METHOD FOR WORKFLOW PROCESSING THROUGH COMPUTER NETWORK

TECHNICAL FIELD OF THE INVENTION

The present invention pertains in general to business communication and in particular to the processing of business transactions between multiple parties.

BACKGROUND OF THE INVENTION

During the past years businesses have incorporated computer systems into many integral portions of their internal operations. In most cases, incorporation of such computer systems has substantially increased the productivity of the business. A business builds its computer operations by adding additional software and hardware to its existing system. This individual business development based on internal requirements results in a business environment in which each business has a highly developed and efficient internal computer operation but such internal development greatly impedes business to business interactivity.

As a business grows, it adopts data formats and communication procedures that are generally not the same as those used by other businesses. When the occasion arises for two businesses to interact, there is often a communication barrier due to the differences in procedures and formats implemented by the two parties. Thus, there is a need for processes and systems to reduce the problems encountered by incompatible communication structures between parties.

A further hindrance to business activity between parties is that a typical business action requires a series of transactions between the parties before reaching the desired result. In many instances these transactions are performed by telephone calls, faxes, letters and e-mail. It is very inefficient for parties to construct a business activities schedule each time two parties need to work with each other.

In view of these communication and business procedural problems that hinder commerce, there exists a need for processes and apparatus to enhance communication between incompatible formats as well as to efficiently process business transactions in the sequences which are required. Further, such a system must be able to accommodate growth, new business activities and new industries without modifying the basic operational structure

by the step by step addition of new information defining the required communications and procedures.

SUMMARY OF THE INVENTION

A selected embodiment of the present invention is a method for workflow processing through a computer system between multiple users. The method begins with receiving an input data file from a first user for initiating a transaction wherein the input data file has multiple units of data. One of a plurality of stored business objects is selected for the transaction. Each of the business objects includes a plurality of states, a plurality of work units each related to at least one of the states, and a plurality of rules related to selected ones of the states and the work units. The selected business object has one of the states selected at one time. Next, one of the work units is selected for the transaction. The validity of the selected work unit is determined by reference to a one of the rules associated with the selected work unit and the selected one of the states of the selected business object. A function specified for the selected work unit is performed if the selected work unit is determined to be valid. The next step is selecting a new state for the selected business object. An output file is created based at least in part on the function performed for the selected work unit. The final step is making an output file available to a second user of the system. The second user is the recipient.

Other aspects of the present invention include a complete business activity conducted by a plurality of repetitions of the method described above, the translation of received files into a standardized format, the use of a standardized internal data format and the translation of output files to formats preselected by the recipients. Still further aspects of the present invention include preprocessing and postprocessing operations.

BRIEF DESCRIPTION OF THE DRAWINGS

For a more complete understanding of the present invention and the advantages thereof, reference is now made to the following description taken in conjunction with the accompanying drawings in which:

Figure 1 is a schematic illustration of a data communications network that is connected to a plurality of users and further connected to a system which processes transactions in accordance with the present invention,

Figure 2 is a schematic state diagram of a sample business object,

Figure 3 is a table defining functionality for the business object shown in Figure 2 including a listing of states, work units, rules, preprocessing steps and postprocessing steps associated with the business object,

Figure 4 is a sequential flow diagram illustrating a general example for the processing steps involving the present invention,

Figure 5 is a transaction schematic illustrating the stages of processing of a transaction in conjunction with particular file types and processing operations,

Figure 6 is a specific example of a business object, namely a flood order, which includes a plurality of states and a plurality of defined work units,

Figure 7 is a table of information defining the business object illustrated in Figure 6, including states, work units, rules, preprocessing steps and postprocessing steps,

Figures 8A and 8B comprise a detailed flow diagram representing an implementation of the present invention,

Figures 9A and 9B represent a transactional description for an example of the present invention involving three stages of processing with specific data types, translations, processing and file delivery,

Figure 10 is an example of a proprietary text format for the example of a flood order,

Figure 11 is an example of a RealXML file as utilized in an embodiment of the present invention for a new flood order,

Figure 12 is an example of a proprietary X.12 format file delivered to a vendor for a new flood order,

Figure 13 is an example of a proprietary format file received from a vendor in response to receiving the output file shown in Figure 12,

Figure 14 is a further schematic state diagram example of a business object for insurance claim processing, and

Figure 15 is a table defining functionality for the business object shown in Figure 14 including a listing of states, work units, rules, preprocessing steps and postprocessing steps.

DETAILED DESCRIPTION

The present invention is directed to a computer system and method for processing business transactions between multiple parties. A computer network 20 for implementation of the present invention is described in reference to Figure 1. The present invention works in conjunction with a communications network 22 which may be, for example, the Internet, the public switched telephone system, or any other communications system. The network 20 includes a plurality of network terminals 24, 26, 28 and 30. The principal functions performed by the present invention are carried out by processors in a computer system 36 which is connected to the communications network 22. The system 36 is connected though a firewall server 38 to the communications network 22. The system 36 further includes a web server 40, a database server 42, a REALMonitor server 44, an SNA server 46 ,and an FTP server 48. The servers 38, 40, 42, 44, 46 and 48 are interconnected by means of a network 58, such as a local area network.

The present invention can be implemented in any desired configuration of servers and any distribution of functions included concentrated in one or a few servers or widely distributed over numerous processors.

In one implementation of the present invention, the terminals 24 and 26 can be used by lenders in a real estate environment, and the terminals 28 and 30 can be utilized by service providers to the lenders.

The present invention comprises a method for processing transactions and providing communications between entities engaged in business activities. For example, the lenders at terminals 24 and 26 have need of services that are supplied by the providers at terminals 28 and 30. In such a real estate environment there may be hundreds of entities corresponding to the lenders and there may be hundreds of entities corresponding to the service providers. In most cases each of these entities has its own data formats and information definitions which do not match with the formats and definitions of other parties. A further aspect of the present invention is the processing of transactions to perform the functions needed to make possible

the necessary commerce between parties. The primary application of the present invention is the provision of commerce between business entities although it may likewise be used with retail entities and consumers.

