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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
23 August 2012 (23.08.2012)

WIPO | PCT

(10) International Publication Number
WO 2012/109771 A1

(51) International Patent Classification:
G06F 11/36 (2006.01)

(21) International Application Number:
PCT/CN2011/000255

(22) International Filing Date:
18 February 2011 (18.02.2011)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: **HEWLETT-PACKARD DEVELOPMENT COMPANY, L. P.** [US/US]; 11445 Compaq Center Drive W., Houston, TX 77070 (US).

(72) Inventor; and

(75) Inventor/Applicant *(for US only)*: **WEI, Yongdong** [CN/CN]; Hewlett-Packard Company, Building 6, No. 690 Bi Bo Road, Zhangjiang Micro-Electric Port, Shanghai 201203 (CN).

(74) Agent: **CHINA PATENT AGENT (H. K.) LTD.**; 22/F, Great Eagle Centre, 23 Harbour Road, Wanchai, Hong Kong Special Administrative Region (CN).

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

Published:

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

(54) **Title**: GENERATING TEST DATA

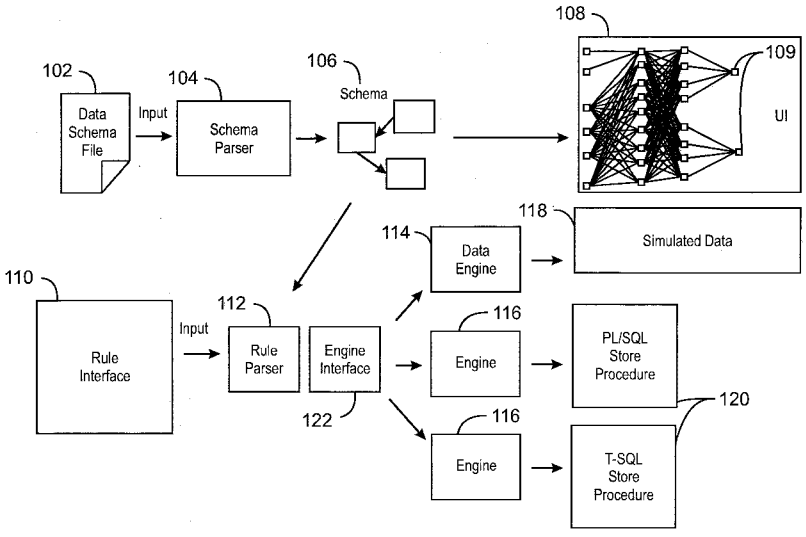


FIG. 1

100

(57) **Abstract**: A method of generating test data is provided herein. The method includes generating a schema comprising a database table. The method also includes receiving a selection of the database table. Additionally, the method includes receiving one or more rule definitions for populating the database table. The method further includes generating a stored procedure for populating the database table based on the rule definitions and the schema.

WO 2012/109771 A1

GENERATING TEST DATA

BACKGROUND

[0001] Using mock data, or test data, is commonplace in many technical areas, such as application testing and report demonstrations. Test data is also useful for agile software development. In some cases, large volumes of test data are used to make testing more meaningful. Test data that approximates real-world production data improves the quality of the testing system. However, in many cases, an actual production environment is not available to provide real-world data for copying or modeling.

[0002] Some large-scale software products include test data generation and feeding mechanisms. Such mechanisms may install or uninstall test data. The feeding mechanisms may input the test data to the testing system. However, these mechanisms are often manual processes. As such, the work may be tedious and error-prone. Some test data tools merely insert data into a database. As such, the tools are run every time test data is needed. Further, software applications delivered with the tools included potentially implicate legal issues relating to whether specific uses are in compliance with license rights provided with the tools. Moreover, users must be trained to learn third party tools and to comply with legal requirements, increasing costs.

BRIEF DESCRIPTION OF THE DRAWINGS

[0003] Certain embodiments are described in the following detailed description and in reference to the drawings, in which:

[0004] Fig. 1 is a block diagram of a system for test data in accordance with an embodiment of the invention;

[0005] Fig. 2 is a block diagram of an interface for test data in accordance with an embodiment of the invention;

[0006] Fig. 3 is a block diagram of an interface for test data in accordance with an embodiment of the invention;

[0007] Fig. 4 is a process flow diagram of a method for generating test data in accordance with an embodiment of the invention;

[0008] Fig. 5 is a block diagram of a system for generating test data in accordance with an embodiment of the invention; and

[0009] Fig. 6 is a block diagram showing a non-transitory, computer-readable medium that stores code for generating test data in accordance with an embodiment of the invention.

DETAILED DESCRIPTION OF SPECIFIC EMBODIMENTS

[0010] As stated previously, feeding test data into databases has useful applications in many areas. Test data is also referred to as mock data herein. One approach to feeding test data involves stored procedures.

[0011] A stored procedure may be a subroutine available to applications accessing a relational database. Stored procedures may actually be stored in a database data dictionary, which enables the applications to call the stored procedures. Stored procedures are technical in nature. As such, a research and development team may develop, for example, SQL stored procedures. Stored procedures may readily store volumes of data into database tables.

