

A MODULE AND METHOD FOR INTEGRATING DATA ACROSS DISPARATE BUSINESS APPLICATIONS

Field of the Invention

This invention relates to a module and method for integrating information or data across disparate business applications. In particular, the module is configured to receive a message from a business application, enhance/transform the message such that when the enhanced/transformed message is received by another disparate business application, the content of the message may be interpreted and subsequently processed by the another disparate business application.

Summary of Prior Art

Existing enterprises have an increasing number of disparate business applications for different purposes, typically from different vendors. These systems may comprise e-commerce platforms, CRM systems, ERP systems, SFA systems, accounting systems, call-center systems, etc. This results in a massive requirement for integration in order to allow internal business processes to run fully automatically.

In addition to these internal requirements of a company, the requirements for integration with external business partners are growing, drastically pushed by growing legal requirements (e.g. Nota Fiscal) and the increasing pressure from large companies (e.g. Walmart) to carry out business transactions completely electronically. In existing enterprises, as more and more vendors, clients and/or customers are brought on board, data or information will have to be shared seamlessly between the enterprise and the various other different business applications. As an example, the enterprise may utilize an internal accounting business applications while a vendor may utilize a commercial accounting business application such as Oracle, SAP, etc. As expected, each of these business applications will store data in their own unique formats, have their individual primary key logics, their individual data customizations and different business logics.

Hence, in order for data to be shared seamlessly between these two disparate accounting systems, the internal accounting system will have to be integrated with the commercial accounting system. This may be done using various types of data-integration methods, for example, using an application integration technique, a legacy migration approach, a data source consolidation method, a master data consolidation approach, a server consolidation approach, or any other software based techniques.

The sharing of data between two disparate business applications may be undertaken for a variety of reasons or initiatives, such as, for example, the migration of legacy databases, consolidation of data sources, consolidation of accounting records, regulatory compliance, Mergers and Acquisitions (M&A), or other Information Technology (IT) initiatives. Regardless of the reason, the initial stage of any data-integration between two disparate business applications typically involves the integration and organization of data. This is usually performed by individual coded logic with the help of a mapping tool.

A mapping tool covers structural mapping only. In other words, it is targeted at the different field names and the arrangement of information inside the message. But even here, there are limits and this often requires additional intervention through extra coding steps. Furthermore, other mapping aspects (such as value mapping, key mapping) and integration aspects (such as incomplete source message) are not covered and have to be solved by individual coding.

Key Mapping is the central aspect of integration as it manages the knowledge about the linkage of each data between different systems. An equally important aspect is the filling of missing data and the mapping of values to meet the different customized settings inside the different systems. These most complicated aspects are not covered by any tool. However, it is mandatory that the Receiver system is able to accept the data and therefore these aspects have to be implemented individually. This is very time-consuming and is also prone to human errors. The reliance on manual inputs increases the implementation time, the quality and the overall costs drastically.

Furthermore, costs will be constantly incurred as any changes or extensions made to the data mapping will require further manual interventions.

For example, in existing integration platforms, in order to determine whether a specific data file associated with the enterprise's internal accounting system exists in a commercial accounting system, the integration platform will typically have to initiate an Application Programming Interface (API) call with the commercial accounting system.

Once identified, another API call will then have to be carried out to retrieve the relevant data file from the commercial accounting system. Conversely, if the target data file is not found in the commercial accounting system, another API call will then have to be initiated to insert and link the data file to the commercial accounting system.

Obtaining the correct data via an API call is only 100% accurate if the search is done via the Receiver's Primary Key. However, this Key is typically not known to the Sender, which

causes an inaccurate search when auxiliary fields are utilized. Another approach proposed by those skilled in the art is to maintain the primary key of the sending system inside the target system which allows the sending system to determine the record via the Sender's Primary Key, which is however an unclean design work-around and not sustainable to cover integration between multiple systems).

However, in order for such an API call to be made in the first place, certain prerequisites have to be fulfilled. During the setup phase, integration consultants will have to determine the fields of the data file that are to be used so that the data file may be uniquely identified using a primary key. As an example, the data file may be identified based on the customers' entries under the fields 'name' and 'city' and these entries may be used as the file's primary key (under the assumption that this combination is sufficiently unique). However, when there are overlaps in the customer entries, such a solution becomes unviable as the entries are no longer distinct and unique.

To address this problem, another approach proposed by those skilled in the art is to initially store the sender's (e.g. the enterprise's internal accounting system) primary key in the receiver's system (e.g. the commercial accounting system). When the integration platform attempts to identify the existence of a data file in the receiver's system, the platform will attempt to retrieve the sender's primary key from the receiver's system. If it is able to do so, it will then be able to retrieve the associated receiver's primary key and the target data file associated with the receiver's primary key. Additionally, if the target data file has not been associated with a receiver's primary key, a new receiver primary key would also have to be individually defined for the data file and this would have to be repeated for all other data files as well.

