

## **SYSTEM AND METHOD FOR IMPLEMENTING AN INTERNATIONAL DEMAND DEPOSIT ACCOUNT BRANCH MIGRATION TOOL**

### **CROSS REFERENCE TO RELATED APPLICATIONS**

**[0001]** The application claims priority to U.S. Provisional Application 62/914,593 (Attorney Docket No. 72167.001792), filed October 14, 2019, the contents of which are incorporated herein in their entirety.

### **FIELD OF THE INVENTION**

**[0002]** The invention relates generally to a system and method for implementing a demand deposit account branch migration tool.

### **BACKGROUND OF THE INVENTION**

**[0003]** International Demand Deposit Account (iDDA) is an application that maintains and runs a Demand Deposit Account business. This may represent a core cash accounting application for International regions (*e.g.*, 122 branches). An International region may cover Europe Middle East and Africa (EMEA), Western Hemisphere (non-domestic US) and Asia branches. iDDA provides multi-currency, multi-branch application processing batch and real time cash postings to customer accounts, calculates interest and provides reporting to customers and Lines of Business (LOBs) for various account types including Nostros.

**[0004]** Financial institutions and other providers may be subject to regulatory requirements that involve migration of accounts and balances off an iDDA Global Platform onto a Local instance of iDDA onshore. Current technologies require a manual approach of closing accounts and opening new ones with new account numbers, customer identifiers, *etc.* There are no products or solutions that facilitate the migration of existing accounts and balances from a Global instance to a Local instance.

[0005] These and other drawbacks exist.

### **SUMMARY OF THE INVENTION**

[0006] According to an embodiment, the invention relates to a system that implements a iDDA branch migration tool. The system comprises: an interactive interface that receives user input via a communication network; and a computer processor coupled to a memory component and the interactive interface and further programmed to perform the steps of: initiating a migration of dictionary, metadata and log files from a global environment to a local branch environment, wherein vocabulary files are copied and database table names are prepared for migration and wherein the log files are updated for dictionary validation; selectively migrating local branch data from the global environment to the local branch environment, wherein redundant global data is cleansed in the local branch environment; cleansing source data in the global environment; and removing local branch data in the global environment.

[0007] According to another embodiment, the invention relates to a computer implemented method that implements a iDDA branch migration tool. The method comprises the steps of: initiating a migration of dictionary, metadata and log files from a global environment to a local branch environment, wherein vocabulary files are copied and database table names are prepared for migration and wherein the log files are updated for dictionary validation; selectively migrating local branch data from the global environment to the local branch environment, wherein redundant global data is cleansed in the local branch environment; cleansing source data in the global environment; and removing local branch data in the global environment.

[0008] The system may include a specially programmed computer system comprising one or more computer processors, interactive interfaces, electronic storage devices, and networks. The computer implemented system and method described herein provide unique advantages to entities,

financial institutions, branch locations, and other users, according to various embodiments of the invention. The Migration Tool of an embodiment of the present invention is directed to providing integrated automated validation with resiliency and operability. The Migration Tool also enables a phased migration process as well as parameterization to specify full copy, definition only or selective copy of the tables involved. An embodiment of the present invention provides a re-useable solution for other upcoming migrations in the future. The Migration Tool facilitates final export/import and provides automated status communications (*e.g.*, emails, messages, *etc.*) at each configurable step within a migration process. In addition, deployment logs may be automatically shared to configured users for quick validation. The Migration Tool also provides parallel execution to promote multitasking. An embodiment of the present invention leverages solutions developed for various (*e.g.*, current and upcoming) regulatory requirements on shoring directives that involve China, India, *etc.*

[0009] These and other advantages will be described more fully in the following detailed description.

#### **BRIEF DESCRIPTION OF THE DRAWINGS**

[0010] In order to facilitate a fuller understanding of the present invention, reference is now made to the attached drawings. The drawings should not be construed as limiting the present invention, but are intended only to illustrate different aspects and embodiments of the invention.

[0011] Figure 1 is a branch migration flow, according to an embodiment of the present invention.

[0012] Figure 2 is an illustration of local migration flows, according to an embodiment of the present invention.

[0013] Figure 3 is an exemplary flow diagram for branch migration, according to an embodiment of the present invention.

[0014] Figure 4 is an exemplary screenshot, according to an embodiment of the present invention.

[0015] Figure 5 is an exemplary flow diagram for branch migration, according to an embodiment of the present invention.

[0016] Figure 6 illustrates an exemplary database portal, according to an embodiment of the present invention.