[0012] Writing stored procedures manually may be difficult and time-consuming. Existing tools typically insert data into a database directly. As such, users run the tools every time test data is used. In an exemplary embodiment, stored procedures are generated automatically. These stored procedures may cover an entire data feeding process. In such an embodiment, an 80% savings in development effort was realized. Test packages typically engineered in a five day period, were engineered in a single day using an exemplary embodiment. Further, the test package of the exemplary embodiment had better quality and coverage than the typical test package.

[0013] There are some tools available, such as DBUnit, EMS Data Generator and SQL Tool Belt, for data feeding. However, these tools can not generate SQL script. Further, users run them every time test data is to be used. Another tool, Turbo Data, is capable of generating SQL script, but is limited to generating INSERT statements. Further, Turbo Data hard-codes all data references. As such, the size of the Turbo Data script may be large, difficult to read and, difficult to maintain.

[0014] In contrast, exemplary embodiments can generate stored procedures that are user-friendly, and easily maintained. Further, exemplary embodiments may include an automatic, graphical stored procedure generator. Exemplary embodiments may also support deleting and recycling of the stored procedures generated. Recycling stored procedures means to re-use the data generated in multiple testing phases. In various embodiments, a stored procedure may be scheduled to run every hour, day, month, etc. At each period specified, the stored procedure may be re-run to re-generate the test data based on a specified rule.

[0015] Fig. 1 is a block diagram of a system 100 for test data in accordance with an embodiment of the invention. The system 100 may enable a user to perform an entire data feeding process. The system 100 may include may include a data schema file 102, a schema parser 104, a schema 106, an interface 108, a rule interface 110, a rule parser 112, a data engine 114, and stored procedure engines 116.

[0016] The schema parser 104 may translate the data schema file 102 for a database into a schema 106. The schema 106 may be used to generate a visual representation, *e.g.*, interface 108, of the tables 109 and relationships in the database. The user may select a table 109 from the interface 108.

[0017] In response, the rule interface 110 may be displayed, where the user may define rules for each column of the selected table. The rule parser 112 then parses these rules. Each type of rule may map to a callback class. The rule parser may call the callbacks that are provided by a registered parser engine, such as the data engine 114 and the stored procedure engines 116.

[0018] The rule interface 110 may enable users to define rules for generating mock data. Rules may include, for example, operators that create numeric values to distinguish different rows of test data. A command, such as "count +-*/ 3" may create values such as (row3, row 6...). Another command, specified "'a'||'bb'" may concatenate strings in the rows of test data. Rules may also include String lists, such as {Beijing; London} and String ranges, e.g, {Location21~30}; functions, such as rand() and cast(); column references, such as [other column name]; and business keys. The business key may be a

primary key of a table. The rule parser 112 may interpret the specified rules to facilitate generating mock data.

[0019] The data engine 114 may generate mock data based on the specified rules. In an exemplary embodiment, the data engine 114 may provide simulated data 118. The simulated data 118 may be a preview of the generated mock data. In various embodiments an interface may be provided enabling the user to select data for insertion to the database.

[0020] In an embodiment, the stored procedure engines 116 may, based on the specified rules, generate stored procedures 120 automatically. The stored procedures 120 may be called by the user to generate test data. As such, the users may populate test data by calling a specific stored procedure 120 from the database. A different stored procedure engine 116 may be provided for different types of databases, *e.g.*, databases that use PL/SQL versus T-SQL. In an exemplary embodiment, the user may extend the stored procedure engines 116 to support additional databases by registering new stored procedure engines 116 for specified databases. The following structured query language (SQL) provides one example of a stored procedure that may be generated by the stored procedure generator:

```
DECLARE VARIABLE_A INTEGER;
DECLARE VARIABLE_B INTEGER;

SET VARIABLE_A = 2
SET VARIABLE_B = 2;

SET TEMP_VARIABLE1 = 0;
WHILE (TEMP_VARIABLE1 < VARIABLE_A)
LOOP
        SET TEMP_STRING = 'Component' + CAST(TEMP_VARIABLE1 + 1 AS
        VARCHAR);
        SET TEMP_VARIABLE2 = 0;
        WHILE...
```

SQL 1

[0021] In an exemplary embodiment, the system 100 may be a full component-based architecture. Further, the schema parser 104 may be extended to support various types of data schema files 102, e.g., schema definitions specific to particular vendors.

[0022] In particular, the system 100 may also include an engine interface 122. With the engine interface 122, the user may register an engine, such as the data engine 114 or one of the stored procedure engines 116. The engine interface 122 may allow for extensions of various embodiments. As such, using the same input (the rules), different outputs may be generated: the simulated data 118, the PL/SQL stored procedure 120, or the T-SQL stored procedure 120.

[0023] Fig. 2 is a block diagram of an interface 200 for test data in accordance with an embodiment of the invention. The user interface 200 may enable the user to select options relevant to the schema parser 104, the rule parser 112, and the data engine 114. As shown, the interface includes icons 202, 204, 206. The icons 202may be clickable to open the rule interface 110. The icon 204 may be clickable to generate a stored procedure 120. The icon 206 may be clickable to manage the stored procedures 120.

