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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
8 February 2007 (08.02.2007)

PCT

(10) International Publication Number
WO 2007/016273 A2

(51) International Patent Classification:
G06F 15/16 (2006.01)

(21) International Application Number:
PCT/US2006/029240

(22) International Filing Date: 28 July 2006 (28.07.2006)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
11/191,358 28 July 2005 (28.07.2005) US

(71) Applicant *(for all designated States except US)*: **JMJ SOFTWARE LLC.** [US/US]; 405 West Sumner Avenue, Spokane, WA 99204 (US).

(72) Inventor: **SMITH, Michael, G.**; 7410 N. Pittsburg Street, Spokane, WA 92217 (US).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **VOSS, Terry, A.** [US/US]; 2403 N. Nettleton St., Spokane, WA 99205 (US). **HOWSER, Martin, W.** [US/US]; 405 West Sumner Avenue, Spokane, WA 99204 (US).

(74) Agent: **SMITH, Michael, G.**; Ivey, Smith and Ramirez, P.O.box 2843, Spokane, WA 99220-2843 (US).

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

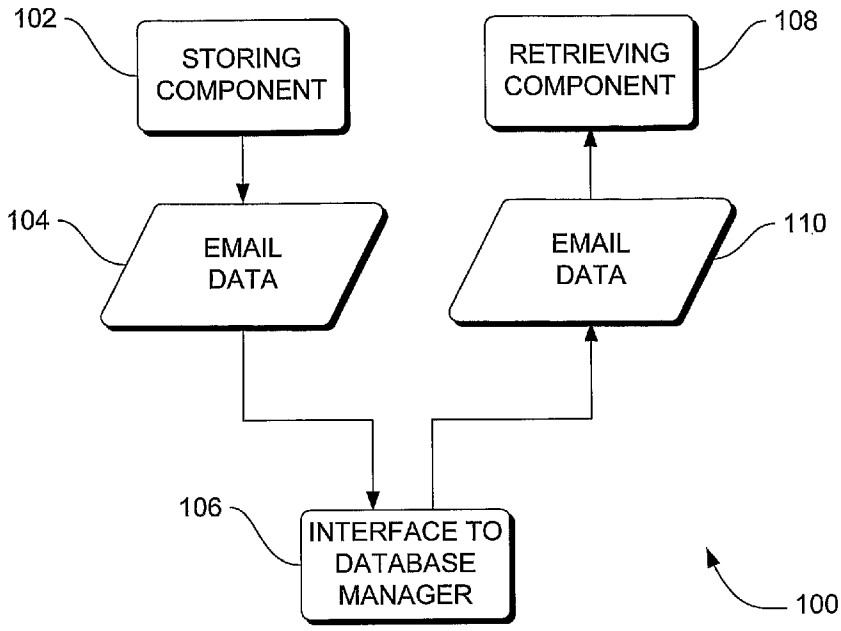
— *as to the applicant's entitlement to claim the priority of the earlier application (Rule 4.17(iii))*

Published:

— *without international search report and to be republished upon receipt of that report*

[Continued on next page]

(54) Title: SYSTEMS, METHODS AND APPARATUS OF AN EMAIL CLIENT



(57) Abstract: Systems, methods and apparatus are provided through which an email client accesses email data through an interface to a database manager, the database manager having a high level security manager.

WO 2007/016273 A2

For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

SYSTEMS, METHODS AND APPARATUS OF AN EMAIL CLIENT

FIELD OF THE INVENTION

[0001] This invention relates generally to client/server systems, and more particularly to email clients.

BACKGROUND OF THE INVENTION

[0002] Email is considered to be the "killer" application that helped create wide-spread popularity and adoption of the Internet. The ability to exchange information and files within a few seconds between any points on Earth is tremendously useful and may have provided significant economic growth in the Western world and possibly to the economies of all nations. Email has affected the structure and organization of corporations, enabling companies to disburse and distribute work and tasks between remote locations to an extent that could only be done with the nearly instant and convenient exchange of data that email provides.

[0003] As discussed above, the commercial value and function of email is tremendous, however, the function of conventional email clients present problems. One problem is the instability of the email clients. Email clients are notorious for ceasing operation without warning. This abrupt cessation is known as "crashing" or "abending." In some cases, an email client while crashing will damage a database of email data that the email client was accessing at the time of the crash that requires a backup copy of the email database to be restored, which typically results in a permanent loss of some email data. In other cases, an email client while crashing can cause the computer to enter into a mode of unoperability, known as "hung" that requires rebooting the computer. All of these problems result in a loss of productivity and high level of dissatisfaction with email clients among the users of the email clients.

[0004] In particular, conventional executable email client files are a binary image of machine executable computer instructions that are loaded into memory of a computer. A

program counter is set to the beginning location of the computer instructions. An operating system of the computer is not involved in overseeing the execution of the computer instructions, other than protections in place around memory management and port I/O. However, the operating system has very little information on what the computer instructions are doing. Therefore, errant computer instructions often wreak havoc on the stability of the operating system and the computer.

[0005] In addition, conventional email clients have very limited ability to prioritize emails that have been received. In particular, one conventional email client has the ability to sort emails in particular folder according to any one of the following fields or criteria: Criteria assigned to the email by the sender (importance), icon, flag status, presence of an attachment, sender, subject, date received and size. This scope of sorting/prioritization often does not readily reveal a particular email that a user is trying to locate. Often, a user will resort to searching the entire database of emails to find a particular email. In some cases, users will save multiple copies of an email on various subdirectories in order to make locating the email easier later on.

[0006] Also, conventional email clients have difficulty displaying email messages that are encoded in hyper text markup language (HTML). To display a HTML encoded email message, conventional email clients invoke a browser. The browser then displays the HTML-encoded email. The invocation of a browser introduces execution errors of the browser into the execution of the email client.

[0007] Furthermore, conventional email clients are limited in the total of email and attachments that can be stored in one email file. For example, on a computer running the Microsoft Windows® operating system, the maximum file size of an email file is often two gigabytes. In that case, the total amount of emails and attachments that can be stored in the email file is two gigabytes. Some users fill up an email file of two gigabytes in a couple of years, and in some unusual situations, a few weeks. Some conventional email clients allow some of the data in the email to be archived, but archival email files must be inconveniently searched separately when the archived email data is retrieved.

[0008] For the reasons stated above, and for other reasons stated below which will become apparent to those skilled in the art upon reading and understanding the present specification, there is a need in the art for an operationally stable email client. There is also a need for improved sorting/prioritization of emails. There is also a need for less invocation of browser errors by an email client. In addition, there is a need for an email client that provides more storage for email data including attachments, without having to resort to archival files.

BRIEF DESCRIPTION OF THE INVENTION

[0009] The above-mentioned shortcomings, disadvantages and problems are addressed herein, which will be understood by reading and studying the following specification.

[0010] In one aspect, an apparatus to provide a reliable apparatus for managing email data comprises an apparatus operable to store at least a portion of the email through an interface to a database manager; the database manager having a high level security manager, and an apparatus operable to retrieve email data through the interface to the database manager. A technical effect of this aspect is reduced corruption of the database, which improves data integrity of the email data.

[0011] In another aspect, an apparatus to provide a reliable apparatus of managing email data, an apparatus operable to store at least a portion of the email through an interface to a relational database manager and an apparatus operable to retrieve email data through the interface to the relational database manager. A technical effect of this aspect is a stable data storage environment for email data.

[0012] In yet another aspect, an apparatus to provide a reliable apparatus of managing email data, the apparatus comprising an apparatus operable to store at least a portion of the email through an interface to a structured-query-language database manager and an apparatus operable to retrieve email data through the interface to the structured-query-language database manager. A technical effect of this aspect is a stable

data storage environment for email data.

[0013] In a further aspect, a method to provide storage for email data including attachments, comprises creating a file structure that is operable to store email attachment data and creating a file structure that is operable to store email data other than the email attachment data. A technical effect of this aspect is greater storage of email data including attachments, without having to resort to archival files.

[0014] Systems, clients, servers, methods, and computer-readable media of varying scope are described herein. In addition to the aspects and advantages described in this summary, further aspects and advantages will become apparent by reference to the drawings and by reading the detailed description that follows.

BRIEF DESCRIPTION OF THE DRAWINGS

[0015] FIG. 1 is a block diagram that provides an overview of a system to provide a reliable apparatus of managing email data;

[0016] FIG. 2 is a diagram of a table layout data structure to manage categories;

[0017] FIG. 3 is a diagram of a table layout data structure to manage persons;

[0018] FIG. 4 is a diagram of a table layout data structure to manage category and person relationships;

[0019] FIG. 5 is a diagram of a data structure to manage category and person relationships;

[0020] FIG. 6 is a diagram of a table layout data structure to manage communications

[0021] FIG. 7 is a diagram of a data structure to manage category and

communications relationships;

[0022] FIG. 8 is a diagram of a data structure to manage persons and communications relationships;

[0023] FIG. 9 is a diagram of a table layout data structure to manage attachments;

[0024] FIG. 10 is a diagram of a data structure to manage communication and attachment relationships;

[0025] FIG. 11 is a diagram of a table layout data structure to manage zip codes;

[0026] FIG. 12 is a diagram of a data structure to manage persons and zip code relationships;

[0027] FIG. 13 is a diagram of a table layout data structure to manage recurrences;

[0028] FIG. 14 is a diagram of a data structure to manage persons and recurrences relationships;

[0029] FIG. 15 is a diagram of a table layout data structure to manage states;

[0030] FIG. 16 is a diagram of a data structure to manage persons and state relationships;

[0031] FIG. 17 is a diagram of a table layout data structure to manage countries;

[0032] FIG. 18 is a diagram of a data structure to manage persons and country relationships;

[0033] FIG. 19 is a diagram of a table layout data structure to manage setups;

[0034] FIG. 20 is a diagram of a table layout data structure to manage priorities;

[0035] FIG. 21 is a diagram of a table layout data structure to manage generations;

[0036] FIG. 22 is a diagram of a table layout data structure to manage email sending defaults;

[0037] FIG. 24 is a diagram of a table layout data structure to manage mdefaults;

[0038] FIG. 24 is a diagram of a table layout data structure to manage mdefaults;

[0039] FIG. 25 is a diagram of a table layout data structure to manage cdefaults;

[0040] FIG. 26 is a block diagram of a data storage architecture that provides a stable data storage environment through a transaction-based architecture;

[0041] FIG. 27 is a block diagram of a database manager that provides a stable data storage environment through a high level security manager;

[0042] FIG. 28 is a block diagram of a database manager that provides a stable data storage environment through relational data management;

[0043] FIG. 29 is a block diagram of a hardware and operating environment in which different embodiments can be practiced;

[0044] FIG. 30 is a flowchart of a method to manage email associations, according to an embodiment;

[0045] FIG. 31 is a flowchart of a method to provide storage for email data including attachments according to an embodiment;

[0046] FIG. 32 is a flowchart of a method to retrieve an email attachment according to an embodiment;

[0047] FIG. 33 is a flowchart of a method to manage auto-receive according to an embodiment;

[0048] FIG. 34 is a data flow diagram to manage attachments according to an

embodiment;

[0049] FIG. 35 is a data flow diagram to manage embedded graphics according to an embodiment;

[0050] FIG. 36 is a flowchart of a method to import an email according to an embodiment;

[0051] FIG. 37 is a data flow diagram to import data into an email client according to an embodiment;

[0052] FIG. 38 is a dataflow diagram to manage email communications according to an embodiment;

[0053] FIG. 39 is a dataflow diagram to manage multiple users according to an embodiment;

[0054] FIG. 40 is a flowchart of a method to create an executable managed code email client according to an embodiment; and

[0055] FIG. 41 is a flowchart of a method to prioritize email communications according to an embodiment.

DETAILED DESCRIPTION OF THE INVENTION

[0056] In the following detailed description, reference is made to the accompanying drawings that form a part hereof, and in which is shown by way of illustration specific embodiments which may be practiced. These embodiments are described in sufficient detail to enable those skilled in the art to practice the embodiments, and it is to be understood that other embodiments may be utilized and that logical, mechanical, electrical and other changes may be made without departing from the scope of the embodiments. The following detailed description is, therefore, not to be taken in a limiting sense.

[0057] The detailed description is divided into six sections. In the first section, a system level overview is described. In the second section, database table embodiments of apparatus are described. In the third section, database manager embodiments are described. In the fourth section, a hardware and operating environment in which embodiments may be practiced is described. In the fifth section, embodiments of methods and dataflow diagrams are described. Finally, in the sixth section, a conclusion of the detailed description is provided.

System Level Overview

[0058] FIG. 1 is a block diagram that provides an overview of a system to provide a reliable apparatus of managing email data. System 100 solves the need in the art for an operationally stable email client

[0059] System 100 includes a storing software component 102 that is operable to store email data 104 through an interface to database manager 106. The database manager 106 provides a stable data storage environment for the email data 104. FIG. 26 below shows an embodiment in which the database manager provides a stable data storage environment through a transaction-based architecture. FIG. 27 below shows an embodiment in which the database manager provides a stable data storage environment through a high level security manager. FIG. 28 below shows an embodiment in which a relational database manager provides a stable data storage environment. In some embodiments, the format of the email data 104 is defined and prescribed in the Request for Comments published by the Internet Engineering Task Force at 1895 Preston White Drive, Suite 100, Reston, VA. 20191.

[0060] System 100 also includes a retrieving software component 108 that is operable to retrieve email data 110 through the interface 106 to a database manager. The storing software component 102 and the retrieving software component 108 are adapted to communicate with the interface 106 to a database manager.

[0061] The interface 106 to a database manager in some embodiments is a separate

component from the storing software component 102 and the retrieving software component 108 as shown to the extent that the interface 106 to a database manager is separately installed on a computer that operates system 100. However, in other embodiments not shown, the interface 106 to a database manager is a component of system 100 to the extent that the interface 106 to a database manager is installed on a computer with the storing software component 102 and the retrieving software component 108.

[0062] Managing email data 104 and 110 through the interface to a database manager 106 provides an operationally stable system of managing email data 104 and 110.

[0063] In some embodiments, the software components 102 and 108 are more generally apparatus, such as computer hardware circuitry. System 100 operates in a multi-processing, multi-threaded operating environment on a computer, such as computer 2902 in FIG. 29.

[0064] While the system 100 is not limited to any particular storing software component 102, retrieving software component 108 or an interface 106 to database manager, for sake of clarity a simplified storing software component 102, retrieving software component 108 and interface 106 to a database manager are described.

Database Table Embodiments

[0065] In the previous section, a system level overview of the operation of an embodiment is described. In this section, the particular database implementations of such an embodiment are described. Describing the database implementations by reference to database table layouts enables one skilled in the art to develop such programs, firmware, or hardware, including such instructions to carry out the methods on suitable computers, executing the instructions from computer-readable media. Table layouts 200-2500 are implemented by software, firmware and/or hardware that is a part of, a computer, such as computer 2902 in FIG. 29.

[0066] FIG. 2 is a diagram of a table layout data structure 200 to manage categories.

[0067] Table layout data structure 200 includes a table representing categories 202. Each record of the category table 202 includes an indexed key field representing an identification of a category 204, a text field representing a description of the category 206 and a text field representing a note on the category 208. Categories are groups or associations. Examples of categories are "Friday evening Volleyball Group" or "Mayoral campaign" or "heaven." Categories are more generally any relationship.

[0068] FIG. 3 is a diagram of a table layout data structure 300 to manage persons.

[0069] Table layout data structure 300 includes a table representing persons 302. Each record of the persons table 302 includes an indexed key field representing an identification of a person 304.

[0070] Other embodiments of the persons table 302 include any combination of the following: A field representing a priority 306 of the person, a field representing a first name 308 of the person, a field representing a middle name 310 of the person, a field representing a last name 312 of the person, a field representing an address 314 of the person, a field representing a city 316 of the person, a field representing a state identification 318 of the person, a field representing a state 320 of the person, a field representing a country 322 a field representing a country other 324 of the person, a field representing a zip code identification 326 of the person, a field representing a zip code 328 of the person, a field representing a title 330 of the person, a field representing a company name 332 of the person, a field representing a business address 334 of the person, a field representing a business city 336 of the person, a field representing a business state identification 338 of the person, a field representing a business state 340 of the person, a field representing a business country 342 of the person, a field representing a business country other 344 of the person, a field representing a business zip code identification 346 of the person, a field representing a business zip code 348 of the person, a field representing a home phone 350 of the person, a field representing a work

phone 352 of the person, a field representing a fax number 354 of the person, a field representing a cell phone number 356 of the person, a field representing an other phone 358 of the person, a field representing a first email address 360 of the person, a field representing a second email address 362 of the person, a field representing a third email address 364 of the person, a field representing a contact 366 of the person, a field representing an Internet URL 368 of the person, a field representing a notes 370 of the person, a field representing an anniversary date 372 and a field representing a birthday date 374.

[0071] FIG. 4 is a diagram of a table layout data structure 400 to manage category and person relationships.

[0072] Table layout data structure 400 includes a table representing category/person relationships 402. Each record of the category/person table 402 includes an indexed key field representing an identification of category/person 404, a field representing a description of the category identification 406 and a text field representing a person identification 408. The table layout data structure 400 that includes a table representing category/persons 402 provides a relationship between categories and persons.

[0073] FIG. 5 is a diagram of a data structure 500 to manage category and person relationships.

[0074] Table layout data structure 500 includes a table representing categories 202, a table representing persons 302 and a table representing category and person relationships 402. The category/person relationships table 402 provides a path through which persons are associated with category data. The persons table 302 has a one-to-many relationship with the category/person relationships table 402 as indicated by the connection 502 and the category table 202 has one-to-many relationship with the category/person relationships table 402 as indicated by the connection 504.

[0075] In table layout data structure 500, the category/person relationships table 402 acts an intermediary table that provides many-to-many associations between the

categories table 202 and the persons table. This provides a many-to-many relationship between persons and categories. Any person, and any number of persons, can be associated with any category or any number of categories.

[0076] FIG. 6 is a diagram of a table layout data structure 600 to manage communications. In some embodiments, the communication is an email communication.

[0077] Table layout data structure 600 includes a table representing communications 602. Each record of the communication table 602 includes an indexed key field representing an identification of a communication 604, a field representing a identification of a person to whom the communication was sent 606, a field representing an identification of a person from whom the communication was sent 608, a field representing a topic of the communication 610, a field representing a read/unread status of the communication 612, a field representing an over importance of the communication 614, a field representing an importance of the person of the communication 616, a field representing an importance of the communication 618, a field representing a subject of the communication 620, a field representing a creation date the communication 622, a field representing a "to" email address of the communication 624, a field representing a "from" email address of the communication 626, a field representing a name of the communication 628, a field representing a reply address of the communication 630, a field representing a contents or body of the communication 632, a field representing a HTML status of the communication 634, a field representing a HTML contents status of the communication 636, a field representing an incoming status of the communication 638, and a field representing a "to do" status of the communication 640. The contents field 632 stores the text of the body of the email communication. The HTML contents 636 stores the HTML tags and text of the email communication.

[0078] FIG. 7 is a diagram of a data structure 700 to manage category and communications relationships.

[0079] Table layout data structure 700 includes a table representing categories 202,

and a table representing communications 602. The category table 202 has a one-to-many relationship with the communications table 602 as indicated by the connection 702.

[0080] FIG. 8 is a diagram of a data structure 800 to manage persons and communications relationships.

[0081] Table layout data structure 800 includes a table representing persons 302, and a table representing communications 602. The persons table 302 has a one-to-many relationship with the communications table 602 as indicated by the connection 802.

[0082] FIG. 9 is a diagram of a table layout data structure 900 to manage attachments. Table layout data structure 900 solves the need in the art for more storage of email data including attachments, without having to resort to archival files.

[0083] Table layout data structure 900 includes a table representing attachments 902. Each record of the attachments table 902 includes an indexed key field representing an identification of an attachment 904, a text field representing a communication identification 906 and a field representing a location of the attachment 908. In some embodiments, the communication is an email. In some embodiments, the location 908 is a full path name such as "C:\\USERFILES\ATTACHMENT04340.DOC".

[0084] In some embodiments, the location 908 is outside the email database. The attachments table 902 provides a means to locate attachments that are stored separately from the email database. Thus the attachments table 900 supports storage of attachments outside of the email database, which allows the only the emails to be stored in the email database, which in turn reduces, if not eliminates, the amount of storage in the email database required to store attachments. This allows many more emails to be stored in the email database and the attachments to be stored elsewhere, such as on the disk drive of the computer, which has a tremendously larger amount of storage space. For example, on a computer running the Microsoft Windows® operating system, the maximum file size of an email database file is often two gigabytes, while the computer will often have ten to sixty gigabytes of unused hard disk space. Storing the attachments in the unused hard

disk space separately from the email database not only provides a means to make the much larger amount of unused disk space available to store attachments, but also provides a means to use the email database only for emails, which allows more emails to be stored in the email database. Thus attachment table 900 solves the need in the art for more storage for email data including attachments, without having to resort to archival files.

[0085] FIG. 10 is a diagram of a data structure 1000 to manage communications and attachment relationships.

[0086] Table layout data structure 1000 includes a table representing attachments 902, and a table representing communications 602. The communications table 602 has a one-to-many relationship with the attachments table 902 as indicated by the connection 1002.

[0087] FIG. 11 is a diagram of a table layout data structure 1100 to manage zip codes.

