Using RosettaNet to build demand management system

Huei-Huang Chen
Department of Computer Science and Engineering
Tatung University
No. 40, Chung-Shan N. Rd, Sec. 3, Taipei, 104, Taiwan, R.O.C.
hhchen@cse.ttu.edu.tw

Hung-Chang Wang
Department of Computer Science and Engineering
Tatung University
No. 40, Chung-Shan N. Rd, Sec. 3, Taipei, 104, Taiwan, R.O.C.
g8706008@cseserv.cse.ttu.edu.tw

Abstract

Computer-to-computer exchange of business information has become an increasingly popular form of electronic commerce. EDI is important because it enables firms to exchange business information faster, more cheaply, and more accurately than is possible using paper-based systems. Take, for instance, logistics, Key areas in the logistics communication channel are likely to include: the order processing system, the demand forecasting, the sales recording system, and the stock reordering system. EDI works like a glue to support the movement of this information through the supply the supply chain. EDI is used in manufacturing, shipping, warehousing, utilities, pharmaceuticals, construction, petroleum, metals, banking, insurance, retailing, government, health care, and textiles, among others. EDI was first used in the transportation industry more than twenty years ago, by ocean, motors, air and rail carriers and the associated shippers, brokers, customs, freight forwarders, and bankers.
But, EDI has not seen widespread acceptance because of some specific limitations such as High cost, Rigid requirements, Partial solutions. Therefore we proposed a architecture that uses Rosettanet as the standard of Data exchange. And UI is unified by using Web-based Application. We hope that the demand management can be developed in the standard environment. EDI will be accepted by the enterprise widely so that the enterprise accelerates product workflow.

I. Introduction

To compete successfully in today's market place, companies need to manage effectively and efficiently the activities of design, manufacturing, distribution, service and recycling of their products and services to their customers. Supply chain management deals with the management of materials, information and financial flows in a network consisting of suppliers, manufacturers, distributors, and customers. The coordination and integration of these flows within and across companies are critical in effective supply chain management.

EDI allows transactions that have required paper-based systems for processing, storage and postage to be replaced and handled electronically-faster and with less room for error. This is why it is sometimes called 'paperless trading'. EDI can thought of as analogous to email, expect that the transactions take place directly between computer systems instead of human beings and, because this, the information needs to be more rigorously structured.

In fact, every enterprise may own a management system. If all system merges becoming a unified system, it is impossible. This is because every enterprise may not hope that the internal data is known by other enterprises.
Internet facilities have relieved the position somewhat. Many companies are now using these open systems to place order with each other, to exchange plans and share data. However, the fundamental problems (of non-standard business process and data definitions) remain. If a screenful of data has to be rekeyed to be process, or cross-reference tables have to be maintained, the electronic revolution will still failed to reach anything like its full potential benefit.

II. Related Works

A. RosettaNet Architecture

The RosettaNet model is intended to enable supply chain business partners to execute interoperable e-business processes by continuously developing, maintaining and distributing partner interface process implementation guidelines.

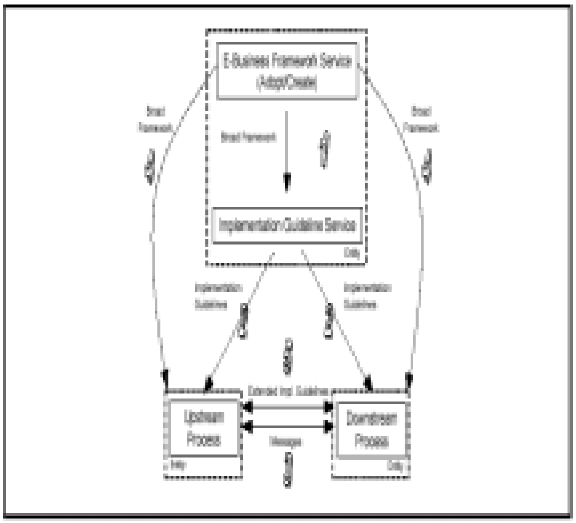


Fig. 1: RosettaNet business model

RosettaNet adopts existing e-business standards, guidelines, or specifications wherever possible and creates new e-business framework specifications where necessary. Typically these frameworks are generic and all-embracing in nature so that they can be used for all types of e-business applications. There are five conceptual parts to the RosettaNet business model:
1. RosettaNet's Partner Interface Process (PIP) teams use these frameworks to create PIP guidelines (labeled "1" in Fig.1) that define how computer systems will cooperatively execute e-business processes in the supply chain. These