[0024] The interface 200 may also include tabs 208, a work area 210, and a display area 212. The display area 212 may be used to display, for example, a schema of database tables 109. Each table 109 may be shown as an icon. In various embodiments a table may be selected along with a tab 208 to work in the work area 210. For example, once a table is selected, the user may select from one of the following tabs 208: "Links, Simulated Data, Definition, and SQL." In response to selecting "Links," the work area 210 may list all tables 109 with a link to the selected table 109. In response to selecting "Simulated data," the work area 210 may show the simulated data 118 for the selected table 109. In response to selecting "Definition," the work area 210 may enable the user to select tables into which the stored procedure 130 inserts test data. In response to selecting "SQL," the work area 210 may display the auto generated SQL script of the stored procedures 120 for the selected table 109.

[0025] Fig. 3 is a block diagram of interfaces 302, 308 for test data in accordance with an embodiment of the invention. The rule interface 302 may enable a user to specify rules for test data generation. The mock data interface 308 may enable a user to preview, modify, and select test data generated by the data engine 114 (Fig. 1).

[0026] As shown, the rule interface 302 may include names 304 of columns in a table selected for test data generation. For each column name, *e.g.*,

"dsi_key_id," the user may specify a function 306. The function 306 may be a rule for generating test data values for the corresponding column.

[0027] The mock data interface 308 may include names 310 of each column specified in the rules interface. The rows 312 may represent each row of test data generated by the data engine 114. As stated previously, the user may use the mock data interface 308 to select, modify test data to be inserted into the database. In one embodiment, the user may make changes to the specified rules after previewing test data in the interface 308.

[0028] After providing rules for a table, the user may choose to calculate the preview data with the data engine 114 to check the preview result in the mock data interface 308. If errors are found, the user may return to the rule interface 302 to modify the rules until an expected result is generated. In one embodiment, the users may select previewed data for direct insertion into the database.

[0029] Fig. 4 is a process flow diagram of a method 300 for test data in accordance with an embodiment of the invention. It should be understood that the process flow diagram is not intended to indicate a particular order of execution. Referring also to Fig. 1, the method 400 may be performed by the schema parser 104 or the stored procedure engine 116.

[0030] The method 400 begins at block 402, where the schema 106 may be generated. The schema 106 may specify one or more database tables 109.

[0031] At block 404, upon receipt of a selection of the schema, the stored procedure 120 may be generated. The selection may specify one or more database tables 109 to be populated with test data. When executed, the stored procedure 120 may populate the database tables 109 specified in the selection. The database tables 109 may be populated with test data that is generated based on rule definitions and the schema. The method 400 is described in greater detail with reference to Fig. 5.

[0032] Fig. 5 is a process flow diagram of a method 500 for test data in accordance with an embodiment of the invention. It should be understood that the process flow diagram is not intended to indicate a particular order of execution. Referring also to Fig. 1, the method 500 may be performed by the

schema parser 104, the rule parser 112, the data engine 114, or the stored procedure engine 116.

[0033] The method 500 begins at block 502, where the schema parser 104 may generate the schema 106. The schema parser 104 may read the data schema files 102 and retrieve the definition of each database table 109. The definitions may be passed to the rule parser 112.

[0034] At block 504, the schema parser 104 may display the schema 106 graphically in the interface 108. The display may occur at the same time the definitions are passed to the rule parser 112. As stated previously, the user may select one or more tables 109 from the interface 108 for test data generation.

[0035] At block 506, the rule parser 112 may receive the schema selection. At block 508, the rule parser 112 may display the rule interface 110, 302. At block 510, the rule parse may receive the rule definitions specified by the user.

[0036] At block 512, the data engine 114 or the stored procedures engine 116 may be invoked based on a user selection. If the data engine 114 is invoked, blocks 514-418 may be repeated until test data is selected for insertion into the database. At block 516, test data may be generated based on the specified rules. At block 518, the mock data interface 308 may be displayed. Once test data is selected, at block 520 the selected test data may be inserted into the database.

[0037] If, instead of the data engine 114, the stored procedures engine 116 is selected, at block 522 the stored procedure 120 may be generated for the appropriate database. At block 524, the stored procedure may be executed.

[0038] Fig. 6 is a block diagram of a system for generating test data in accordance with an embodiment of the invention. The system is generally referred to by the reference number 600. Those of ordinary skill in the art will appreciate that the functional blocks and devices shown in Fig. 6 may comprise hardware elements, software elements, or some combination of software and hardware. The hardware elements may include circuitry. The software elements may include computer code stored on a non-transitory, computer-readable medium.

[0039] Additionally, the functional blocks and devices of the system 600 are but one example of functional blocks and devices that may be implemented in an

embodiment of the invention. Those of ordinary skill in the art would readily be able to define specific functional blocks based on design considerations for a particular electronic device.

[0040] The system 600 may include servers 602, 604, in communication over a network 630. The server 604 may be similarly configured to the server 602. For example, the server 602 may include one or more processors 612, which may be connected through a bus 613 to a display 614, a keyboard 616, one or more input devices 618, and an output device, such as a printer 620. The input devices 618 may include devices such as a mouse or touch screen. The server 602 may also be connected through the bus 613 to a network interface card 626. The network interface card 626 may connect the database server 602 to the network 630.

[0041] The network 630 may be a local area network, a wide area network, such as the Internet, or another network configuration. The network 630 may include routers, switches, modems, or any other kind of interface device used for interconnection. In one example embodiment of the invention, the network 630 may be the Internet.