A significant term used in conjunction with the present invention is that of a "business object." A business object is defined to be a multi-function business process in which the functions are related, the process has a plurality of states, but is in only one state at a time, and has restrictions on what functions may be performed at each state. The restrictions are referred to herein as business rules. The business object includes one or more work units which define specific functionality within the business object. In a selected embodiment, the business object is implemented as a program "object" using an object oriented programming language. Such programming languages include C++ and Visual Basic. Object oriented programming in general is described in "Object Oriented Analysis and Design With Applications," Second Edition, by Grady Booch, Copyright 1994, published by B. Benjamin/Cummings Publishing Company. This book is incorporated herein by reference.

The business objects referred to herein can be implemented in non-object oriented programming languages, but still include the same structural elements and operations as described herein. The business object principally describes the context of the business interaction between parties. It is not a model for the internal operation of an individual business.

A generic example of a business object , referred to as "BUSINESS OBJECT #1" is shown in Figures 2 and 3. Referring to Figure 2, business object #1 (BO#1) has 5 states which are 70, 72, 74, 76 and 78 that represent respective states A, B, C, D and E. One of the states of the business object #1 has a functional operation 80 associated with the state 72 (B).

Business object #1 has work units 82, 84, 86, 88, 90, 92 and 94 which are identified respectively as WU#1, WU#2, WU#3, WU#4, WU#5, WU#6, and WU#7 in Figures 2 and 3.

As shown in Figure 3, the defined system functionality is maintained in a data store 98, such as a system memory or disk. The store 98 includes a block 100 which identifies the business object #1 and indicates that it has states A, B, C, D and E. Each state represents a stage in the processing of the business object. The store 98 includes a block 102 for

identifying the work units #1 through #7 (82-94) that are associated with business object #1. Each work unit further includes a definition of its specific functionality.

As shown in Figure 2, WU#1 (82) defines a function that results in a transition between states A and B. WU#2 (84) defines a function that results in a transition between states B and E and WU#3 (86) defines a function that results in a transition between states A and E. WU#4 (88) defines a function that results in a transition between states A and C and WU#5 (90) defines a function that results in a transition between states C and E. WU#6 (92) defines a function that results in a transition between states C and D, while WU#7 (94) defines a function associated only with state B of business object #1.

There is further included in the store 98 for the business object #1 a block 104 which includes a set of rules identified as R#1, R#2, R#3 and R#4 that are associated with the business object #1. The rules define requirements and limitations on what may be done in the corresponding business object. Each rule is defined with respect to a specific environment. As shown, R#1 is associated with state A of business object #1. R#2 is associated with WU#1. R#3 is associated with state B and WU#2 concurrently, and R#4 is associated with state C. However, the work units and rules in blocks 102 and 104 are associated specifically with business object #1, and its defined states.

The store 98 further includes a group of preprocessing and postprocessing steps which are shown respectively in blocks 106 and 108. Block 106 illustrates preprocessing steps #1 and #2. Each processing step has as defined association with entities within the store 98 and is related to one of the business objects defined within the store 98.

In the present invention the parties involved in either sending or receiving communication are referred to as users. In particular, a user that starts a transaction is referred to as an initiator and one that receives the result of that transaction is defined as a recipient. More specifically, such as shown in Figure 1, an example of an initiator may be a lender and an example of a recipient may be a service provider or vendor. However, when a service provider starts an action, that service provider can be the initiator and one of the lenders can be the recipient.

As shown in Figure 3 within store 98, each preprocessing operation is defined for a specific environment of a particular business object and includes a specific functionality. In

store 98, preprocessing step #1 is implemented for WU#1. Likewise, all of the other defined preprocessing steps within the store 98 have a defined environment and a defined functionality.

A preprocessing step is defined for a particular entity in most cases. A postprocessing step is generally defined for a particular entity.

In a similar fashion, there are postprocessing steps within block 108. Each postprocessing step likewise corresponds to a specific business object, has a defined environment and a defined functionality. When the defined environment exists for a specific postprocessing entity, the corresponding functionality for that entity is performed.

The store 98 shown in Figure 3 can include a substantial number of business objects and associated with each of these objects are work units, rules, preprocessing steps and postprocessing steps. As shown in this figure, a business object #2 has states A, B and C. Associated with this business object is a block 112 for defining the work units, a block 114 defining the associated rules, a block 118 for the associated preprocessing steps and a block 120 for the postprocessing steps. Preprocessing and postprocessing steps may not be present for all business objects.

Referring to Figures 1, 2 and 3, a business object, such as #1, is defined for a particular relationship which may occur between any two of the users set up for the system 20. The store 98 is included within the system 36 and more specifically within the server 42. The system 36 stores a plurality of the business objects, such as shown in Figure 3, to enable a very large number of business transactions to be conducted between the users. For each business object there is included the defined work units, rules and any associated preprocessing and postprocessing steps.

When a user, such as the lender at terminal 24, is acting as an initiator and begins a transaction, it sends an input file via the network 22 to the system 36. This file, as further defined below, results in selection of a business object, and causes a sequence of steps to be performed by the system 36 as defined by the selected business object, such as shown in Figure 2. In general, the receipt of a file from an initiator results in the establishment of a given state for the selected business object, for example state A, or after functional operations are performed the transition of the business object from one state to another. In many cases,

this state transition, or creation, results in the routing of a communication to a recipient. In general, the process of the business object continues with further receipts of file communications from users and associated with each is generally a transition between states of the business object. This continues until a termination object is reached which in most cases indicates the completion of a business activity between two parties, but can also indicate the termination of the activity before completion.