guidelines narrow the general information frameworks into detailed specifications that must be embraced by all members who wish to conduct e-business with RosettaNet-compliant partners.
2. The implementation guidelines are provided to companies who wish to conduct e-business according to the RosettaNet's specifications (labeled "2" in Fig. 1).
3. Guidelines are used to validate the information exchanged between companies (labeled "3" in Fig. 1). These guidelines can also be used to create the content that is exchanged and to support tools used to create and manage content in each company's internal system.
4. RosettaNet intends to allow companies to extend the implementation guideline for their own individual needs. Companies can extend the implementation guideline according to the broad framework (labeled "4" in Fig. 1). These extensions cannot override those specified by RosettaNet.
5. The extended implementation guidelines are then exchanged between companies (labeled "5" in Fig. 1). This then allows companies to validate these message extensions during exchange.

A unique aspect of the RosettaNet e-business model is that all guidelines and translations will be distributed as machine-readable documents. This will allow companies to quickly configure their RosettaNet-compliant applications to execute and validate new or updated PIP specifications.

B. XML Architecture

The XML language, XML namespaces, and the Document Object Model are World Wide Web Consortium (W3C) recommendations, the final stage in the W3C development and approval process. Because of these fully stable specifications, developers can start tagging and exchanging their data in the XML format. XML offers a robust solution as the underlying architecture for data in three-tier architectures.

XML can be generated from existing databases using a scalable three-tier model. With XML, structured data is maintained separately from the business rules and the display. Data integration, delivery, manipulation, and display are the steps in the underlying process as summarized in the following Fig. 2.

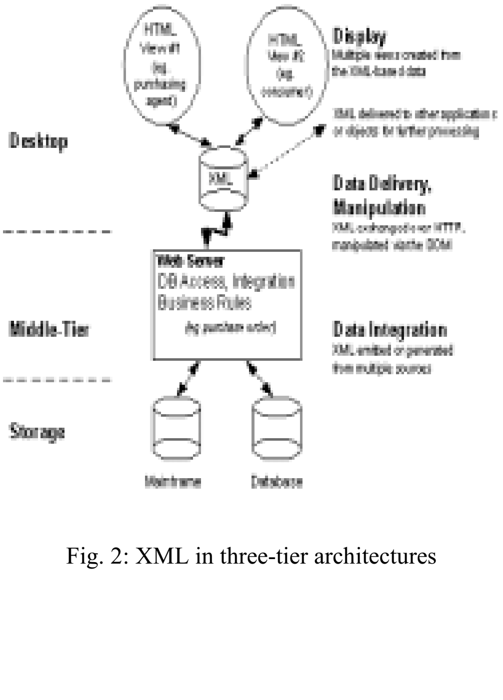


Fig. 2: XML in three-tier architectures

An XML document does not by itself specify whether or how its information should be displayed. The XML data merely contains the facts (such as who ordered which books at which prices). HTML is an ideal display language for presenting this data to an end user. For example, an employee of an online bookstore can visit a Web page to find a list of order entries. From the server, the individual data records are expressed in XML. However, the data is presented to the employee through an HTML page. To construct this Web page, either the Web server or the Web browser will need to convert the XML data records into an HTML presentation, such as a table.

III. The System Architecture of demand management

A. The System Architecture

Our task is based on RosettaNet specification and then utilizes object-orientation to establish a mechanism, which can be used in demand management. In the process of building up, we will comply with RosettaNet architecture. We hope that the new industry-standard can unify data exchange. It will eliminate from incompatible data exchange currently.

According to PIP's definition, whole system must comprise the following elements:

Service Components. Service component classes represent software components that can both initiate service requests to other services and respond to requests from other services. Service components have methods that are invoked during component interaction.

Agent Components. Agent component classes represent software components that can only initiate service requests but cannot respond to service requests.

In order to have a communication's method between Agent Component and Service Component，RosettaNet specified PIP(Partner Interface Process) which is used as standard . The RosettaNet PIP architecture comprises two fundamental parts.