[0042] The server 602 may have other units operatively coupled to the processor 612 through the bus 613. These units may include non-transitory, computer-readable storage media, such as storage 622. The storage 622 may include media for the long-term storage of operating software and data, such as hard drives. The storage 622 may also include other types of non-transitory, computer-readable media, such as read-only memory and random access memory.

[0043] The storage 622 may include the software used in embodiments of the present techniques. In an embodiment of the invention, the storage 622 may include a database 624 and a test data system 628. The test data system 628 may enable a user to populate the database 624 with test data. Alternatively, the test data system 628 may generate stored procedures 120 that populate the database 624 with test data. The test data system 628 may include interfaces for selecting tables, specifying rules, and previewing generated test data.

[0044] Fig. 7 is a block diagram of a tangible, machine-readable medium that stores code adapted to generate test data according to an exemplary

embodiment of the present invention. The tangible, machine-readable medium is generally referred to by the reference number 700.

[0045] The tangible, machine-readable medium 700 may correspond to any typical storage device that stores computer-implemented instructions, such as programming code or the like. For example, the storage device may include a hard disk drive, a magnetic disk drive, e.g., to read from or write to a removable magnetic disk, and an optical disk drive, e.g., for reading a CD-ROM disk or to read from or write to other optical media.

[0046] The storage device may be connected to a system bus by a storage device interface, such as a hard disk drive interface, a magnetic disk drive interface and an optical drive interface. For example, the storage device may be the storage 622.

[0047] Although the description of computer-readable media above refers to a hard disk, a removable magnetic disk and a CD-ROM disk, it should be appreciated that other types of media that are readable by a computer system and that are suitable to the desired end purpose may be used, such as magnetic cassettes, flash memory cards, digital video disks, Bernoulli cartridges, and the like.

[0048] When read and executed by a processor 702 via a communication path 704, the instructions stored on the tangible, machine-readable medium 700 are adapted to cause the processor 702 to generate test data according to an exemplary embodiment of the present invention, as described herein.

[0049] A schema 706 may be displayed graphically. The schema 706 may include a database table. The schema 706 may be selectable by selecting a visual representation of the database table. A rules interface may be displayed in response to receiving a selection of the schema 706. The rules interface may receive rule definitions 708 for populating the database table. A stored procedure 610 may be generated to populate the database table. The database table may be populated with test data generated based on the rule definitions 708 and the schema 706.

CLAIMS

What is claimed is:

1. A method of generating test data, comprising:
generating a schema comprising a database table; and
upon receipt of a selection of the schema, generating a stored procedure for populating the database table, wherein populating the database table comprises generating test data based on rule definitions and the schema.

2. The method recited in claim 1, comprising populating the database table using the stored procedure.

3. The method recited in claim 1, comprising causing a graphical display of the schema, wherein the selection comprises clicking a visual representation of the database table.

4. The method recited in claim 1, comprising causing a display of a rules interface in response to receiving the selection, wherein the rules interface receives the rule definitions.

5. The method recited in claim 1, comprising generating one or more rows for the database table, wherein the rows are generated based on the rule definitions and the schema.

6. The method recited in claim 5, comprising causing a display of the one or more rows.

7. The method recited in claim 6, comprising receiving a selection of one of the rows.

8. The method recited in claim 7, comprising receiving a request for the selected row, wherein the request specifies one of:
a modification to the selected row;

a deletion of the selected row;
an insertion of the selected row into the database table; or
combinations thereof.

9. The method recited in claim 8, comprising one of:
modifying the selected row based on the request;
deleting the selected row based on the request;
inserting the selected row into the database table based on the request;
or
combinations thereof.

10. A computer system for generating test data, comprising:
a memory storing instructions;
a processor configured to execute the instructions to:
generate a schema comprising a database table; and
upon receipt of a selection of the schema, generate a stored procedure to populate the database table, wherein the database table is poulated with test data that is generated based on rule definitions and the schema.

11. The computer system recited in claim 10, wherein the processor is configured to populate the database table using the stored procedure.

12. The computer system recited in claim 10, wherein the processor is configured to cause a display of a rules interface in response to receiving the selection, wherein the rules interface receives the rule definitions.

13. The computer system recited in claim 10, wherein the processor is configured to:
generate one or more rows for the database table, wherein the rows are generated based on the rule definitions and the schema; and
cause a display of the one or more rows.

14. A non-transitory, computer-readable medium comprising machine-readable instructions executable by a processor to generate test data, wherein the machine-readable instructions, when executed by the processor, cause the processor to:

generate a schema comprising a database table;

cause a graphical display of the schema, the schema being selectable by selecting a visual representation of the database table;

cause a display of a rules interface in response to receiving a selection of the schema, wherein the rules interface receives rule definitions for populating the database table;

generate a stored procedure to populate the database table, wherein the database table is populated with test data generated based on the rule definitions and the schema; and

populate the database table using the stored procedure.