The data structure associated with the present invention, such as shown in Figures 2 and 3, provides definition and modularity to enable an efficient and economic implementation of a very wide range of business transactions between large numbers of users. The system 36 can be expanded in increments to add additional business objects as these are required by users of the system. A great advantage of this approach is that entirely new types of business activities and new types of data formats can be incorporated as defined elements within the business objects and added to the store 98 of the system 36 without the need for building a specific system for each type of business transaction for each industry, or for designing systems specifically for the unique data formatting and communication techniques utilized by particular business within an industry.

A basic flow diagram representing the processes, in general, of the present invention is shown in Figure 4. This is described also in reference to Figures 1, 2 and 3. In a first block 132, the system 36 receives a user input data file (IDF) from an initiator, for example the lender having the terminal 24 shown in Figure 1. In block 134, the system 36 reads the input data file and selects a particular business object which has been specified for this file. This corresponds to the business objects which have been previously recorded in the store 98 as shown in Figure 3. The file further includes specific data which is utilized by the system 36 in block 136 to populate the selected business object and thereby create an instance of that object.

As shown in block 138, the system 36 examines the input data file, or the folder in which it is stored, to determine a specified work unit which is to be executed in the current transaction.

A business process as used herein typically includes a plurality of transactions, with each transaction generally corresponding to one of the states of the business object. The flow

diagram shown in Figure 4 represents the processing of one transaction which occurs within a business object.

In block 140 the system 36 determines by examining the store 98 whether any preprocessing steps should be implemented. If so, the functionality corresponding to each of these preprocessing steps is performed. A preprocessing step is primarily for acquiring additional data. Next, at block 150, the system 36 performs the defined functionality for the selected work unit.

After the specific functionality for the selected work unit has been performed, the system 36 at block 152 loads the database with a file that includes all of the relevant portions of the information within the input data file received from the user as well as any additional information and data which has been developed during the preprocessing and performance of the selected work unit.

In block 154, the system performs the defined postprocessing which is associated with the current environment that exists at the time of entering this block. The transaction may require the generation of a certain output, this is performed in block 156 and the output is sent to a specified recipient at block 158.

A postprocessing step is primarily for sending a confirmation or acknowledgement that something has been revised or an action has been performed.

In general, the processing of a transaction as shown in Figure 4 will result in either the identification of an initial state of a business object or a transition within the business object from one state to another. The resulting state may be an interim state which will be followed by further transitions or it may be a termination state which indicates the completion of the functions necessary for the particular business object. Preprocessing may not be performed in all transactions and likewise postprocessing may not be performed in all transactions.

An example of the present invention is now presented within the context of processing real estate transaction documents. The setting involves a large group of lenders who provide loans for real estate properties and service providers who supply specific services that are required within the real estate field. A significant aspect of the present example is that the users, both requestors and recipients, likely utilize different formats in

their businesses for performing real estate functions. At the present time, this incompatibility hampers the processing of real estate activities, but this problem can be handled within the present invention. As time progresses, it is possible that certain industries, such as the real estate industry, will standardize on activity formats for files and reduce this problem. In the present example, a customer of a lender, such as associated with terminal 24 shown in Figure 1, has requested a loan to purchase a specific property. The lender, referred to in the start of this example as the initiator, must perform multiple activities before it can grant the loan for the property. One step in the process, which represents this example, is the determination by the lender as to the status of the property in question concerning flooding. Such a determination is frequently an important step in the loan evaluation process. To answer this question, the lender submits what is termed a "flood order" to determine a flood classification status for the particular property. Within the United States there are many different service providers (vendors) who provide this information. Some vendors operate only in specific regions and national vendors may only work with certain lenders, rather than the universe of all lenders. Since many lenders operate on a national basis, the processing of a single flood order, within the context of thousands of such orders being processed, can be relatively complex.

In a further description of the present invention there is shown a transaction schematic in Figure 5 for the processing of such real estate orders in a broad context, one aspect of which can be the flood order of the present example. The transaction schematic in Figure 5 has three stages. These are stage 168 for input and translation, stage 170 for transaction processing, and stage 172 for translation/output/delivery. In this example, the lender is referred to as the initiator (I). Further referring to Figure 5, the initiator places an order, such as the floor order, in stage 168. The initiator transmits a file from his terminal 24 through the network 22 to the system 36. This can be either a proprietary format file 174 or an XML file 176. The received proprietary file is placed in a one of a set of folders that are reserved for the particular initiator. The initiator places the proprietary file in a particular folder associated with the specific work unit and business object for the required action.

Should the file be in a proprietary format, it is processed through a translator 178 to produce an XML file 180. The file 176 corresponds to the file 180.

In the present example of real estate processing, a standardized file format has been selected which utilizes Extensible Mark-up Language (XML). XML is further described in "Extensible Markup Language (XML) 1.0," W3C Recommendation 10-February-1998 (41 pages), which is incorporated herein by reference. XML describes a class of data objects called XML documents and partially describes the behavior of computer programs which process them. XML is a form of SGML, the Standard Generalized Mark-up Language which is specified by ISO 8879. By definition, XML documents conform to SGML documents. A particular structure for an XML document file has been selected for defining the data required in the real estate processing associated with the defined business objects. Non-XML files are translated into standardized XML files for processing in the system 36.

Further referring to Figure 5, at stage 170, a business object is selected as defined by the data within the file submitted by the initiator for a formatted XML file. The business object is defined by its folder for a proprietary file. The initiator selects the appropriate folder. For each transmitted file (transaction) a work unit is further identified. The work unit is specified in an XML file but is again specified by the selected folder for a proprietary file. This is processed within stage 170 by the business logic processor 182. This is done in conjunction with a system data base 184 which corresponds to the store 98 shown in Figure 3. The data base 184 includes a collection of business objects and associated work units and rules previously described in reference to Figures 2, 3 and 4. Preprocessing step 186 is performed in accordance with the current environment, principally to collect additional information. After the preprocessing step 186 is completed, the current file information is stored in the data base 184. Next, postprocessing 188 is performed as determined by the current environment.