However, this approach is limited to systems that provide options for user-defined-fields, so that the fields in the receiver system may not be misused to incorrectly store the sender's primary key. Such a solution can work under the mentioned restrictions, but when more than two business applications are to be integrated, this causes the overall complexity of the solution to increase drastically along with its associated reliability and costs.

For the above reasons, those skilled in the art are constantly striving to come up with a module and method that is capable of integrating data across disparate business applications in an automated, efficient and seamless manner.

Summary of the Invention

The above and other problems are solved and an advance in the art is made by systems and methods provided by embodiments in accordance with the invention.

A first advantage of embodiments of modules and methods in accordance with the invention is that the invention is able to determine, without manual inputs, the defined primary keys of a message received from a business application. The determined primary keys are then appended to the message, the corresponding primary key of the another business application is determined and subsequently provided to another business application so that content in the message from the business application may be seamlessly mapped to another message in the another business applications' format.

A second advantage of embodiments of modules and methods in accordance with the invention is if it is determined that the message from the business application has not been stored in another business application, the invention is able to determine the primary keys of the another business application that is to be tagged to the message based on the predetermined behaviour of the another business application.

A third advantage of embodiments of modules and methods in accordance with the invention is that the handling of primary keys is not limited to the primary keys of the main object, but also covers secondary keys related to business objects of any level.

A fourth advantage of embodiments of modules and methods in accordance with the invention is that is the invention processes messages in a fully automated manner, based on predetermined definitions, and mappings of customized values which may differ from one business application and another business application.

A fifth advantage of embodiments of modules and methods in accordance with the invention is that the invention enhances the incoming message from the business application in a fully automated manner and does so based on predefined definitions to complete the data for the another business application.

The above advantages are provided by embodiments of a module in accordance with the invention operating in the following manner.

According to a first aspect of the invention, a module for integrating information shared between a first business application and a second business application is disclosed, wherein the first and second business applications comprise disparate business applications, where the module comprises: a processing unit; and a non-transitory media readable by the processing unit, the media storing instructions that when executed by the processing unit cause the processing unit to: receive a first-message from the first business application; retrieve local-first primary keys L1-Keys associated with the first-message; for each of the retrieved local-first primary keys L1-Keys that has been assigned a global key GKey, identify

a local-second primary key L2-Key that has been paired with the local-first primary key L1-Key, and retrieve system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key; for each of the retrieved local-first primary keys L1-Keys that has not been assigned a global key GKey, generate and assign a global key GKey to the local-first primary key L1-Key to form a (Gkey, L1-Key) entry, receive a local-second primary key L2-Key generated for the (Gkey, L1-Key) entry, and retrieve system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key; append all the retrieved local-first primary keys L1-Keys, their associated global keys GKeys, and the respective retrieved system identities to the first-message; provide the appended first-message to an integration logic module, wherein the integration logic module is configured to: map content associated with each of the local-first primary keys L1-Keys to content in a second-message, and provide the second message to the second business application.

With regard to the first aspect of the invention, the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated by the module based on a Global Unique Identifier (GUID), a random number, a primary key generation algorithm in combination with a number range, or a primary key generation algorithm in combination with a defined logic.

With regard to the first aspect of the invention, the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated as part of a response by the second business application in response to an Application Programming Interface (API) call from the first business application, whereby the generated response is provided to the first business application.

With regard to the first aspect of the invention, the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated by the module based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having a similar time-stamp as a first API that was sent from the first business application to the second business application.

With regard to the first aspect of the invention, the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated by the module based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a

second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having fields that match predefined requirements.

According to a second aspect of the invention, a method for integrating information shared between a first business application and a second business application is disclosed, wherein the first and second business applications comprise disparate business applications, the method comprises the steps of: receiving, using a module, a first-message from the first business application; retrieving, using the module, local-first primary keys L1-Keys associated with the first-message; for each of the retrieved local-first primary keys L1-Keys that has been assigned a global key GKey, identifying, using the module, a local-second primary key L2-Key that has been paired with the local-first primary key L1-Key, and retrieving system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key; for each of the retrieved local-first primary keys L1-Keys that has not been assigned a global key GKey, generating and assigning, using the module, a global key GKey to the local-first primary key L1-Key to form a (Gkey, L1-Key) entry, receiving a local-second primary key L2-Key generated for the (Gkey, L1-Key) entry, and retrieving system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key; appending, using the module, all the retrieved local-first primary keys L1-Keys, their associated global keys GKeys, and the respective retrieved system identities to the first-message; providing, using the module, the appended first-message to an integration logic module, wherein the integration logic module maps content associated with each of the local-first primary keys L1-Keys to content in a second-message, and provides the second message to the second business application.