15. The non-transitory, computer-readable medium recited in claim 14, wherein the machine-readable instructions comprise machine-readable instructions that cause the processor to:

generate one or more rows for the database table, wherein the rows are generated based on the rule definitions and the schema;

cause a display of the rows;

receive a selection of one of the rows;

receive a request for the selected row, wherein the request specifies one of:

a modification to the selected row;

a deletion of the selected row;

an insertion of the selected row into the database table; or

combinations thereof; and

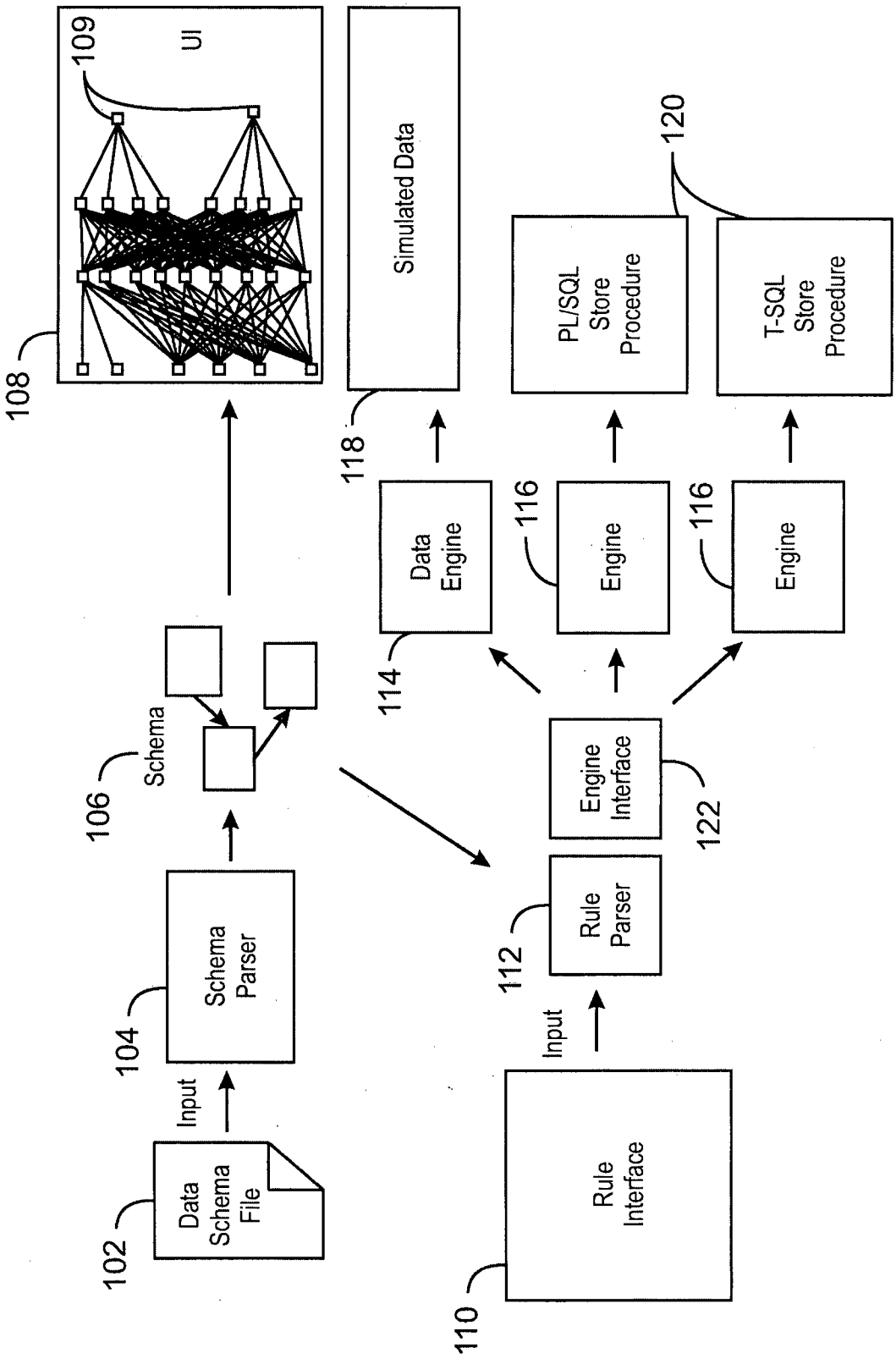
perform, based on the request, one of:

modifying the selected row;

deleting the selected row;

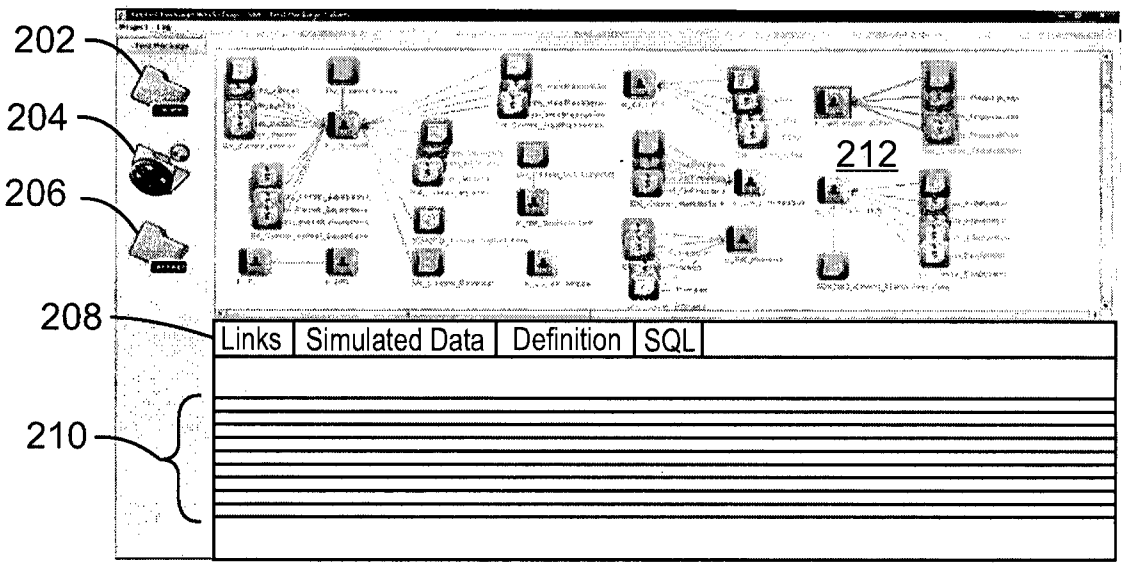
inserting the selected row into the database table; or