1. A business process model. This model captures business roles and their interactive functional activities, the information that is exchanged when performing these interactive activities, and the sequence in which these interactions take place.

2. A distributed information system design. This design specifies the agent and service software components, together with their information exchange and message protocols, that can either replace or support the roles in the business process model.

The PIP specifies an e-business applications layer that uses protocols in layers used to implement interoperable computer systems on the WWW. eBusiness data interchange applications are categorized according to the categories that Valerie Leyland describes in her book entitled "Electronic Data Interchange as showed Fig. 3. These applications, although adhering to similar computer communications principles, present different issues for consideration during their analysis, specification, design and implementation.

The e-business applications layer comprises service components that respond to requests from other services components. These services, required for e-business in the IT supply-chain, are defined as RosettaNet executes its development strategy.

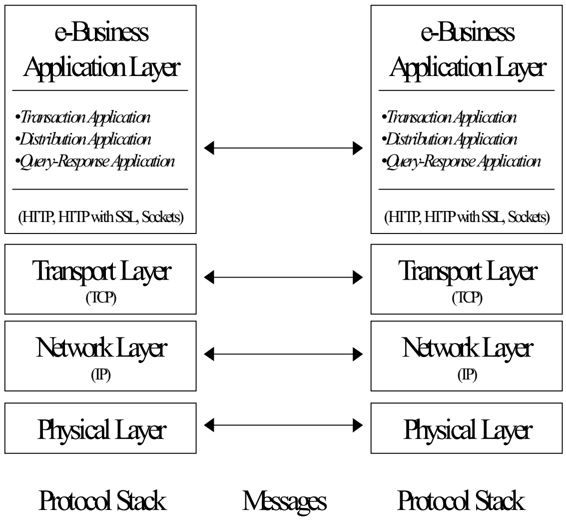


Fig. 3: The e-business applications layer

Each service responds to one or more requests using an agreed language-based method of communication called a protocol. Application specific protocols are built on top of these service protocols. The service protocols are themselves built on top of other protocols as shown in Fig. 4. (Note that I have left out the Internet network and physical protocols for simplicity.)

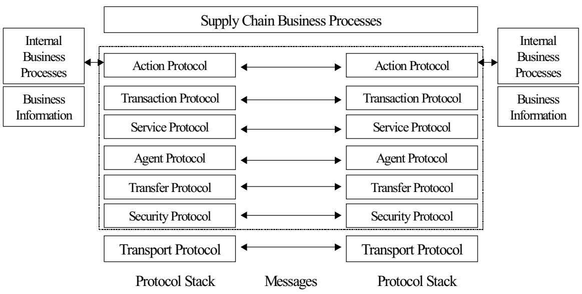


Fig. 4: Service protocols

B. Message Protocol

RosettaNet defined two protocol categories.

- Application protocols.
- PIP protocols.

Application protocols can be reused in any other protocol specification. There is one PIP protocol for each PIP blueprint. A protocol that is used in all other protocol specifications is the control protocol. This protocol captures all the action, transaction and process protocol specifications for acknowledgement controls and for exception handling controls that are specified in the PIP blueprints. All of the application protocol specifications have at least one abstract service component specified in their software component designs. There may be other abstract component, action, transaction, activity and protocol classes. A PIP protocol specification must have concrete classes that can conceptually be implemented. abstract semantics, categorization and aggregation. Based on past studies, we can determine the former three relationships easily without controversy, but we have to apply the domain knowledge from DBA to check the two cases of disjoint generalization, and no relationship. About categorization and aggregation, almost no existing methodologies take them into account. We hope we can get more relationships, as we said above, about integration through process models. We apply attribute equivalence to conform the suspect relationships between two objects derived from process model.

We will list several important protocols as following:

"Manage Purchase Order" specification

The purchase order management process comprises the creation, change and cancellation of a purchase order. A Buyer or buying organization initially creates a purchase order and sends the request to fulfill the order to a Seller. The Seller acknowledges acceptance of the purchase order by returning a substantive purchase order acceptance Business Document. A Buyer can then initiate a purchase order change or cancel the purchase order. The Fig 5 shows overall workflow.