With regard to the second aspect of the invention, the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of: generating using the module, a Global Unique Identifier (GUID), a random number, a primary key generation algorithm in combination with a number range, or a primary key generation algorithm in combination with a defined logic.

With regard to the second aspect of the invention, the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of: generating, by the second business application, the local-second primary key L2-Key as part of a response to an Application Programming Interface (API) call from the first business application, whereby the generated response is provided to the first business application.

With regard to the second aspect of the invention, the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of: generating, using the module, the local-second primary key L2-Key based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having a similar time-stamp as a first API that was sent from the first business application to the second business application.

With regard to the second aspect of the invention, the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of: generating, using the module, the local-second primary key L2-Key based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having fields that match predefined requirements.

Brief Description of the Drawings

The above and other problems are solved by features and advantages of a system and method in accordance with the present invention described in the detailed description and shown in the following drawings.

Figure 1 illustrating a block diagram of a system for integrating information across disparate business applications in accordance with embodiments of the invention;

Figure 2 illustrating a block diagram representative of processing systems providing embodiments in accordance with embodiments of the invention;

Figure 3 illustrating a comparison between an exemplary message in the format of a first business application and the same message in the format of a second business application;

Figure 4 illustrating exemplary unique values that may be used as primary keys, secondary keys, links to other business objects, a determination result and the mapping of a customized value in the exemplary message in accordance with embodiments of the invention.

Figure 5 illustrates a process flow of the linking of a message that is in the format of a first business application to a message that has the format of a second business application in accordance with embodiments of the invention;

Figure 6 illustrating a process flow of the updating of an existing message that has the format of a second business application based on the content of a message that is in the format of a first business application in accordance with embodiments of the invention;

Figure 7 illustrating a process flow of the integration of two disparate business applications whereby one system comprises a sender's business application and the other system comprises a receiver's business application in accordance with embodiments of the invention;

Figure 8 illustrating a process flow of the generation of a local primary key for the receiver's business application in accordance with a first embodiment of the invention;

Figure 9 illustrating a process flow of the generation of a local primary key for the receiver's business application in accordance with a second embodiment of the invention; and

Figure 10 illustrating a process flow of the generation of a local primary key for the receiver data management system in accordance with a third embodiment of the invention.

Detailed Description

This invention relates to a module and method for integrating information or data across disparate business applications. In particular, the module is configured to receive a message from a business application, enhance the message such that any integration logic may easily transform the enhanced message into a format that can be easily processed by another disparate business application. In embodiments of the invention, the processing of the message may comprise the retrieval of the message from the other business application or may comprise the insertion of the message into the format of the other business application.

The present invention will now be described in detail with reference to several embodiments thereof as illustrated in the accompanying drawings. In the following description, numerous specific features are set forth in order to provide a thorough understanding of the embodiments of the present invention. It will be apparent, however, to one skilled in the art, that embodiments may be realised without some or all of the specific features. Such embodiments should also fall within the scope of the current invention. Further, certain process steps and/or structures in the following may not have been described in detail and the reader

will be referred to a corresponding citation so as to not obscure the present invention unnecessarily.

Further, one skilled in the art will recognize that many functional units in this description have been labelled as modules throughout the specification. The person skilled in the art will also recognize that a module may be implemented as circuits, logic chips or any sort of discrete component. Still further, one skilled in the art will also recognize that a module may be implemented in software which may then be executed by a variety of processors. In embodiments of the invention, a module may also comprise computer instructions or executable code that may instruct a computer processor to carry out a sequence of events based on instructions received. The choice of the implementation of the modules is left as a design choice to a person skilled in the art and does not limit the scope of this invention in any way.

Figure 1 illustrates a block diagram of the system for the integration of disparate business applications in accordance with embodiments of the invention. System 100 comprises of business application 105, business application 115 and integration module 110. It should be noted that business applications 105 and 115 both comprise disparate business application which are sitting on top or installed on top of database systems whereby each business application stores data according to their own proprietary format, have their data customized in their own way, have business objects tailored to their unique methods and process data according to their own proprietary rules. Business applications 105 and 115 provide their own logic, and may be exposed using various User Interfaces (UIs) and Application Programming Interfaces APIs such as, but are not limited to, SAP S/4, Oracle E-Business Suite, salesforce.com, Quickbooks, Shopify, Amazon, Magento, Xero , and all other types/forms of business applications. It should be noted that the designations used to describe the business applications throughout this description are not to be construed as limiting the invention to specific numbers and/or types of business applications and these designations are selected purely for the convenience of the reader.

