INTEGRATED DATA SHARING SYSTEM

Field of the Invention

The invention relates to the sharing of data between disparate computer systems, in particular to a data sharing system which simplifies the consolidation and control of information to create documents on disparate computer systems on a network.

Background and Prior Art

Currently, stand-alone systems developed by different vendors are used for different applications. Each system has its own logic and databases. When several of these stand-alone systems are to be used together, it is important that these stand-alone systems are able to share data, so that common data fields need not be re-entered for each system.

At present, in order to generate documents created by disparate systems, the user may have to log into each of the separate stand-alone systems to key in information, especially for document-based systems. Forms have to be subsequently generated and routed to back-end entities. Nevertheless, each of these systems may require the same information for common fields. For example, in a system for dealing with cargo insurance, information like the "Sender" field and the "Addressee" field can be used in different systems, such as an export permit application system, goods description system and so on. There may thus be a number of systems that require the same information.

Usually, the computer programmers would aggregate all fields into a single common database to consolidate the information for access by the different

systems or each system may include logic to the effect that if there is a change in the definition of a particular field, it will require a corresponding update in other systems. However, these methods are often cumbersome and can be inefficient.

It is therefore an object of the invention to aim to provide a more efficient means to implement an integrated data sharing system and to seek to avoid some of the disadvantages of the prior art.

Accordingly, the invention provides a system for allowing sharing of data from disparate sources, comprising a user interface that enables a user to access the disparate sources, an intermediary component operationally linked between the user interface and the disparate sources, and a knowledge component of the intermediary component adapted to allow bi-directional triggering of data flow between one or a plurality of sources and another of said sources.

Using the invention it is possible to create an environment that will more easily integrates all the features of each of the disparate sources, or systems, while allowing easier maintenance and facilitate the adding of additional applications.

It will be convenient to hereinafter describe the invention in greater detail by reference to the accompanying drawings which illustrate one embodiment of the invention in an Internet-based environment. The particularity of the drawings and the related description is not to be understood as superseding the generality of the broad identification of the invention as defined by the claims.

Brief description of the Drawings

Figure 1 is an architectural overview of the major components of a preferred embodiment of a system according to the invention.

Figure 2 is an overview of the process for populating data between different document types, the different document types essentially being used by different applications.

Figure 3 is a system flowchart illustrating the navigation between document types in the system.

Figure 4 is a flowchart illustrating the process of detecting any changes in the data and the provision of user alerts in the system.

Figure 5 is a sample database definition table to indicate the field mappings between different applications.

Figure 6 is a sample database definition table showing the relationships between data from different documents and the auto-learning function of the system.

Detailed description of the preferred embodiment of the invention

Figure 1 is an architectural overview of the major components of the preferred embodiment of the invention. A user 1 is able to access a series of disparate computer systems or applications 40-48 through a user interface, via an intermediary component 5. The intermediary component 5 comprises the major functional components of the system, and preferably combines an alert component 10, a navigation component 20 and

knowledge component 30 to integrate the distributed disparate applications 40-48.

The alert component 10 detects changes in the data fields and preferably provides user alerts upon occurrence of the changes. The navigational component 20 navigates between different document types in the system and is capable of inferring a document flow, if any. The knowledge component 30 consolidates relationships between the various data fields and populates data for different document types in the system. The programming language is preferably an object-oriented language, such as Java programming language. The system preferably forms a single integrated entity on a web deployment environment comprising a web server and possibly multiple application servers.

Figure 2 is an overview of the process for populating data between different document types, the different document types essentially being used by different applications. Document-related and field-related knowledge are consolidated at the knowledge component 30 outside the disparate applications 40-48 by the use of a table of field mappings (see Figure 5). The knowledge component 30 in Figure 1 separates the inter-document field relationships and related processing logic. It will be appreciated that different document types are still residing in the different applications 40-48. As discussed below, the knowledge component contains the mappings and dependencies between the various fields, and the transformation logic for each pair of document types. It allows mappings from one or multiple document types to another document type. This construction allows the bi-directional trigger of data flow, by the use of object-oriented programming, wherein the objects are capable of containing data as well as procedure to manipulate the data.

The knowledge component 30 will first determine the type of target document from the input in the user session. It will then obtain the source documents, and retrieve the data relationships and dependencies from a table of field mappings (see Figure 5). Once relevant fields are identified, the target document is populated with data from the source documents. Preferably, the system is implemented in a World-Wide Web based Internet environment. Thus, the target document may be put into a HyperText Transfer Protocol (HTTP) session on the Internet. The Uniform Resource Locator (URL) address of the target application is then retrieved and the target application is informed to further process the target document.