[0088] Table layout data structure 1100 includes a table representing zip codes 1102. Each record of the zip code table 1102 includes an indexed key field representing an identification of a zip code 1104, a field representing a zip code 1106, a field representing a city of the zip code and a field representing state of the zip code 1110.

[0089] FIG. 12 is a diagram of a data structure 1200 to manage persons and zip code relationships.

[0090] Table layout data structure 1200 includes a table representing persons 302, and a table representing zip codes 1102. The zip code table 1102 has a one-to-many relationship with the persons table 302 as indicated by the connection 1202.

[0091] FIG. 13 is a diagram of a table layout data structure 1300 to manage recurrences.

[0092] Table layout data structure 1300 includes a table representing recurrences 1302. Each record of the recurring table 1302 includes an indexed key field representing an identification of a recurrence 1304.

[0093] Other embodiments of the recurring table 1302 include any combination of the following: A field representing a person identification 1306, a field representing a period of recurrence 1308, a field representing a type of recurrence 1310, a field representing a day of the week of the recurrence 1312, a field representing a which of the recurrence 1314, a field representing a which first of the recurrence 1316, a field representing a month of the recurrence 1318, a field representing a lead time of the recurrence 1320, a field representing a short description of the recurrence 1322, a field representing a long description of the recurrence 1324, a field representing a from email address of the recurrence 1326, a field representing a first date of the recurrence 1328, a field representing a second date of the recurrence 1330, a field representing a start time of the recurrence 1332, a field representing an end time of the recurrence 1334, a field representing a maximum status of the recurrence 1336, a field representing a maximum occurrences of the recurrence 1338, a field representing a last date that the occurrence was generated 1340, and a field representing a number of remaining occurrences 1342.

[0094] FIG. 14 is a diagram of a data structure 1400 to manage persons and recurrences relationships.

[0095] Table layout data structure 1400 includes a table representing persons 302, and a table representing recurrences 602. The persons table 302 has a one-to-many relationship with the recurrences table 602 as indicated by the connection 1402.

[0096] FIG. 15 is a diagram of a table layout data structure 1500 to manage states.

[0097] Table layout data structure 1500 includes a table representing states 1502. Each record of the states table 1502 includes an indexed key field representing an identification of a state 1504, a text field representing an abbreviation of the state 1506 and a field representing a name of the state 1508.

[0098] FIG. 16 is a diagram of a data structure 1600 to manage persons and state relationships.

[0099] Table layout data structure 1600 includes a table representing persons 302, and a table representing states 1502. The state table 1502 has a one-to-many relationship with the persons table 302 as indicated by the connection 1602.

[00100] FIG. 17 is a diagram of a table layout data structure 1700 to manage countries.

[0100] Table layout data structure 1700 includes a table representing countries 1702. Each record of the country table 1702 includes an indexed key field representing an identification of a country 1704, and a field representing a name of the country 1706.

[0101] FIG. 18 is a diagram of a data structure 1800 to manage persons and country relationships.

[0102] Table layout data structure 1800 includes a table representing persons 302, and a table representing countries 1702. The country table 1702 has a one-to-many relationship with the persons table 302 as indicated by the connection 1802.

[0103] FIG. 19 is a diagram of a table layout data structure 1900 to manage setups.

[0104] Table layout data structure 1900 includes a table representing setups 1902. Each record of the setup table 1902 includes an indexed key field representing an identification of a setup 1904.

[0105] Other embodiments of the setup table 1902 include any combination of the following: A field representing a login 1906, a field representing an email address 1908, a field representing a first mail server 1910, a field representing a first Microsoft user name 1912, a field representing a first Microsoft password 1914, a field representing a second mail server 1916, a field representing a second Microsoft user name 1918, a field representing a second Microsoft password 1920, a field representing a third mail server

1922 a field representing a third Microsoft user name 1924, a field representing a third Microsoft password 1926, a field representing a standalone status 1928, a field representing a password 1930, a field representing a first name 1932, a field representing a middle name 1934, a field representing a last name 1936, a field representing a local area code 1938, a field representing a start up date 1940, a field representing a date path 1942, a field representing a document path 1944, a field representing a server name 1946, a field representing a super administrator (SA) password 1948, a field representing a database name 1950, and a field representing user information 1952.

[0106] FIG. 20 is a diagram of a table layout data structure 2000 to manage priorities.

[0107] Table layout data structure 2000 includes a table representing priorities 2002. Each record of the priority table 2002 includes an indexed key field representing an identification of a priority 2004, and a field representing a description of the priority 2006.

[0108] FIG. 21 is a diagram of a table layout data structure 2100 to manage generations.

[0109] Table layout data structure 2100 includes a table representing generations 2102. Each record of the priority table 2102 includes an indexed key field representing an identification of a generation 2104, and a field representing a late generation date of the priority 2106.

[0110] FIG. 22 is a diagram of a table layout data structure 2200 to manage email sending defaults (edefaults).

[0111] Table layout data structure 2200 includes a table representing edefaults 2202. Each record of the edefaults table 2202 includes an indexed key field representing an identification of an edefault 2204.

[0112] Other embodiments of the edefaults table 2202 include any combination of the following: A field representing a blind-carbon-copy (BCC) 2206, a field representing a priority of the edefault 2208, a field representing a format of the edefault 2210, a field representing a delivery report 2212, a field representing a read report 2214, a field representing a use introduction 2216, a field representing an introduction 2218, a field representing a use compose 2220, a field representing a use signature 2222 a field representing a signature 2224, and a field representing a from email address 2226.

[0113] FIG. 23 is a diagram of a table layout data structure 2300 to manage contacts.

[0114] Table layout data structure 2300 includes a table representing contacts 2302. Each record of the contacts table 2302 includes an indexed key field representing an identification of a contact 2304.

[0115] Other embodiments of the contacts table 2302 include any combination of the following: A field representing a first name 2306, a field representing a last name of the contact 2308, a field representing a middle name of the contact 2310, a field representing a name 2312, a field representing an email address 2314, a field representing an address 2316, a field representing a street 2318, a field representing a city 2320, a field representing a zip code 2322 a field representing a state 2324, and a field representing a region 2326.

[0116] FIG. 24 is a diagram of a table layout data structure 2400 to manage email receiving defaults (mdefaults).

[0117] Table layout data structure 2400 includes a table representing mdefaults 2402. Each record of the mdefaults table 2402 includes an indexed key field representing an identification of a mdefault 2404.

[0118] Other embodiments of the mdefaults table 2402 include any combination of the following: A field representing a direction 2406, a field representing a first date of the mdefault 2408, a field representing a second date of the mdefault 2410, a field

representing a connection 2412, a field representing a done status 2414, a field representing a find status 2416, a field representing a findin status 2418, and a field representing days back 2420.

[0119] FIG. 25 is a diagram of a table layout data structure 2500 to manage contact defaults (cdefaults).

[0120] Table layout data structure 2500 includes a table representing contact default information 2502. Each record of the contact default table 2502 includes an indexed key field representing an identification of a contact default 2504, a field representing a contact default header 2506, a field representing data of the contact default and a field representing at least one target of the contact default 2510.

Database Manager Implementation

[0121] Referring to FIGS. 26-28, particular implementations of database managers and data architectures are described in conjunction with the system overview in FIG. 1 and the methods described in conjunction with FIGS. 30-41

[0122] FIG. 26 is a block diagram of a data storage architecture 2600 that provides a stable data storage environment through a transaction-based architecture. In a transaction-based database manager, a bifurcated file structure improves data integrity.

[0123] Transactions 2602 are stored separately from the database 2604 which allows the transactions 2602 to be backed up separately from the database 2604. The separate data storage locations provide the bifurcated structure. This reduces the chance that corruption in the database 2604 will affect the transactions 2602. When the database 2604 becomes corrupted, a backup copy (not shown) of that database 2604 is restored, and then the transactions 2602 are applied to the database 2604 by a transaction-based database manager 2606, bringing the database 2604 to a fully updated state. In other embodiments, the transactions 2602 are rolled back from the corrupted database 2604 by the transaction-based database manager 2604. Thus the state of the database 2604 is

returned to a state prior to the corruption. The ability to back out transactions 2602 and update the database 2604 with transactions 2602 greatly reduces the chance that data will be permanently lost, which improves data integrity of the database 2604.

[0124] Apparatus 2600 solves the need in the art to improve data integrity, and thus operational stability of an email client that accesses the data storage architecture 2600.

[0125] FIG. 27 is a block diagram of a database manager 2700 that provides a stable data storage environment through a high level security manager.

[0126] Database manager 2700 includes a high level security manager 2702 that ensures that data is written to appropriate portions of a database (not shown), and only to appropriate portions of the database, and also ensures that the data is written only under required authorizations. This reduces the chances of corruption of the database, which improves data integrity of the database.

[0127] FIG. 28 is a block diagram of a database manager 2800 that provides a stable data storage environment through relational data management. The relational database manager 2800 is adapted to operate in conjunction with tables, such as tables 200-2500.

[0128] Apparatus components can be embodied as computer hardware circuitry or as a computer-readable program, or a combination of both. In another embodiment, the systems, methods and apparatus are implemented in an application service provider (ASP) system.

[0129] More specifically, in the computer-readable program embodiment, the programs can be structured in managed or unmanaged code, or in an object-orientation using an object-oriented language such as Java, Smalltalk or C++, and the programs can be structured in a procedural-orientation using a procedural language such as COBOL or C. The software components communicate in any of a number of means that are well-known to those skilled in the art, such as application program interfaces (API) or interprocess communication techniques such as remote procedure call (RPC), common

object request broker architecture (CORBA), Component Object Model (COM), Distributed Component Object Model (DCOM), Distributed System Object Model (DSOM) and Remote Method Invocation (RMI). The components execute on as few as one computer as in computer 2902 in FIG. 29, or on at least as many computers as there are components.

Hardware and Operating Environment

[0130] FIG. 29 is a block diagram of a hardware and operating environment 2900 in which different embodiments can be practiced. The description of FIG. 29 provides an overview of computer hardware and a suitable computing environment in conjunction with which some embodiments can be implemented. Embodiments are described in terms of a computer executing computer-executable instructions. However, some embodiments can be implemented entirely in computer hardware in which the computer-executable instructions are implemented in read-only memory. Some embodiments can also be implemented in client/server computing environments where remote devices that perform tasks are linked through a communications network. Program modules can be located in both local and remote memory storage devices in a distributed computing environment.

[0131] Computer 2902 includes a processor 2904, commercially available from Intel, Motorola, Cyrix and others. Computer 2902 also includes random-access memory (RAM) 2906, read-only memory (ROM) 2908, and one or more mass storage devices 2910, and a system bus 2912, that operatively couples various system components to the processing unit 2904. The memory 2906, 2908, and mass storage devices, 2910, are types of computer-accessible media. Mass storage devices 2910 are more specifically types of nonvolatile computer-accessible media and can include one or more hard disk drives, floppy disk drives, optical disk drives, and tape cartridge drives. The processor 2904 executes computer programs stored on the computer-accessible media.

[0132] Computer 2902 can be communicatively connected to the Internet 2914 via a

communication device 2916. Internet 2914 connectivity is well known within the art. In one embodiment, a communication device 2916 is a modem that responds to communication drivers to connect to the Internet via what is known in the art as a "dial-up connection." In another embodiment, a communication device 2916 is an Ethernet® or similar hardware network card connected to a local-area network (LAN) that itself is connected to the Internet via what is known in the art as a "direct connection" (e.g., T1 line, etc.).

[0133] A user enters commands and information into the computer 2902 through input devices such as a keyboard 2918 or a pointing device 2920. The keyboard 2918 permits entry of textual information into computer 2902, as known within the art, and embodiments are not limited to any particular type of keyboard. Pointing device 2920 permits the control of the screen pointer provided by a graphical user interface (GUI) of operating systems such as versions of Microsoft Windows®. Embodiments are not limited to any particular pointing device 2920. Such pointing devices include mice, touch pads, trackballs, remote controls and point sticks. Other input devices (not shown) can include a microphone, joystick, game pad, satellite dish, scanner, or the like.

[0134] In some embodiments, computer 2902 is operatively coupled to a display device 2922. Display device 2922 is connected to the system bus 2912. Display device 2922 permits the display of information, including computer, video and other information, for viewing by a user of the computer. Embodiments are not limited to any particular display device 2922. Such display devices include cathode ray tube (CRT) displays (monitors), as well as flat panel displays such as liquid crystal displays (LCD's). In addition to a monitor, computers typically include other peripheral input/output devices such as printers (not shown). Speakers 2924 and 2926 provide audio output of signals. Speakers 2924 and 2926 are also connected to the system bus 2912.

[0135] Computer 2902 also includes an operating system (not shown) that is stored on the computer-accessible media RAM 2906, ROM 2908, and mass storage device 2910, and is and executed by the processor 2904. Examples of operating systems include

Microsoft Windows®, Apple MacOS®, Linux®, UNIX®. Examples are not limited to any particular operating system, however, and the construction and use of such operating systems are well known within the art.

[0136] Embodiments of computer 2902 are not limited to any type of computer 2902. In varying embodiments, computer 2902 comprises a PC-compatible computer, a MacOS®-compatible computer, a Linux®-compatible computer, or a UNIX®-compatible computer. The construction and operation of such computers are well known within the art.

[0137] Computer 2902 can be operated using at least one operating system to provide a graphical user interface (GUI) including a user-controllable pointer. Computer 2902 can have at least one web browser application program executing within at least one operating system, to permit users of computer 2902 to access an intranet, extranet or Internet world-wide-web pages as addressed by Universal Resource Locator (URL) addresses. Examples of browser application programs include Netscape Navigator® and Microsoft Internet Explorer®.

[0138] The computer 2902 can operate in a networked environment using logical connections to one or more remote computers, such as remote computer 2928. These logical connections are achieved by a communication device coupled to, or a part of, the computer 2902. Embodiments are not limited to a particular type of communications device. The remote computer 2928 can be another computer, a server, a router, a network PC, a client, a peer device or other common network node. The logical connections depicted in FIG. 29 include a local-area network (LAN) 2930 and a wide-area network (WAN) 2932. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets, extranets and the Internet.

[0139] When used in a LAN-networking environment, the computer 2902 and remote computer 2928 are connected to the local network 2930 through network interfaces or adapters 2934, which is one type of communications device 2916. Remote

computer 2928 also includes a network device 2936. When used in a conventional WAN-networking environment, the computer 2902 and remote computer 2928 communicate with a WAN 2932 through modems (not shown). The modem, which can be internal or external, is connected to the system bus 2912. In a networked environment, program modules depicted relative to the computer 2902, or portions thereof, can be stored in the remote computer 2928.

[0140] Computer 2902 also includes power supply 2938. Each power supply can be a battery.

Method Embodiments

[0141] In the previous section, table layouts are described. In this section, the particular methods and dataflow diagrams of such embodiments are described by reference to a series of flowcharts. Describing the methods by reference to a flowchart and dataflow diagrams enables one skilled in the art to develop such programs, firmware, or hardware, including such instructions to carry out the methods on suitable computers, executing the instructions from computer-readable media. Similarly, the methods performed by the server computer programs, firmware, or hardware are also composed of computer-executable instructions. Methods 3000-4100 are performed by a program executing on, or performed by firmware or hardware that is a part of, a computer, such as computer 2902 in FIG. 29.

[0142] FIG. 30 is a flowchart of a method 3000 to manage email associations, according to an embodiment.

[0143] Method 3000 includes receiving an email 3002 and determining 3004 whether or not the sender of the email is in a database. In some embodiments, the determination is performed by comparing a sender email address contained in the email to a table in a database, such as the person table 300 in FIG 3. For example, the fields EMAIL 360, EMAIL1 360 and/or EMAIL3 364 in the person table 300 in FIG 3 can be compared to the sender's email address to determine if the sender is in the database.

[0144] If the sender of the email is in the database, then a connection is created 3006 in the database between email and the person.

[0145] FIG. 31 is a flowchart of a method 3100 to provide storage for email data including attachments according to an embodiment. Method 3100 solves the need in the art for more storage of email data including attachments, without having to resort to archival files.

[0146] Method 3100 includes creating 3102 a file structure that is operable to store email attachment data. In some embodiments, that email attachment file structure is a file stored on a disk drive of a computer performing method 3100.

[0147] Method 3100 also includes creating 3104 a file structure that is operable to store email data other than the email attachment data. One such data structure created by action 3104 is a record in the communication table 602 in FIG. 6 above.

[0148] Method 3100 further includes creating 3106 a data structure that is operable to associate each attachment data with an email data. One such data structure created by action 3106 is a record in the attachment table 902 in FIG. 9 above.

[0149] Thereafter, method 3100 includes populating 3108 the file structure that is operable to store email data other than the email attachment data with an email data file other than email attachment data.

[0150] If the email includes an attachment 3110, then method 3100 includes populating 3112 the file structure that is operable to store email attachment data with the email attachment file and populating 3114 the attachment table 900 with an entry related to the email attachment file in the email attachment file structure. Actions 3110-3114 are repeated for as many attachments that are associated with the email data.

[0151] Thereafter, in some embodiments, method 3100 includes retrieving 3116 the email data file from the file structure that is operable to store email data other than the

email attachment data and method 3100 includes retrieving 3118 the email attachment data file from the file structure that is operable to store email attachment data. One embodiment of retrieving 3118 is described below in FIG. 32.

[0152] In essence, method 3100 provides a manner of the storing an attachment of an email, providing on a disk drive of a computer performing method 3100, creating an association of the attachment to the email and later retrieving the attachment in reference to the association.

[0153] In some embodiments of method 3100 in FIG. 31 above and method 3200 in FIG. 32 below, the subject of each method is an embedded graphic file instead of an attachment. An email communication may be associated with an embedded graphic file. In these embodiments, the graphic file is treated substantially similar to the attachment; more specifically, the embedded graphic file is stored on a disk drive of a computer performing method 3100, an association between the embedded graphic file and the email is created and later the embedded graphic file is retrieved from the disk drive in reference to the association.

[0154] FIG. 32 is a flowchart of a method 3200 to retrieve an email attachment according to an embodiment. Method 3200 solves the need in the art for more storage of email data including attachments, without having to resort to archival files. Method 3200 is one embodiment of retrieving 3118 an email attachment in FIG. 3100 above.

[0155] Method 3200 includes receiving 3202 a communication identification and retrieving 3204 a location associated with the communication identification. In some embodiments, the communication identification is retrieved from a data structure that associates the attachment data with the email data such as an entry in the attachment table 900. Thereafter, method 3200 includes retrieving 3206 the attachment from a file structure that is operable to store email attachment data in reference to the location.

[0156] FIG. 33 is a flowchart of a method 3300 to manage auto-receive according to an embodiment. Method 3300 provides a means of determining from a user a number of

parameters pertinent to the email client and performing periodic queries for emails that have not yet been downloaded from an email server to a computer executing the email client.

[0157] Method 3300 includes receiving 3302 an indication of the status of checkbox that represents a binary condition. Various embodiments of the data of the auto-receive status include ON/OFF, TRUE/FALSE, CHECKED/UNCHECKED and 0/1. The auto-receive status indicates whether or not auto-receiving of emails is enabled. The indication of the status can be solicited from the user via a checkbox via a graphical user interface.

[0158] Method 3300 also includes receiving 3304 an integer value representing the frequency of auto-receiving emails. The units of time of the frequency is predetermined and communicated to a user via a graphical user interface.

[0159] Method 3300 also includes storing 3306 the auto-receive status and the auto-receive frequency. In some embodiments, the auto-receive parameters are stored in the mdefault table 2402 in FIG. 24. Thereafter, the auto-receive data is available for access and reference by the email client.

[0160] FIG. 34 is a data flow diagram 3400 to manage attachments according to an embodiment. Data flow diagram 3400 solves the need in the art for more storage of email data including attachments, without having to resort to archival files.

[0161] Data flow diagram 3400 includes an attachment 3402 of an email that is stored on a mass storage device 3404 in data file structure that is separate from a body of the email. Data describing the attachment 3402 (including the location of the attachment 3402 on the mass storage device 3404) is stored in a record in an attachment table 902 and data describing the email is stored in a record of a communication table 602 (including the location of the record in the attachment table 902. Thus, the attachment 3402 is stored separately from the email on the mass storage device 3402 and the attachment 3402 is associated with the email through the communication table 602. In

some embodiments, the location of the attachment 3402 is stored in the communication table 602.

[0162] Thus the data flow diagram 3400 supports storage of attachments outside of the email database, which allows the only the emails to be stored in the email database, which in turn reduces, if not eliminates, the amount of storage in the email database required to store attachments. This allows many more emails to be stored in the email database and the attachments to be stored elsewhere, such as on the disk drive of the computer, which has a tremendously larger amount of storage space. For example, on a computer running the Microsoft Windows® operating system, the maximum file size of an email database file is often two gigabytes, while the computer will often have ten to sixty gigabytes of unused hard disk space. Storing the attachments in the unused hard disk space separately from the email database not only provides a means to make the much larger amount of unused disk space available to store attachments, but also provides a means to use the email database only for emails, which allows more emails to be stored in the email database. Thus data flow diagram 3400 solves the need in the art for more storage for email data including attachments, without having to resort to archival files.

[0163] FIG. 35 is a data flow diagram 3500 to manage embedded graphics according to an embodiment. Data flow diagram 3500 solves the need in the art for more storage of email data including embedded images, without having to resort to archival files.