[0017] Figure 7 is an exemplary flow diagram for selective migration, according to an embodiment of the present invention.

[0018] Figure 8 is an exemplary flow diagram for migration cleansing, according to an embodiment of the present invention.

[0019] Figure 9 is an exemplary flow diagram for branch removal, according to an embodiment of the present invention.

#### **DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT(S)**

[0020] The following description is intended to convey an understanding of the present invention by providing specific embodiments and details. It is understood, however, that the present invention is not limited to these specific embodiments and details, which are exemplary only. It is further understood that one possessing ordinary skill in the art, in light of known systems and methods, would appreciate the use of the invention for its intended purposes and benefits in any number of alternative embodiments, depending upon specific design and other needs.

[0021] An embodiment of the present invention is directed to a Migration Tool that migrates accounts in an efficient and effective manner. The Migration Tool enables a user to select

and copy key tables within an iDDA Global platform to a local instance. In addition, the Migration Tool is transparent to clients with existing account numbers being retained.

**[0022]** According to an embodiment of the present invention, the migration process further maintains historic data to support back value dated transactions. The Migration Tool may include validation to ensure account creation is successful and balances have been migrated correctly. An embodiment of the present invention may provide an end-to-end process that is generic and reusable to migrate any branch at a future date and time. In addition, certain target locations may have different requirements (*e.g.*, regulatory, country specific, *etc.*) which an embodiment of the present invention may account for during the migration process. With an embodiment of the present invention, data required for a new branch setup may be migrated from Global and then set-up as a new branch.

**[0023]** According to an exemplary implementation, the Migration Tool may be developed using Info-Basic, Shell Scripts and PL/SQL (Procedural Language/Structured Query Language). Info-Basic may represent a front-end that provides a menu-driven approach. Shell scripts may act as an interface between the front-end to the back-end PL/SQL. PL/SQL programming may run in a database system performing the migration. Other components, services and variations thereof may be implemented. An exemplary solution may leverage native database administrator (DBA) utilities such as a database self-service portal for a final export/import of tables. The database self-service portal for application development teams provides a self-service web portal for commodity based tasks. Self-service automated tasks reduce user requests for tickets and demand on the infrastructure database teams. In addition, the database self-service portal may provide exporting of dictionary, metadata of dictionary and log files as well as exporting of source data. Various

database self-service portals, services and tools may be implemented in accordance with the embodiments of the present invention.

**[0024]** Automated validation steps may be provided at specific phases of execution. For example, automated status communications (*e.g.*, emails, messages, *etc.*) or other notifications may be issued on completion of key steps within the migration process. Deployment logs may be automatically shared to specified users for validation and audit. In addition, an embodiment of the present invention provides parallel execution to promote multitasking. The Migration Tool may be configured to allow flexibility in approaches to specify full copy, definition only or selective copy of the tables involved. Other subsets may be identified. For improved transparency and ease of validation, an embodiment of the present invention may include a communication feature to send automated emails or notifications with information on executions performed as well as migration logs to the configured users/stakeholders. Other status and relevant information may be provided in various formats.

**[0025]** The Migration Tool may include an automated menu driven interface that displays the steps executed where the migration validation may be integrated and automated. The branch migration tool may further include an interface and/or other GUI front end.

**[0026]** An embodiment of the present invention seeks to reduce and/or eliminate compliance issues. With an embodiment of the present invention, other branch data is not forwarded into a new target environment. For example, in the new target environment, an application server and database server only contains relevant (target environment) details and would therefore be light. In this exemplary implementation, legacy of (target environment) specific history data may be maintained including contents of the application server. In addition, the same account numbers may be maintained. There is little to no risk of back value transactions

as the account ladder may be maintained. In this example, the forward transaction queue may happen as business as usual (BAU). With this, there is little to no client impact. Moreover, an embodiment of the present invention may avoid any requirement of a staging environment and requiring a double copy (*e.g.*, global to staging and then staging to target environment).

**[0027]** In an exemplary scenario, there may be approximately 115,000 tables to be migrated from a global environment to a target environment or database. Out of the 11500 tables, there may be 3800 tables that are dictionary tables. For example, each table may have thousands and thousands of rows/columns of data including customer account information, *etc.* Accordingly, an embodiment of the present invention may perform migration in multiple phases followed by a cleansing phase in the global environment. This would reduce the risk of performing an entire data migration on a single instance and further reduce migration time significantly. With this approach, there would be no deployment performed between two phases of migration (in the source). During migration, input channels may be brought down to prevent any accidental input.