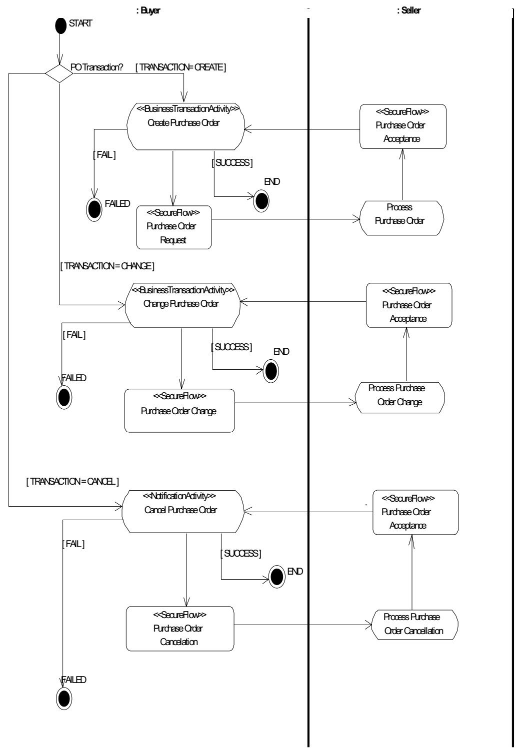


Fig. 5: The Activity diagram of purchase order

The purpose of the PIP is to specify the purchase order management process between trading partners. The management process includes the creation, change and cancellation Business Document. All purchase order acknowledgements of acceptance are "substantive acceptance". A substantive acceptance returns some part of the original business document without modifications.

<u>The start state is comprised of the following conditions.</u>

- Either a requisition exists for which a purchase order must be created or a purchase order exists that must be changed or canceled.
- The transaction property (TRANSACTION) must be set to either Create, Change or Cancel.

<u>End States are comprised of one or more conditions:</u>

END

- Purchase Order exists.
- Purchase Order changed.
- Purchase Order canceled.

FAILED

- Purchase Order does not exists.
- Purchase Order change has not occurred.

- Purchase Order has not been canceled

"Distribute Order Status" specification

The status of a product order is distributed on an unsolicited basis after a purchase order is created. The status of a product order informs a Buyer of the fulfillment and/or shipping status of the products in the order. For example, products in the purchase order request may be backordered, shipped or the entire purchase order may have been canceled. The Fig 6 shows overall workflow.

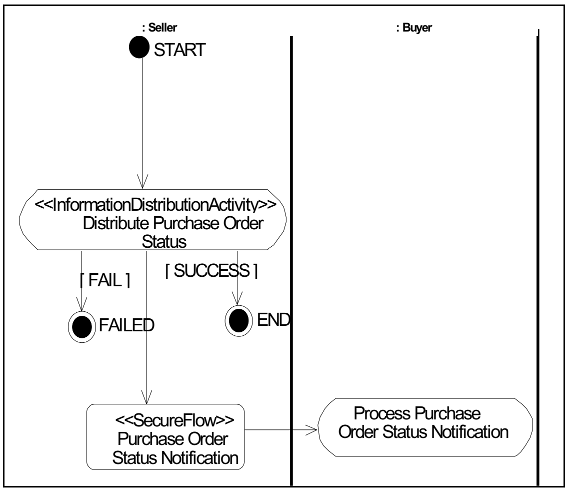


Fig.6: The Activity diagram of "distribute order status"

The start state is comprised of the following conditions:

- A destination address for Purchase Order Status exists.
- At least one open Purchase Order exists.

End States are comprised of one or more conditions:

END

- Purchase Order Status exists.

FAILED

- Purchase Order Status does not exist.

"Query Order Status" specification

The status of a product order is requested after a purchase order is created. The status of a purchase order informs a requesting partner of the fulfillment and shipping status of the products in the order.

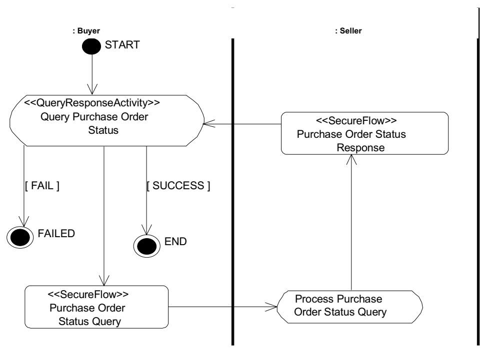


Fig. 7: The Activity diagram of "Query order status"

For example, products in the purchase order request may be backordered, shipped, or the entire purchase order may have been canceled. The Fig 7 shows the whole workflow.