As a result of the processing performed during the transaction processing state 170, an XML file 190 is produced. In the example being described, an order delivery is necessary to a selected recipient. If this recipient utilizes the XML standard format, the file 190 is passed through a transfer 192 and delivered by either an FTP (File Transfer Protocol) method or via E-mail to the recipient (R), which in the present example is a selected service provider.

If the selected recipient does not utilize the standard XML file, the file 190 is provided to a translator 194 to produce a proprietary format file 196 that is also transmitted

by either FTP or E-mail to the selected recipient. For the example being described, the documentation provided to the recipient is a flood order which identifies in a format compatible with the recipient the request for a flood order, an identification of the property, the initiator and all other required information. At this point, it is the responsibility of the recipient to provide a response. This step defines the particular vendor to be an initiator (I) in accordance with the present description.

Continuing with the example of the flood order, reference is made to Figures 6 and 7 for defining business object # 7 which represents the flood order business process. The flood order process (business object #7) has states 220, 222, 224, 226 and 228, which are respectively for this business object the states A, B, C, D, and E. The business object #7 includes work units 234, 236, 238, 240, 242, 244 and 246 which correspond respectively to WU#1, WU#2, WU#3, WU#4, WU#5, WU#6 and WU#7. Further referring to Figure 6, the flood order business object #7 has state 220(A) which represents the state at which a new order has been placed by a requestor. This is performed by work unit 234 (WU#1). When the vendor has performed the operations required to confirm receipt of the order, that is, executing the work unit 236 (WU#2), business object #7 is transferred to state 222 representing "confirmed order." At this state it may be necessary for either of the users (requestor or vendor) to attach a document to the current file. This is accomplished by executing work unit 246 (WU#7). The execution of this work unit does not transition the business object from one state to the other, but modifies the information established for a given state, which in this case is the attachment of a document to the file.

In general, a party causes a work unit to be executed by submitting to the system 36 a file which either identifies the work unit or is placed in a folder which identifies the selected business object and work unit.

It may be such that the vendor cannot process the requested order or desires not to process the requested order. If this should occur, the vendor executes work unit 238 (WU#3) by submitting a file to the system 36 referencing this work unit and the business object is transferred to the termination state 224. This is a state of "rejected order." At this state the requestor has been notified that the order has been rejected and the processing of the business object #7 is terminated.

The vendor may execute work unit 240 (WU#4) and transfer back to the requestor all the information required for the flood order thereby completing the order at termination state 226 of business object #7.

A work unit 242 can be executed only when the business object is in the state 222 to permit either the requestor or vendor to cancel the current order. If either party invokes this work unit, the business object #7 is transitioned to termination state 228(E) thereby terminating the processing of the business object #7. A still further option for cancellation is that the requestor may execute work unit 244(WU#6) to cancel an order established at state 220 thereby also moving the business object to termination state 228.

Referring to Figure 7, the system of the present example has a store 252 which includes a plurality of business objects and specifically the defined business object #7. As shown in store 252 there is a block 254 for identifying the specific business object (BO#7) and the states of that business object.

Figure 7 illustrates the storage 252 with the business object #7 states defined in block 254, the work units defined in block 256, the rules associated with the business object #7 defined in block 258, the preprocessing steps, if any, defined in block 260 and the postprocessing steps, if any, defined in block 262. There may be any number of business objects preceding or following business object #7 in storage 252.

In reference to business object #7 in Figure 7, WU#1 is related only to state A of business object #7. This is the work unit for new order placement. In reference to Figure 5, this comprises transmitting a file, either XML or proprietary, by the requestor to the system 36 for the transaction processing stage 170 and then through the stage 172 for delivery to the recipient (vendor) in the format previously defined for receipt by that particular vendor. In general, the execution of a work unit results in one pass from left to right through the transaction schematic shown in Figure 5 with the result that a new state is established for the currently executing business object, such as business object #7 shown in Figure 6.

WU#2 represents the process of confirming receipt of the order by the vendor in which case the vendor becomes the initiator, referring to Figure 5, and transmits a file through the stages 168, 170 and 172 for delivery to the recipient, which in this transaction is the lender. This results in a transition of the business object #7 from state 220(A) to state 222

(B) for a confirmed order. The file submitted by the vendor references the instance of the specific business object #7 being executed.

WU#3 represents the process of rejecting the order by the vendor. In that case the vendor is the initiator (Figure 5) for sending a formatted file which is processed through the stages 168, 170 and 172 for providing to the recipient (lender) information stating that the specific order had been rejected. This results in the business object #7 transferring from state 220(A) to termination state 224(C).

WU#4 in Figure 7 represents function which results in a transition from state B to state D. The execution of this work unit comprises the transmission of a file from the vendor through the processing described in the transaction schematic in Figure 5 from left to right to provide a completion of the order in an appropriate format to the recipient (lender). This results in the business object #7 being changed to the termination state 226 (D) for a completed order.

WU#5 can be initiated by either the requestor or vendor to cancel an existing, confirmed order. This results in a transition from state 222 (B) to state 228 (E).

WU#6 is the action of canceling an order by the requestor before it has either been rejected by the vendor or confirmed by the vendor. This results in a transition from state 220(A) to termination state 228(E).

WU#7 does not perform the function of transitioning from one state to another in business object #7, but instead represents the function of attaching a particular document to the flood order in progress so that the users can reference this document.

In Figure 7, there is a listing of the rules which represent restrictions on the functions and activities that can be performed with respect to the business object #7. The rules which are relevant to this particular business object are described as follows:

R#1 states that the only valid work unit when an object instance does not exist and needs to be created is WU#1.