[0164] Data flow diagram 3500 includes an embedded image 3502 of an email that is stored on a mass storage device 3404 in data file structure that is separate from the email. Data describing the embedded image 3502 (including the location of the embedded image 3502 on the mass storage device 3404) is stored in a record in an attachment table 902 and data describing the email is stored in a record of a communication table 602 (including the location of the record in the attachment table 902. Thus, the embedded image 3502 is stored separately from the email on the mass storage device 3502, the embedded image 3502 is associated with the email through the

communication table 602. In some embodiments, the location of the embedded image 3502 is stored in the communication table 602.

[0165] Thus the data flow diagram 3500 supports storage of embedded images outside of the email database, which allows the only the emails to be stored in the email database, which in turn reduces, if not eliminates, the amount of storage in the email database required to store embedded images. This allows many more emails to be stored in the email database and the embedded images to be stored elsewhere, such as on the disk drive of the computer, which has a tremendously larger amount of storage space. For example, on a computer running the Microsoft Windows® operating system, the maximum file size of an email database file is often two gigabytes, while the computer will often have ten to sixty gigabytes of unused hard disk space. Storing the embedded images in the unused hard disk space separately from the email database not only provides a means to make the much larger amount of unused disk space available to store embedded images, but also provides a means to use the email database only for emails, which allows more emails to be stored in the email database. Thus data flow diagram 3500 solves the need in the art for more storage for email data including embedded images, without having to resort to archival files.

[0166] FIG. 34 and FIG. 35 show that embedded images and attachments can be managed identically, using the same data structures. In some embodiments, an embedded image 3502 from an email is stored in a same subdirectory as an attachment 3402 from the same email.

[0167] FIG. 36 is a flowchart of a method 3600 to import an email communication according to an embodiment.

[0168] Method 3600 includes receiving and storing 3602 raw text of an email and creating a record in a communication table, such as communication table 602. In some embodiments, the received email in one of a plurality of emails in a .mbx file, a .csv file or an .eml file.

[0169] Method 3600 also includes storing 3604 the email data in a communication table, such as communication table 602 that is described above. In some embodiments, the storing 3604 includes creating an entry in the communication table and copying the email data to the entry.

[0170] Method 3600 also includes associating 3606 the stored email data with a person. One example of associating 3606 includes updating a field in the stored communication entry indicating a person indicated by a "emailfromaddress" who is represented in a entry in a person table 302.

[0171] Method 3600 also includes associating 3608 the stored email data with a category. One example of associating 3608 includes updating a field in the stored communication entry indicating a category that is described in an entry in a category table 202.

[0172] FIG. 37 is a data flow diagram 3700 to import data into an email client according to an embodiment.

[0173] Data flow diagram 3700 includes contact data 3702 and email data 3704 that is received by the email client (not shown) in association with a particular user of the email client, such as User-A 3706. In some embodiments, the contact data 3702 and the email data 3704 is in the same file format as the received email in FIG. 36, such as in one of a plurality of emails in a .mbx file, a .csv file or an .eml file.

[0174] Depending upon which of a number of databases User-A 3706 is logged into, such as SQL Server Database-A 3708 or SQL Server Database-B 3710, the contact data 3702 and email data 3704 will be stored into the database that the user is logged into. Any number of users, such as User-B, User-C or User-D can be associated with any number of databases, such as Database-C, Database-D or Database-E. In some embodiments, the method of FIG. 36 is performed during the storing in FIG. 37.

[0175] FIG. 38 is a dataflow diagram 3800 to manage email communications

according to an embodiment. Dataflow diagram 3800 provides a flexible hierarchy of categories, the hierarchy providing an unlimited number of levels within the hierarchy of categories. The hierarchy is accomplished by providing a linked list of records in a category table.

[0176] In the embodiment of linked category records shown in FIG. 38, all but a final parent record links to the immediate parent record of each category record. In the example shown in FIG. 38, final parent record for category table record #A 3802 is linked by a child category table record #B 3804. This pattern repeats for as many successive child category table records as necessary. More specifically, parent record for category table record #B 3804 is linked by a child category table record #C 3806, parent record for category table record #C 3806 is linked by a child category table record #D 3808.

[0177] In some embodiments, dataflow diagram 3800 includes a treeview control (not shown) to manage the hierarchy of category records. The treeview control recursively traverses the hierarchy. In a communication table, records are linked to category records, that are in turn linked to other category records through a parent filed that contains the Id number of the patent category. In a person table, records are linked to categories, but the categories are linked to parent categories through the parent field of the category record.

[0178] FIG. 39 is a dataflow diagram 3900 to manage multiple users according to an embodiment.

[0179] Dataflow diagram 3900 includes a User-A 3706 who accesses a Server Database-A 3708 and/or SQL Server Database-B 3710. Dataflow diagram also includes User-B 3902 who accesses the Server Database-B 3710 and/or SQL Server Database-C 3904.

[0180] In some embodiments, the User-A 3706 is named "admin" and the User-B 3902 is named "user."

[0181] Dataflow diagram 3900 shows two different embodiments of database access. In the first embodiment, User-A 3706 has exclusive access to Server Database-A 3708 and User-B 3902 has exclusive access to SQL Server Database-C 3904. In the second embodiment, User-A 3706 and User-B 3902 share access to SQL Server Database-B 3710.

[0182] When a user is added to a setup table, such as setup table 1902, a database name, servername and password are added to that user for a new database or one that already exists. If a non-existent database is specified, that database is created, and then the database is connected to servername and password.

[0183] In some embodiments, any person can have multi-databases: personal, business, etc. For example, one user can have and access a personal database of emails and people, and then at some point, leave the program, and log into another business database that is there database also, (separate from the personal database), so no mixing of emails. Access to each database is through the login that allows entry. In another example, any set of users can share one database in which different users can use the same database by just both submitting the login credentials to the email client. In another example, all users can have separate databases, in which each user has their own database so that no other user could delete their emails. In another example, each user can import to their database connecting emails to any topic, in which a user logs into any database, that user will have the authorization to import into that database. In another example, when a user logs in as a user with access to a database, all importing by the email client directed by the user do applies to the database the user has the ability to which to connect.

[0184] FIG. 40 is a flowchart of a method 4000 to create an executable managed code email client according to an embodiment. Method 4000 solves the need in the art for an operationally stable email client.

[0185] Managed code is executable computer instructions whose execution is managed by a runtime module, such as the Microsoft .NET Framework Common

Language Runtime (CLR). The runtime module refers to an architecture of cooperation between natively executing instructions and the runtime module. This architecture specifies that at any point of execution, the runtime module may stop an executing processor (CPU) and retrieve information specific to the current CPU instruction address. Information that must be query-able generally pertains to runtime state, such as register or stack memory contents.

[0186] Method 4000 includes encoding 4002 source code of an email client in an intermediate language (IL). An IL describes how the information is to be encoded, and programming languages that target the runtime module emit the correct encoding. One embodiment of an IL is the Common Language Infrastructure (CLI) Standard published by the International Organization for Standardization (ISO) in Geneva Switzerland, of which the CLR is the primary commercial implementation.

[0187] In addition, infrastructure data such as associated metadata, or symbolic information that describes all of the entry points and the constructs exposed in the IL (e.g., methods, properties) and their characteristics, are encoded 4004 in the IL.

[0188] The IL instructions are compiled 4006 into native executable instructions. Because the compiling 4006 is performed by the managed execution environment (or, in some embodiment, by a runtime-aware compiler that accesses information on how to target the managed execution environment), the managed execution environment can provide information about what function the instructions will perform. The managed execution environment can insert traps and appropriate garbage collection hooks, exception handling, type safety, array bounds and index checking. For example, such a compiler creates stack frames so that a garbage collector can run in the background on a separate thread, constantly traversing the active call stack, finding all the roots, identifying all of the live objects. In addition because the IL instructions have information describing type safety, an execution engine will maintain type safety eliminating some programming mistakes that often lead to security holes. The managed execution provides an operationally stable email client.

[0189] FIG. 41 is a flowchart of a method 4100 to prioritize email communications according to an embodiment. Method 4100 solves the need in the art for improved sorting/prioritization of emails.

[0190] Method 4100 includes receiving 4102 an email communication. A numeric priority field is set to numeric "0" in all digits of the priority field. The leading digit (the first digit) represents the overall importance of the email communication. A second digit represents the person importance of the email communication, and a third digit represents the email priority of the email communication. Some embodiments of the priority field comprise 3 digits, as the digits described above, other embodiments include a subset of the above described digits, other embodiments includes the above three digits and other digits, and other embodiments include a subset of the above three digits and other digits.

[0191] A determination 4104 as to whether or not the email communication is a "To Do" communication or whether or not the email communication was created from a compose screen of the email client. If either determination is true, the first digit in a numeric priority field is set 4106 to numeric 1. If both determinations 4104 are false, then a determination 4108 is performed as to whether or not the value in the "to" or the "from of the email communication is found in a database of email addresses other than the owner of the email client, the owner of the email client being indicated in the 1st entry in the database in some embodiments. If determination 4108 is true, then the first digit in the numeric priority field is set 4110 to numeric 2.

[0192] Thereafter a determination 4112 as to whether or not the email communication is related to a person in the database. If the determination 4112 is true, then the second digit in the priority field is set 4114 to the priority of that person, otherwise the second digit in the priority field is set 4116 to 5.

[0193] Thereafter, a determination 4118 as to whether or not the email communication has a priority is made. If the determination 4118 is true, then the third digit in the priority field is set 4120 to the priority of the email communication. If the

determination 4118 is false, then the third digit in the priority field is set 4122 to an average, middle, representation, such as numeric 3.

[0194] Thus, a priority of each email communication is set or determined. The emails can be displayed by the email client in descending numeric order indicated by the priority field.

[0195] In some embodiments, methods 3000-4100 are implemented as a computer data signal embodied in a carrier wave that represents a sequence of instructions which, when executed by a processor, such as processor 2904 in FIG. 29, cause the processor to perform the respective method. In other embodiments, methods 3000-4100 are implemented as a computer-accessible medium having executable instructions capable of directing a processor, such as processor 2904 in FIG. 29, to perform the respective method. In varying embodiments, the medium is a magnetic medium, an electronic medium, or an optical medium.

Conclusion

[0196] An email client is described. Although specific embodiments have been illustrated and described herein, it will be appreciated by those of ordinary skill in the art that any arrangement which is calculated to achieve the same purpose may be substituted for the specific embodiments shown. This application is intended to cover any adaptations or variations. For example, although described in procedural terms, one of ordinary skill in the art will appreciate that implementations can be made in an object-oriented design environment or any other design environment that provides the required relationships.

[0197] In particular, one of skill in the art will readily appreciate that the names of the methods and apparatus are not intended to limit embodiments. Furthermore, additional methods and apparatus can be added to the components, functions can be rearranged among the components, and new components to correspond to future enhancements and physical devices used in embodiments can be introduced without

departing from the scope of embodiments. One of skill in the art will readily recognize that embodiments are applicable to future communication devices, different file systems, and new data types.

[0198] The terminology used in this application is meant to include all object-oriented, database and communication environments and alternate technologies which provide the same functionality as described herein.

CLAIMS

We claim:

1. An apparatus to provide a reliable apparatus for managing email data, the apparatus comprising:

a first client apparatus operable to store at least a portion of the email data through an interface to a database manager; the database manager having a high level security manager; and

a second client apparatus operable to retrieve email data through the interface to the database manager,

wherein the first client apparatus and the second client apparatus are operable to communicate directly to the interface to the database manager, the apparatus not including a component between the first client apparatus and the second client apparatus and the interface to the database manager.

2. The apparatus of claim 1 further comprising:

an apparatus operable to receive the at least a portion of the email data from a network.

3. A apparatus to provide a stable email client program, the apparatus comprising:

an interface to a database manager, the database manager having a high level security manager; and

a client program operably coupled to the database manager, the client program operable to receive the at least a portion of the email data from a network, store the at least a portion of the email data through the interface to the database manager and retrieve the at least a portion of the email data through the interface to the database manager,

wherein the client program is operable to communicate directly to the interface of the database manager, the apparatus not including a component between

the client program and the interface to the database manager.

4. A apparatus to provide a reliable apparatus of managing email data, the apparatus comprising:

an executable managed code apparatus operable to store at least a portion of the email data through an interface to a relational database manager; and

an executable managed code apparatus operable to retrieve the at least a portion of the email data through the interface to the relational database manager.

5. A apparatus to provide a reliable apparatus of managing email data, the apparatus comprising:

an executable managed code apparatus operable to store at least a portion of the email data through an interface to a structured-query-language database manager; and

an executable managed code apparatus operable to retrieve at least a portion of the email data through the interface to the structured-query-language database manager.

6. A method to manage email associations, the method comprising:

receiving an email;

determining that a sender of the email is stored in a database; and

creating an association in the database between the email and the sender.

7. The method of claim 7, wherein the determining further comprises:

comparing a sender email address contained in the email to a table in a database.

8. A method to provide storage for email data including attachments, the method comprising:

creating a first file structure that is operable to store email attachment data; and

creating a second file structure that is operable to store email data other than the email attachment data.

9. The method of claim 8 further comprising:
populating the second file structure with an email data file other than email attachment data.

10. The method of claim 9 further comprising:
retrieving the email data file from the second file structure.

11. The method of claim 9 further comprising:
creating a data structure that is operable to associate each attachment data with an email data.

12. The method of claim 9 further comprising:
populating the first file structure with an email attachment file.

13. The method of claim 12 further comprising:
retrieving the email attachment data file from the first file structure.

14. The method of claim 13, wherein the retrieving further comprises:
receiving a communication identification;
retrieving the location associated with the communication identification from a data structure that associates the attachment data with the email data; and
retrieving the attachment from the first file structure in reference to the location.

15. A method to manage at least one attachment of an email communication, the method comprising:
storing a body of the email communication in a first data file structure; and

storing the at least one attachment in a second data file structure.

16. The method of claim 21 further comprising:
storing data describing the location of the body of the email communication; and
storing data describing the location of the least one attachment.

17. The method of claim 22, wherein the storing data describing the location of the body of the email communication further comprises:
storing data describing the location of the body of the email communication in a record of a communication table, and
wherein the storing data describing the location of the least one attachment further comprises:
storing data describing the location of the least one attachment in a record in an attachment table.

18. The method of claim 21 further comprising:
storing data describing the location of the first data file structure; and
storing data describing the location of the second data file structure.

19. A method to manage at least one embedded image of an email communication, the method comprising:
storing a body of the email communication in a first data file structure; and
storing the at least one embedded image in a second data file structure.

20. The method of claim 25 further comprising:
storing data describing the location of the body of the email communication; and
storing data describing the location of the least one embedded image.

21. The method of claim 26, wherein the storing data describing the location of the

body of the email communication further comprises:

storing data describing the location of the body of the email communication in a record of a communication table, and

wherein the storing data describing the location of the least one embedded image further comprises:

storing data describing the location of the least one embedded image in a record in an embedded image table.

22. The method of claim 25 further comprising:

storing data describing the location of the first data file structure; and

storing data describing the location of the second data file structure.

23. A method to import an email communication, the method comprising:

receiving raw text of an email communication;

storing the raw text of an email communication; and

storing the raw text of the email communication in a record of a communication table; wherein the record of the communication table further comprises:

a field storing data representing an indexed key field representing an identification of a communication; and

a field storing data representing identification of a person to whom the communication was sent.

24. The method of claim 23 further comprising:

associating the stored raw text of the email communication with a person.

25. The method of claim 24, wherein the associating the stored raw text of the email communication with the person further comprises:

updating a field in the stored communication entry indicating a person indicated by a "emailfromaddress" who is represented in a entry in a person table.

26. The method of claim 23 further comprising:
associating the stored raw text of the email communication with a category.

27. The method of claim 26, wherein the associating the stored raw text of the email communication with the category further comprises:
updating a field in the stored communication entry indicating a category that is described in an entry in a category table.

28. The method of claim 23, wherein the storing the raw text of the email communication in a record of a communication table further comprises:
creating an entry in the communication table; and
copying the raw text email data to the entry;

29. A method to create an executable managed code email client, the method comprising:
encoding source code of an email client into intermediate language instructions;
encoding infrastructure data into intermediate language instructions, the infrastructure data further comprising associated metadata, and symbolic information that describes all of the entry points and the constructs exposed in the intermediate language; and
compiling the intermediate language instructions into native executable instructions.

30. The method of claim 29, wherein the compiling further comprises:
creating stack frames so that a garbage collector can run in the background on a separate thread, constantly traversing the active call stack, finding all the roots, identifying all of the live objects

31. The method of claim 29, wherein the intermediate language instructions further comprise:

information describing type safety.

32. The method of claim 29, wherein the intermediate language further comprises:

an intermediate language that conforms to the Common Language Infrastructure (CLI) Standard.

33. A method to prioritize an email communication, the method comprising:

setting a leading digit in a numeric priority field to numeric 1, if the email communication is a "To Do" communication or if the email communication was created from a compose screen of an email client;

setting the leading digit in the numeric priority field to numeric 2, if a "to" or a "from" field of the email communication is found in a database of email addresses other than the owner of the email client;

setting a second digit in the numeric priority field to a priority of a person, if the email communication is related to a person in the database;

setting the second digit in the numeric priority field to a numeric 5, if the email communication is not related to a person in the database;

setting a third digit in the numeric priority field to a priority of the email communication, if the email communication includes a priority; and

setting the third digit in the numeric priority field to an average value, if the email communication includes no priority.

34. The method of claim 33, wherein the setting the third digit in the numeric priority field to an average value further comprises:

setting the third digit in the numeric priority field to a numeric 3.

35. A method to prioritize an email communication, the method comprising:

setting a digit in a numeric priority field to an overall importance of the email communication;

setting a digit in a numeric priority field to a person importance of the email communication; and

setting a digit in a numeric priority field to an email importance of the email communication.