<u>The start state is comprised of the following conditions:</u>

- Purchase Order exists.

<u>End states are comprised of one or more conditions:</u>

END

- Purchase Order Status exists.

FAILED

- Purchase Order Status does not exist.

"Query Product Information" specification

There are numerous points in the supply chain at which product information is necessary. Product information is a category of information that is necessary to sell or buy the product. Product information does not include the technical specification of the product. The Fig 8 shows the whole workflow.

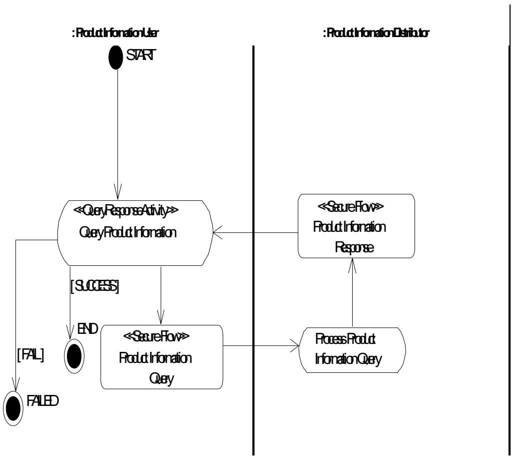


Fig. 8: The Activity diagram of "Query Product Information"

The RosettaNet query/response design pattern allows information to be requested as an XML document and returned as an XML document that is not necessary tabular in nature (it is often a hierarchical structure). The product information query is expressed as a query structured as follows:

Product Information Query
    Product Information Query
        Product Line Item
            Product Description
                Global Product Identifier "001234567890"
                Product Name. Free Form Text

<u>The start state is comprised of the following conditions:</u>

- Query parameters and constraints exist.

<u>End states are comprised of one or more conditions:</u>

END

- Product Information is received.

FAILED

- Product Information is not received.

C. Agent Component mechanism

To realize agent component, we must classify and analyze PIP's message protocols. As a result of the object-oriented analysis on PIP model, we build our object model of the demand management.

E represents Agent collection

E=(P,R,B,M,T)

P: Product Information agent: To respond to the request of Product Information.

R: Order agent: To respond to the request of orders.

B: PurchaseOrder agent: To respond to the request of purchasing order.

M: Message agent: To respond to the request of messages.

T: Task agent: To respond to the request of assigned task such as tracing order, and so on.
The Fig.8 shows overall structure. Basically, every object corresponds to every PIP protocol. We hope that the method will simplify system's complexity and be implemented easily. Next to sections, we will explain those objects how to work and their structure.

D. Service component mechanism

Service component can both initiate service requests to other services and respond to requests from other services and agents. A RosettaNet service is a service defined by the RosettaNet consortium for the IT supply-chain.

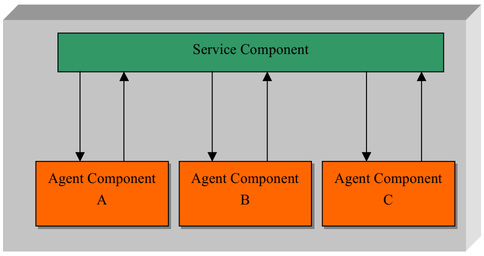


Fig. 9: The relationship of Service and Agent

The RosettaNet Service protocol message is the business message that is exchanged between two RosettaNet entities (Services and/or Agents) as showed Fig 9.

To reach the objective, we decompose all message protocol to become separate service object according to the type of protocols. In fact, this work is not hard. We only make sure that the implementation of all objects complies with PIP's specification. Every service object basically is independent.