At the application level 40-48, the target application retrieves a populated document from the user session. The target application processes the data and allows the user to update the data.

Figure 3 is a system flowchart illustrating the navigation between document types in the system. It demonstrates the navigational flexibility of the navigation component 20 in controlling the sequence of document flow. The system has the ability to branch out, if necessary, from pre-determined document flow sequences. It also has the ability to track a document sequence by the use of a definition table (see Figure 6).

The navigation component 20 of the system first obtains the status of a sequence flow. If there is an available user-defined document flow, the system infers the next possible target document type based upon the user-defined document flow. Otherwise, the system will infer the next possible target document type based upon the past experience of the user, in accordance with the table in Figure 6 showing the relationships between data from different documents recorded based upon prior activity of the user working on the system. The user has the option to accept the

suggested target document type. Otherwise, the user would select a target document type he requires.

The status of the sequence flow is then updated for later reference in Figure 6. Thereafter, the workflow proceeds back to the knowledge component 30 (see Figure 2). The navigational component 20 of the system thus determines, where applicable, the next target document type in flow sequence, based on user-defined document flow or past transaction data.

Figure 4 is a flowchart illustrating the process of detecting any changes in the data and the provision of user alerts in the system. The system will detect data changes in various document types in each of the applications once these changes occur. Thereafter, it will alert all users that have earlier made use of these data by reading the definition table (see Figure 6) which has recorded the previous target documents created by users of the system, and the source documents used to populate them.

The application 40-48 would first inform the alert component 10 of data changes to any of the documents. The alert component 10 assesses the definition table listing the different documents to trace and locate other documents that contain data that was derived from the changed document. The alert component 10 then retrieves the associated user-ids from the table. The alert component 10 is capable of alerting the affected users via e-mails, messages or any other communication mechanisms.

Figure 5 is a sample database definition table to indicate the field mappings between different applications. The values of the fields are preferably inserted by the system administrator before the system is used. The use of the table for the retrieval of data relations and dependencies will be illustrated with reference to the following sample table 1A, which illustrates one of the ways in which the table may be used:

Table 1A:

Source Application Id	Source Bind Name	Source BindSeq No	Source Doc Type	Source Field Name	Source FieldType	Target Application Id	Target DocType	Target FieldName	Target FieldType	Special Rule Id
App A	Address	0	Doc A	Single_ Field C	Section D	App C	Doc C	Single_ Field F	Section A	
App A		-1	Doc A	Field E	Group B	App C	Doc C	Field M	Group D	Sr1
App B		-1	Doc B	Single_ Field A	Section A	App A	Doc A	Single_ Field G	Section C	
App B		-1	Doc B	Single_ Field C	Section E	App A	Doc A	Single_ Field B	Section F	
App A	Address	1	Doc A	Single_ Field D	Section D	App C	Doc C	Single_ Field F	Section A	

Table 1A contains five hypothetical document field relationships. Assuming that the source document type is *Doc A* and target document type is Doc C (i.e. the user is populating *Doc C* with data from *Doc A*), the document field relationships between these 2 documents can be retrieved by the following means:

Search the database for rows where

column **SourceDocType = Doc A** AND column **TargetDocType = Doc C**

Using the values from Table 1A, this database query will yield rows 1, 2 and 5.

From row 1, the system infers that Field C in Section D of Doc A is related to Field F in Section A of Doc C. The prefix "Single_" of SourceDocType indicates that Field C is a single field (possibly a textbox) in Section A of Doc A. For such single fields, the system only needs to map a single value to the target document.

From row 2, the system infers that Field E in Group B of Doc A is related to Field M of Group D of Doc C. Since SourceDocType is not prefixed with the value "Single_", it implies Field E consists of a group of values (possibly a column of values in a table). For such group field, the system needs to map a group of values to the target document.

The value of "Address" in SourceBindName of both row 1 and 5 indicates that values of Fields C and D in Section D of Doc A is to be joined to form a single value and be mapped to Field F of Section A in Doc C. The placement priority between Fields C and D in the joined result will be determined through the value in BindSeqNo. In this case, Field C has a value of 0 and Field D has a value of 1. This indicates that value of Field C should come before that of Field D. Thus, the result will be

<value of Field C><value of Field D>

In addition, the value of "Sr1" in field "Special Rule ID" indicates that there are some unique field mapping processing that is needed to be done, but cannot be captured entirely in the database schema according to the table of field mappings. The additional field mapping processing logic is contained in a separate object, where each logic is identified by the special rule ID (which in this case is "Sr1").