R#2 states that the only valid work units for state A are WU#2, WU#3, WU#6

R#3 states that the only valid work units for state B are WU#4 and WU#5

The business object #7 can further include a preprocessing #1 step which occurs in the environment condition of the transaction to perform the function of data manipulation to achieve changes in the current business object context prior to work unit execution to achieve the required business logic. Preprocessing step #1 for the recurrence of work unit #1 is to verify the address for the order. Preprocessing step #2 for work unit #1 is to select a vendor using a predetermined procedure established by the lender.

A postprocessing step #1 in block 262 is performed when a specific environment condition of the "transaction" is reached. This environment is the occurrence of work unit #4. this postprocessing comprises an electronic funds transfer (EFT) operation to pay for the flood order.

A detailed process flow representing an example of the present invention is shown in Figures 8A and 8B, and are further described in respect to the example of a flood order.

Following a start block 276, entry is made to a block 278 to detect the presence of an input file in a folder of an initiator. At block 280 an inquiry is made to determine if the format of the received file is XML, a previously determined configuration. If the answer is Yes, transfer is made to block 282 for examining the received file and determining the particular business object therein that is identified within the file. An XML file specifies the business object and the folder holding a proprietary file corresponds to the selected business object and work unit. This business object corresponds to one of the business objects in the system store, such as 252 in Figure 7. For the present example, the selected business object is the business object #7. Continuing with Figure 8A, should the answer in block 280 be negative, transfer is made to block 284 for translating the received file into the desired XML format. This is done by the translator 178 shown in Figure 5. Following block 284 transition is made to the block 282.

Following block 282, block 286 is executed to construct or populate a particular instance of the business object which has been determined and selected in block 282. Continuing to block 288, the input file is further examined to determine which work unit associated with the selected business object has been selected. This corresponds to one of the defined work units such as those in block 256 in Figure 7. The work unit is specified in an original XML file and corresponds to the file folder for an original proprietary file.

At this point the system 36 has received the input file from the initiator, selected a business object as specified in this file, populated that business object and determined the work unit specified by the initiator. Transfer is then made to the block 290 to determine if the specified work unit is valid for the current state or status of the business object. This refers to the states A, B, C, D, E or to the status of not having entered a state. This status is determined by the business rules associated with the selected business object, such as a rule defined in the block 258 of store 252 in Figure 7. If the determination is made in performing block 290 that the current work unit is not valid for the current state or status, the No exit is taken to a block 298 for generating an error message that is transmitted to the initiator followed by transition to an end block 300.

If the current work unit is determined to be valid in execution of block 290, the Yes exit is taken to block 296 to determine if any preprocessing steps should be performed. The current environment is compared to the defined preprocessing step environments listed in block 260 in Figure 7. If any of the listed preprocessing step environments match the current environment, the Yes exit is taken to block 302 for performing the specified one or more preprocessing steps. If the current environment does not match any of the stored environment conditions, the No exit is taken for transfer to a question block 304. Block 304 corresponds to a further one of the rules in block 258 for defining the validity of the data that exists at this stage of the processing. If a determination is made that there is invalid data at this point, the No exit is taken to a block 306 for generating an appropriate error message and sending it to the initiator, followed by transfer to an end block 308.

The validity of data is determined by a set of rules for specific types of data. For example, a data field for a ZIP code should have exactly five or nine digits and should not have any alphabetic characters. A flood rating which may be A, B or C cannot have any other letter and can have only one letter. For a report having a flood hazard, there must be text in a comment field.

If the data is determined to be valid at block 304, the Yes exit is taken to block 310 for performing the requested functions associated with the specified work unit. This comprises the actions performed by the business logic processor 182 (Figure 5) as defined for the business work unit.

Following block 310, entry is made to question block 312 to determine if there is a change in status for the selected business object. If so, the Yes exit is taken to block 314 to set the new status or state of the business object. This corresponds to designating a new state of the business object, such as shown in Figure 6. If in block 312, there is no requirement for a change, the No exit is taken to question block 316. Within this block a determination is made if any type of delivery, such as a file, is required to a recipient. If such a delivery is required, the Yes exit is taken to block 318 to determine the required format for the delivery, then to block 320 to determine the transport mechanism for the delivery, such as FTP or E-mail. Next, transition is made to block 322 to perform the transaction delivery as now specified as to format and mechanism. After block 322, transfer is made to a question block 324 which is also entered if a No response is made to block 316.

Within question block 324, an inquiry is made to determine if the present environment status matches that of any of the postprocessing steps listed in block 262 in Figure 7. If the answer is Yes, transfer is made to block 326 to perform the specified postprocessing steps. If not, the No exit is taken for transfer to block 328 which is also entered following block 326. Within block 328 an acknowledgment is created for the initiator if such action is required at this stage and any created acknowledgment is forwarded to the initiator. A final transfer is made to the end block 330 to complete this transaction which, in most cases, has resulted in a state transfer for the currently selected business object.

Continuing with the example of the flood order, reference is now made to Figures 5, 6, 7, 8A and 8B. Assuming that a requestor has submitted a new order (Figure 6) and executed work unit 234, the business object #7 will be in state 220 (A). In this state the vendor has been notified that an order has been placed with it and it now has the specific data for that order. At this point in time the initiator can be either the requestor or the vendor as defined in the business object #7 (Figure 6). Assuming for the present example that the vendor wishes to confirm the order, the vendor will be designated as the initiator for the next transaction and it will submit an input file as shown in stage 168 in Figure 5 to a folder corresponding to that particular vendor. System 36 will then execute block 278 in Figure 8A to detect the input file and subsequently determine if this file is in the correct XML format in block 280. If not, a translation step will be performed at block 284 by the translator 178 shown in Figure 5.