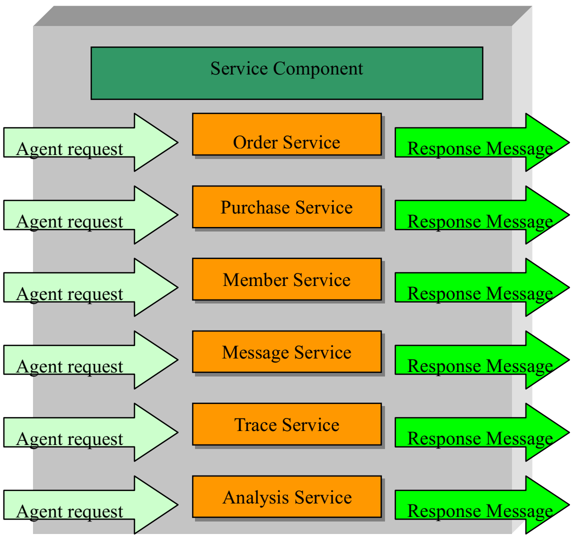


Fig. 10: The work diagram of Service component

In the architecture, every Service component is independent. But a Service component can be called by another Service component or a Agent Component. Logically, we use 'SP' to represent whole Service component.
SP= (SO, SP, SL, SM, ST, SA)
SO: Order Service: To process order service.
SP: Purchase Service: To process purchase-order service.
SL: Member Service: To process works about the management of members such as login, joining new member and so on.
SM: Message Service: To process messages that be sent and received.
ST: Trace Service: To process routines a enterprise assigned
SA: Analysis Service: To process Web-based OLAP Service

Following explains the relationship between Agent and Service Agent. Service Component responds to communicate between Agents as showed Fig. 10.

E. Complete workflow

We will explain how the architecture works through Use Case in the following section. Meanwhile, we will select one of all request and service component to explain how to work together.

Use Case	The management of Purchase order
Related Actor	Member, Message Service, Login Service, Order Service, ProductInformation Service
Description	The member requests “Purchase order service”
Workflow	<The member requests “Purchase management service” >The system displays the menu of Purchase management <The member selects the one of the menu -If the member select the following item: **New Purchase (A-1)** **Update Purchase (A-2)** Cancel Purchase (A-3) Query Purchase (A-4)
Sub-Workflow	A-1 >The system displays the screen of “New Purchase” <User inputs data >User send data to agent >The Agent calls Purchase Service to create a purchase order >Agent responds to the message of the result A-2 >The system displays the screen of “Update Purchase” <User inputs condition >The system displays wanted item <User selects the item of wanted update. >The system displays update screen. <Users updates data in the screen. > User send updated data to agent >The Agent calls Purchase Service to update the purchase order >Agent responds to the message of the result A-3 >The system displays the screen of “Cancel Purchase” <User inputs condition >The system displays wanted item <User selects the item of wanted update. >The system displays complete data of wanted deletion <If user chooses following menu **Yes (B-1)** **No (B-2)** A-4

	>The system displays the screen of “Query Purchase” <User inputs condition >The system displays matched results
	B-1
	>The Agent calls Purchase Service to update the purchase order >Agent responds to the message of the result
	B-2
	Cancel the transaction

Tab. 1: Use case of “Purchase order”

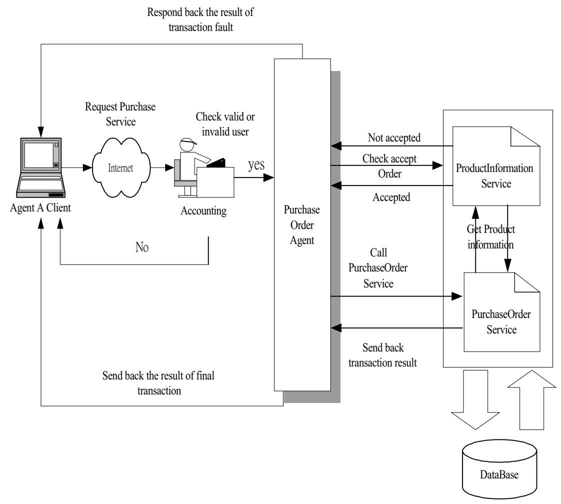


Fig. 11: The workflow diagram of “purchase order”

IV. Implementation

The networked application specified in this document is built on the web protocols and thus exchange information with each other using web servers. There are two methods for transferring RosettaNet Objects between web servers: a) the server-to-server method for directly exchanging information between two web servers, and b) the server-browser-server method for indirectly exchanging information between web servers via a web browser.