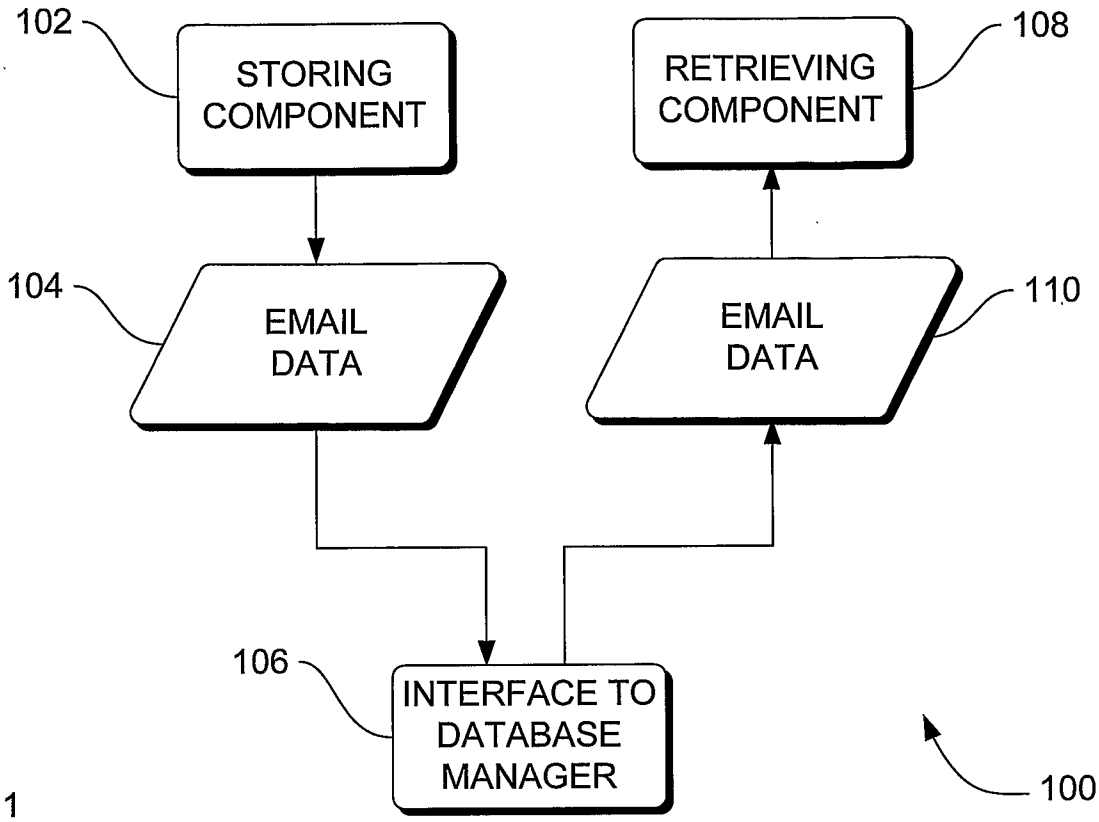

102
STORING COMPONENT
108
RETRIEVING COMPONENT
104
EMAIL DATA
110
EMAIL DATA
106
INTERFACE TO DATABASE MANAGER
100

FIG. 1

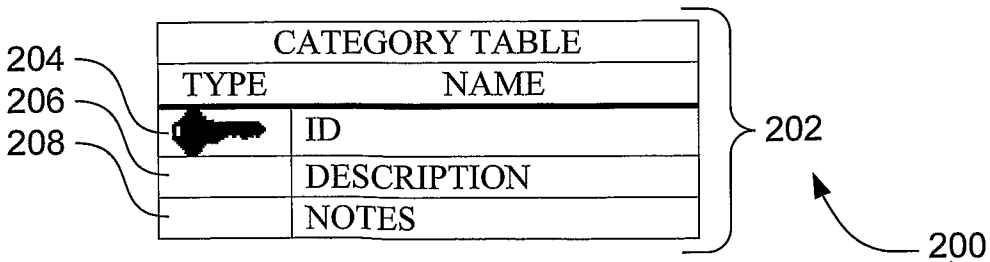

CATEGORY TABLE
TYPE
NAME
ID
DESCRIPTION
NOTES
204
206
208
202
200

FIG. 2

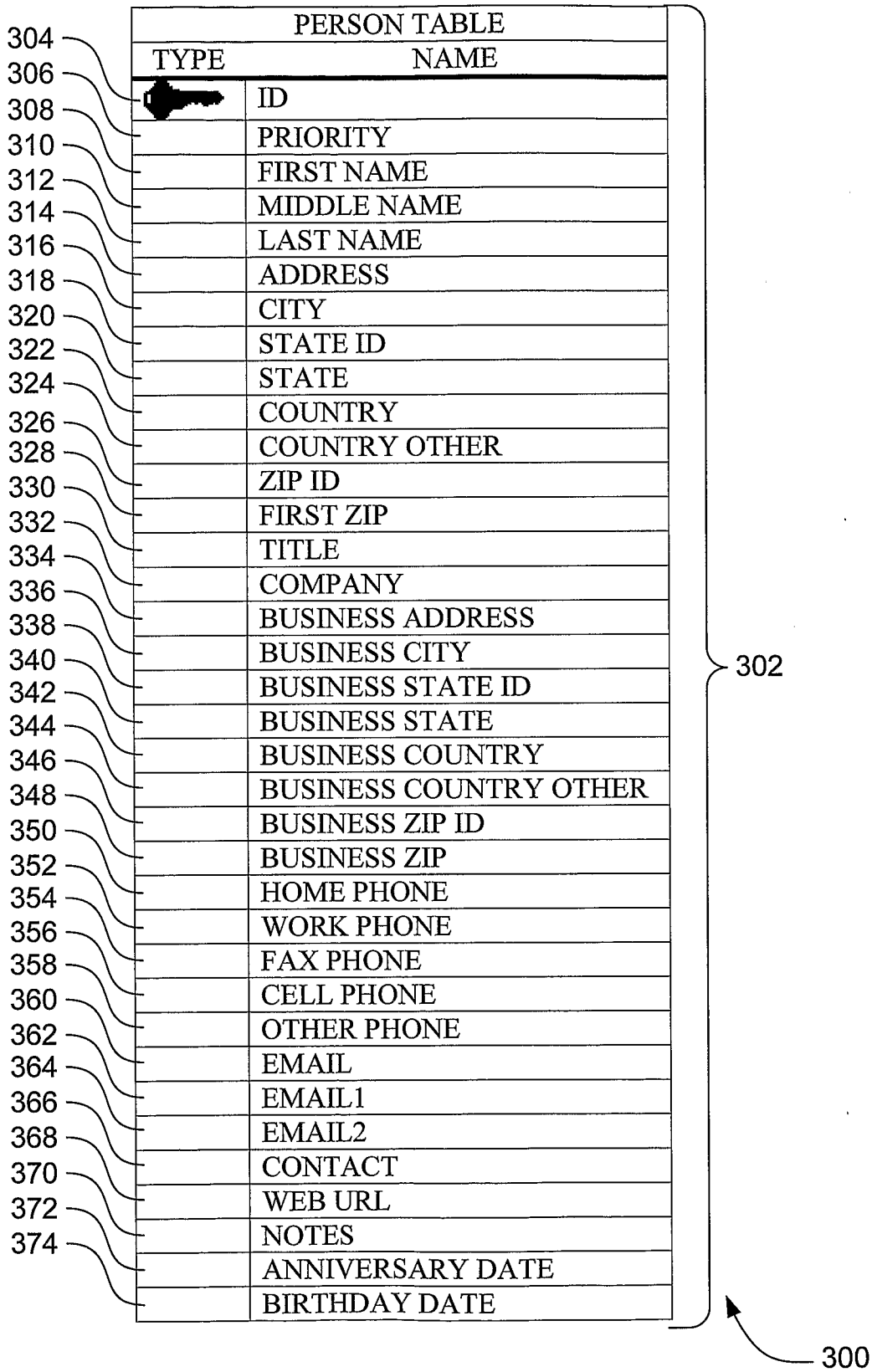

PERSON TABLE
TYPE
NAME
ID
PRIORITY
FIRST NAME
MIDDLE NAME
LAST NAME
ADDRESS
CITY
STATE ID
STATE
COUNTRY
COUNTRY OTHER
ZIP ID
FIRST ZIP
TITLE
COMPANY
BUSINESS ADDRESS
BUSINESS CITY
BUSINESS STATE ID
BUSINESS STATE
BUSINESS COUNTRY
BUSINESS COUNTRY OTHER
BUSINESS ZIP ID
BUSINESS ZIP
HOME PHONE
WORK PHONE
FAX PHONE
CELL PHONE
OTHER PHONE
EMAIL
EMAIL1
EMAIL2
CONTACT
WEB URL
NOTES
ANNIVERSARY DATE
BIRTHDAY DATE
304
306
308
310
312
314
316
318
320
322
324
326
328
330
332
334
336
338
340
342
344
346
348
350
352
354
356
358
360
362
364
366
368
370
372
374
302
300

FIG. 3

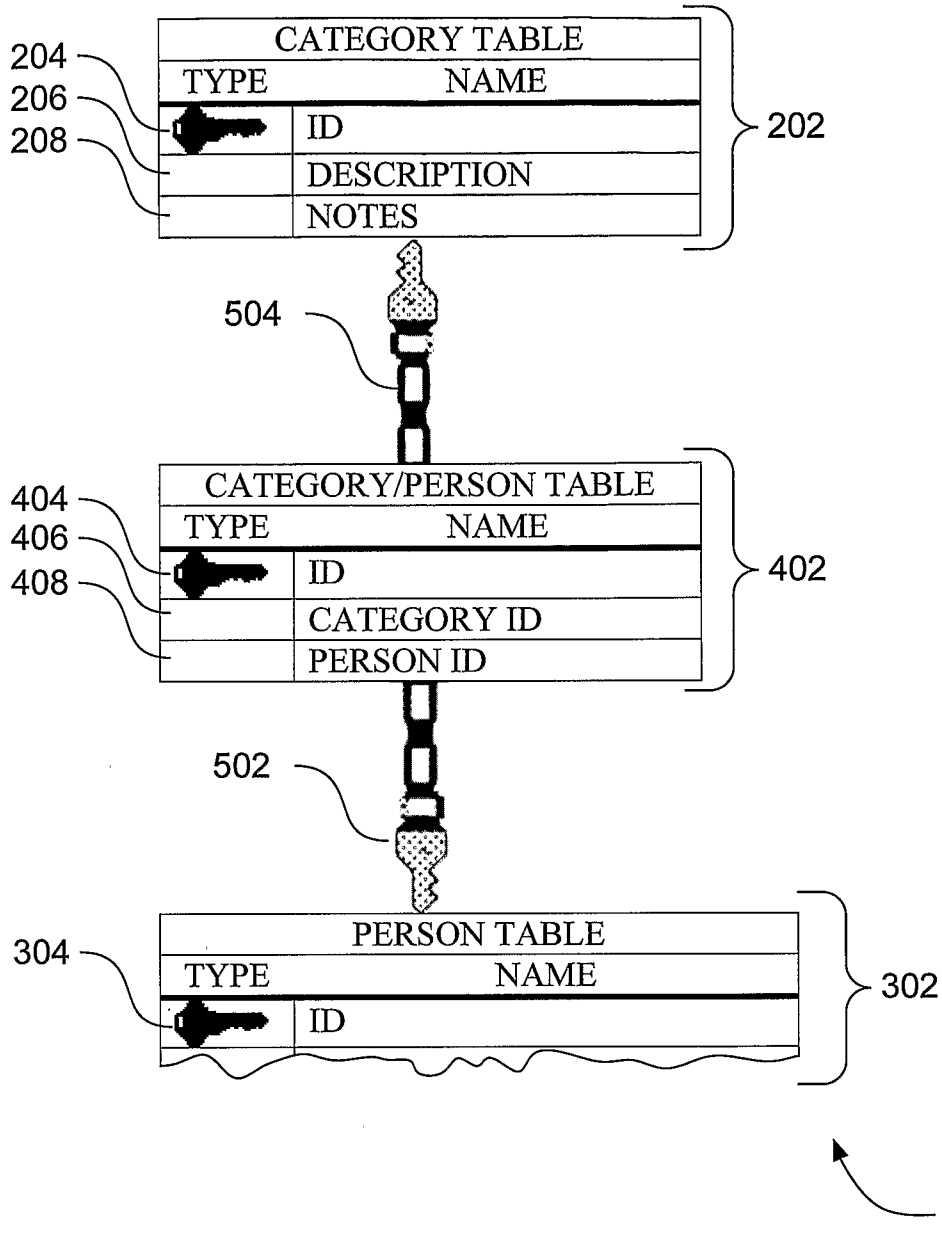

CATEGORY TABLE
TYPE
NAME
ID
DESCRIPTION
NOTES
204
206
208
202
504
CATEGORY/PERSON TABLE
TYPE
NAME
ID
CATEGORY ID
PERSON ID
404
406
408
402
502
PERSON TABLE
TYPE
NAME
ID
304
302
500

FIG. 5

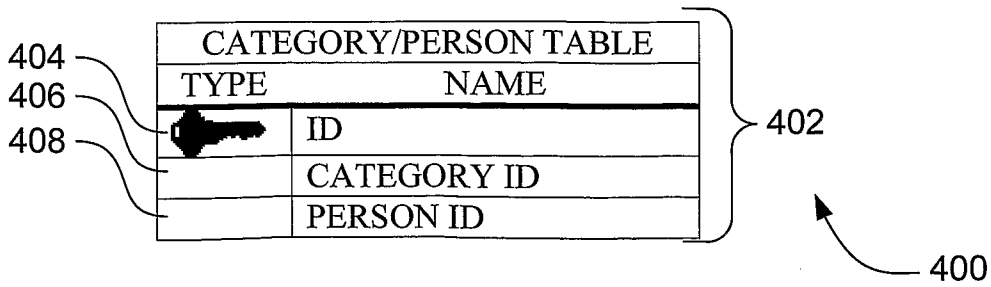

CATEGORY/PERSON TABLE
TYPE
NAME
ID
CATEGORY ID
PERSON ID
404
406
408
402
400

FIG. 4

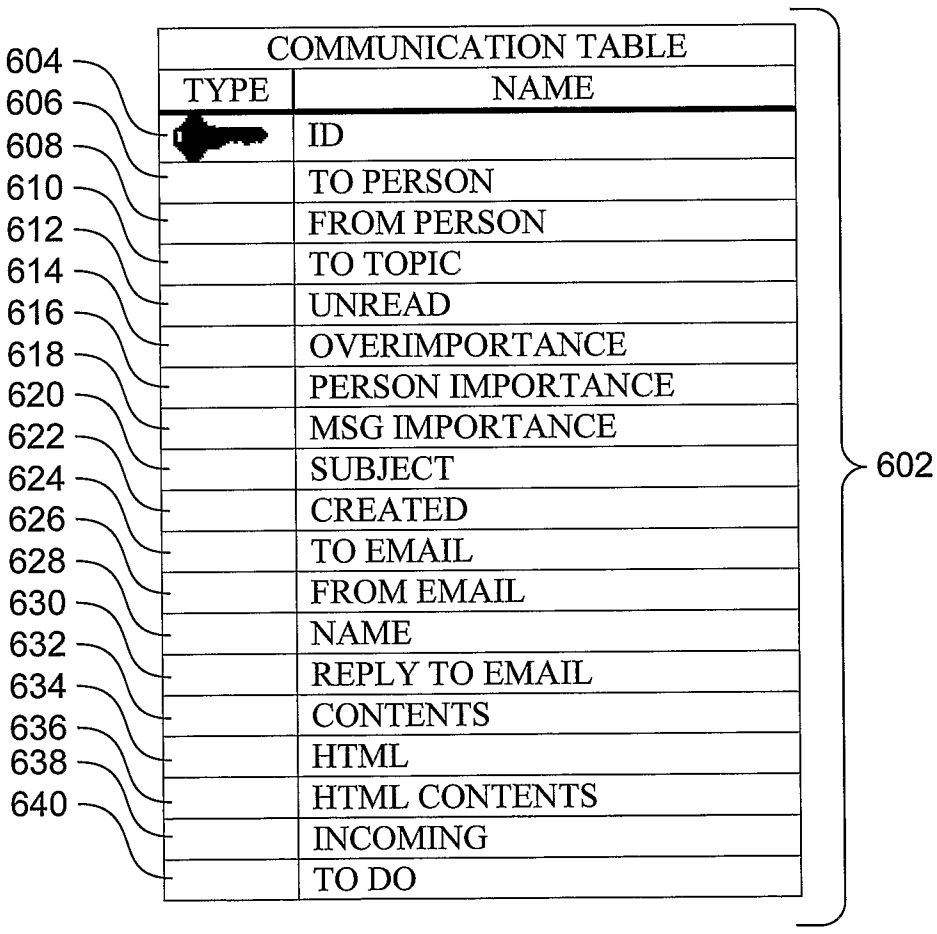

COMMUNICATION TABLE
TYPE
NAME
ID
TO PERSON
FROM PERSON
TO TOPIC
UNREAD
OVERIMPORTANCE
PERSON IMPORTANCE
MSG IMPORTANCE
SUBJECT
CREATED
TO EMAIL
FROM EMAIL
NAME
REPLY TO EMAIL
CONTENTS
HTML
HTML CONTENTS
INCOMING
TO DO
604
606
608
610
612
614
616
618
620
622
624
626
628
630
632
634
636
638
640
602
600

FIG. 6

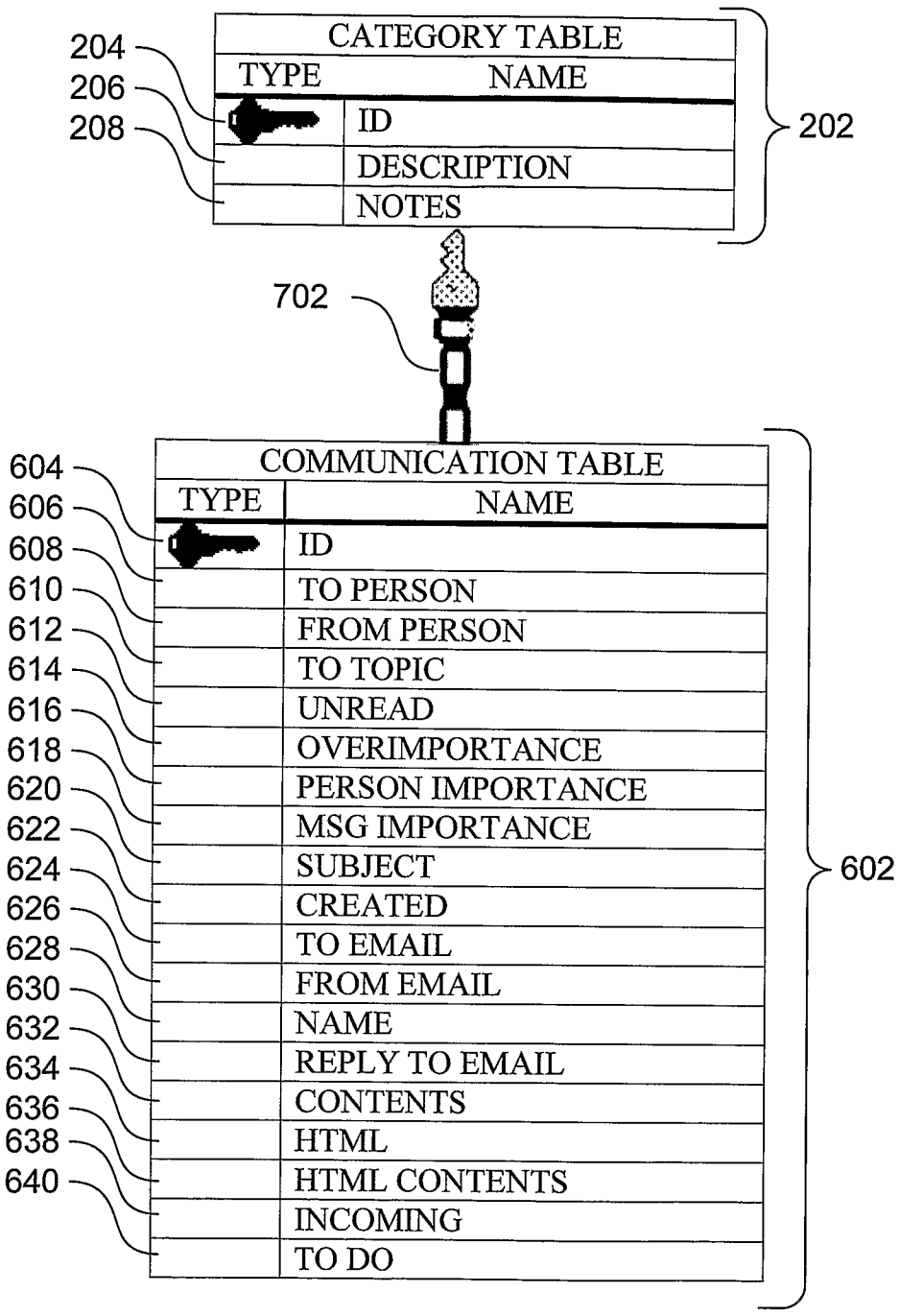

CATEGORY TABLE
TYPE
NAME
ID
DESCRIPTION
NOTES
204
206
208
202
702
COMMUNICATION TABLE
TYPE
NAME
ID
TO PERSON
FROM PERSON
TO TOPIC
UNREAD
OVERIMPORTANCE
PERSON IMPORTANCE
MSG IMPORTANCE
SUBJECT
CREATED
TO EMAIL
FROM EMAIL
NAME
REPLY TO EMAIL
CONTENTS
HTML
HTML CONTENTS
INCOMING
TO DO
604
606
608
610
612
614
616
618
620
622
624
626
628
630
632
634
636
638
640
602

FIG. 7

700

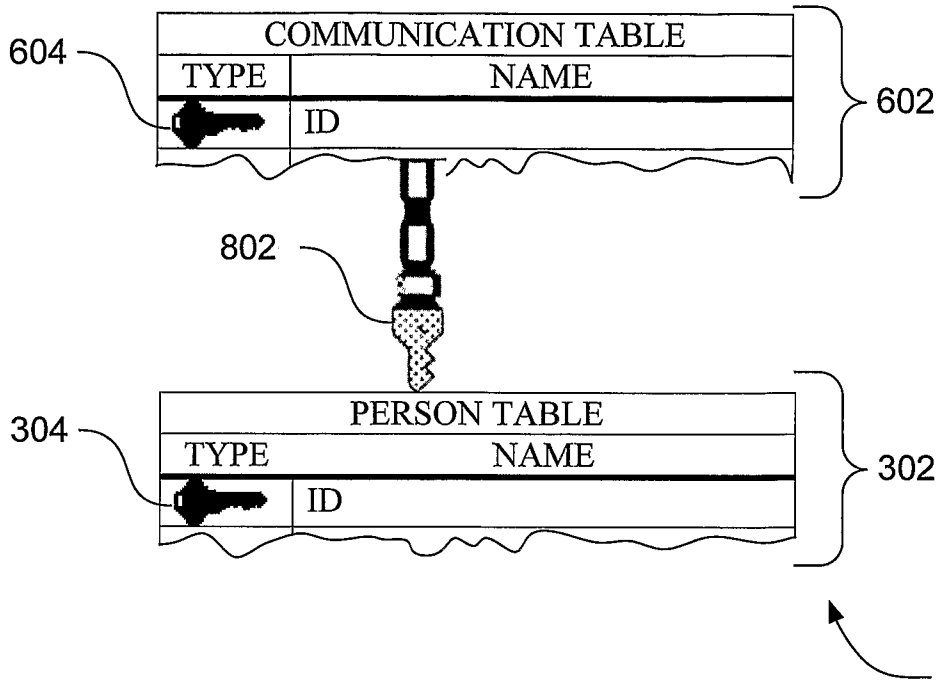

COMMUNICATION TABLE
TYPE
NAME
ID
604
602
802
PERSON TABLE
TYPE
NAME
ID
304
302
800

FIG. 8

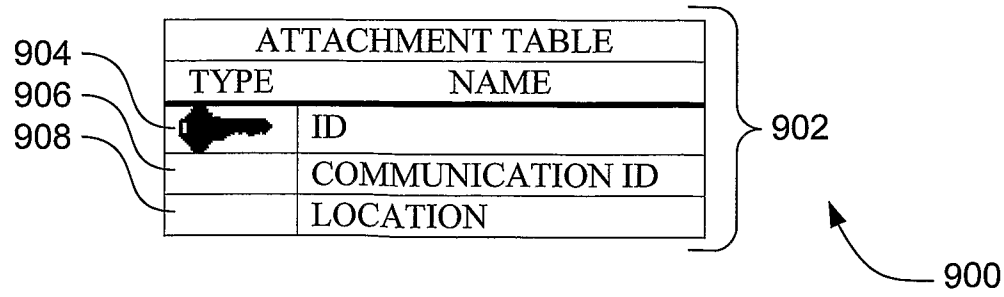

ATTACHMENT TABLE
TYPE
NAME
ID
COMMUNICATION ID
LOCATION
904
906
908
902
900

FIG. 9

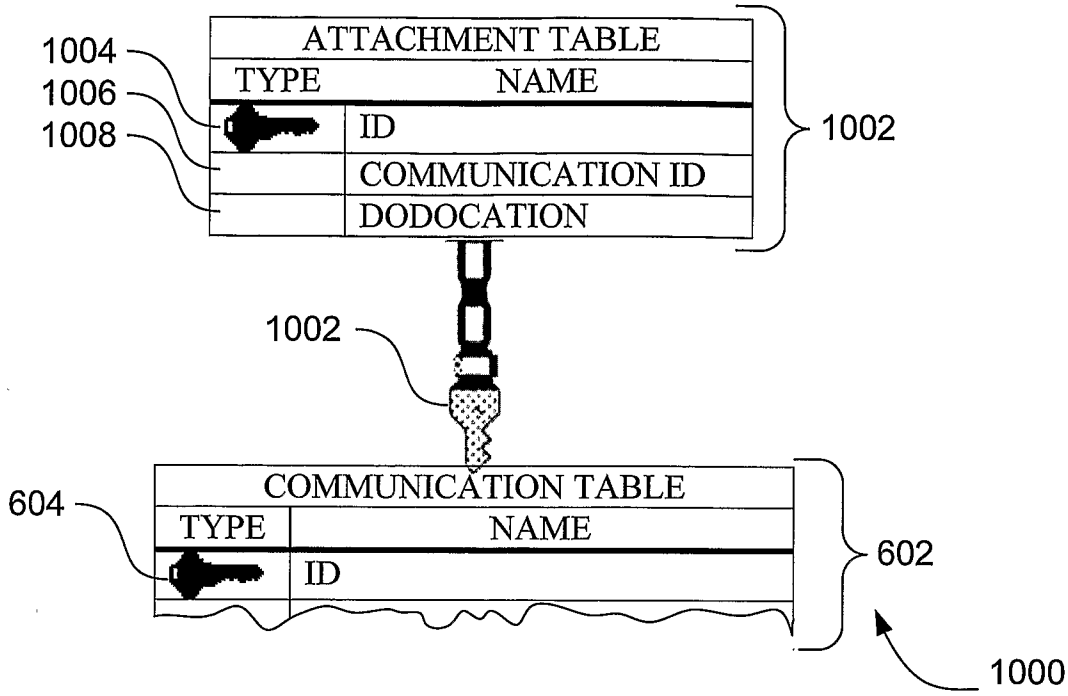

ATTACHMENT TABLE
TYPE
NAME
ID
COMMUNICATION ID
DODOCATION
1004
1006
1008
1002
1002
COMMUNICATION TABLE
TYPE
NAME
ID
604
602
1000

