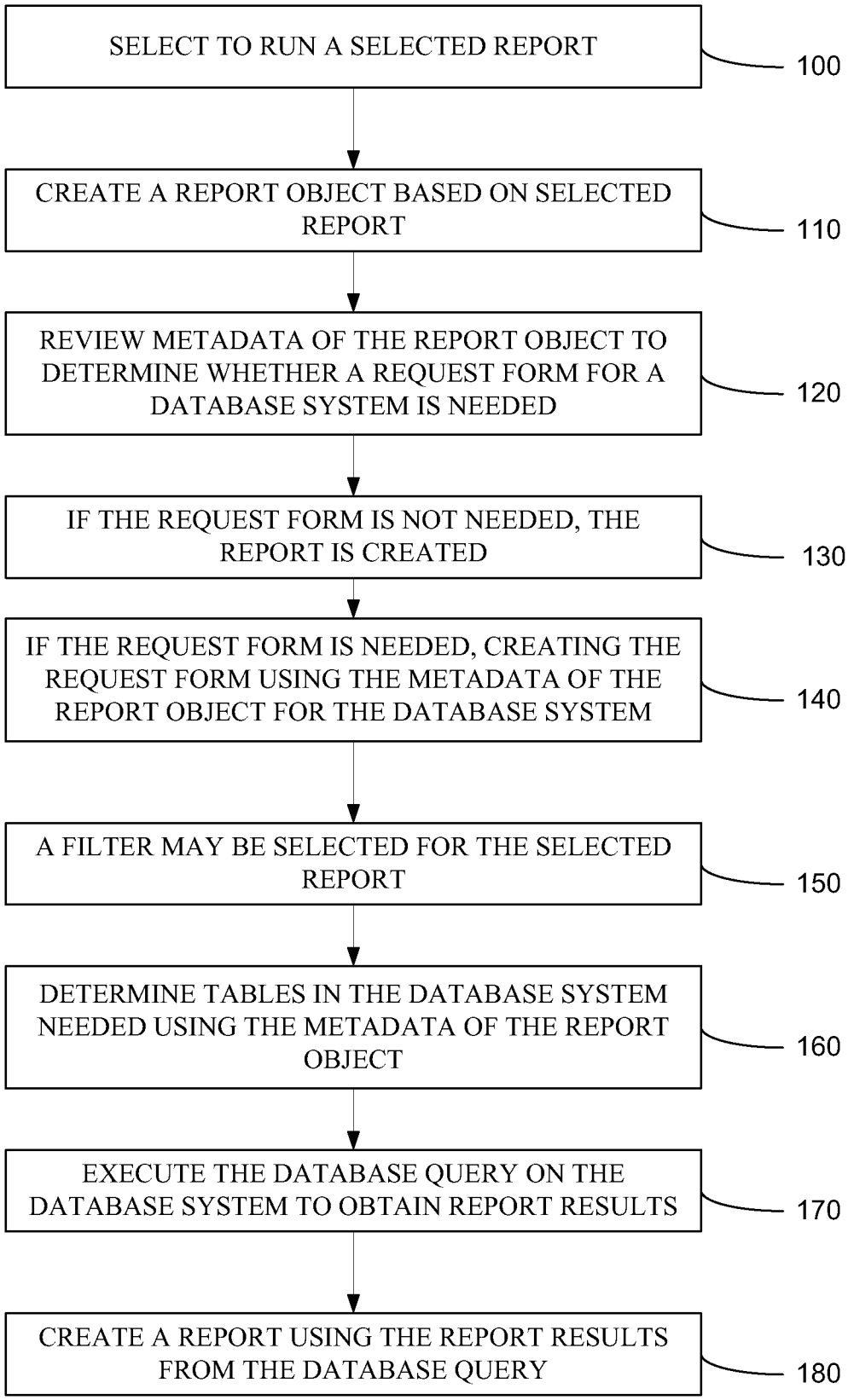

SELECT TO RUN A SELECTED REPORT
100
CREATE A REPORT OBJECT BASED ON SELECTED REPORT
110
REVIEW METADATA OF THE REPORT OBJECT TO DETERMINE WHETHER A REQUEST FORM FOR A DATABASE SYSTEM IS NEEDED
120
IF THE REQUEST FORM IS NOT NEEDED, THE REPORT IS CREATED
130
IF THE REQUEST FORM IS NEEDED, CREATING THE REQUEST FORM USING THE METADATA OF THE REPORT OBJECT FOR THE DATABASE SYSTEM
140
A FILTER MAY BE SELECTED FOR THE SELECTED REPORT
150
DETERMINE TABLES IN THE DATABASE SYSTEM NEEDED USING THE METADATA OF THE REPORT OBJECT
160
EXECUTE THE DATABASE QUERY ON THE DATABASE SYSTEM TO OBTAIN REPORT RESULTS
170
CREATE A REPORT USING THE REPORT RESULTS FROM THE DATABASE QUERY
180