**[0028]** Figure 1 is a branch migration flow, according to an embodiment of the present invention. According to an embodiment of the present invention, migration data may be provided in phases. As shown in Figure 1, Phase 1 represents Migration of Dictionary, Metadata and Log Files. Phase 2 represents Migration of Data tables and Target Cleansing. Phase 3 represents Cleansing of data in a Source. Phase 4 represents Branch Removal in the Source. As shown in Figure 1, Source 102 is represented as Global and Target 104 is represented as the Local. In this example, branch migration is from Source 102 to Target 104.

**[0029]** In Phase 1 of migration, the tool may extract, import and validate various tables and data. This may include Dictionary tables, Migration log tables and Metadata of Data tables to be migrated. As shown in Figure 1, Source 102 exports dictionary and migration log tables (step

1a). Target 104 may then import Dictionary tables and log tables (step 1b). Target may run automated validation of imported dictionary tables (step 1c). In addition, Source 102 may export metadata of data tables (step 1d). This may exclude certain information such as statistics, for example. Target 104 may then import Metadata (step 1e). Target 104 may rename Metadata of binary XML non-selective tables (step 1f). Target 104 may run automated validation of import of Phase 1 (step 1g).

**[0030]** In Phase 2 of migration, the tool may perform Data tables migration, *e.g.*, export, import and validate. In addition, Data tables relevant to Branch Migration (*e.g.*, local instance) may include: Full Copy – Entire contents of these tables will be copied over to the new instance; Definition Only – Only the Metadata/Table structure needs to be copied to the new instance; and Selective Copy – Only the Branch specific data needs to be copied over. Other variations in selective portions of data and files may be applied.

**[0031]** As shown in Figure 1, Source 102 may initiated an automated selective migration (step 2a). This may include exporting data, which may include a full copy and/or selective portions (step 2b). Certain information may be included, such as statistics, for example. Target 104 may then import Data Tables (step 2c). This may involve renaming selective tables (step 2d). Target 104 may run an automated validation of imported data tables (step 2e).

**[0032]** As shown in Figure 1, Phase 2 may include Target Cleansing. Table statistics may be gathered (step 2f). Automated target cleansing may be initiated (step 2g). This may then lead to Post Validation Testing (PVT) (step 2h).

**[0033]** In Phase 2 of Branch migration, the required Metadata and Dictionary may be moved to Target (Onshoring branch). Target Cleansing may be performed to remove Global data. This may ensure that the new branch instance contains the required branch data. Cleansing of

Source data in Target may be performed to avoid Global users having access to on-shored Branch instance, *etc.* According to an exemplary illustration, cleansing in the Target environment may involve the following: INTERCO.PARAMETER - remove global data from Target; user table cleansing - remove global data from Target; TSA.SERVICE cleansing - remove Global server name from Target and scripts to update batch for FATCA, Bridger, IDCERT and REGD. An embodiment of the present invention provides a parameter-based configurable way to remove redundant data.

**[0034]** Global Cleansing may be initiated after validation of phase 2. Once the branch is migrated from Global, migrated branch specific data may be cleansed. Cleansing of Branch migrated data from Global may be a mandatory activity to remove unused data. Global cleansing may be performed in user data, in Batch records *etc.* Once the migration of local instance is successfully completed, the Migration Tool may perform cleansing on the Source. This may include: Automated Source Cleansing and Validation of Source Cleansing.

**[0035]** As shown in Figure 1, Phase 3 may involve initiating an automated source cleansing process (step 3a) at Source 102. Phase 3 may further involve validation of source cleansing (step 3b) and PVT of source cleansing (step 3c).

**[0036]** Phase 4 may be an optional phase that performs based on a decision to remove all the tables of on-shored branch (*e.g.*, local instance) from a global instance. Phase 4 may delete (*e.g.*, drop/remove) all the on-shored branch tables. As shown in Figure 1, Phase 4 may involve initiating an automated branch removal (step 4a). Phase 4 may further involve automated validation of branch removal (step 4b) and PVT of branch removal (step 4c).

**[0037]** Figure 2 is an illustration of local migration flows, according to an embodiment of the present invention. As shown in Figure 2, Global system 202 may migrate branch data to Target

(local) system 204. The migration process may occur in phases where Metadata 210, Dictionary Tables 212, Log 214 and Migration Log 216 are respectively migrated to Metadata 220, Dictionary Tables 222, Log 224 and Migration Log 226 of Target system 204. Once the migration process is complete, the migrated data may be removed or deleted from Global 202.