As for integration module 110, this module comprises of various components and a processing system for implementing embodiments of the invention. Figure 2 illustrates an exemplary processing system 200 that may be provided within module 110. One skilled in the art will recognize that the exact configuration of processing system may be different, the exact configuration may vary and Figure 2 is provided by way of example only.

In embodiments of the invention, each of the modules may comprise controller 201 and user interface 202. User interface 202 is arranged to enable manual interactions between a user and each of these modules as required and for this purpose includes the input/output

components required for the user to enter instructions to provide updates to each of these modules. A person skilled in the art will recognize that components of user interface 202 may vary from embodiment to embodiment but will typically include one or more of display 240, keyboard 235 and track-pad 236.

Controller 201 is in data communication with user interface 202 via bus 215 and includes memory 220, processor 205 mounted on a circuit board that processes instructions and data for performing the method of this embodiment, an operating system 206, an input/output (I/O) interface 230 for communicating with user interface 202 and a communications interface, in this embodiment in the form of a network card 250. Network card 250 may, for example, be utilized to send data from these modules via a wired or wireless network to other processing devices or to receive data via the wired or wireless network. Wireless networks that may be utilized by network card 250 include, but are not limited to, Wireless-Fidelity (Wi-Fi), Bluetooth, Near Field Communication (NFC), cellular networks, satellite networks, telecommunication networks, Wide Area Networks (WAN) and etc.

Memory 220 and operating system 206 are in data communication with CPU 205 via bus 210. The memory components include both volatile and non-volatile memory and more than one of each type of memory, including Random Access Memory (RAM) 220, Read Only Memory (ROM) 225 and a mass storage device 245, the last comprising one or more solid-state drives (SSDs). Memory 220 also includes secure storage 246 for securely storing secret keys, or private keys. One skilled in the art will recognize that the memory components described above comprise non-transitory computer-readable media and shall be taken to comprise all computer-readable media except for a transitory, propagating signal. Typically, the instructions are stored as program code in the memory components but can also be hardwired. Memory 220 may include a kernel and/or programming modules such as a software application that may be stored in either volatile or non-volatile memory.

Herein the term “processor” is used to refer generically to any device or component that can process such instructions and may include: a microprocessor, microcontroller, programmable logic device or other computational device. That is, processor 205 may be provided by any suitable logic circuitry for receiving inputs, processing them in accordance with instructions stored in memory and generating outputs (for example to the memory components or on display 240). In this embodiment, processor 205 may be a single core or multi-core processor with memory addressable space. In one example, processor 205 may be multi-core, comprising—for example—an 8 core CPU. In another example, it could be a cluster of CPU cores operating in parallel to accelerate computations.

With reference to Figure 1, when system 105 generates a message 120 (which is in the format of business application 105) for business application 115, message 120 will be first intercepted and processed into a suitable format for system 115 by integration module 110. If message 120 were to be directly sent to system 115, system 115 would not be able to process message 120 as its format would not be compatible with the format utilized by business application 115.

An exemplary format of a message sent by business application 105 is illustrated as message-format 305 in Figure 3 while the format adopted by business application 115 is illustrated as message-format 310. As can be seen from these two message-formats, they both utilize different data structures, they tailor data differently, and they assign and handle primary keys in their own unique styles. Therefore, it is widely accepted that it would be a challenge to generate message-format 310 directly from message-format 305.

One skilled in the art will recognize that a primary key/ local primary key is defined as a unique identifier that is associated with a particular record. Its main use is to uniquely identify a record or specific content within a message or a table.

Figure 4 illustrates local primary keys 410a-d as found in invoice 400 (i.e. message format 310 in Figure 3). In this example, although the message is shown to be in the format of an invoice, one skilled in the art will recognize that the message may comprise of any other types of documents without departing from the invention.

It can be seen that in invoice 400, primary key 410a comprises (A8B226576C); primary key 410b comprises (4789990.A); primary key 410c comprises (itm013668); and primary key 410d comprises (itm003445).

In this example, the function of primary key 410a is to provide a document identifier to invoice 400 such that this invoice 400 may be identified simply by calling primary key 410a. Similarly, primary key 410b functions as a link to an identity of a customer of invoice 400, e.g. location 425a and address 425b, such that when primary key 410b is called, it will return location 425a and address 425b. As for primary key 410c, in this example, it acts as a link to the details of a particular item, e.g. item_details 430; and similarly, primary key 410d acts as a link to the details of another item, e.g. item_details 435. Hence, when primary keys 410c and 410d are called, it will return the details of items 430 and 435 respectively.