FIG. 10

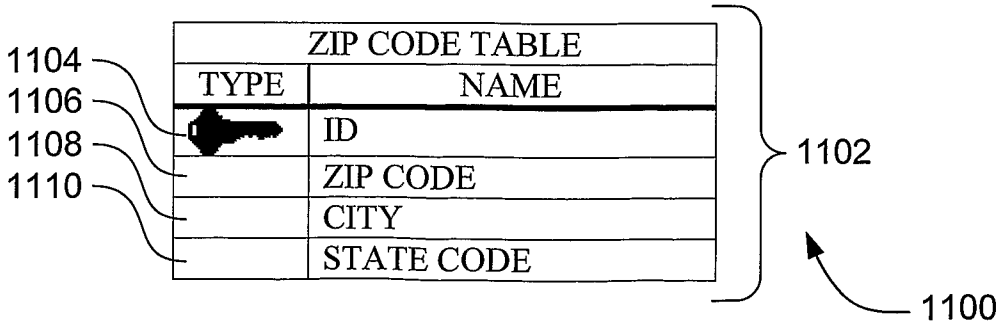

ZIP CODE TABLE
TYPE
NAME
ID
ZIP CODE
CITY
STATE CODE
1104
1106
1108
1110
1102
1100

FIG. 11

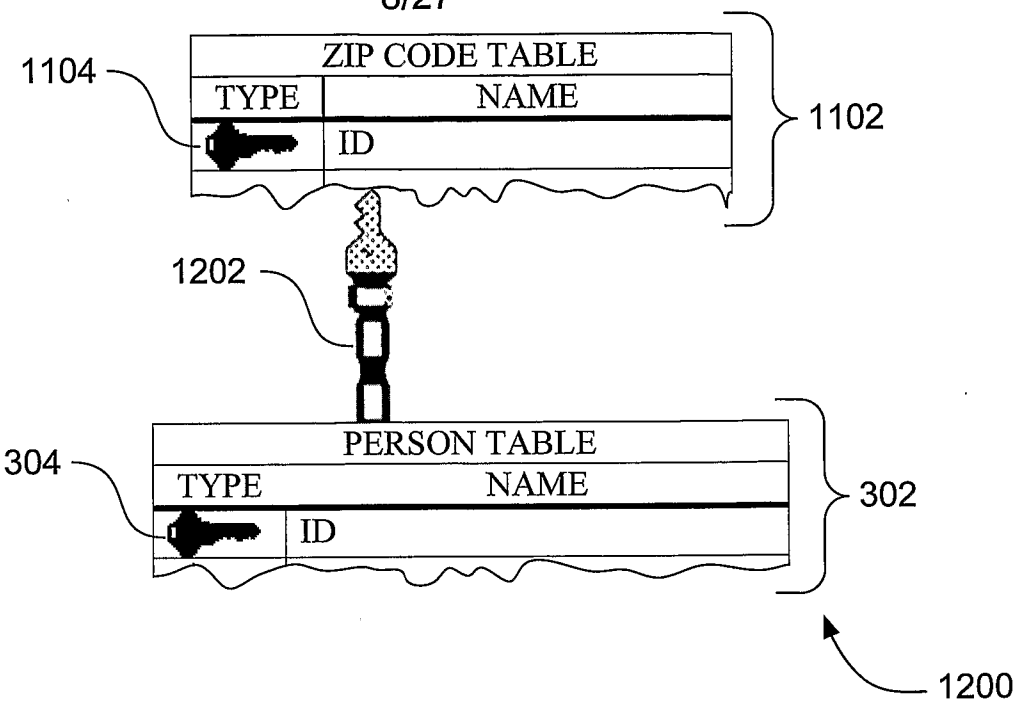

ZIP CODE TABLE
TYPE
NAME
ID
1104
1102
1202
PERSON TABLE
TYPE
NAME
ID
304
302
1200

FIG. 12

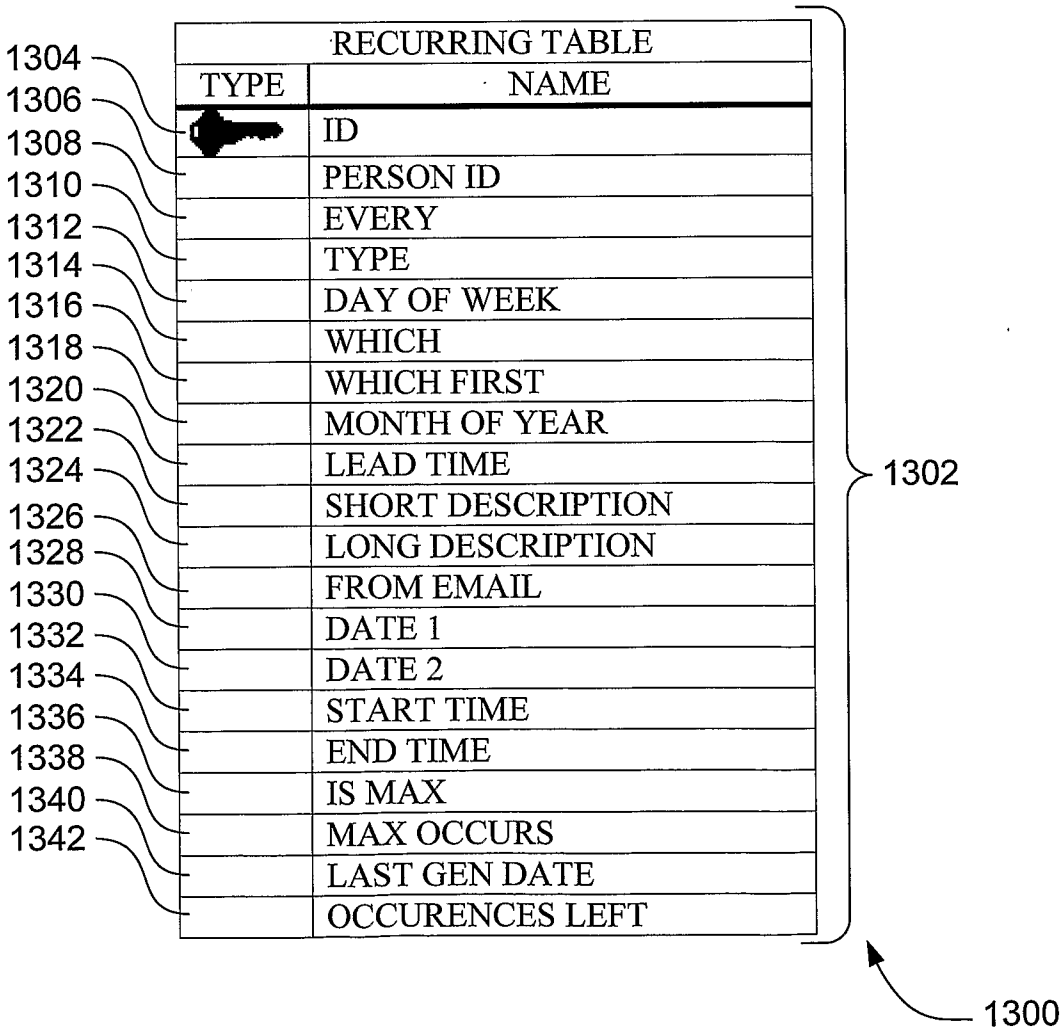

RECURRING TABLE
TYPE
NAME
ID
PERSON ID
EVERY
TYPE
DAY OF WEEK
WHICH
WHICH FIRST
MONTH OF YEAR
LEAD TIME
SHORT DESCRIPTION
LONG DESCRIPTION
FROM EMAIL
DATE 1
DATE 2
START TIME
END TIME
IS MAX
MAX OCCURS
LAST GEN DATE
OCCURENCES LEFT
1304
1306
1308
1310
1312
1314
1316
1318
1320
1322
1324
1326
1328
1330
1332
1334
1336
1338
1340
1342
1302
1300