**[0038]** Figure 3 is an exemplary flow diagram for branch migration, according to an embodiment of the present invention. At step 310, dictionary, metadata and log files may be migrated from a global environment to a new local instance. Database objects have various descriptive attributes. Information about a particular schema object may be obtained by performing a DESCRIBE operation. The result may be accessed as an object of a Metadata class by passing object attributes as arguments to the various methods of the Metadata class. Additional details are shown in Figures 5 and 6. At step 312, selective migration of local branches from global to new local instance may be followed by cleansing of redundant (*e.g.*, user, parameter, batch, *etc.*), global data in local instance. Additional details are shown in Figure 7. At step 314, cleansing in a global environment may be performed. Additional details are shown in Figure 8. At step 316, branch removal (local) in the global environment may be performed. Additional details are shown in Figure 9. While the process of Figure 3 illustrates certain steps performed in a particular order, it should be understood that the embodiments of the present invention may be practiced by adding one or more steps to the processes, omitting steps within the processes and/or altering the order in which one or more steps are performed.

**[0039]** Figure 4 is an exemplary screenshot, according to an embodiment of the present invention. Figure 4 illustrates a menu driven approach. The interactive interface may be agnostic of the branch, database, and schema. Further, the interface provides repeatable steps and may support easily configurable dependencies.

**[0040]** Figure 4 illustrates an interface to address pre-requisites, such as scripts and configurations. As shown in Figure 4, the interface may provide Migration Home Directory 410, Configuration for Source and Target Cleansing, Tablespace, User 412, Log File Directories 414 and Scripts for invoking certain packages 416. Migration Home Directory 410 may include a path name for export. Configuration information at 412 may include configuration data for source, tablespace, target and user. Log File Directories 414 may include log information for archives, spool log and Unix log. Scripts 416 may relate to validation, XML driver log, flat table data build, migrate list, old validation, pre validation, rename, validation script, validation source script, validation target script, *etc.*

**[0041]** Another interface at 404 may provide Configuration for Cleansing Target 420, User for Migration Process 422, Configuration for Cleansing at Source 424 and Data and Index Table Space for Migration 426. Configuration for Cleansing Target 420 may include target configuration data. Configuration for Cleansing at Source 424 may include source configuration data relating to user, batch, *etc.* Data and Index Table Space for Migration 426 may identify tables and/or other files/data for migration.

**[0042]** Figure 5 is an exemplary flow diagram for branch migration, according to an embodiment of the present invention. According to an exemplary scenario, a sequence of steps may be executed. Step 510 may create an execution identifier (ID) for a migration run. For example, an execution ID may represent a unique number for a migration activity. Step 512 may build metadata for migration. Step 514 may perform a pre-validation to check unwanted files and check consistency. Step 516 may copy vocabulary files (VOC) and prepare a database table name versus table name file for migration. Step 518 may build flat table and migration metadata for migration. Step 520 may update log file for dictionary validation. For example, migration log

table may be built for the purpose of validating dictionary tables. Step 522 may export dictionary, metadata and log files. Step 524 may export metadata for non-dictionary tables.

**[0043]** Additional steps may include one or more of the following: transferring an exported dump from Source to Target via a database self-service portal; importing Dictionary and Migration log tables in the Target via a database self-service portal; performing an Automated Validation of Dictionary tables import in Target; building a Migration log table for the purpose of validating the Metadata tables in Source; exporting Metadata of Data Tables (excluding Statistics) in Source; transferring the exported dump from Source to Target via a database self-service portal; importing Metadata (excluding Statistics) in Target via a database self-service portal; renaming Metadata of Binary XML non selective tables in Target; and performing Automated Validation of Metadata import in Target.

**[0044]** While the process of Figure 5 illustrates certain steps performed in a particular order, it should be understood that the embodiments of the present invention may be practiced by adding one or more steps to the processes, omitting steps within the processes and/or altering the order in which one or more steps are performed.

**[0045]** Figure 6 illustrates an exemplary database portal, according to an embodiment of the present invention. Figure 6 may represent an exemplary database self-service portal. A user may identify Refresh Type 610, Refresh Mode 612 and Data Consistency 614. A user may further identify Source Database 616, Source Schema List 618, Source Directory 620 and Table Filter Criteria 622. The database portal shown by Figure 6 may be used to export dictionary, metadata of dictionary and log files as well as export metadata of non-dictionary tables.

**[0046]** Refresh Type 610 may be selected from options such as refresh schema, refresh table, export schema and export table. Refresh Mode 612 may include all, metadata and data.