In other embodiments of the invention, 410b, 410c and 410d may be identified as secondary keys whereby each of these secondary keys are associated with individual

business objects. Reference 415 illustrates a mapping of a customization value while 425a and 425b represent enhancements that were added to the message.

The 'action=update' feature 450, illustrates the determination result - which is the result of the invention's logic to detect if the message is already available in receiver system (update) or if the message is not available in the receiver system (insert).

Figure 5 illustrates a process flow showing the linking of message 501 that is in the format of first business application 105 to a message (not shown) that has the format of second business application 115 in accordance with embodiments of the invention.

For all messages, that are sent from application 105 to application 115, all the information about the various local primary keys (and their functions) will be first generated and stored within module 110. With the next appearance of the already passed messages, sent from application 105 to application 115, these stored information will be used.

In operation, message 501 will be initially intercepted and received by integration module 110. For this example, as the intention is to link message 501 to a message in application 115, module 110 will retrieve a local primary key, L1-Key (i.e. "43557C0A8B2269715"), associated with the document identity of message 501 from message 501. It should be noted that at this initial stage, the data in message 501 only comprises the local primary key L1-Key. Module 110 then determines at step 552 whether the retrieved local primary key L1-Key has been assigned with a global key GKey and as a result has previously been paired with a local primary key associated with a message of second business application 115.

In this example, at step 552, as the local primary key L1-Key has not been assigned with a global GKey (and has not been paired with a local primary key associated with a message of the second business application), module 110 will proceed to generate a new GKey and will link this new GKey to the L1-Key. Module 110 will also retrieve the unique identifier (SysId1) of the Business Application 105 and link this SysId1 to the GKey-L1-Key pair resulting in the entry GKey-SysId1-L1-Key being entered into module 110. All this information is then appended to message 501 at this step. This enhances the information contained within message 501. In this embodiment of the invention, module 110 proceeds to generate, at step 554, a local primary key L2-Key (that is associated with a message of the second business application) for local primary key L1-Key and will store the key, that is paired with the previously generated GKey into database 505. The enhanced message 501 is then provided to any integration logic to prepare and send the message to application 115.

In an embodiment of the invention, the local primary key L2-Key may be generated directly by module 110. In this embodiment the local primary key L2-Key may be generated based on a Global Unique Identifier (GUID); based on a random number of a particular format and length; based on a primary key generation algorithm followed by a number range having a specific format and length; or based on a primary key generation algorithm followed by a defined logic.

In another embodiment of the invention, the local primary key L2-Key may be generated by business application 115 and then subsequently provided to module 110. In this embodiment, module 110 is only made aware of the local primary key L2-Key that is to be paired with the local primary key L1-Key only after it has received it from business application 115. As such, in this embodiment, the local primary key L2-Key may be determined by a “response” approach, by “a post call with timestamp” approach or by a “post call with filter” approach.

Under the “response” approach, module 110 will perform a synchronous insert-API call of message 501 (transformed to 115 format) to application 115. Upon receiving this API call, system 115 will generate an appropriate local primary key L2-Key and subsequently provide this L2-Key to module 110 as part of the response to the API call. Upon receiving the response from application 115, module 110 is then able to identify the L2-Key inside the response data, based on definitions, describing from where to pick the primary key in the response data of this document type.

When the “post call with timestamp” approach is adopted, module 110 will similarly perform an insert-API call of message 501 to system 115. However, under this approach, application 115 will not respond to such an API call with a generated primary key. After a predetermined period of time has lapsed, module 110 will perform another API call to application 115 to retrieve all document types along with their corresponding documents that were attempted to be “inserted” into application 115 over a particular period. This means that if this particular period were to be set to the period/time-range that the original insert-API call of message 501 was sent to application 115, this would result in application 115 retrieving all the documents that are of the same document-type as message 501. The bundle of documents having the same document-type as message 501 would then be provided to module 110. Upon receiving these documents, module 110 will then identify the recently inserted document and will pick the corresponding primary key L2-Key from this document, using definitions which will describe where the primary key is to be obtained from this document type.

Under the “post call with filter” approach, as application 115 will not respond to an insert-API call from module 110 with a generated primary key and does not support the retrieval of all document types by timestamps along with their corresponding documents that were attempted to be “inserted” into application 115 over a particular period. Module 110 will have to perform another API call to application 115 to retrieve all documents that are of the same document type as message 501, filtered to before inserted identifier fields (e.g. name, total amount, last updated, etc. (based on definitions for this system)). The documents having the same document-type as message 501, filtered by identifier fields would then be provided to module 110. Upon receiving these documents, module 110 will then identify the recently inserted document and will pick the corresponding primary key L2-Key from this document, using definitions which will describe where the primary key is to be obtained from this document type.