At step 282, a determination is made for the business object identified in the input file from the initiator (the vendor in this case). This identifies an existing business object that has previously been selected and populated. Transfer is then made through step 286 to step 288 for determining the work unit specified with in the input file received from the vendor. For the present example this will be WU#2 in block 256 shown in Figure 7. At block 288, the rules for business object #7 will be referenced to determine if this work unit is valid at this state. Rule #2 shows that this work unit is valid at this state. The Yes exit is taken from block 290 to determine if any preprocessing steps must be executed. For the current environment, there are no such preprocessing steps and the No exit is taken to block 304 for determining if all of the existing data within the file being processed is valid. Assuming that the data is valid, the Yes exit is taken from block 304 to block 310 to perform the selected work unit, that is, work unit WU#2 (Figure 7). This is performed by sending the received, and possibly translated, XML file as file 190 from the business logic processor 182.

After the required file format is determined for the recipient (in this case the lender), a transfer is made to either the transfer file block 192 (Figure 5) or a translator 194 for producing either a proprietary file or conveying the existing XML file. The transmission mechanism of either FTP or E-mail is also determined for the particular recipient and the recipient is then sent the appropriate file.

Referring further to Figure 8B, entry is made to question block 312 to determine if a status change is required for the existing business object, which is business object #7 for the present example. For this example, the answer to this is Yes and transfer is made to block 314 in which the state of business object #7 within processor 182 is changed from state 220(A) to state 222(B). This is the state in which the order has been confirmed to the lender.

Further referring to Figure 8B, blocks 316-322 are processed to select a format and mechanism and then effect a file delivery to the recipient (the lender).

At the next step entry is made to block 324 to determine if any postprocessing steps are required. This is done by comparing the present environment to the listed postprocessing steps in block 262 of Figure 7. If there is a match, the Yes exit is taken to block 326 for performing the specified postprocessing steps. In the state for the present example, no postprocessing steps are required. Finally, in block 328, optionally, an acknowledgment may

be sent to the initiator, in this case the vendor, that confirmation of the order has been completed.

A still more detailed description of an example of the present invention described in reference to the flood order example is shown in Figures 9A and 9B in addition to Figures 10, 11 and 12. The described example pertains to the real estate industry and in particular to the processing of a "flood order" by a lender to determine if a specified property has a flood rating. The system shown in Figures 9A and 9B is divided into three stages. These are the input and translation stage 352, order processing stage 354 and the output and delivery stage 356. One method of input to the system is through Web clients 358, 360 and 362 which are part of a Web farm 364 that communicates in a conventional manner to a web server 366 which is a part of the system 36.

Files can be transmitted in any one of many techniques to the system server. These include a TransXML file 368, an X.12 file 370, and a proprietary format file 372. The web server 366 operates in conjunction with the program 382 to produce a specified format XML files which are transferred as a RealXML file 384.

The files 368, 370, 372 are deposited in a server in folders corresponding to the respective initiators and selected work units for proprietary files. These are monitored by a program REALMONITOR 386 which identifies the format of the file and transfers the preformatted XML file 368 directly to the RealXML file 384, the X. 12 file to a translator 388 which in turn provides the file 384 and transfers the proprietary file 372 to a translator 390 which produces the resulting XML file 384.

However, the present invention is not limited to this one industry, but is generally applicable to business processing between multiple entities in any industry.

The RealXML file 384 is a defined format XML file 391 which is input to a program 392 termed "RealXMLDb." This is the object program associated with the defined XML file. The program 392 works in conjunction with a database 394 which stores the system functionality, such as shown in Figure 3, as well as the current XML file.

The program 392 utilizes the data stored in the database 394 to perform optional preprocessing steps 396 and 398. When these steps are completed, the current state of the formatted XML file is stored in the data base 394.

At this point the RealXMLDb program 392 performs the work unit specified in the received file, and if required, performs an order delivery 400 which comprises accessing a REALDelivery object program 402 (Figure 9B).

The RealXMLDb program 392 further performs any defined optional postprocessing in steps 404 and 406.

Further referring to Figure 9B the program 402 (REALDelivery) responds to the requirement of the recipient to generate and transmit a RealXML file 416 or invoke an exporter program 418. The RealXML file 416 is transferred through a TransXML export operation 420 to convey a TransXML file 422 to the designated recipient.

If the designated recipient has specified that it is to receive files in the X.12 format, the exporter 418 transfers the existing RealXML file from program 402 to a translator 423 for producing an X.12 file 424.

Should the designated recipient have specified a proprietary format, the exporter 418 transfers the standardized XML from program 402 file to a proprietary format translator 426 which produces a proprietary format file 428.

The proprietary format file 372 for the flood order example is shown in a detailed example in Figure 10. The formatted XML file 391 is shown in a specific embodiment in Figure 11. The X.12 file 424 is shown in detail in Figure 12.

The processing of the flood order example explained above is described in still further detail in reference to Figures 9A, 9B, 10, 11 and 12. In this example, a lender, such as a bank, sends a flood order to the system by submitting a text file in its own pre-specified layout. This text file is a proprietary format file 372 shown in Figure 10. This file is sent via FTP to the lender's specified input folder on the system FTP server, such as 48 shown in Figure 1.

The REALMONITOR program 386 detects that the lender's file is present in a specific folder in the server and moves the file to a processing folder. The program 386

examines the file and detects that the file is in a proprietary text format and calls for an appropriate translator object, which is translator 390, to perform the function of converting the lender's text file 372 into a RealXML file in the specified format utilized by the system. The produced XML file is file 391 shown in Figure 11. File 391 has a unique identification that represents the specific instance of the selected business object.

This file is then sent to the RealXMLDb program 392 where it is determined that the file belongs to the submitting lender and calls specific programs (objects) that contain the preprocessing steps required for this particular lender's flood orders. One preprocessing step specifies that when this particular lender places an order for flood, the system should automatically select a vendor from a list of approved vendors. This is based on a set of rules that the lender has previously supplied. The completion of the preprocessing results in the specification of a particular vendor to receive the order. After preprocessing is complete, the order file is then loaded into the system database 394.