Other options may be available. Data consistency may be enforced at 614. Source Database 616 may be selected by name, identifier and/or other criteria. At Source Schema List 618, a schema name may be entered in a schema mapping input. Source Directory 620 may be selected from a list of available data paths. Table Filter Criteria 622 may enter filters and/or other conditions.

**[0047]** Figure 7 is an exemplary flow diagram for selective migration, according to an embodiment of the present invention. Selective Migration of data may be performed on tables and/or other subset of data, files, *etc.* For example, selective migration of the tables may be grouped into packages (*e.g.*, 4 packages, *etc.*) based on their functionality/domain. This may include: User Package; Company Packages; Customer Packages and Account Packages.

**[0048]** According to an exemplary scenario, a sequence of steps may be executed. Step 710 may perform a selective copy of a user package for migration in a source environment. Step 712 may perform a selective copy of a company package for migration. Step 714 may perform a selective copy of customer package for migration. Step 716 may perform a selective copy of accounts package for migration. Step 718 may perform a copy of definitions package for migration. Step 720 may update log files for data validation in the source environment. Step 722 may validate data in the source environment. Step 724 may export the source data. This may involve exporting data and migrating log tables from the source.

**[0049]** Additional steps may involve importing Dictionary and Migration log tables in the target environment and initiating automated validation of the import of Dictionary tables into the target environment.

**[0050]** While the process of Figure 7 illustrates certain steps performed in a particular order, it should be understood that the embodiments of the present invention may be practiced by

adding one or more steps to the processes, omitting steps within the processes and/or altering the order in which one or more steps are performed.

**[0051]** Figure 8 is an exemplary flow diagram for migration cleansing, according to an embodiment of the present invention. According to an exemplary scenario, a sequence of steps may be executed. Step 810 may back-up cleaning tables in a source environment. Step 812 may initiate cleansing USER in a global/source server. This may refer to cleansing redundant user data. Step 814 may initiate cleansing BATCH in global/source server. This may involve cleansing redundant batch data and/or end of day processing data. According to an exemplary illustration, this may include cleansing BANK.REGD.HIST, BANK.REGD.BAL and BANK.REGD in a global/source server. Step 816 may restore USER in global/source server. This may be initiated when there is a need to restore user data and/or bring the user data back to business as usual (BAU). This may be for no-go scenarios, such as not being able to go live which would involve having global/source server restore data back to its original state. Step 818 may restore BATCH in a global/source server. This may be initiated when there is a need to restore batch or end of day processing data and/or bring the data back to business as usual (BAU). This may be for no-go scenarios. According to an exemplary illustration, this may include restoring BANK.REGD.HIST, BANK.REGD.BAL and BANK.REGD in a global/source server. Additional steps may involve stopping all AUTO services to run the cleansing and then starting all AUTO services. Additional steps may include running a P.BANK.AD.AUTO.DEPLOY program to deploy data records. While the process of Figure 8 illustrates certain steps performed in a particular order, it should be understood that the embodiments of the present invention may be practiced by adding one or more steps to the processes, omitting steps within the processes and/or altering the order in which one or more steps are performed.

**[0052]** Figure 9 is an exemplary flow diagram for branch removal, according to an embodiment of the present invention. According to an exemplary scenario, a sequence of steps may be executed. Step 910 may remove Local from INTERCO.PARAMETER 'SYSTEM.' Step 912 may rename and remove migrated Tables and VOC (Vocabulary). Step 914 may remove Local Selective tables from a Global environment. Step 916 may perform a validation of Local Selective table, Rename and Drop of Local tables and deleting of VOCs. Step 918 may initiate an auto validate of selective table migrations. Step 820 may initiate an auto validation of Tables and VOC. While the process of Figure 9 illustrates certain steps performed in a particular order, it should be understood that the embodiments of the present invention may be practiced by adding one or more steps to the processes, omitting steps within the processes and/or altering the order in which one or more steps are performed.

**[0053]** Other embodiments, uses, and advantages of the invention will be apparent to those skilled in the art from consideration of the specification and practice of the invention disclosed herein. The specification and examples should be considered exemplary only, and the scope of the invention is accordingly not intended to be limited thereby.

**[0054]** The foregoing examples show the various embodiments of the invention in one physical configuration; however, it is to be appreciated that the various components may be located at distant portions of a distributed network, such as a local area network, a wide area network, a telecommunications network, an intranet and/or the Internet. Thus, it should be appreciated that the components of the various embodiments may be combined into one or more devices, collocated on a particular node of a distributed network, or distributed at various locations in a network, for example. As will be appreciated by those skilled in the art, the components of the various

embodiments may be arranged at any location or locations within a distributed network without affecting the operation of the respective system.