One skilled in the art will recognize that other methods and/or techniques may be used at step 554 to generate a suitable local primary key L2-Key for local primary key L1-Key and that the generation of the L2-Key is not specifically limited to the approaches and/or embodiments described above.

Once a local primary key L2-Key associated with a message of the second database has been generated at step 554, module 110 then proceeds to add a new entry into database 505, pairing the already available GKey (which is paired with L1-Key) with the new L2-Key. At step 554, module 110 then retrieves the unique identifier (SysId2) (which is used to identify Business Applications used by system 115) and will link this with the L2-Key. At this stage, it is useful to note that the entry for the primary key of the incoming message (L1-Key-SysId1-GKey) has been previously entered into module 110 and now, an entry for the primary key of the inserted message (L2-Key-SysId2-GKey) is created in database 505, which may be provided within module 110 or may be located in a remote location and communicatively connected to module 110. All this information is then appended to message 501 at step 558. This enhances the information contained within message 501 for further processing.

It should be noted that although the description above only discloses the appending of a single set of local primary keys L1-Key and L2-Key, the retrieved system identities and the assigned GKey, one skilled in the art will recognize that two or more such sets (comprising of local primary keys, the associated retrieved system identities and the assigned global keys GKeys) may be generated and processed as described above. The above steps ensure that the pairing of all the relevant local primary keys in message 501 (which is in the format of system 105) would be performed by module 110 before appended message 501 (which is enriched with all the sets of information) are provided to an integration logic for further

processing into the appropriate format for business application 115. With the use of the key entry format (GKey-SysId-LKey), any number of systems may be logically linked.

Figure 6 illustrates a process flow showing the updating of the content in message 601 (that is in the format of first business application 105) to the content in a corresponding message (not shown) that has the format of second business application 115 in accordance with embodiments of the invention.

In operation, message 601 will be initially intercepted and received by integration module 110. For this example, as the intention is to update the contents of message 601 to contents in a corresponding message in application 115, module 110 will retrieve a local primary key, L1-Key (i.e. "43557C0A8B2269715"), associated with the document identity of message 601 from message 601.

Module 110 then determines at step 652 whether the retrieved local primary key L1-Key has been assigned with a global key GKey. In this example, as the local primary key L1-Key has been assigned with a global GKey, i.e. GKey '2000' and its SysID1 "0010000101", module 110 then proceeds to identify the local primary key L2-Key via its SysID2 that has been previously paired with the local primary key L1-Key via the GKey '2000'. In this example, the paired local primary key L2-Key would comprise "A8B226576C" and SysID2 is "0010000102". It is useful to note that in this example, the L2-Key would be the primary key that is associated with the document identity of a message of the second business application.

Set 656 is then appended by module 110 to message 601 at step 658. This enhances the information contained within message 601.

Integration logic (not shown) will leverage this information to translate the message into the format and conventions, required by business application 115. Module 110 then provides the transformed message 601 to business application 115. Upon receiving the transformed message 601, application 115 may then proceed without any further logic related to integration as the message is completely in the proper format of application system 115. It should be noted that although the description above only discloses the appending of a single set of local primary keys L1-Key and L2-Key, the retrieved system identities and the assigned GKey, one skilled in the art will recognize that two or more such sets (comprising of local primary keys, the associated retrieved system identities and the assigned global keys GKeys) may be generated and appended to the message, sent to any integration logic, and processed accordingly, afterwards handover to application 115 so that multiple contents may be mapped simultaneously.

In yet another embodiment of the invention, if module 110 determines that certain information is missing from a message that is sent from system 105 to system 115 (e.g. only business partner primary key (43557C0A8B2269715) is in the message, but address of business partner is missing, but is needed to handover to system 115), module 110 may obtain this missing information by an API call to the system 105 based on the business-partner-primary key. Module 110 obtains the information to retrieve the data using the primary key and the primary key may be obtained based on definitions that inform module 110 where the primary key is to be found in the message and from the definitions of the API. The required data is then retrieved and added in a special section in the appended message.

Figures 7-10 sets out exemplary processes that may be performed in module 110 for integrating information, e.g. a first-message, shared between a first business application and a second business application, wherein the first and second business applications comprise disparate business applications.

Process 700 begins at step 702 with process 700 receiving a first-message from the first business application. Process 700 then retrieves local-first primary keys L1-Keys (this includes the actual Primary Key of the main business object (410a) and also all Secondary Primary Keys which are in the message, linking to other object (410b, 410c, 410d)) from the received first-message at step 704.