combinations thereof.


Data Schema File
102
Input
Schema Parser
104
Schema
106
UI
109
108
Rule Interface
110
Input
Rule Parser
112
Engine Interface
122
Data Engine
114
Engine
116
Engine
116
Simulated Data
118
PL/SQL Store Procedure
120
T-SQL Store Procedure

100

FIG. 1



200

FIG. 2

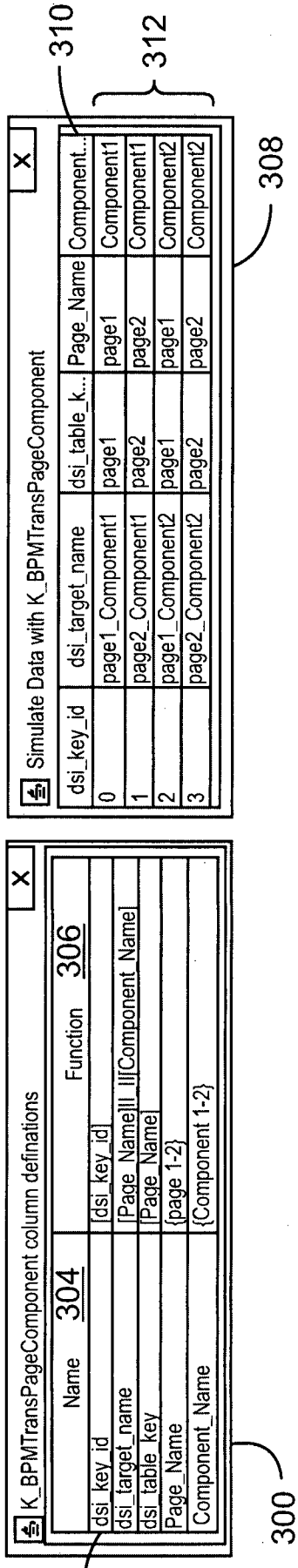
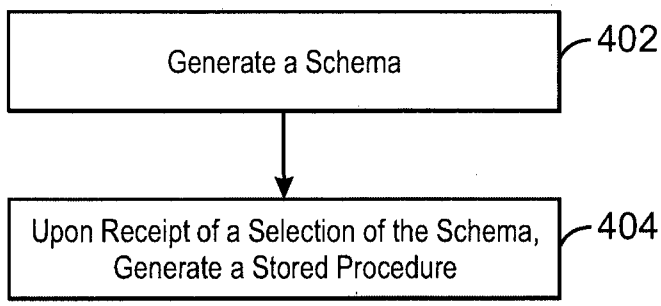
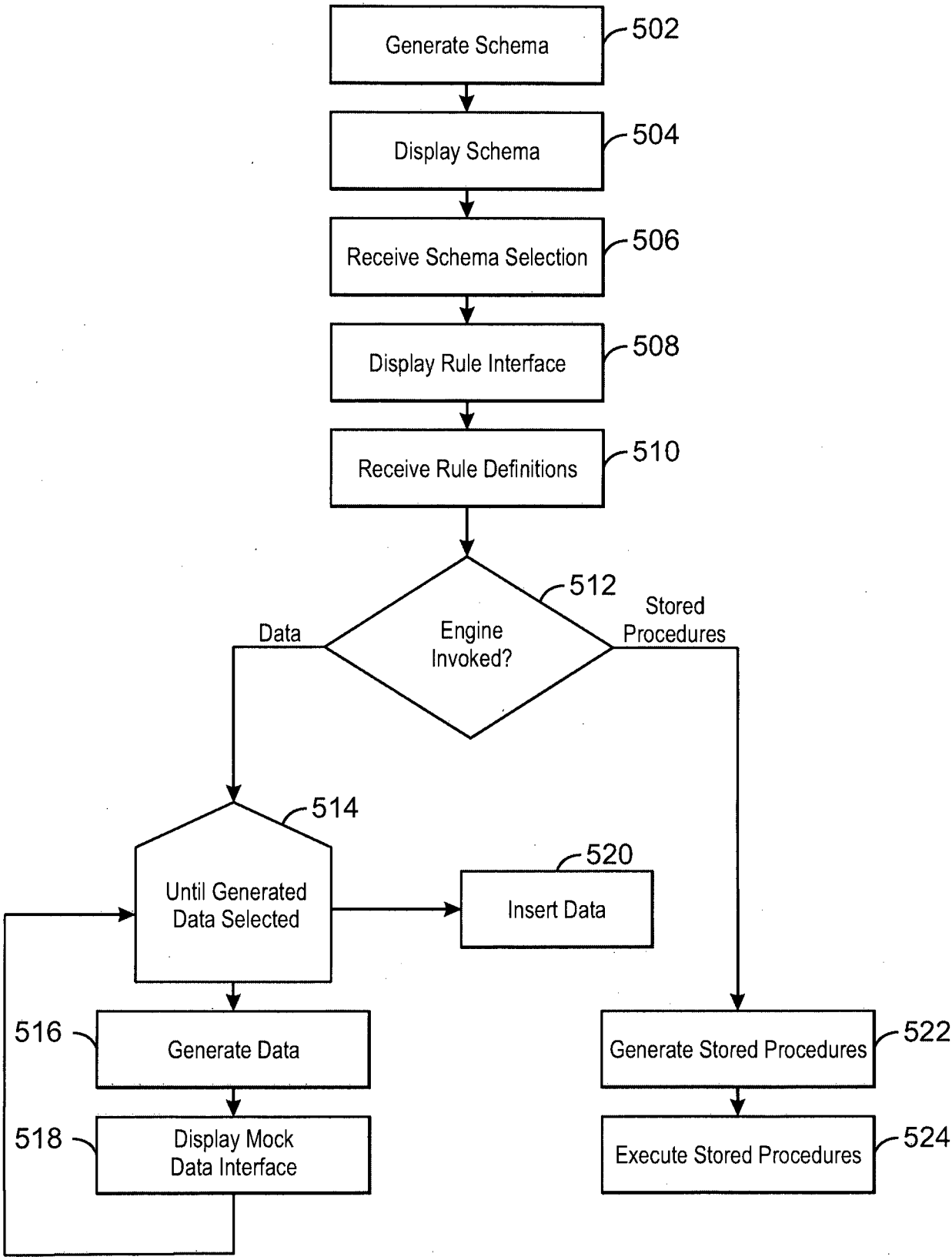