**[0055]** As described above, the various embodiments of the present invention support a number of communication devices and components, each of which may include at least one programmed processor and at least one memory or storage device. The memory may store a set of instructions. The instructions may be either permanently or temporarily stored in the memory or memories of the processor. The set of instructions may include various instructions that perform a particular task or tasks, such as those tasks described above. Such a set of instructions for performing a particular task may be characterized as a program, software program, software application, app, or software.

**[0056]** It is appreciated that in order to practice the methods of the embodiments as described above, it is not necessary that the processors and/or the memories be physically located in the same geographical place. That is, each of the processors and the memories used in exemplary embodiments of the invention may be located in geographically distinct locations and connected so as to communicate in any suitable manner. Additionally, it is appreciated that each of the processor and/or the memory may be composed of different physical pieces of equipment. Accordingly, it is not necessary that the processor be one single piece of equipment in one location and that the memory be another single piece of equipment in another location. That is, it is contemplated that the processor may be two or more pieces of equipment in two or more different physical locations. The two distinct pieces of equipment may be connected in any suitable manner. Additionally, the memory may include two or more portions of memory in two or more physical locations.

**[0057]** As described above, a set of instructions is used in the processing of various embodiments of the invention. The servers may include software or computer programs stored in the memory (*e.g.*, non-transitory computer readable medium containing program code instructions executed by the processor) for executing the methods described herein. The set of instructions may be in the form of a program or software or app. The software may be in the form of system software or application software, for example. The software might also be in the form of a collection of separate programs, a program module within a larger program, or a portion of a program module, for example. The software used might also include modular programming in the form of object oriented programming. The software tells the processor what to do with the data being processed.

**[0058]** Further, it is appreciated that the instructions or set of instructions used in the implementation and operation of the invention may be in a suitable form such that the processor may read the instructions. For example, the instructions that form a program may be in the form of a suitable programming language, which is converted to machine language or object code to allow the processor or processors to read the instructions. That is, written lines of programming code or source code, in a particular programming language, are converted to machine language using a compiler, assembler or interpreter. The machine language is binary coded machine instructions that are specific to a particular type of processor, *i.e.*, to a particular type of computer, for example. Any suitable programming language may be used in accordance with the various embodiments of the invention. For example, the programming language used may include assembly language, Ada, APL, Basic, C, C++, COBOL, dBase, Forth, Fortran, Java, Modula-2, Pascal, Prolog, REXX, Visual Basic, JavaScript and/or Python. Further, it is not necessary that a single type of instructions or single programming language be utilized in conjunction with the

operation of the system and method of the invention. Rather, any number of different programming languages may be utilized as is necessary or desirable.

**[0059]** Also, the instructions and/or data used in the practice of various embodiments of the invention may utilize any compression or encryption technique or algorithm, as may be desired. An encryption module might be used to encrypt data. Further, files or other data may be decrypted using a suitable decryption module, for example.

**[0060]** In the system and method of exemplary embodiments of the invention, a variety of “user interfaces” may be utilized to allow a user to interface with the mobile devices or other personal computing device. As used herein, a user interface may include any hardware, software, or combination of hardware and software used by the processor that allows a user to interact with the processor of the communication device. A user interface may be in the form of a dialogue screen provided by an app, for example. A user interface may also include any of touch screen, keyboard, voice reader, voice recognizer, dialogue screen, menu box, list, checkbox, toggle switch, a pushbutton, a virtual environment (*e.g.*, Virtual Machine (VM)/cloud), or any other device that allows a user to receive information regarding the operation of the processor as it processes a set of instructions and/or provide the processor with information. Accordingly, the user interface may be any system that provides communication between a user and a processor. The information provided by the user to the processor through the user interface may be in the form of a command, a selection of data, or some other input, for example.

**[0061]** The software, hardware and services described herein may be provided utilizing one or more cloud service models, such as Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure-as-a-Service (IaaS), and/or using one or more deployment models such as public cloud, private cloud, hybrid cloud, and/or community cloud models.

**[0062]** Although the embodiments of the present invention have been described herein in the context of a particular implementation in a particular environment for a particular purpose, those skilled in the art will recognize that its usefulness is not limited thereto and that the embodiments of the present invention can be beneficially implemented in other related environments for similar purposes.