Figure 6 is a sample database definition table showing the relationships between data from different documents and the learning function of the system to aid the navigational component 30. The use of the definition table will now be illustrated for documents relationship and automatic learning with reference to the following sample table 2A, which illustrates one of the ways in which the table may be used.

The values in table 2A are inserted by the system's computer program to record the activities of each user according to the target document created and the source document from which data had been exported. The table may be used by both the navigational component and the alert component according to the described embodiment.

Table 2A

UserId	Source Application ID	Source DocType	Source DocID	TargetApplication ID	Target DocType	Target DocID	Relationship identifier
Usr1	App1	TypeA	DocID1	App2	TypeB	DocID2	R1
Usr1	App2	TypeB	DocID2	App3	TypeC	DocID3	R1
Usr1	App3	TypeC	DocID3	App4	TypeD	DocID4	R1
Usr1	App1	TypeA	DocID8	App2	TypeB	DocID9	R2

Table 2A illustrates some sample data of the document relationship table. From the data, the relationship module can form the following document relationships:

There are 2 sets of relationships, identified by R1 and R2. R1 consists of the following relationship (note that each document is uniquely identified by its document identifier): **DocID1 -> DocID2 -> DocID3 -> DocID4**, which means that the document with **document id = DocID1**, is the source/parent document for this relationship. DocID4 imports its data from DocID3, DocID3 imports its data from DocID2 and DocID2 imports its data from DocID1. The system knows that DocID1, DocID2, DocID3 and DocID4 are related as they share the same relationship identifier. On the other hand, R2 consists of : **DocID8 -> DocID9**

From the same table, the navigational component can deduce that there are two transactions where data from document of TypeA is exported to document of TypeB. There is one occurrence each where data from document of TypeB is exported to document of TypeC and from document of TypeC to document of TypeD. If a document of TypeA is created, the learning engine can deduce with some probability that document of TypeB will be created next.

It will be understood that the term "document" used herein refers to any input data whether in hard form or software form.

The invention described herein is susceptible to variations, modifications and/or additions other than those specifically described and it is to be understood that the invention includes all such variations, modifications and/or additions which fall within the spirit and scope of the above description.

CLAIMS

1. A system for allowing sharing of data from disparate sources, comprising:

a user interface that enables a user to access the disparate sources, an intermediary component operationally linked between the user interface and the disparate sources, and

a knowledge component of the intermediary component adapted to allow bi-directional triggering of data flow between one or a plurality of sources and another of said sources, wherein the system may be capable of inferring the next possible target document from one or more records of document sequence flow information recorded from previous activities of the user.

2. A system according to Claim 1, the sources comprising disparate computer systems, or applications, data being shareable between them the knowledge component having a table of field mappings indicating the relationships between different fields of documents used by the disparate computer systems or applications.

3. A system according to Claim 2, wherein the table of field mappings includes information on source documents and target documents sharing one or more fields.

4. A system according to Claim 3 wherein the table of field mappings include transformation logic between each pair of documents to indicate the manner in which one or more fields of the source document is to populate one or more fields of the target document.

5. A system according to Claim 4, wherein a target application further processes the target document after the target document is populated with data.

6. A system according to any one of Claims 2 to 5, which includes a rule identification field to indicate a unique field mapping processing that cannot be captured entirely in the table of field mapping.

7. A system according to any one of Claims 2 to 6, wherein the table of field mappings are created prior to the system being used.

8. A system according to any one of Claims 2 to 7, comprising a navigation component for facilitating the user to navigate between different document types in the disparate computer systems.

9. A system according to Claim 8, comprising a definition table accessible by the navigational component, wherein the definition table contains information on source documents and target documents created by a user using said source document when the user is accessing the system.

10. A system according to Claim 9, wherein the definition table includes a relationship identifier field to indicate the relationship between source documents and target documents used by the user.

11. A system according to Claim 9 or 10, wherein from the definition table, the navigational component is capable of deducing the sequence of documents likely to be created by the user.

12. A system according to any one of Claims 2 to 11, which includes an alert component to detect changes in the data and to alert the user of changes in the data of documents which have been created.

13. A system according to Claim 12, when dependant on Claim 10, wherein the alert component assesses the definition table to locate target documents created by the user that contain data derived from the changed data.

14. A system according to Claim 12 or 13, wherein the alert component informs users of changes to data via e-mails, messages or other communication mechanisms.