K_BPMTransPageComponent column defination
Name 304
Function 306
dsi_key_id
{dsi_key_id}
dsi_target_name
{Page_Name}||_||{Component_Name}
dsi_table_key
{Page_Name}
Page_Name
{page 1-2}
Component_Name
{Component 1-2}
300
302
Simulate Data with K_BPMTransPageComponent
dsi_key_id
dsi_target_name
dsi_table_k...
Page_Name
Component...
0
page1_Component1
page1
page1
Component1
1
page2_Component1
page2
page2
Component1
2
page1_Component2
page1
page1
Component2
3
page2_Component2
page2
page2
Component2
308
310
312

FIG. 3


Generate a Schema
402
Upon Receipt of a Selection of the Schema, Generate a Stored Procedure
404

400

FIG. 4


Generate Schema
502
Display Schema
504
Receive Schema Selection
506
Display Rule Interface
508
Receive Rule Definitions
510
Engine Invoked?
512
Data
Stored Procedures
Until Generated Data Selected
514
Insert Data
520
Generate Data
516
Display Mock Data Interface
518
Generate Stored Procedures
522
Execute Stored Procedures
524

500

FIG. 5

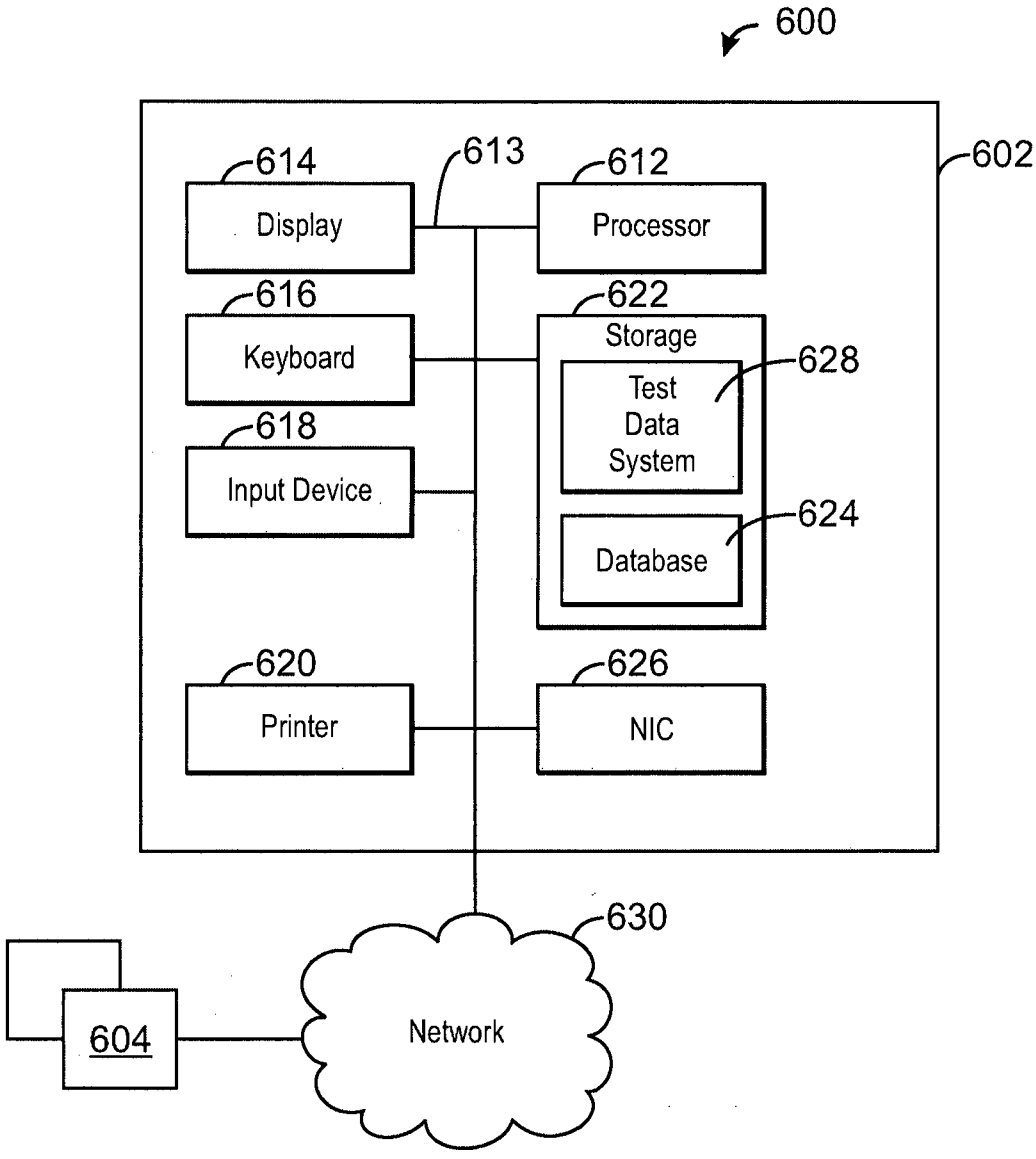

600
602
614
Display
613
612
Processor
616
Keyboard
622
Storage
628
Test
Data
System
618
Input Device
624
Database
620
Printer
626
NIC
630
Network
604

FIG. 6

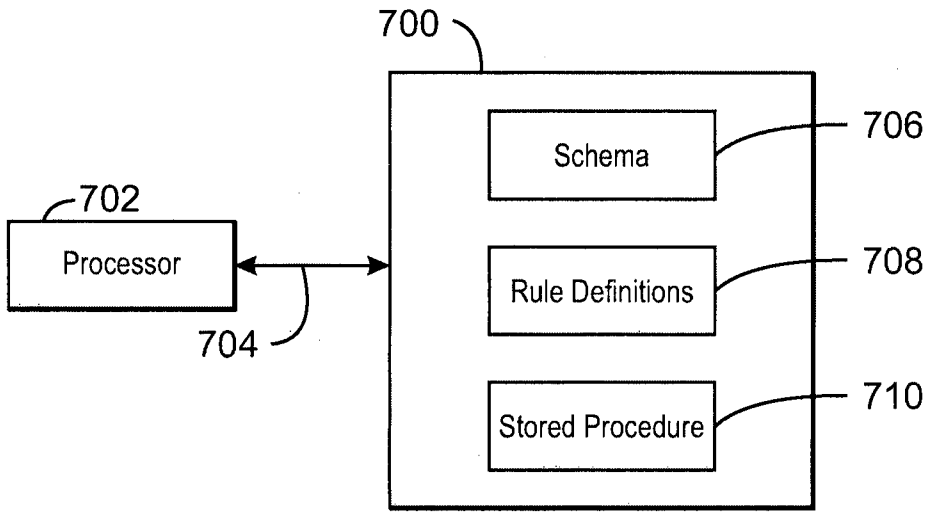

700
Schema
706
702
Processor
Rule Definitions
708
704
Stored Procedure
710

FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2011/000255

A. CLASSIFICATION OF SUBJECT MATTER