FIG. 1

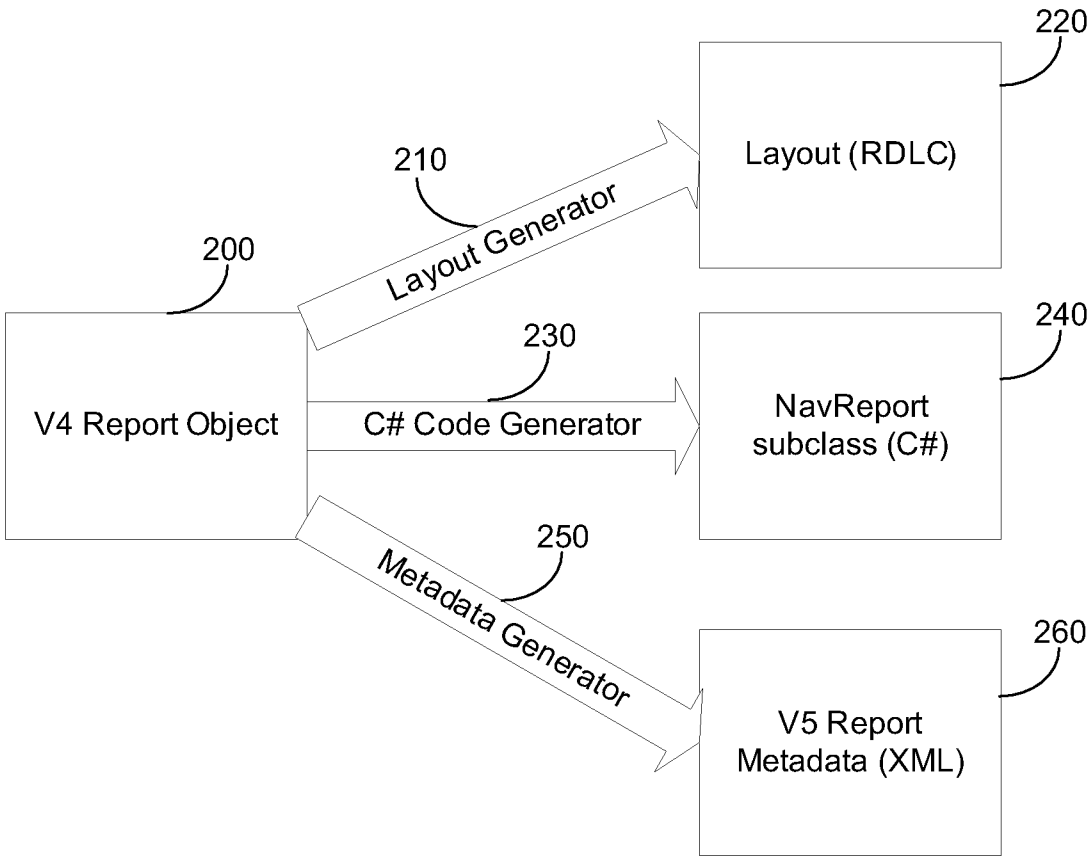

220
Layout (RDLC)
210
Layout Generator
200
V4 Report Object
230
C# Code Generator
240
NavReport
subclass (C#)
250
Metadata Generator
260
V5 Report
Metadata (XML)