After the file has been loaded, the RealXMLDb program 392 examines the postprocessing steps specified for this particular lender and this particular type of order. This particular lender has specified that one postprocessing step is the automatic placement of a credit report order upon completion of the flood order.

The principal function to be performed by the submission of the flood order is the submission of the order to the selected vendor. This is a function specified for the work unit within the file 372. The RealXMLDb program calls the order delivery program 400 for performing the further steps required to complete the delivery. The order delivery program 400 checks the data base 394 to determine what format and what transport mechanism the selected vendor requires for new orders. In this example the selected vendor prefers that the X.12 format be used for flood orders and that the order be placed in the output folder in the system FTP server 48. The delivery program 400 calls the translator 422 to convert the file into the desired format (X.12) for the selected vendor. The translator 422 then stores the selected vendor's file 424 in the vendor's output folder at the FTP server 48, thus making it available to the vendor. The vendor could specify that it be sent directly to the vendor.

The proprietary text format file 372 shown in Figure 10 includes all of the information needed by a vendor for issuing a flood status report for a specific property. When this file is

submitted by the lender, it is sent to a folder that corresponds to the particular work unit of a particular business object. In the example, this is the work unit "new order placement" in business object #7 (order). The system 36 identifies the business unit and work unit by the folder in which it is located.

The file 391 in Figure 11 is an example of the standardized XML file used in the system 36 of the present invention. The business object is specified in the field "BO name" as "order." The work unit is specified in the "Identity action" field as "create." The data elements for the business objects are specified after the fields "tag name." The file 424 shown in Figure 12 is in a preprocessing format selected in advance by the vendor. A portion of this file is a unique identification which references the specific instance of the selected business object. This identification is the term "CESARG948321893652."

After the vendor has picked-up or received the output file 424, the vendor can confirm that the order has been received by submitting an input file 430 as shown in Figure 13. This file includes the identification number which was in the received file 424. this is in the first line which is identified as "Order Identification." The system 36 uses this identification number in file 430 to associate the file with the specific instance of the selected business object which in the example is business object #7.

Although the present invention has been described above with reference to the field of real estate transactions, it is equally applicable to communication and processing of business activities between business in other fields and industries. The modular structure of the present invention allows the addition of business objects with the corresponding work units, rules, preprocessing and postprocessing steps as needed to permit the system to process business activities. The addition of another business object to the system 36 is illustrated in Figures 14 and 15. This example is for the processing of automobile insurance claims. The entities involved in such processing include large insurance underwriting companies, insurance issuers, authorized automobile repair shops, police agencies and state motor vehicle agencies. A series of transactions between multiple ones of these parties are required to process an automobile insurance claim. A business object #248 for such processing is shown in Figure 14 and the business object with its associated work units, rules, preprocessing and postprocessing are shown in Figure 15.

The business object #248 in Figure 14 has states 440(A), 442(B), 444(C), 446(D), 448(E) and 450(F). There are further included work units 452(WU#1), 454(WU#2), 456(WU#3), 458(WU#4), 460(WU#5), 462(WU#6), and 464(WU#7).

Referring to Figure 15, a storage 470 includes a block 472 for identifying the business object #248 with its specified states. A block 474 specifies the work units corresponding to the business object #248. A block 476 specifies the rules associated with the business object #248. A block 478 describes the preprocessing steps associated with the business object #248 and a block 480 describes the postprocessing steps associated with the business object #248.

After the business object #248 has been prepared, it is loaded into the system 36 and the system can then process input files from users for performing the functionality of this new business object. This illustrates the modularity of the present invention.

In summary, the present invention provides a structure and procedure for reducing communication and business procedure difficulties between businesses, thereby reducing costs and time expended, which results in increased productivity.

Although several embodiments of the invention have been illustrated in the accompanying drawings and described in the foregoing Detailed Description, it will be understood that the invention is not limited to the embodiments disclosed, but is capable of numerous rearrangements, modifications and substitutions without departing from the scope of the invention.

WHAT IS CLAIMED IS:

1. A method for workflow processing through a computer system between multiple users, comprising the steps of:

receiving an input data file from a first user for initiating a transaction, said input data file having multiple units of data,

selecting one of a plurality of stored business objects for said transaction, each of said business objects including a plurality of states, a plurality of work units each related to at least one of said states, and a plurality of rules related to selected ones of said states and said work units, wherein said selected business object has one of said states selected at one time,

selecting one of said work units for said transaction,

determining the validity of said selected work unit by reference to a one of said rules associated with the selected work unit and the selected one of said states of said selected business object,

performing a function specified for said selected work unit if said selected work unit is determined to be valid,

selecting a new state for said selected business object,

creating an output file based at least in part on said function performed for said selected work state, and

making said output file available to a second user of said system.

2. A method for workflow processing as recited in Claim 1 including the step of performing a preprocessing operation to collect information related to said one business object prior to said step of creating said output file.

3. A method for workflow processing as recited in Claim 1 including the step of performing a postprocessing operation to communicate information to a party after said step of performing a function specified by said selected work unit.

4. A method for workflow processing as recited in Claim 1 including the step of storing data associated with said input file in a database.

5. A method for workflow processing as recited in Claim 1 wherein said business object is an object as used in object oriented programming.

6. A method for workflow processing through a computer system between multiple users, comprising the steps of:

receiving an input data file from a first user for a transaction, said input data file having multiple units of data,

selecting one of a plurality of stored business objects for said transaction based on data in said input data file, each of said business objects including a plurality of states, a plurality of work units each related to at least one of said states, and a plurality of rules related to selected ones of said states and said work units, wherein said selected business object has one of said states selected at one time,

selecting one of said work units for said transaction based on data in said input data file,

determining the validity of said selected work unit by reference to a one of said rules associated with the selected work unit and the selected one of said states of said selected business object,

performing a function specified for said selected work unit, if said selected work unit is determined to be valid,

selecting a new state for said selected business object,

creating an output file based at least in part on said function performed for said selected work state, and

making said output file available to a second user of said system.