FIG. 13

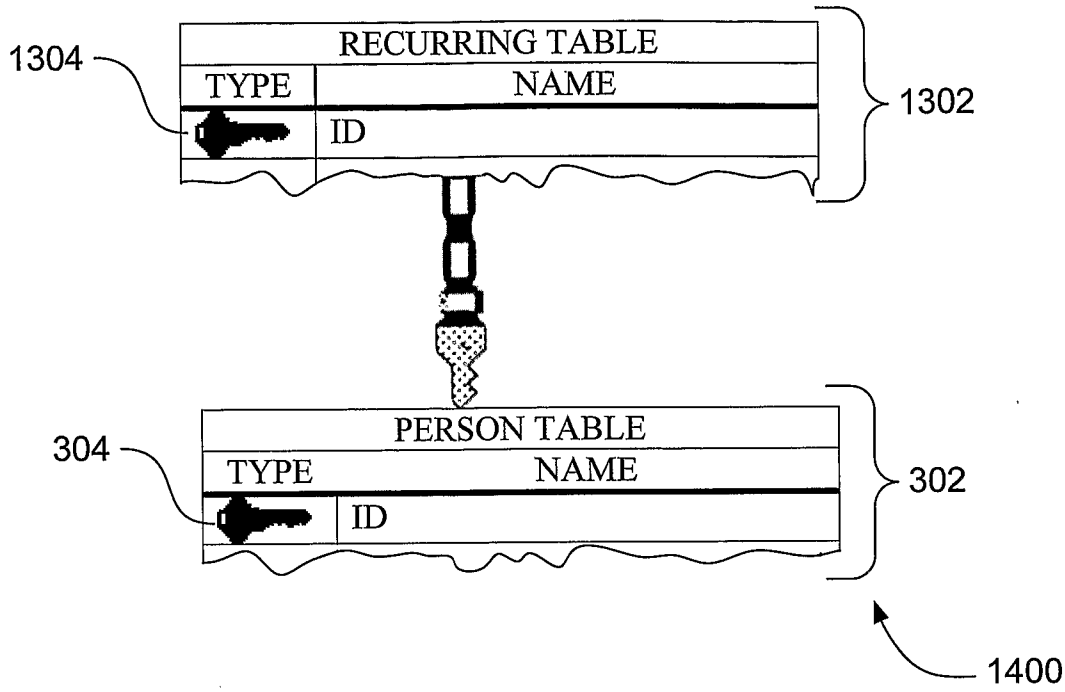

RECURRING TABLE
TYPE
NAME
ID
1304
1302
PERSON TABLE
TYPE
NAME
ID
304
302
1400

FIG. 14

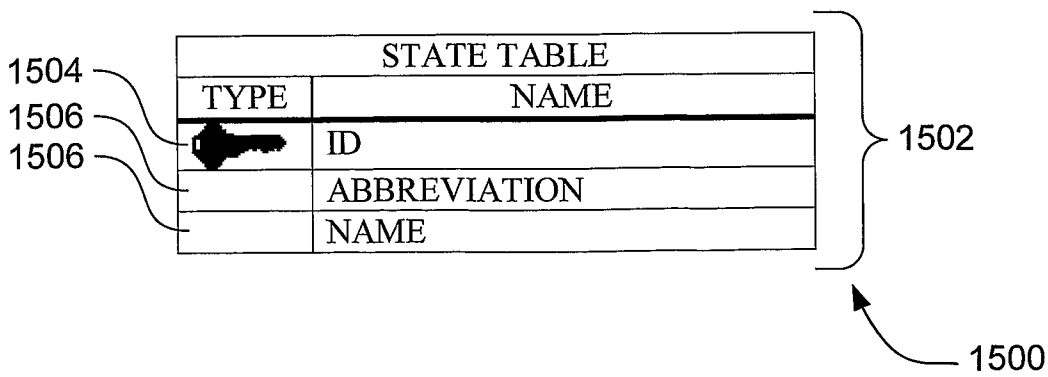

STATE TABLE
TYPE
NAME
ID
ABBREVIATION
NAME
1504
1506
1506
1502
1500

FIG. 15

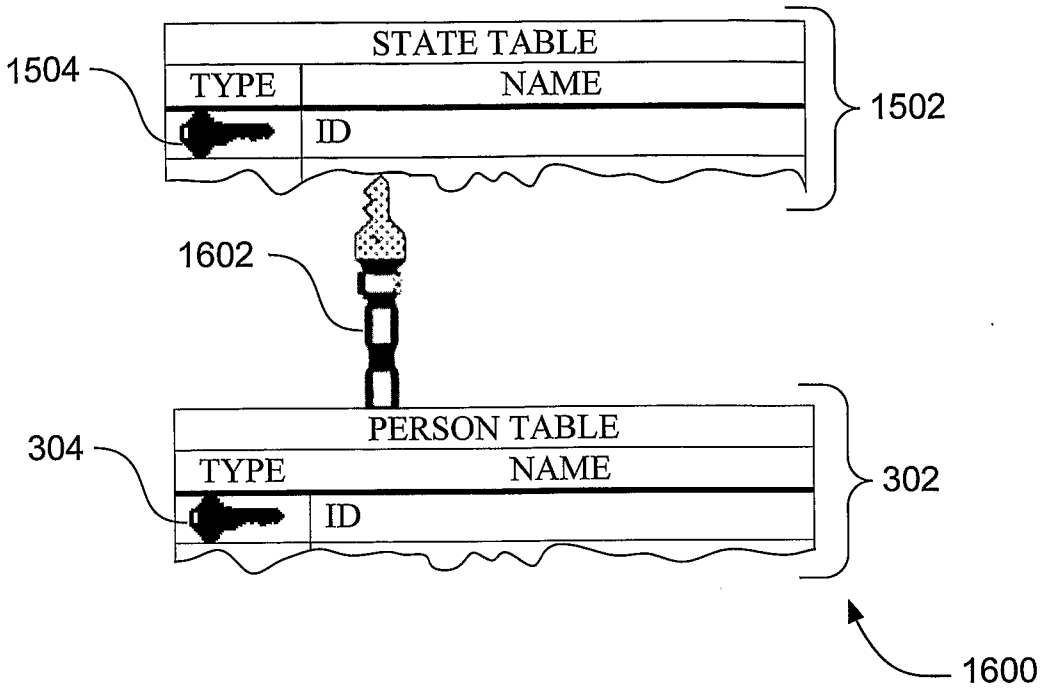

STATE TABLE
TYPE
NAME
ID
1504
1502
1602
PERSON TABLE
TYPE
NAME
ID
304
302
1600

FIG. 16

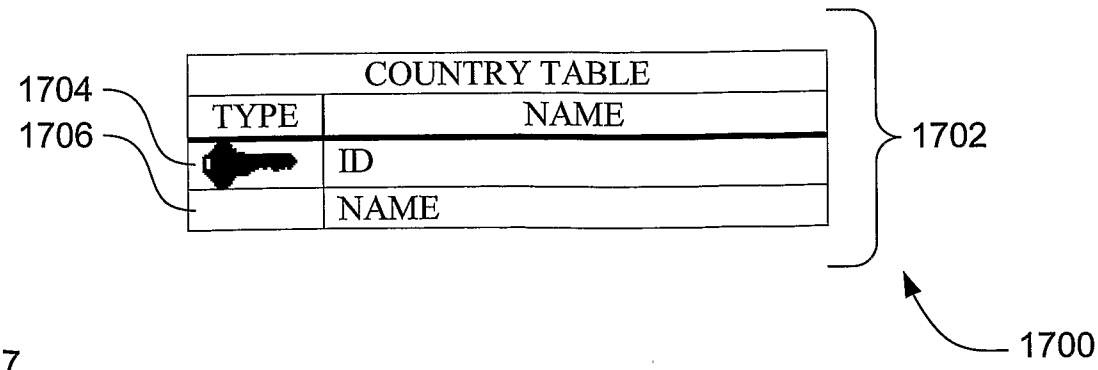

COUNTRY TABLE
TYPE
NAME
ID
NAME
1704
1706
1702
1700

FIG. 17

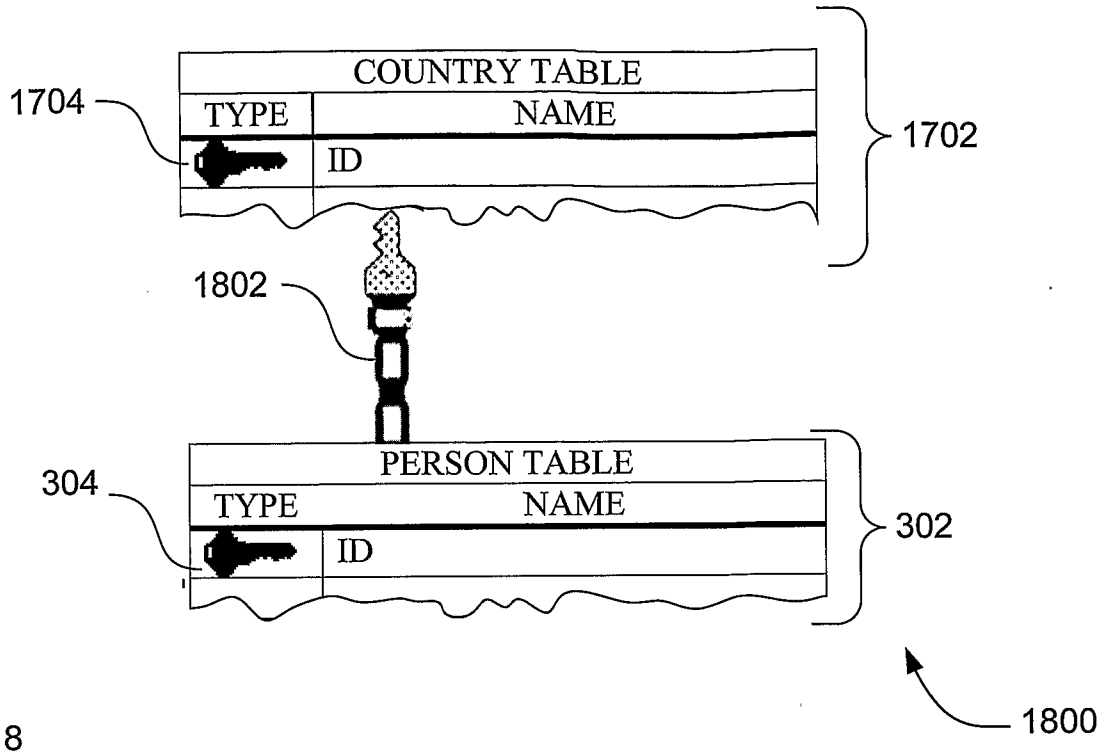

COUNTRY TABLE
TYPE
NAME
ID
1704
1702
1802
PERSON TABLE
TYPE
NAME
ID
304
302
1800

FIG. 18

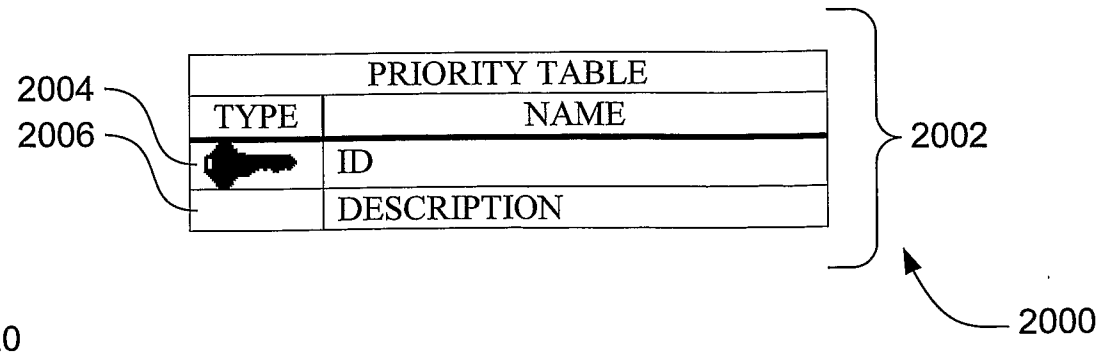

PRIORITY TABLE
TYPE
NAME
ID
DESCRIPTION
2004
2006
2002
2000

FIG. 20

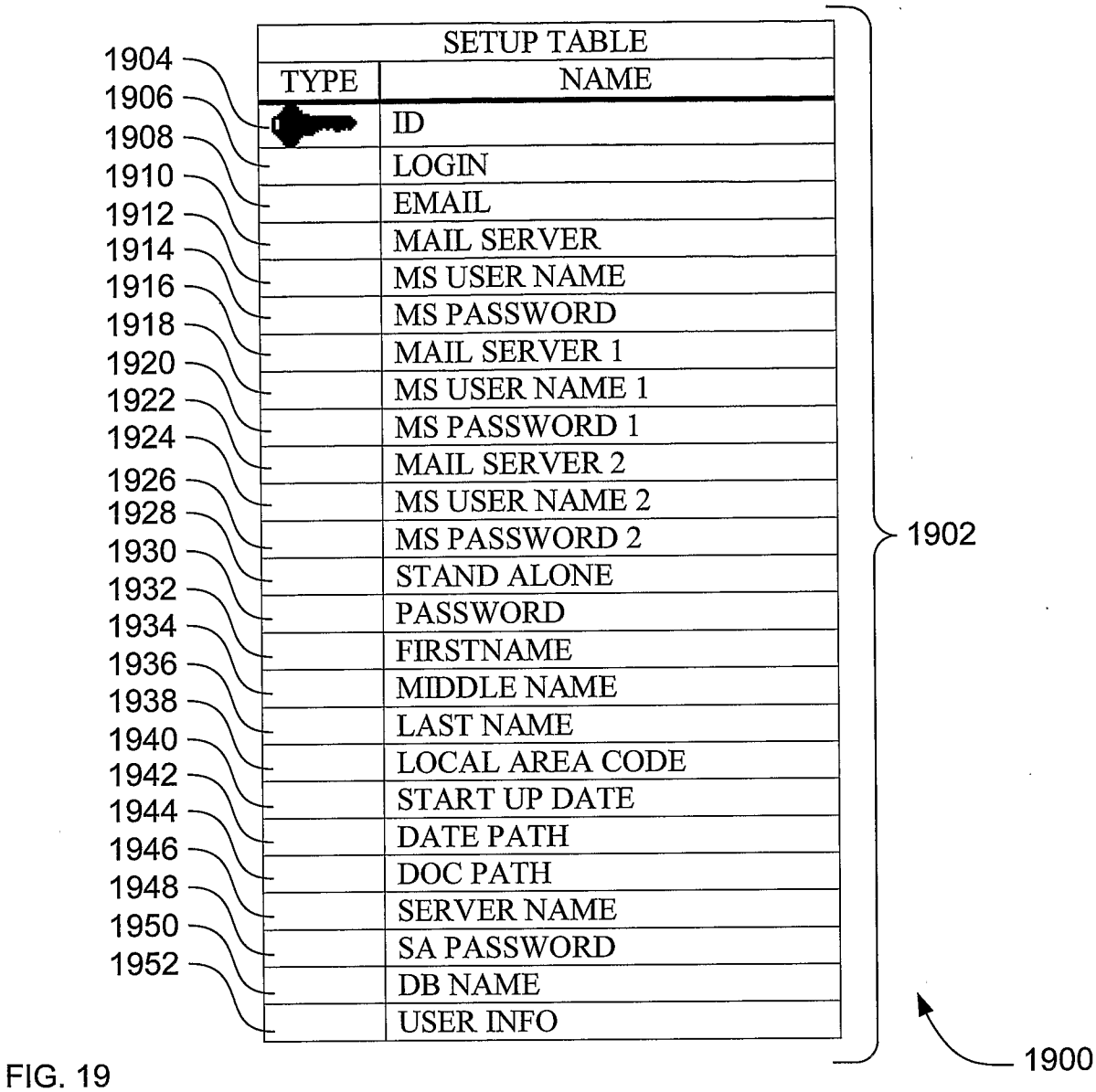

SETUP TABLE
TYPE
NAME
ID
LOGIN
EMAIL
MAIL SERVER
MS USER NAME
MS PASSWORD
MAIL SERVER 1
MS USER NAME 1
MS PASSWORD 1
MAIL SERVER 2
MS USER NAME 2
MS PASSWORD 2
STAND ALONE
PASSWORD
FIRSTNAME
MIDDLE NAME
LAST NAME
LOCAL AREA CODE
START UP DATE
DATE PATH
DOC PATH
SERVER NAME
SA PASSWORD
DB NAME
USER INFO
1904
1906
1908
1910
1912
1914
1916
1918
1920
1922
1924
1926
1928
1930
1932
1934
1936
1938
1940
1942
1944
1946
1948
1950
1952
1902
1900

FIG. 19

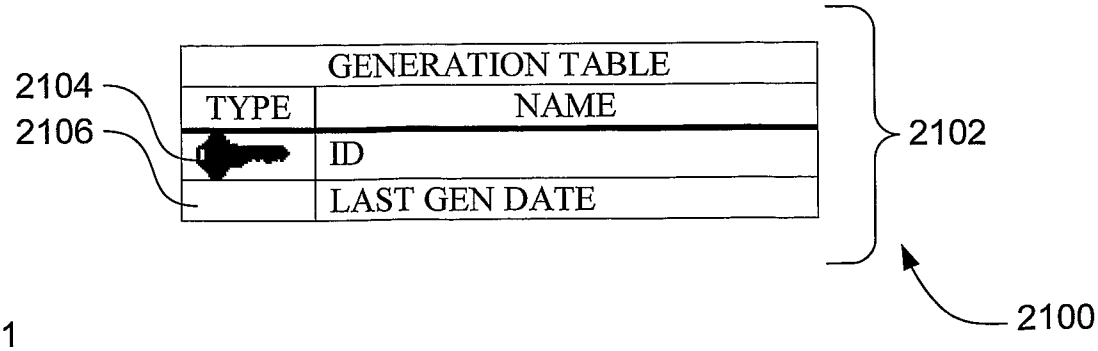

GENERATION TABLE
TYPE
NAME
ID
LAST GEN DATE
2104
2106
2102
2100

FIG. 21

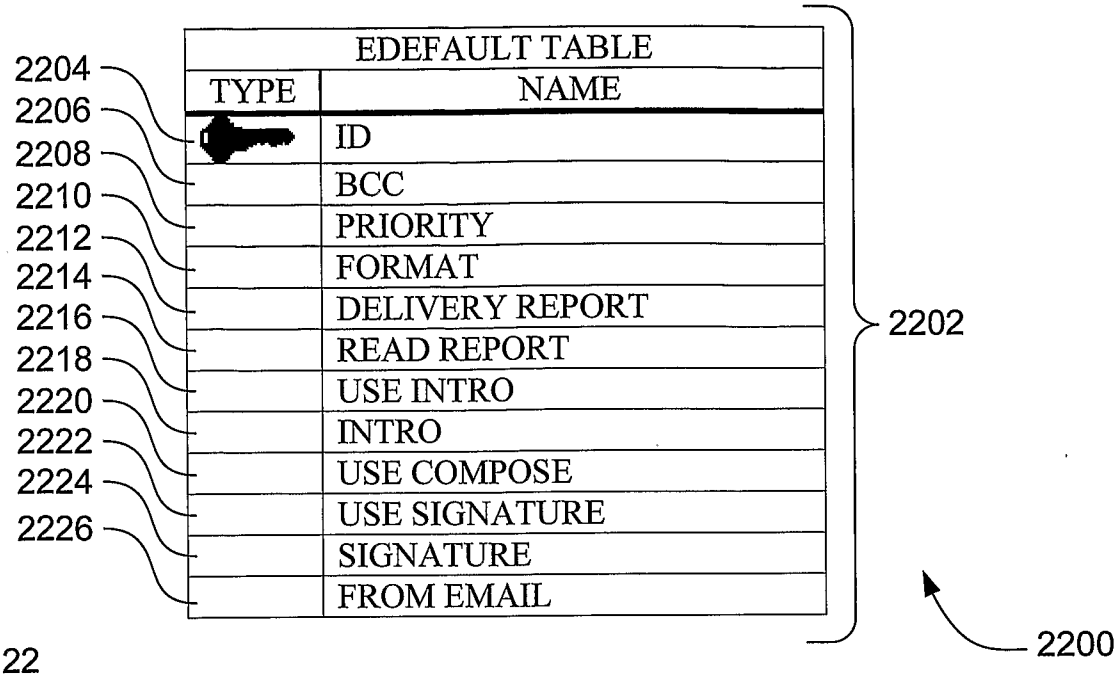

EDEFAULT TABLE
TYPE
NAME
ID
BCC
PRIORITY
FORMAT
DELIVERY REPORT
READ REPORT
USE INTRO
INTRO
USE COMPOSE
USE SIGNATURE
SIGNATURE
FROM EMAIL
2204
2206
2208
2210
2212
2214
2216
2218
2220
2222
2224
2226
2202
2200

FIG. 22

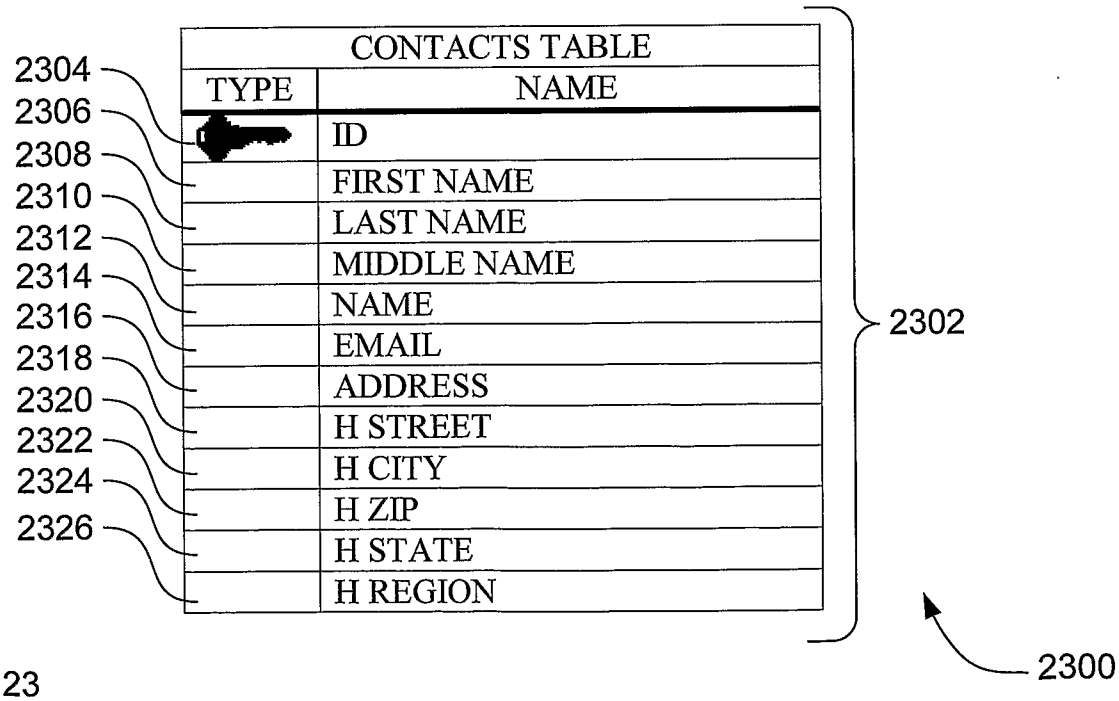

CONTACTS TABLE
TYPE
NAME
ID
FIRST NAME
LAST NAME
MIDDLE NAME
NAME
EMAIL
ADDRESS
H STREET
H CITY
H ZIP
H STATE
H REGION
2304
2306
2308
2310
2312
2314
2316
2318
2320
2322
2324
2326
2302
2300

FIG. 23

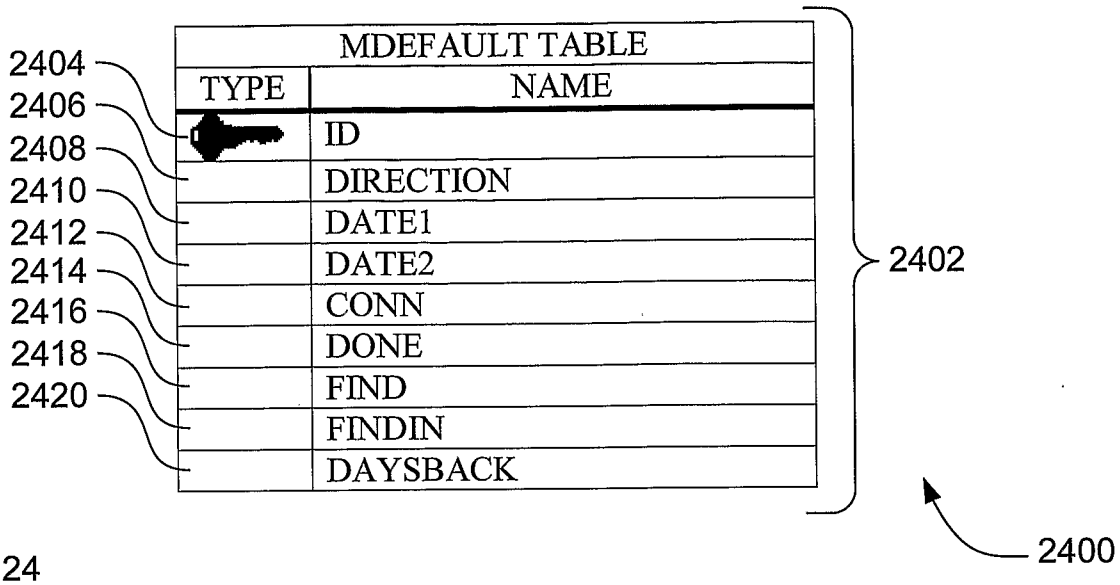

MDEFAULT TABLE
TYPE
NAME
ID
DIRECTION
DATE1
DATE2
CONN
DONE
FIND
FINDIN
DAYSBACK
2404
2406
2408
2410
2412
2414
2416
2418
2420
2402
2400