FIG. 2

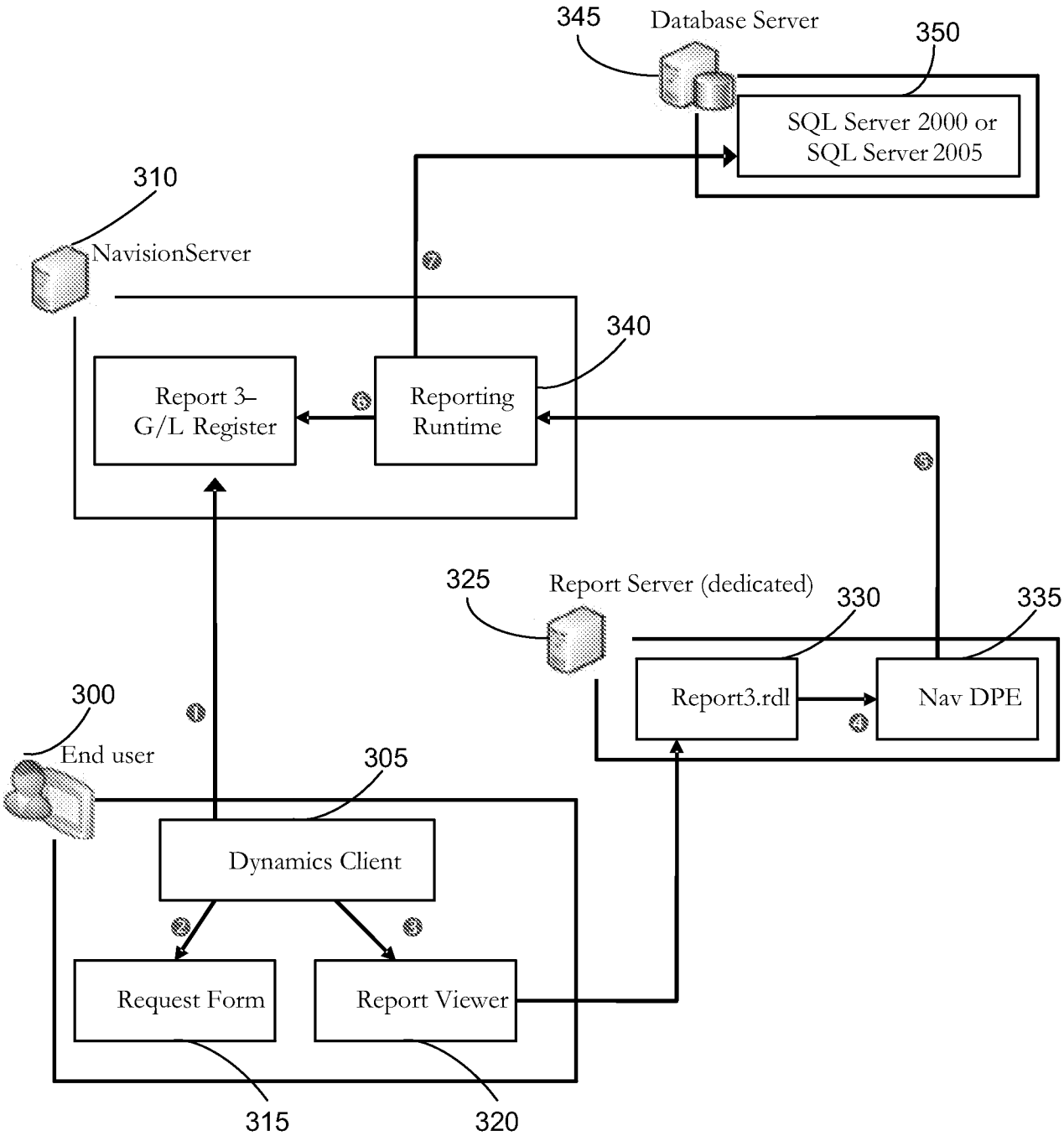

345
Database Server
350
SQL Server 2000 or
SQL Server 2005
310
NavisionServer
340
Report 3–
G/L Register
Reporting
Runtime
325
Report Server (dedicated)
330
335
Report3.rdl
Nav DPE
300
End user
305
Dynamics Client
Request Form
Report Viewer
315
320