G06F11/36 (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: G06F11/-

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 practicable, search terms used)
CNABS,CNKI,VEN,DWPI: database, data, schema, test, generat+, table, populat+, model, rule, relation, definition, program, procedure, method, code

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 2008301813 A1 (MICROSOFT CORP.) 04 Dec. 2008 (04.12.2008) The whole document	1-15
A	US 2006224777 A1 (INT BUSINESS MACHINES CORP.) 05 Oct. 2006 (05.10.2006) The whole document	1-15
A	US 2006230083 A1 (MICROSOFT CORP.) 12 Oct. 2006 (12.10.2006) The whole document	1-15
A	CN 1752945 A (INST SOFTWARE CAS) 29 Mar. 2006 (29.03.2006) The whole document	1-15

☐ 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 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
25 Oct. 2011 (25.10.2011)	**10 Nov. 2011 (10.11.2011)**
Name and mailing address of the ISA/CN The State Intellectual Property Office, the P.R.China 6 Xitucheng Rd., Jimen Bridge, Haidian District, Beijing, China 100088 Facsimile No. 86-10-62019451	Authorized officer ZENG, Wei Telephone No. (86-10)62411695

Form PCT/ISA /210 (second sheet) (July 2009)

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2011/000255

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
US2008301813A1	04.12.2008	US7926114B2	12.04.2011
US2006224777A1	05.10.2006	NONE	
US2006230083A1	12.10.2006	NONE	
CN1752945A	29.03.2006	NONE	