FIG. 24

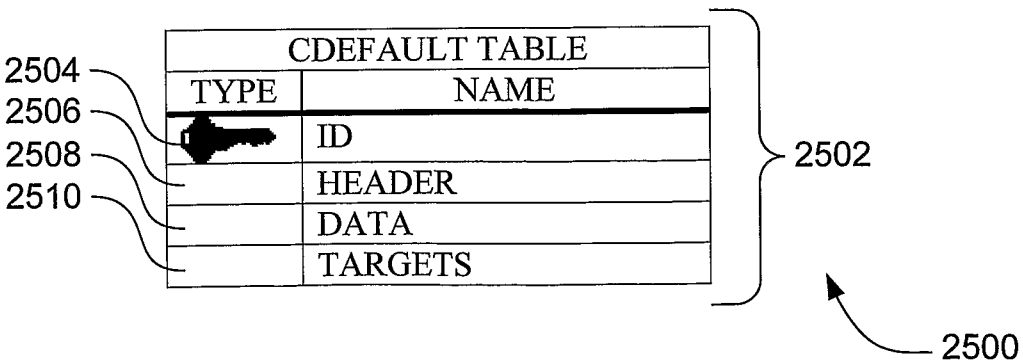

CDEFAULT TABLE
TYPE
NAME
ID
HEADER
DATA
TARGETS
2504
2506
2508
2510
2502
2500

FIG. 25

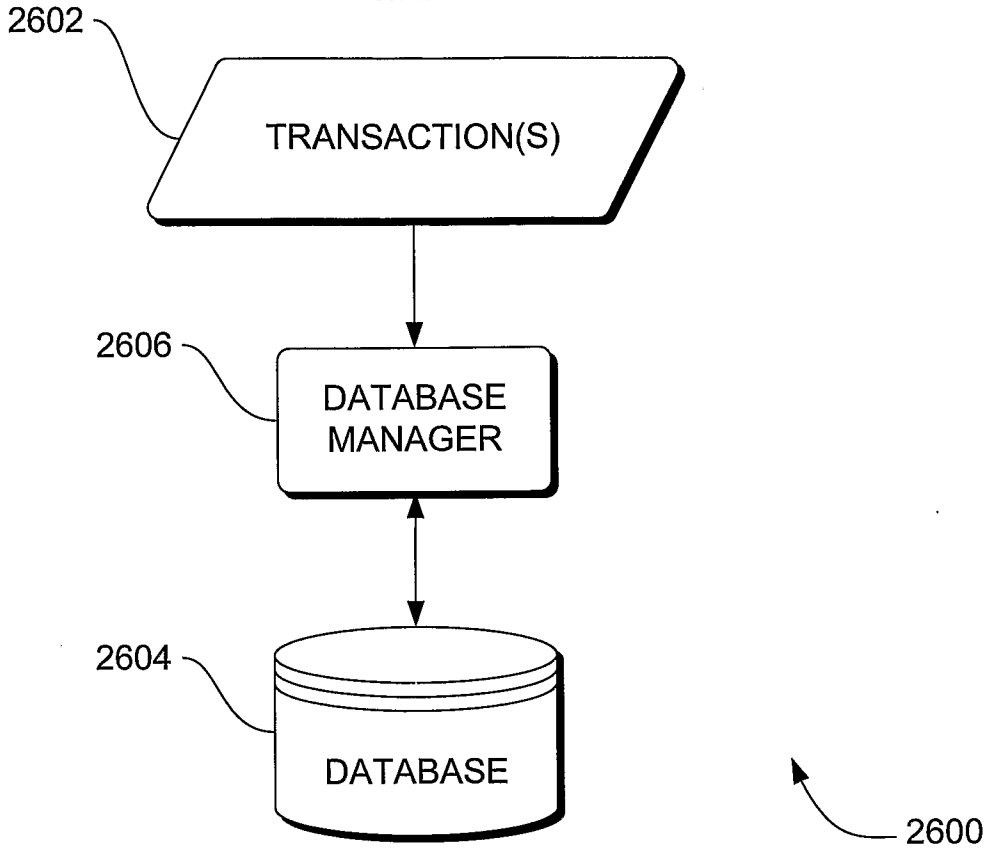


FIG. 26

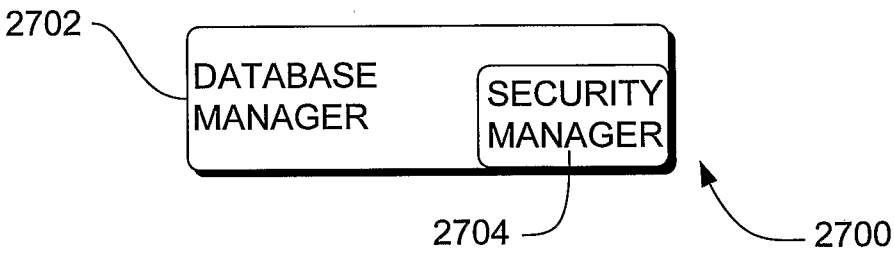


FIG. 27

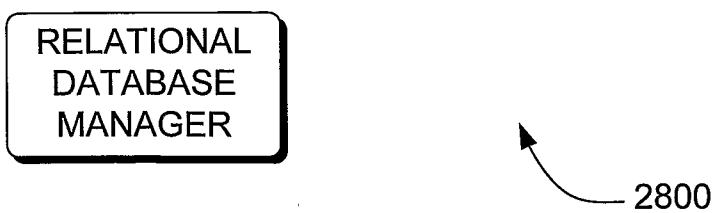


FIG. 28

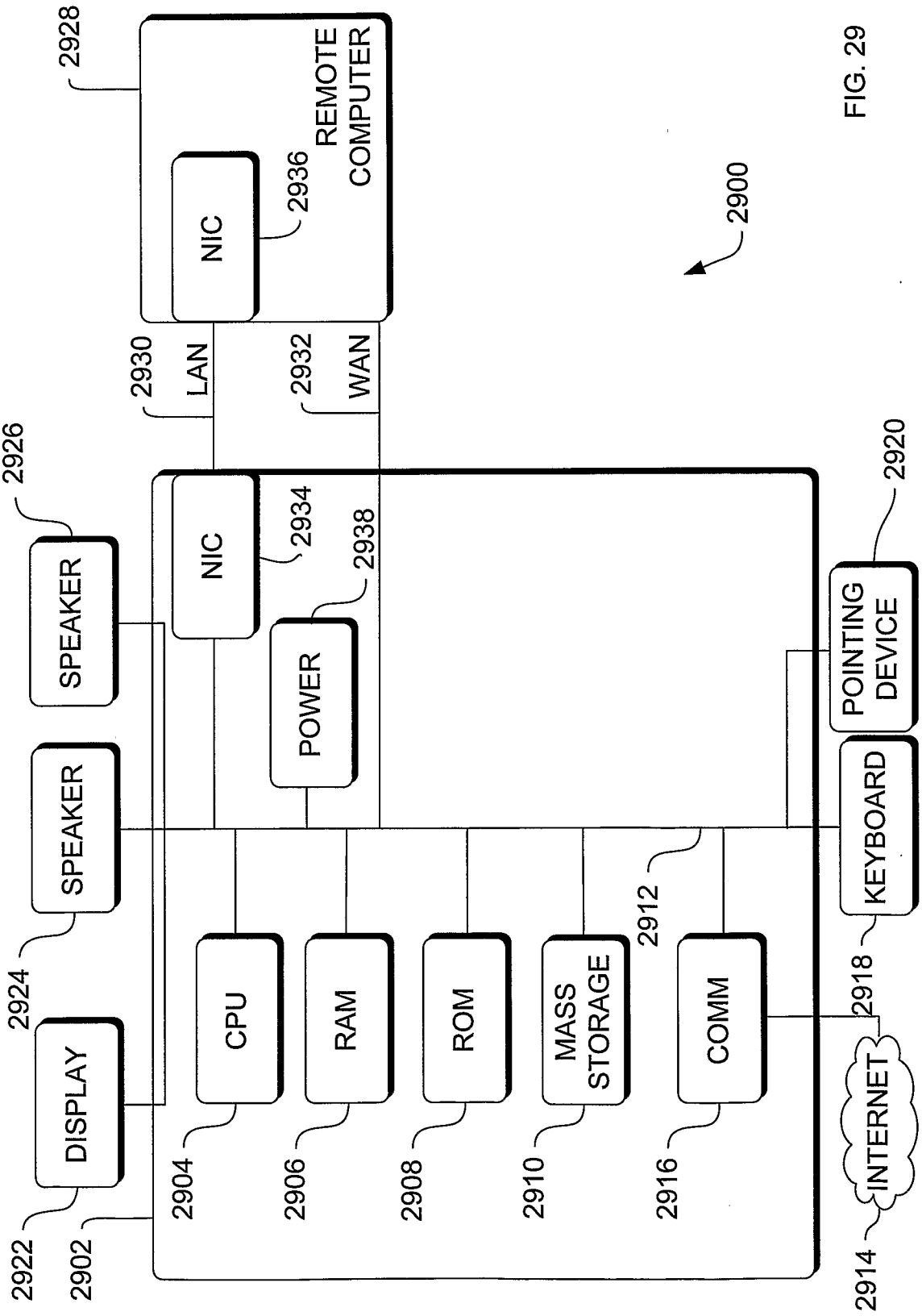

2928
REMOTE COMPUTER
NIC
2936
2900
2930
LAN
2932
WAN
2926
SPEAKER
NIC
2934
2938
POWER
2920
POINTING DEVICE
SPEAKER
KEYBOARD
2924
DISPLAY
CPU
RAM
ROM
MASS STORAGE
COMM
2912
2918
INTERNET
2922
2902
2904
2906
2908
2910
2916
2914

FIG. 29

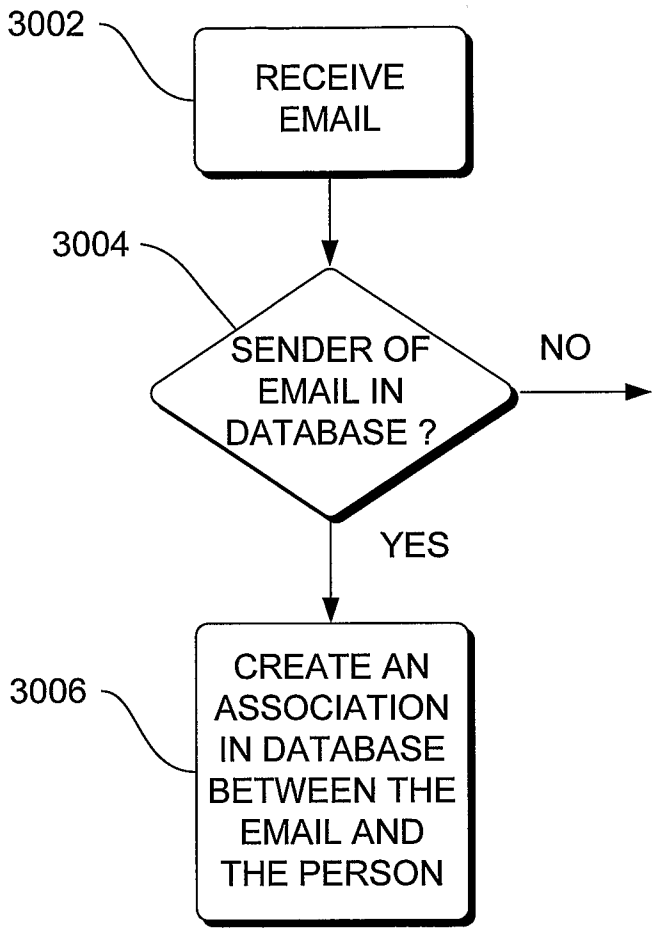

3002
RECEIVE EMAIL
3004
SENDER OF EMAIL IN DATABASE ?
NO
YES
3006
CREATE AN ASSOCIATION IN DATABASE BETWEEN THE EMAIL AND THE PERSON

FIG. 30

3000

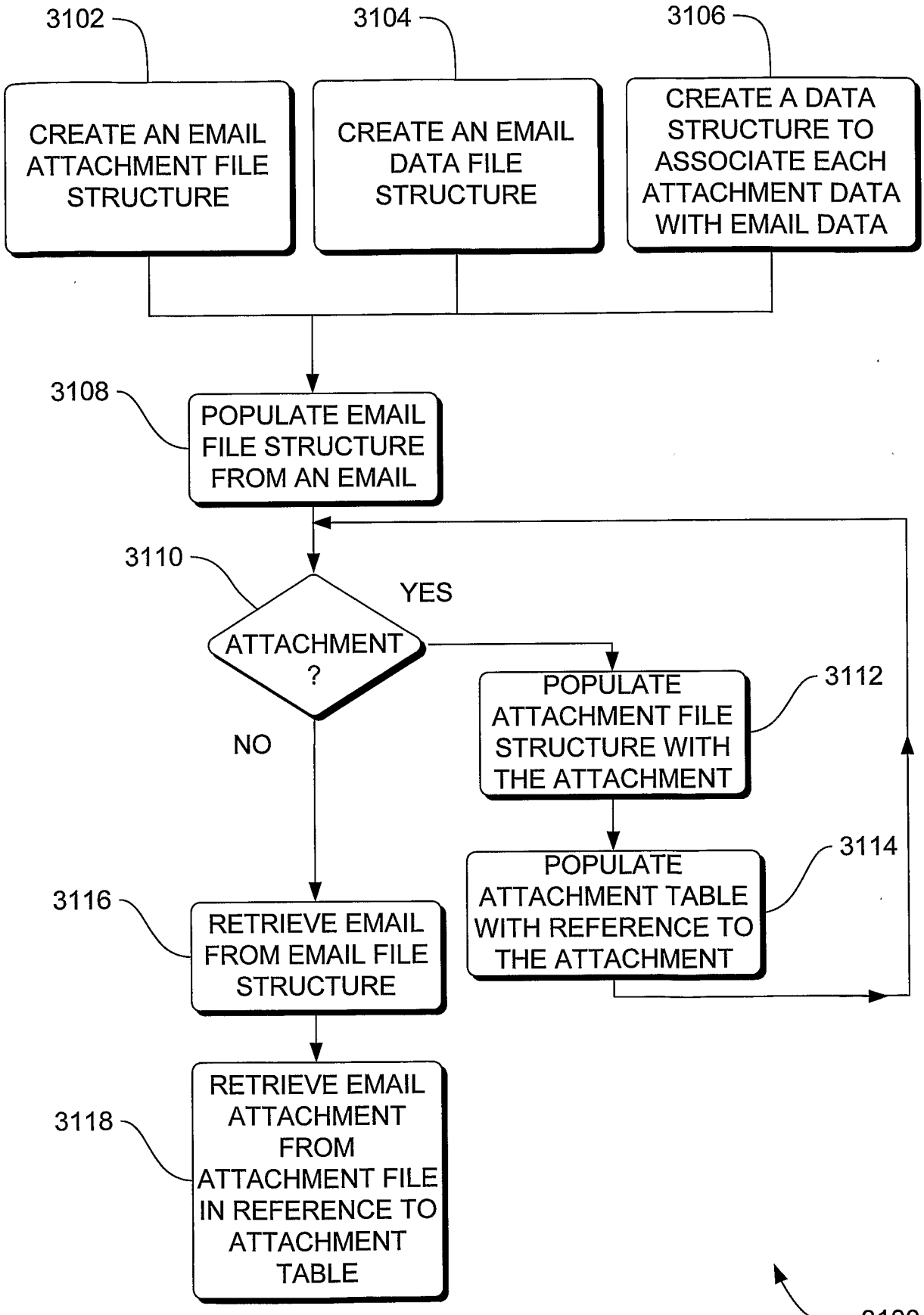

3102
CREATE AN EMAIL ATTACHMENT FILE STRUCTURE
3104
CREATE AN EMAIL DATA FILE STRUCTURE
3106
CREATE A DATA STRUCTURE TO ASSOCIATE EACH ATTACHMENT DATA WITH EMAIL DATA
3108
POPULATE EMAIL FILE STRUCTURE FROM AN EMAIL
3110
ATTACHMENT ?
YES
NO
3112
POPULATE ATTACHMENT FILE STRUCTURE WITH THE ATTACHMENT
3114
POPULATE ATTACHMENT TABLE WITH REFERENCE TO THE ATTACHMENT
3116
RETRIEVE EMAIL FROM EMAIL FILE STRUCTURE
3118
RETRIEVE EMAIL ATTACHMENT FROM ATTACHMENT FILE IN REFERENCE TO ATTACHMENT TABLE
3100

FIG. 31

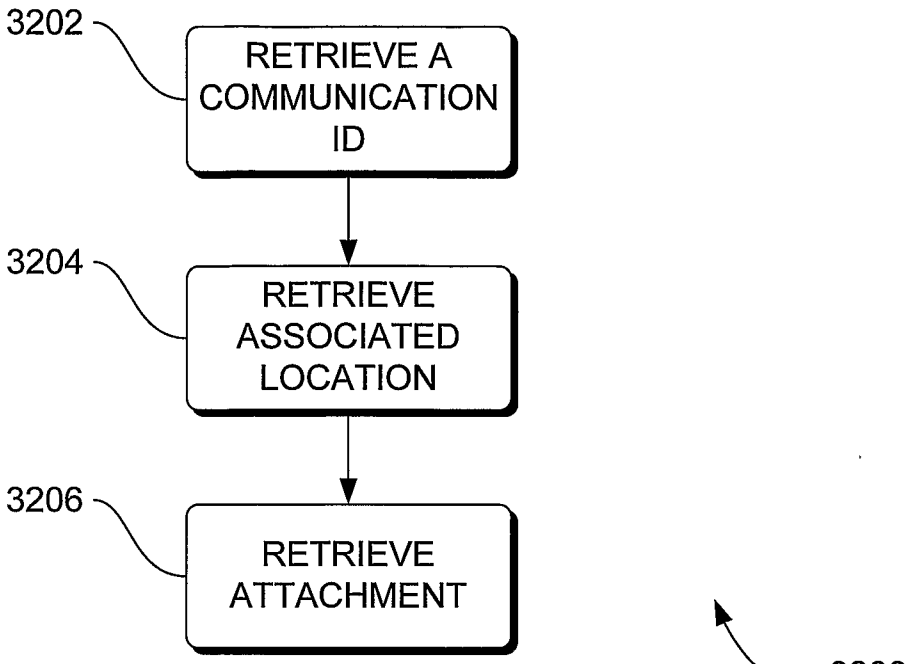

3202
RETRIEVE A COMMUNICATION ID
3204
RETRIEVE ASSOCIATED LOCATION
3206
RETRIEVE ATTACHMENT
3200

FIG. 32

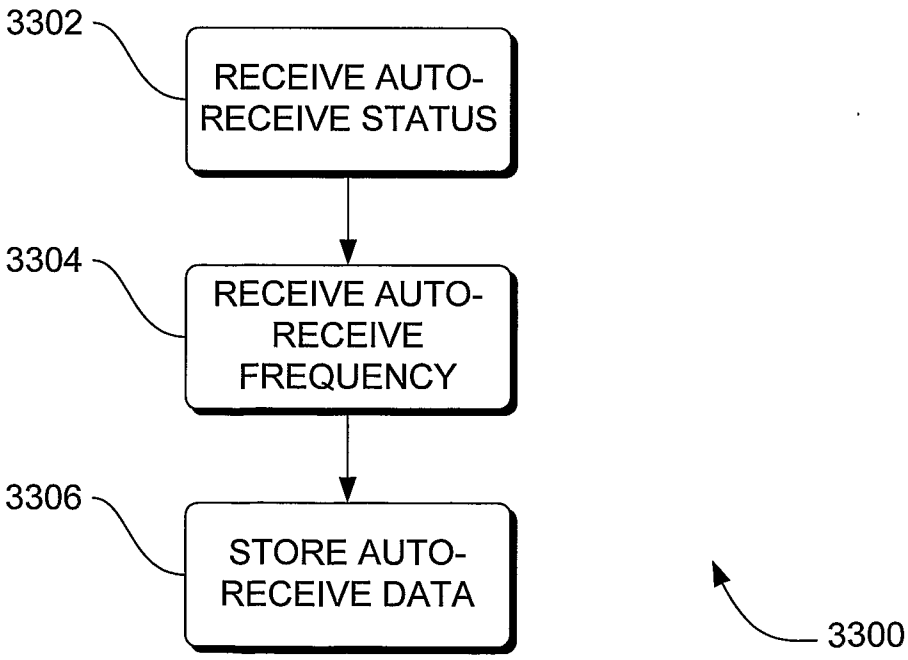

3302
RECEIVE AUTO-RECEIVE STATUS
3304
RECEIVE AUTO-RECEIVE FREQUENCY
3306
STORE AUTO-RECEIVE DATA
3300

FIG. 33

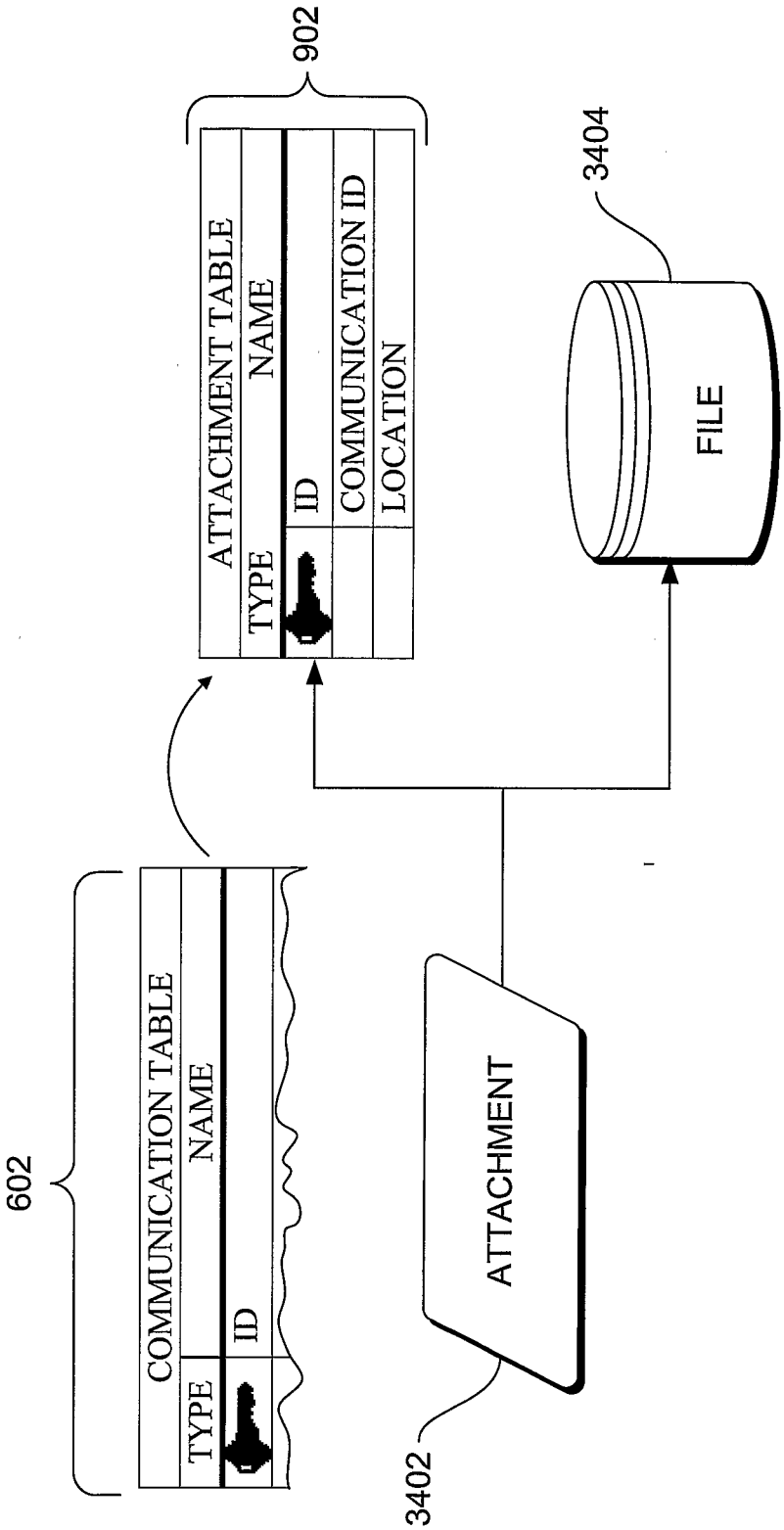

902
ATTACHMENT TABLE
TYPE
NAME
ID
COMMUNICATION ID
LOCATION
3404
FILE
602
COMMUNICATION TABLE
TYPE
NAME
ID
ATTACHMENT
3402

FIG. 34

3400

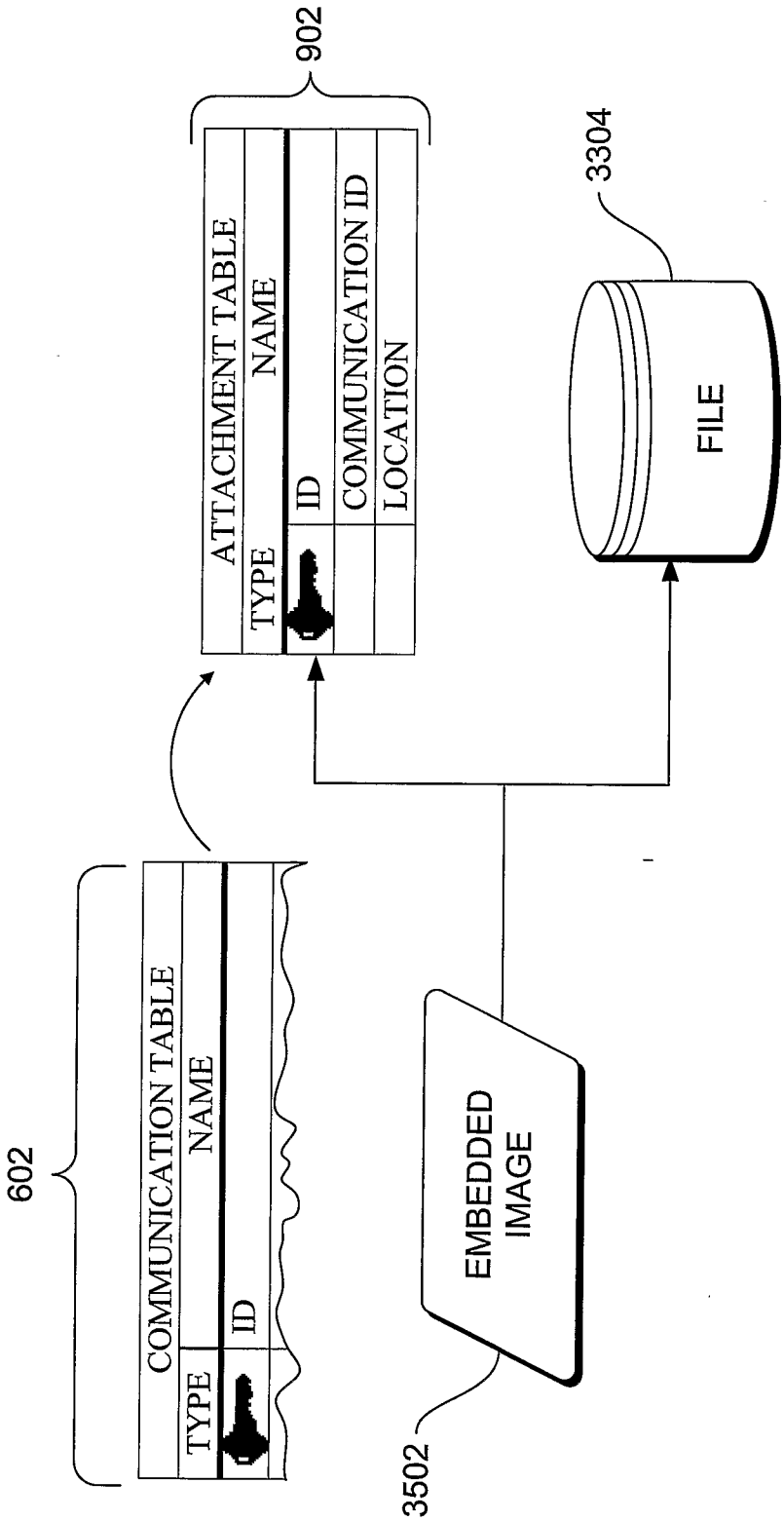

902
ATTACHMENT TABLE
TYPE
NAME
ID
COMMUNICATION ID
LOCATION
3304
FILE
602
COMMUNICATION TABLE
TYPE
NAME
ID
EMBEDDED IMAGE
3502