Server-to-Server Transfer

An application that transfers this RosettaNet Object to a remote web server via a local web server requests the HTTP protocol to transfer the object as content using the HTTP/1.0 POST request to a target URL. The recipient receives the HTTP request and immediately checks the HTTP headers. If the content-type or transfer encoding is improper, or if the content length fails to match the actual length of the entity body, the recipient returns a 400 (BAD REQUEST) response. If the request is accepted for processing by upper layers in the protocol, a 200 (OK) response will be returned immediately as an acknowledgement of message receipt.

If a sender does not receive a response to the request, then the application must retry the POST method until a response is returned. A receiver application must handle duplicate messages. The method of handling duplicates is not specified.

Server-Browser-Server Transfer

An application can also transfer a RosettaNet Object to a remote web server via a local web server and a web browser. In this case the application requests the HTML User Agent Protocol to first wrap the object as the value for a "VALUE" attribute in a hidden field that is part of an HTML form, before transfer by HTTP as in the server-to-server transfer method. The web browser then forwards the RosettaNet Object by requesting the CGI Agent Protocol to first wrap the object as the value of a name-value pair, before transfer by HTTP as in the server-to-server transfer method.

An application can also transfer a RosettaNet Object to a remote web server via a local web server and a web browser. In this case the application requests the HTML User Agent Protocol to first wrap the object as the value for a "VALUE" attribute in a hidden field that is part of an HTML form, before transfer by HTTP as in the server-to-server transfer method. The web browser then forwards the RosettaNet Object by requesting the CGI Agent Protocol to first wrap the object as the value of a name-value pair, before transfer by HTTP as in the server-to-server transfer method.

a. RosettaNet Agent Protocol. This protocol comprises rules and conventions that govern the exchange of request and response messages between RosettaNet services.

b. CGI Agent Protocol. This protocol comprises rules and conventions that govern the exchange of messages from applications that request services via a common gateway interface to RosettaNet services.

Fig. 12: The work screen of the workshop Fig. 13: The work screen of order management Fig: 14. The work screen of product information

Fig 12

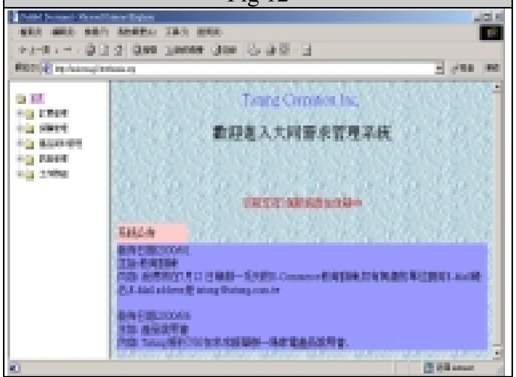


Fig. 13

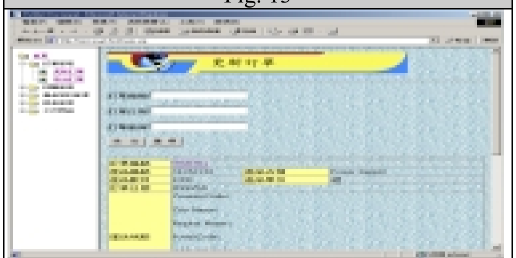
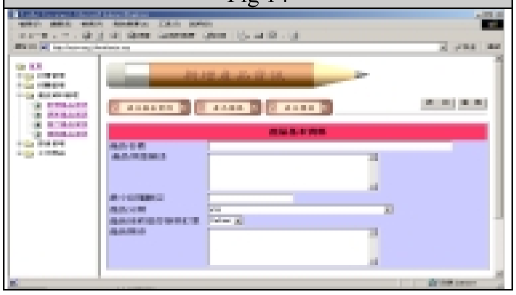


Fig 14



V. Conclusion

The explosive growth of electronic commerce across the IT industry's supply chain has fundamentally changed the way in which products and services are bought and sold. This change holds tremendous potential, promising to enable businesses to strengthen existing customer relationships, identify and capitalize on new revenue opportunities, and create operational efficiencies.

Unfortunately, however, supply-chain misalignments stemming from its lack of global business process standards have created inefficiencies that not only prevent the industry from realizing these benefits, but also critically impede businesses' ability to compete and survive. The IT industry sorely needs to develop, promote, and adopt such standards, thereby reducing supply-chain misalignments, and enabling its members to reap the full benefits of the new digital economy.