Process 700 will proceed to step 706 for each of the retrieved local-first primary keys L1-Keys that has been assigned a global key GKey. Conversely, process 700 will proceed to step 712 for each of the retrieved local-first primary keys L1-Keys that has not been assigned a global key GKey

At step 706, process 700 will proceed to retrieve global keys GKeys associated with all the local-first primary/second keys L1-Keys as obtained from step 704. At step 708, process 700 will identify and retrieve all existing L-Keys of other systems (i.e. other than the local-first primary key L1-Key) which are associated with the existing GKeys. Process 700 also retrieves the first business application's relevant SysIds, i.e. SysId1s, and also the second business application's relevant SysIds, i.e. SysId2s. These retrieved SysIds are then appended to the GKey-L1-Key pairs and GKey-L2-Key pairs to form (GKey-SysId1-L1-Key) and (GKey-SysId2-L2-Key) entries. This takes place at step 710. These entries are then appended to the first-message and subsequently provided to any integration logic to transform the message accordingly and handover to the second business application.

Conversely, at step 712 (where each of the retrieved local-first primary keys L1-Keys that has not been assigned a global key GKey), process 700 will generate new GKeys for

each of these retrieved L1-Keys that do not have an associated GKey. The new GKeys are then linked with their associated L1-Keys. SysID1 is also retrieved and added to these GKey-L1-Key pairs. The GKey-SysID1-L1-Key entries are then appended to the first message.

In an embodiment of the invention, at step 716, process 700 receives local-second primary keys L2-Keys that are generated by the second business application for each of the (L1-Key-GKey-SysID1) entries after the GKey-SysID1-L1-Key entries have been appended to the first message and sent (after transforming by any integration logic) to the second business application. Process 700 then pairs the received entries accordingly with the associated GKey to form (L2-Key-GKey-SysID2) entries.

In another embodiment of the invention, at step 716, process 700 receives local-second primary keys L2-Keys that are generated by module 110 for each of the (L1-Key-GKey-SysID1) entries as the second business application has authorized module 110 to generate the local-second primary keys L2-Keys for each of the entries. Process 700 then pairs the entries accordingly with the associated GKey to form (L2-Key-GKey-SysID2) entries. The (GKey-SysID1-L1-Key) entries and the (L2-Key-GKey-SysID2) entries are then appended to the first message and sent to an integration logic to be transformed. With these actions the transformed message was sent to the second business application.

Regardless of either embodiment, these (L2-Key-GKey-SysID2) entries are then stored in the module for further processing and this takes place at step 718.

Process 700 then ends.

Figure 8 illustrates an exemplary flowchart of process 800 for generating a local-second primary key L2-Key for a retrieved local-first primary key L1-Key in accordance with an embodiment of the invention, i.e. step 716 of process 700. Process 800 begins at step 802 with ~~process 800~~ sending an API call to the second business application. Upon receiving the response from the second business application at step 804, process 800 then proceeds to step 806. At this step, process 800 will select an appropriate local-second primary key L2-Key from the response to be paired with the local-first primary key L1-Key. Process 800 then ends.

Figure 9 illustrates an exemplary flowchart of process 900 for generating a local-second primary key L2-Key for a retrieved local-first primary key L1-Key in accordance with another embodiment of the invention, i.e. step 716 of process 700. Process 900 begins at step 902 by sending an API call to the second business application. However, in this embodiment, as the second business application did not respond the generated primary key to the API call

sent at step 902, process 900 then proceeds to send another API call to the second business application to retrieve target documents having a particular timestamp. This takes place at step 904. This causes all the documents in the second business application that are of the same document-type as the first-message and in the requested timeframe to be sent to module 110 and these documents are received by process 900 at step 906. Process 900 then identifies a suitable document from the received response and based on the identified document, process 900 then retrieves and selects the suitable local-second primary key L2-Key to be paired with the local-first primary key L1-Key. Process 900 then ends.

Figure 10 illustrates an exemplary flowchart of process 1000 for generating a local-second primary key L2-Key for a retrieved local-first primary key L1-Key in accordance with yet another embodiment of the invention, i.e. step 716 of process 700. Process 1000 begins at step 1002 by sending an API call to the second business application. However, in this embodiment, as the second business application did not respond to the API call sent with a generated primary key at step 1002, process 1000 then proceeds to send another API call to the second business application to retrieve target documents having the same document-type, and filtered by identifier fields (e.g. name, total amount, last updated, etc. (based on definitions for this system)). This takes place at step 1004. This causes all the documents in the second business application that are of the same document-type as the first-message and fitting to the defined filter to be sent to module 110 and these documents are received by process 1000 at step 1006. Process 1000 then identifies the suitable document from the received response and based on the identified document, process 1000 then retrieves and selects the suitable local-second primary key L2-Key to be paired with the local-first primary key L1-Key. This takes places at step 1008 Process 1000 then ends.