7. A method for workflow processing as recited in Claim 6 including the step of performing a preprocessing operation to collect information related to said one business object prior to said step of creating said output file.

8. A method for workflow processing as recited in Claim 6 including the step of performing a postprocessing operation to communicate information to a party after said step of performing a function specified by said selected work unit.

9. A method for workflow processing as recited in Claim 6 including the step of storing data associated with said input file in a database.

10. A method for workflow processing as recited in Claim 6 wherein said business object is an object as used in object oriented programming.

11. A method for workflow processing through a computer system between multiple users, comprising the steps of:

receiving an input data file from a first user for a transaction, said input data file having multiple units of data and stored in one of a plurality of folders associated with said first user,

selecting one of a plurality of stored business objects for said transaction based on said folder in which said input data file was stored, each of said business objects including a plurality of states, a plurality of work units each related to at least one of said states, and a plurality of rules related to selected ones of said states and said work units, wherein said selected business object has one of said states selected at one time,

selecting one of said work units for said transaction based on said folder in which said input data file was stored,

determining the validity of said selected work unit by reference to a one of said rules associated with the selected work unit and the selected one of said states of said selected business object,

performing a function specified for said selected work unit if said selected work unit is determined to be valid,

selecting a new state for said selected business object,

creating an output file based at least in part on said function performed for said selected work state, and

making said output file available to a second user of said system.

12. A method for workflow processing as recited in Claim 11 including the step of performing a preprocessing operation to collect information related to said one business object prior to said step of creating said output file.

13. A method for workflow processing as recited in Claim 11 including the step of performing a postprocessing operation to communicate information to a party after said step of performing a function specified by said selected work unit.

14. A method for workflow processing as recited in Claim 11 including the step of storing data associated with said input file in a database.

15. A method for workflow processing as recited in Claim 11 wherein said business object is an object as used in object oriented programming.

16. A method for workflow processing through a computer system between multiple parties, comprising the steps of :

detecting an input file from a first user, said input file stored in one of a plurality of folders associated with said first user, and said input data file having a plurality of data units,

creating a standardized format file using data from said input data file and information associated with said one of said folders in which said input file was stored, said created standardized format file including the identification of one of a plurality of stored business objects and identification of one of a plurality of stored work units associated with said one business object, each of said stored business objects including:

(a) a plurality of states, wherein said selected business object has one of said states selected at one time,

(b) a plurality of work units each related to at least one of said states, and

(c) a plurality of rules related to selected ones of said states and said work units,

performing a function associated with said one work unit,

changing the state of said one business object,

determining a selected file format for a second user which is to receive an output file for said transaction,

creating said output file in said selected file format using data associated with said business object, and

making said output file available to said second user.

17. A method for workflow processing as recited in Claim 16 including the step of performing a preprocessing operation to collect information related to said one business object prior to said step of creating said output file.

18. A method for workflow processing as recited in Claim 16 including the step of performing a postprocessing operation to communicate information to a party after said step of performing a function specified by said selected work unit.

19. A method for workflow processing as recited in Claim 16 including the step of storing data associated with said input file in a database.

20. A method for workflow processing as recited in Claim 16 wherein said business object is an object as used in object oriented programming.

21. A method for processing transactions between parties by use of a computer system, comprising the steps of:

receiving a first input data file from a first user for initiating a transaction, said input data file having multiple units of data,

selecting a business object from a plurality of stored business objects, each of said stored business objects including:

(a) a plurality of states, wherein said selected business object has one of said states selected at one time,

(b) a plurality of work units each related to at least one of said states, and

(c) a plurality of rules related to selected ones of said states and said work units,

selecting a first one of said work units,

performing a function defined for said first work unit,

establishing a first one of said states as a function of said first work unit,

producing a first output file based on said selected business object,

making said first output file available to a recipient,

receiving a first input file from said recipient,

selecting a second one of said work units in response to said first input file from said recipient,

changing to a second one of said states as a function of said second work unit,

producing a first output file based on said selected business object, and

making said second output file available to said first user based on said selected business object and said second work unit.

22. A method for processing transactions as recited in Claim 21 including the step of performing a preprocessing operation to collect information related to said selected business object prior to said step of producing a first output file.

23. A method for processing transactions as recited in Claim 21 including the step of performing a postprocessing operation to communicate information to a party after said step of performing a function specified by said selected work unit.

24. A method for processing transactions as recited in Claim 21 including the step of storing data associated with said input file in a database.

25. A method for processing transactions as recited in Claim 21 wherein said business object is an object as used in object oriented programming.

26. A computer system for providing workflow processing between a plurality of parties, comprising,

a communications system for sending files to and receiving files from said parties,

a system storage having therein a plurality of business objects, each said business including:

(a) a plurality of states,

(b) a plurality of work units each related to at least one of said states, and

(c) a plurality of rules related to selected ones of said states and said work units,

a processor for receiving an input file via said communication system from a first one of said parties and relating one of said business objects to said input file, for selecting one of said work units, processing a function defined for said one work unit, and establishing one of said states of said one business object, and

said processor coupled to said communication system for making available an output file to a second one of said parties, wherein said output file is a function of said one business object and said input file.

27. A computer system for providing workflow processing as recited in Claim 26 including a preprocessing operation stored in said system storage and associated with a specific state of a specific one of said business objects.

28. A computer system for providing workflow processing as recited in Claim 26 including a postprocessing operation stored in said system storage and associated with a specific state of a specific one of said business objects.

29. A computer system for providing workflow processing as recited in Claim 26 including a plurality of translators for converting input files having predetermined formats into a standardized file format for use by said system.

30. A computer system for providing workflow processing as recited in Claim 26 including a plurality of translators for converting a standardized file format used by said system into predetermined output formats.