FIG. 35

3500

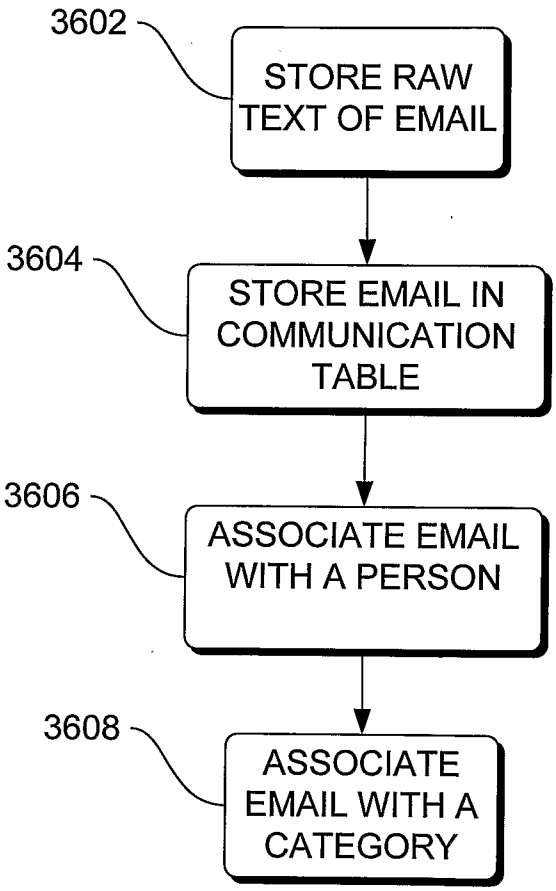

3602
STORE RAW TEXT OF EMAIL
3604
STORE EMAIL IN COMMUNICATION TABLE
3606
ASSOCIATE EMAIL WITH A PERSON
3608
ASSOCIATE EMAIL WITH A CATEGORY

FIG. 36

3600

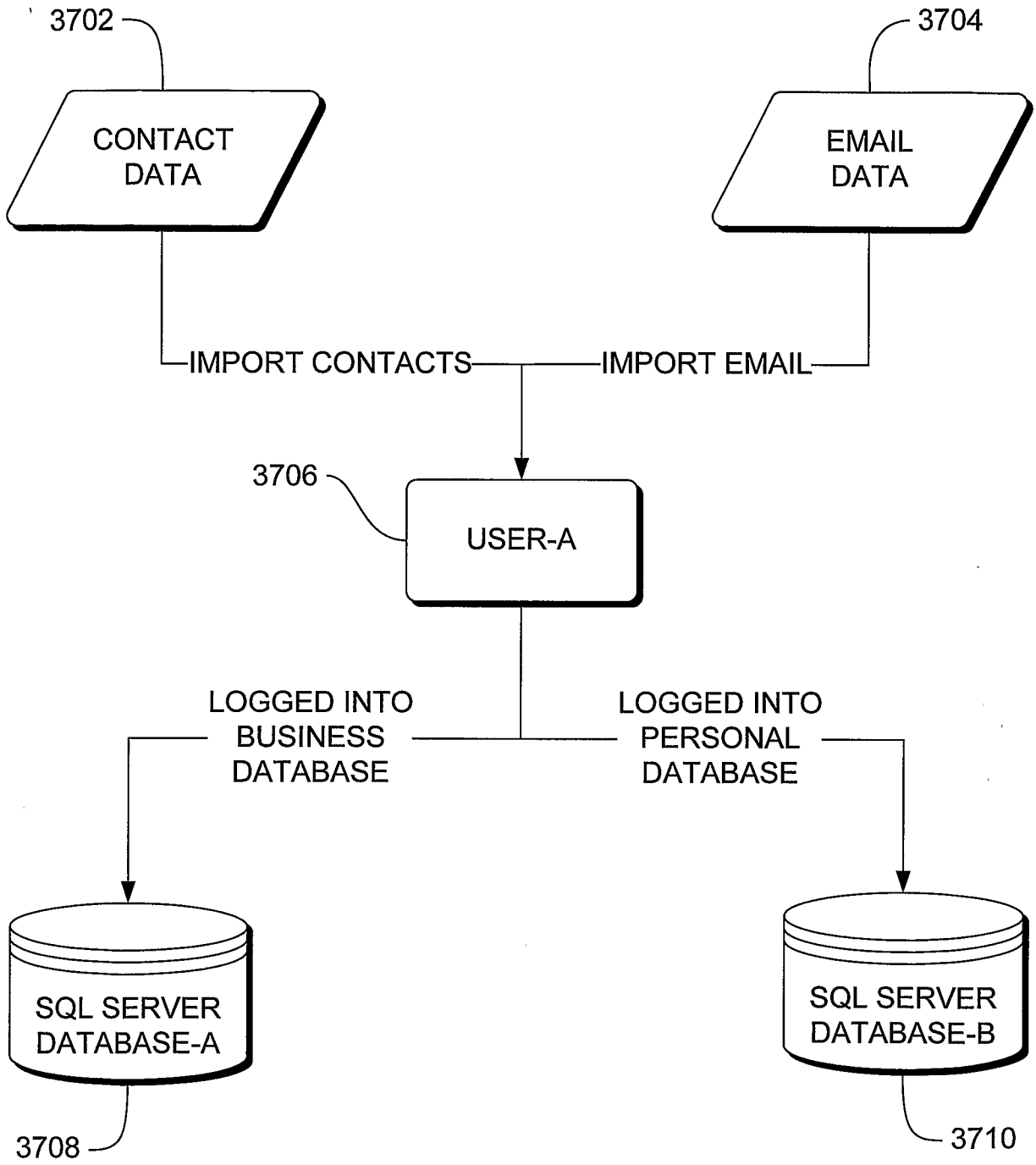

3702
CONTACT DATA
3704
EMAIL DATA
IMPORT CONTACTS
IMPORT EMAIL
3706
USER-A
LOGGED INTO BUSINESS DATABASE
LOGGED INTO PERSONAL DATABASE
SQL SERVER DATABASE-A
SQL SERVER DATABASE-B
3708
3710
3700

FIG. 37

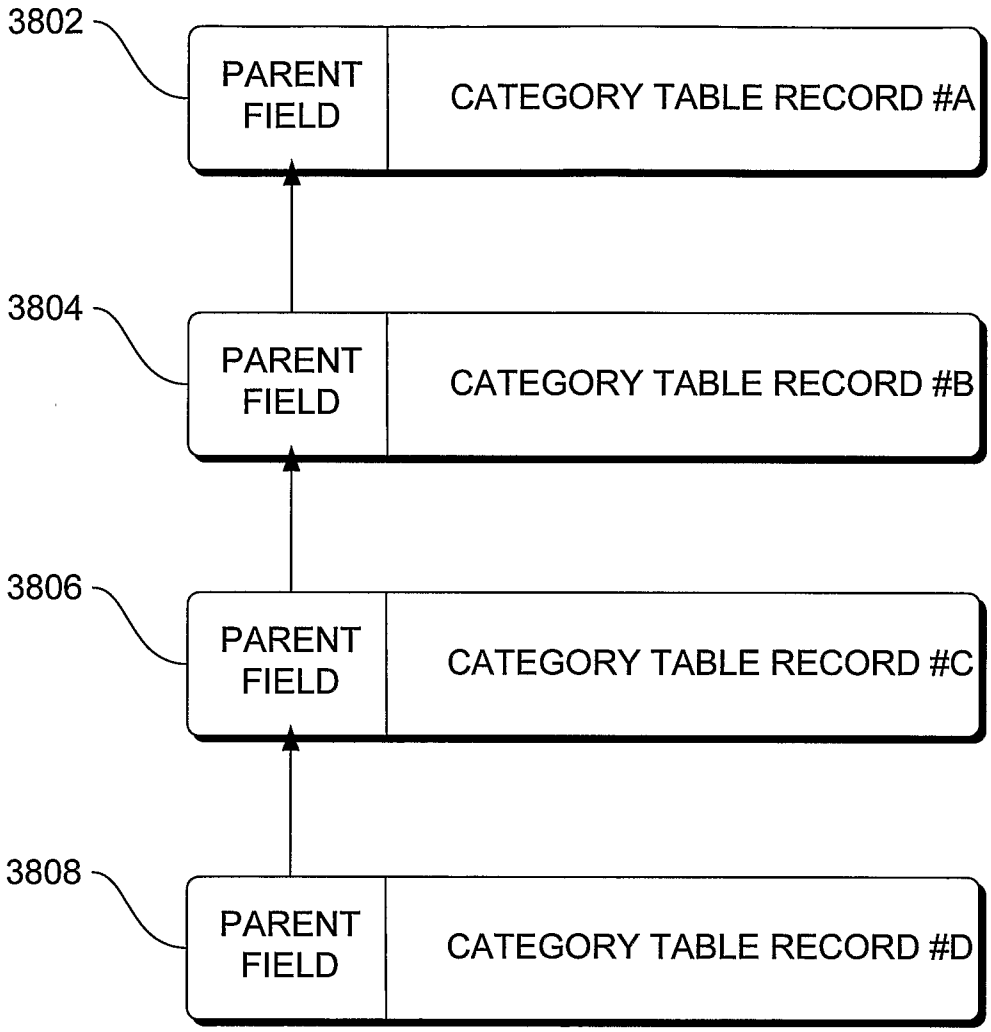

3802
PARENT FIELD
CATEGORY TABLE RECORD #A
3804
PARENT FIELD
CATEGORY TABLE RECORD #B
3806
PARENT FIELD
CATEGORY TABLE RECORD #C
3808
PARENT FIELD
CATEGORY TABLE RECORD #D
3800

FIG. 38

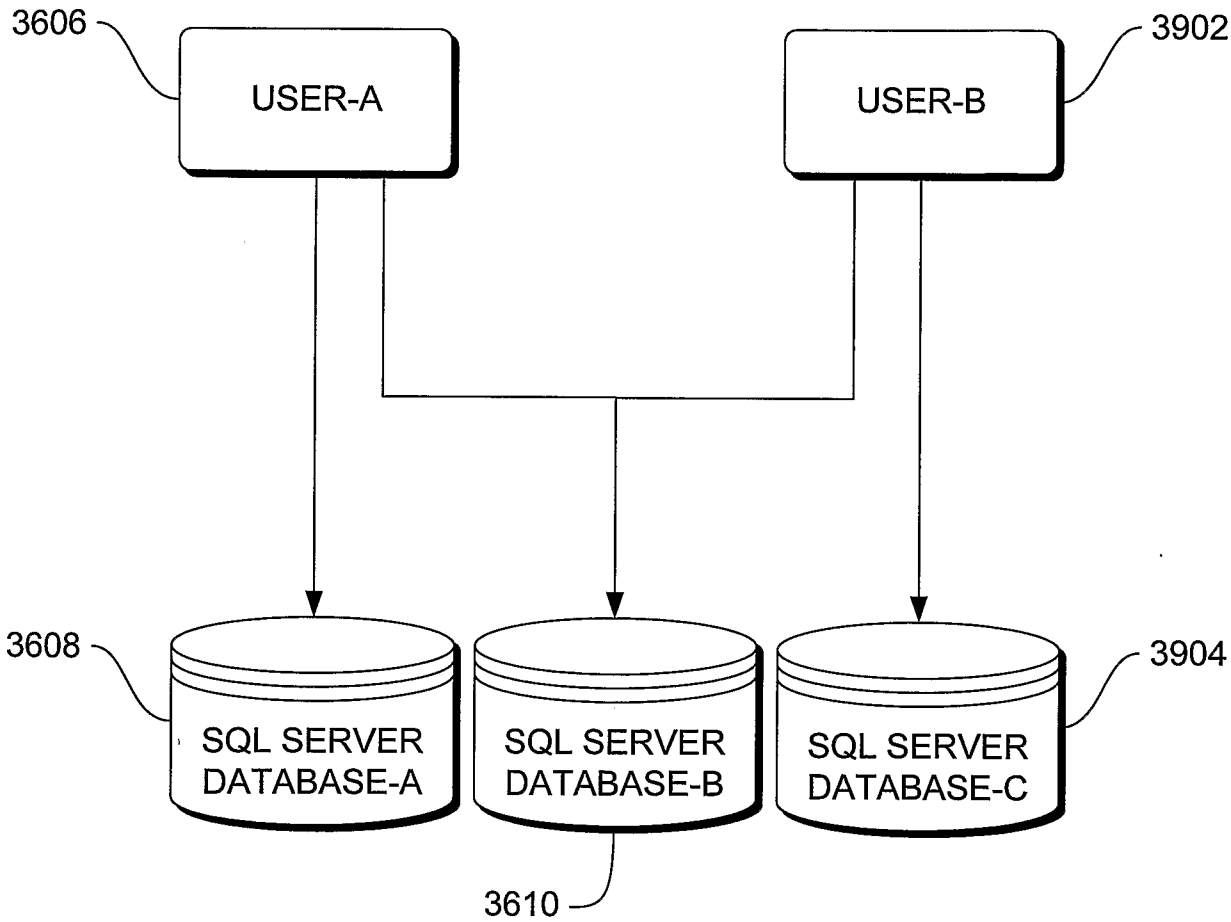

3606
USER-A
USER-B
3902
3608
SQL SERVER DATABASE-A
SQL SERVER DATABASE-B
SQL SERVER DATABASE-C
3904
3610
3900

FIG. 39

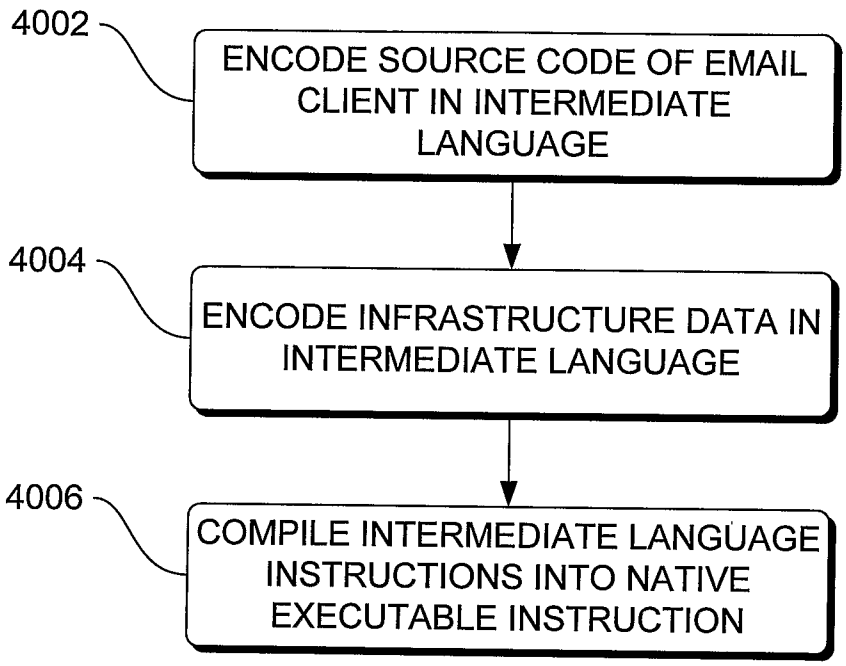

4002
ENCODE SOURCE CODE OF EMAIL CLIENT IN INTERMEDIATE LANGUAGE
4004
ENCODE INFRASTRUCTURE DATA IN INTERMEDIATE LANGUAGE
4006
COMPILE INTERMEDIATE LANGUAGE INSTRUCTIONS INTO NATIVE EXECUTABLE INSTRUCTION
4000

FIG. 40

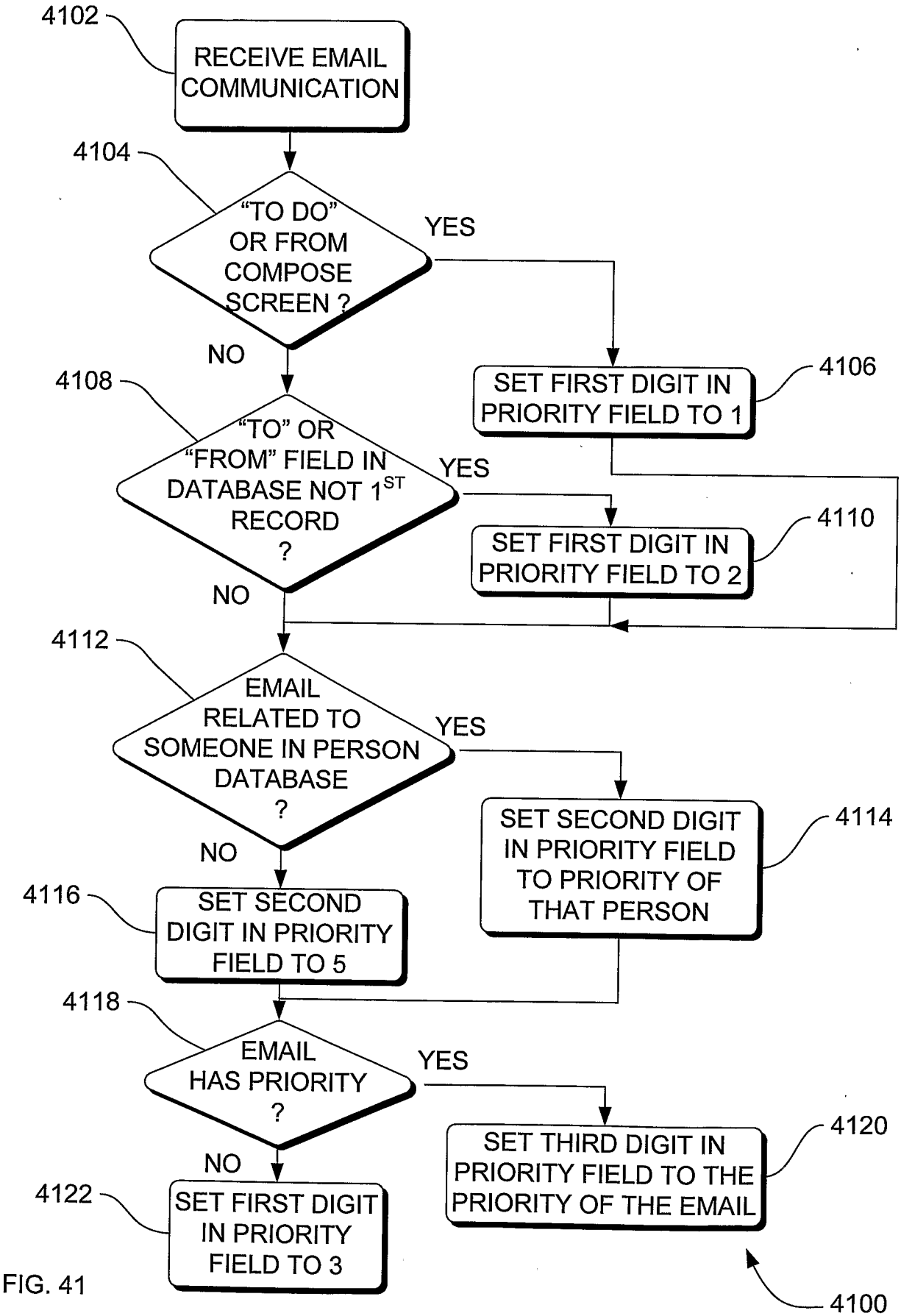

4102
RECEIVE EMAIL COMMUNICATION
4104
"TO DO" OR FROM COMPOSE SCREEN ?
YES
NO
4106
SET FIRST DIGIT IN PRIORITY FIELD TO 1
4108
"TO" OR "FROM" FIELD IN DATABASE NOT 1ST RECORD ?
YES
NO
4110
SET FIRST DIGIT IN PRIORITY FIELD TO 2
4112
EMAIL RELATED TO SOMEONE IN PERSON DATABASE ?
YES
NO
4114
SET SECOND DIGIT IN PRIORITY FIELD TO PRIORITY OF THAT PERSON
4116
SET SECOND DIGIT IN PRIORITY FIELD TO 5
4118
EMAIL HAS PRIORITY ?
YES
NO
4120
SET THIRD DIGIT IN PRIORITY FIELD TO THE PRIORITY OF THE EMAIL
4122
SET FIRST DIGIT IN PRIORITY FIELD TO 3
4100

FIG. 41