Numerous other changes, substitutions, variations and modifications may be ascertained by one skilled in the art and it is intended that the present invention encompass all such changes, substitutions, variations and modifications as falling within the scope of the appended claims.

CLAIMS:

1. A module for integrating information shared between a first business application and a second business application, wherein the first and second business applications comprise disparate business applications, the module comprising:

- a processing unit; and

- a non-transitory media readable by the processing unit, the media storing instructions that when executed by the processing unit cause the processing unit to:

 - receive a first-message from the first business application;

 - retrieve local-first primary keys L1-Keys associated with the first-message;

 - for each of the retrieved local-first primary keys L1-Keys that has been assigned a global key GKey, identify a local-second primary key L2-Key that has been paired with the local-first primary key L1-Key, and retrieve system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key;

 - for each of the retrieved local-first primary keys L1-Keys that has not been assigned a global key GKey, generate and assign a global key GKey to the local-first primary key L1-Key to form a (Gkey, L1-Key) entry, receive a local-second primary key L2-Key generated for the (Gkey, L1-Key) entry, and retrieve system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key;

 - append all the retrieved local-first primary keys L1-Keys, their associated global keys GKeys, and the respective retrieved system identities to the first-message;

 - provide the appended first-message to an integration logic module,

 - wherein the integration logic module is configured to:

 - map content associated with each of the local-first primary keys L1-

 - Keys to content in a second-message, and

 - provide the second message to the second business application.

2. The module according to claim 1 wherein the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated by the module based on a Global Unique Identifier (GUID), a random number, a primary key generation algorithm in combination with a number range, or a primary key generation algorithm in combination with a defined logic.

3. The module according to claim 1 wherein the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated as part of a response by the second business application in response to an Application Programming Interface (API) call from the first business application, whereby the generated response is provided to the first business application.

4. The module according to claim 1 wherein the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated by the module based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having a similar time-stamp as a first API that was sent from the first business application to the second business application.

5. The module according to claim 1 wherein the local-second primary key L2-Key generated for the (Gkey, L1-Key) entry is generated by the module based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having fields that match predefined requirements.

6. A method for integrating information shared between a first business application and a second business application, wherein the first and second business applications comprise disparate business applications, the method comprising:

receiving, using a module, a first-message from the first business application;

retrieving, using the module, local-first primary keys L1-Keys associated with the first-message;

for each of the retrieved local-first primary keys L1-Keys that has been assigned a global key GKey, identifying, using the module, a local-second primary key L2-Key that has been paired with the local-first primary key L1-Key, and retrieving system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key;

for each of the retrieved local-first primary keys L1-Keys that has not been assigned a global key GKey, generating and assigning, using the module, a global key GKey to the local-first primary key L1-Key to form a (Gkey, L1-Key) entry, receiving a local-second primary key L2-Key generated for the (Gkey, L1-Key) entry, and retrieving system identities associated with the local-first primary key L1-Key and the local-second primary key L2-Key;

appending, using the module, all the retrieved local-first primary keys L1-Keys, their associated global keys GKeys, and the respective retrieved system identities to the first-message;

providing, using the module, the appended first-message to an integration logic module, wherein the integration logic module maps content associated with each of the local-first primary keys L1-Keys to content in a second-message, and provides the second message to the second business application.

7. The method according to claim 6 wherein the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of:

generating using the module, a Global Unique Identifier (GUID), a random number, a primary key generation algorithm in combination with a number range, or a primary key generation algorithm in combination with a defined logic.

8. The method according to claim 6 wherein the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of:

generating, by the second business application, the local-second primary key L2-Key as part of a response to an Application Programming Interface (API) call from the first business application, whereby the generated response is provided to the first business application.

9. The method according to claim 6 wherein the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of:

generating, using the module, the local-second primary key L2-Key based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having a similar time-stamp as a first API that was sent from the first business application to the second business application.

10. The method according to claim 6 wherein the generating of the local-second primary key L2-Key for the (Gkey, L1-Key) entry comprises the step of:

generating, using the module, the local-second primary key L2-Key based on a bundle of documents received from the second business application, the bundle of documents being sent by the second business application to the first business application in response to a second continuous API call from the first business application, whereby the bundle of documents comprise documents having a same document-type as the first-message and having fields that match predefined requirements.