Attempts to develop and implement such global standards have failed in the past. However, the chaos resulting from increased internet connectivity, combined with the emergence of channel assembly; IT products' growing dynamism and complexity; deflationary pricing; and mounting industry frustration with ad hoc and/or proprietary business processes and electronic implementations, has made the need for a solution greater now than ever before. Hence, at present, the conditions for standardization efforts are especially favorable. Such efforts will be particularly

The adoption of global business process standards in the IT industry will greatly ameliorate the problems discussed above. Most generally, it will move the industry away from the misalignments of the "smokestack" model, thereby enabling information to flow efficiently across the IT supply chain. This will result in substantial benefits for the specific IT industry players. Manufacturers will gain access to accurate and timely inventory information, enabling them rapidly to assemble products according to precise user specifications. Distributors will disseminate clearer, standardized product descriptions, thereby reducing return rates, and will more fully capture critical data coming back up the supply chain from end-users and resellers. Resellers will reduce their level of investment in back-office operations, allowing them to focus on sales and service, and facilitating the entry of new players into the supply chain. And end-users will make more precise purchase requests and informed buying decisions. In short, the development, promotion, and adoption of global business process standards will maximize the impact of electronic commerce on the IT industry, and enable companies and consumers to begin reaping the myriad benefits of the new digital economy. Business-process standardization thus offers the supply-chain solution that today's IT industry so sorely needs. Truly, it is a project whose time has come.

VI. Reference

[1] D. Taylor, Business Engineering with Object Technology, Wiley, New York, 1995.C. Batini, M. Lenzerini, and S. B. Navathe, “A comparative analysis of methodologies for database schema integration,” ACM Computing Surveys, Vol. 18, No. 4, pp. 323-364, Dec. 1986.

[2] Frank Leymann and Dieter Roller , Production Workflow, Prentice Hall, 2000

[3] Gabriel Sanchez Gutierrez, "The WIDE Project: Final Report,"WIDE Consortium http://dis.sema.es/projects/WIDE/ , May 31, 1999.

[4] Workflow Management Coalition (WfMC), Coalition Overview, URL: http://www.aiai.ed.ac.uk:80/WfMC/overview.html.

[5] G. Alonso, U. Fiedler, C. Hagen, A. Lazcano, H. Schuldt, N. Weiler, "Wise: Business to Business E-Commerce," 9th International Workshop on Research Issues on Data Engineering (RIDE-VE'99), Sydney, Australia, March 23-24, 1999.

[6] Norman W. Paton and Oscar Diaz, "Active Database systems," ACM Computing Surveys, vol. 31, no. 1, pp. 63-103, March. 1999.

[7] Letizia Tanca and Piero Fraternali, "A structured approach for the definition of the semantics of active databases," ACM Transactions on Database Systems, vol. 30, no. 4, pp. 414-471, Decemner. 1995.

[8] Ravi Kalakota and Marcia Robinson , e-Business - Roadmap for Success, Addison-Wesley, 1999

[9] "Workflow Management Coalition Specification : Workflow Management Coalition The workflow Reference Model," WfMC White Paper http://www.aiim.org/wfmc/mainframe.htm, Jan 19, 1995.

[10] P. G. Fakas and B. Karakostas, A workflow management system based on intelligent collaborative objects, Information and Software Technology of the Elsevier Science (1999).

[11] RosettaNet, URL: http://www.rosettanet.org

[12] P. G. Atsushi Inamoto, Agent oriented system approach for workflow automation, Information and Software Technology of the Elsevier Science (1999).

[13] Work Group 1, "Workflow Management Coalition Interface 1: Process Definition Interchange Q&A and Example," WfMC White Paper http://www.aiim.org/wfmc/mainframe.htm, Jan 1, 1999.

[14] Mike Anderson and Rob Allen, "Workflow Interoperability - Enabling E-Commerce," WfMC White Paper http://www.aiim.org/wfmc/mainframe.htm, April 1, 1999.

[15] Work Group 1, "Workflow Management Coalition Interface 1: Process Definition Interchange Process Model," WfMC White Paper http://www.aiim.org/wfmc/mainframe.htm, Nov 12, 1998.