FIG. 3

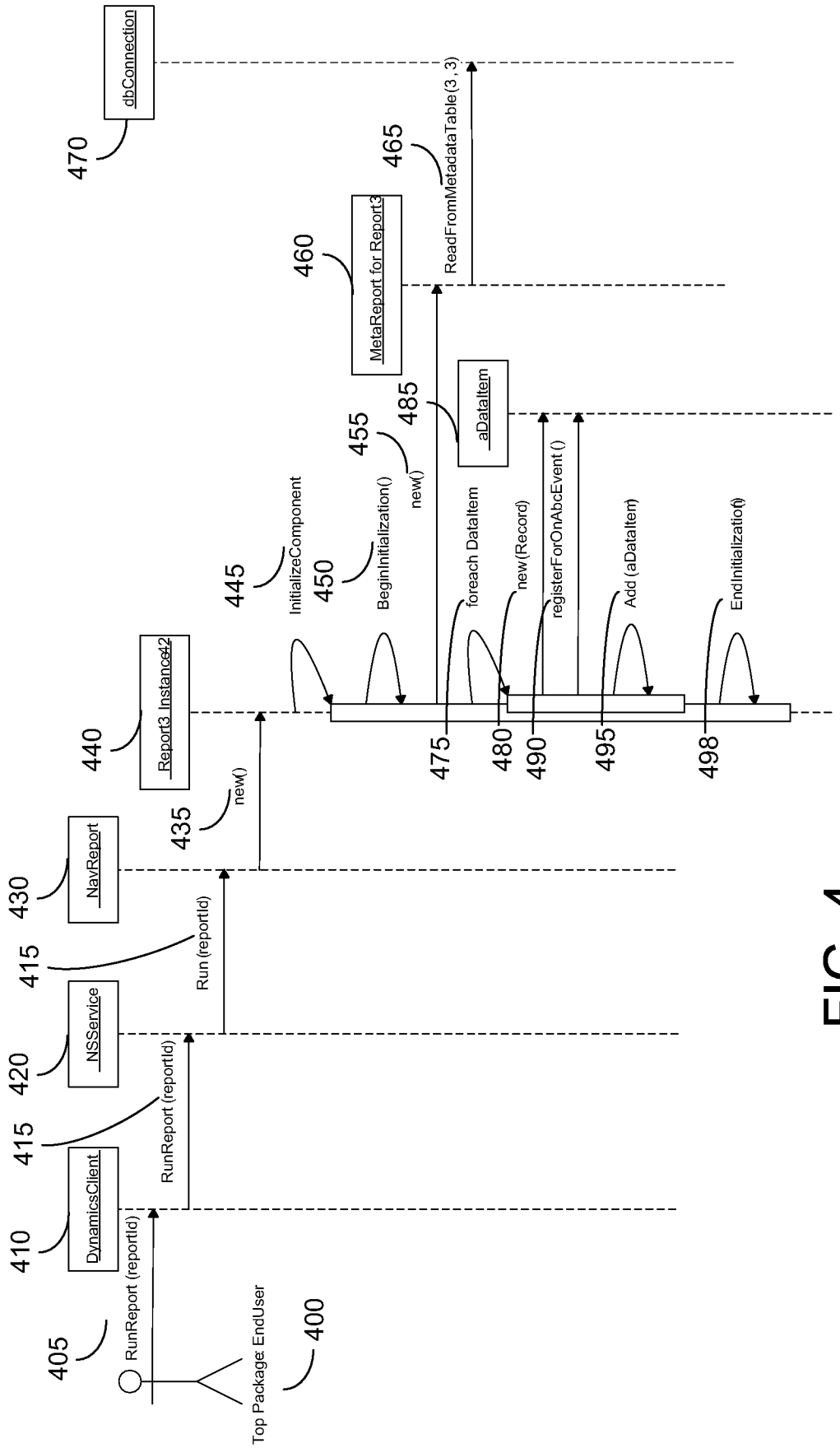

405
Top Package: EndUser
400
RunReport (reportId)
410
DynamicsClient
415
RunReport (reportId)
420
NSService
415
Run (reportId)
430
NavReport
435
new()
440
Report3 Instance#2
445
InitializeComponent
450
BeginInitialization()
455
new()
460
MetaReport for Report3
465
ReadFromMetadataTable(3, 3)
470
dbConnection
485
aDataItem
475
foreach DataItem
480
new(Record)
490
registerForOnAbcEvent ()
495
Add (aDataItem)
498
EndInitialization()

FIG. 4

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US2007/077272

A. CLASSIFICATION OF SUBJECT MATTER

G06Q 10/00(2006.01)i, G06Q 99/00(2006.01)i, G06F 17/30(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC 8 G06F 15/00, G06F 7/00, G06F 15/40, H04M 3/22, G06Q 10/00

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

Korean utility models and applictions for utility models since 1975.
Japanese utility models and application for utility models since 1975.

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKIPASS(KIPO internal) & keyword: REPORT, GENERATION, DATABASE, METADATA

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2002/0013786 A1 (MACHALEK, R.) 31 January 2002 See abstract; claim 1; claim 5; claim 6	1-20
Y	US 7,051,038 B1 (YEH, A. E. C. et al.) 23 May 2006 See abstract; claim 1; claim 11	1-20
A	US 5,404,295 A (KATZ, B. et al.) 4 April 1995 See abstract	1-20
A	US 5,546,455 A (JOYCE, R. et al.) 13 August 1996 See abstract	1-20

☐ 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 application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of 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
21 JANUARY 2008 (21.01.2008)	**21 JANUARY 2008 (21.01.2008)**

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
920 Dunsan-dong, Seo-gu, Daejeon 302-701,
Republic of Korea

Facsimile No. 82-42-472-7140

Authorized officer

YUK, SEONG WON

Telephone No. 82-42-481-8213

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/US2007/077272

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US20020013786A1	31.01.2002	US2007143661AA US07185279B2	21.06.2007 27.02.2007
US07051038B1	23.05.2006	None	
US05404295A	04.04.1995	None	
US05546455A	13.08.1996	None	