The Development and Evaluation of Exception Handling Mechanisms for Order Fulfillment process Based on BPEL4WS

Fu-ren Lin  
National Tsing-hua University  
frlin@mx.nthu.edu.tw

Hsiang-chin Chang  
National Sun Yat-sen University  
u4370843@cc.ncu.edu.tw

ABSTRACT

The advance of Internet technology fosters the order fulfillment process in a supply chain across heterogeneous information systems. In order to monitor states between partners in the process, this study develops exception handling mechanisms based on Web service stack. After detecting exceptions, the first step is to make use of the fault and compensation handlers provided by the BPEL4WS (Business Process Execution Language for Web Services) specification to roll back planned or even executed business processes belonging to the same scope to the original state. The second step is to utilize the resource searching mechanisms, including BE4WS and BCME, to find suitable substitutes to make a replacement or find fitting partners to cooperate with the original one. The third step is to take advantage of the extensibility, flexibility and scalability of BPEL4WS business process to adjust and reform a new one to handle unexpected situations. This study uses the LCD supply chain as an example to evaluate the proposed exception handling mechanisms, and obtains encouraging performance. The major contribution of this research is the initiative efforts on developing exception handling mechanisms based on Web service technologies to improve the exception handling performance.

Keywords: Supply chain management; Web service, BPEL4WS; exception handling; resource management.

1. Introduction

Business processes increasingly span organizational boundaries along with the progress of e-commerce; therefore, workflow technology needs to be extended to support such cross-organizational processes (Luo, et al. 1998). Another reason for driving the collaboration of companies is that companies focus on their core competence and outsource second-priority activities to other organizations. Moreover, the growing complexity of products requires co-makership relations between organizations. On account of reasons above, workflow management systems have been used to support enterprise-wide business processes, such as workflow creation, enactment, administration and monitoring of these business processes in cross-organizational settings (Chen and Meixell, 2003; Chiu, Li, and Karlapalem, 1999; Luo, et al., 1998).

Constantly changing business environment might induce unexpected situations treated as exceptions, which acquire special attention to change the business processes accordingly (Chiu, Li, and Karlapalem, 1999). The quick responses to exceptions rely on seamless information exchange and coherent control among business partners via compatible information infrastructure. Due to the heterogeneous information systems used by partners in a supply chain, interoperability between legacy systems is essential to efficiently handle exceptions. Web-service technology co-sponsored mainly by IBM, Microsoft, and other organizations enhances dynamic business process interoperability through standard protocols to link loosely coupled and heterogeneous systems. Service-oriented, loosely
coupled, and distributed component-based application architecture realized by the stack of Web-services standards, such as SOAP (Simple Object Access Proposal), WSDL (Web Service Definition Language), UDDI (Universal Description, Discovery and Integration), and BPEL4WS (Business Process Execution Language for Web Services), strengthens e-business processes throughout the value Web (Gottschalk, et al., 2002).

In order to cope with uncertainties in business processes, BPEL4WS specification provides the fault and compensation handlers to detect exceptions and to make up shortage immediately. The business processes are reformed automatically in order to deal with exceptions. This study uses BPEL4WS as the standard to model, deploy, execute, and manage inter-organizational business processes by the specific platform provided by vendors. The objective of this study is to propose a framework composed of reusable Web services by BPEL4WS standard in order to efficiently exchange information across companies, centrally monitor the business process execution and quickly search new substitutes or partners to remedy the shortage of resources and then effectively reorganize a whole new process when exceptions happen. This study models LCS supply chain and evaluates the exception handling performance based on BPEL4WS specification. The results show that the deployment of Web services with BPEL4WS can enhance the dynamics of inter-organizational business processes.

2. Web Service Stack and BPEL4WS

2.1 Web Service Stack

Web services are eXtensible Markup Language (XML) applications mapped to programs, objects, databases, or comprehensive business functions. By using XML to describe structure and content, Web services can provide an interface to data on legacy systems, or between heterogeneous platforms over Intranet, Extranet, or the Internet (Jorgensen, 2002). Web service stack contains four main components: SOAP, WSDL, UDDI and BPEL4WS (Gottschalk, 2002).

SOAP makes it possible for applications written in different languages to run on different platforms to make remote procedure calls (RPC) effectively, even through firewalls. The interface of a Web service and its service interactions are described by WSDL. WSDL describes technical details of a Web service, and users can access the Web services through the specifications specified inside. UDDI initiated by Microsoft, IBM, and Ariba4, does not only consist of defining a data structure standard for all business-related descriptions of services (i.e., service publication), but it also contains all mechanisms that allow the service requestor to gain access to the service publication and service description; thus, it is also in charge of the service discovery layer. BPEL4WS assumes the graph meta-model of WSFL, based on a particular derivation of Petri nets, and the algebraic meta-model of XLANG, which derives from the Pi calculus.

This study includes four crucial parts for executing exception handling on Web services architecture. First, each business process is implemented as a reusable XML-based Web service for multiple-platform information exchange, and these Web services are integrated by the BPEL4WS standard for completing specific jobs. The second part is BE4WS (Business Explore for Web Services) used to find the appropriate type of services and descriptions from several UDDI repositories by one single request, XML-based USML (UDDI Search Markup Language) files. The third part is BCME (Business Criteria Matchmaking Engine) used to store dynamic resources’ information, like suppliers, and it is convenient for clients to search requirements based on business criteria efficiently and effectively. The last part is the platform following BPEL4WS standard to organize, orchestrate and adjust business processes formed by Web services, and monitoring the information and exception flow inside.
2.2 Workflow Formation and Management

Workflows are composed of pieces of activities triggered by cases and these activities are executed in a specific sequence (van der Aalst, 1998). A workflow includes three critical elements: case, resources for manufacturing and process. Workflow management system is also called “logistic control system”, so as implied by the name, it requires protocols to compose and direct processes inside. This problem can be solved in two paradigms: choreography and orchestration (Muslera, 2003).

The orchestration approach treats the composition of Web services like an orchestra, where exists a system playing the role of the director and coordinating the execution of the Web services. It is a more individualistic point of view since Web services do not need to know that they are part of a higher level of business processes. Since only the central coordinator must know this, a unique document is needed for the process to emphasize the steps of the process.

The choreography approach sees the Web services like a group of dancers, and each of them knows exactly when to execute and with whom to interact. This is a more collaborative approach since all the participants need a document describing their roles in the interaction. These documents focus more on the exchanged messages of the process.

2.3 Exception Handling

Workflow combines several business activities triggered by cases (e.g., orders) and uses resources to finish jobs. From this point of view, whatever is missed in these three elements would cause exceptions to occur. The resource exception can be defined as the gap between the expected and the fulfilled resource, e.g., unexpected delivery days and unavailable resources. Take LCM industry as an example, the critical materials and components rely on the import, and it takes a long time to deliver the goods. The resource exception causes panel manufacturers very difficult to meet the market requirements. The long delivery time may also delay the response which in turn cannot provide goods on-time. For a LCM supplier, this unexpected situation is the shortage of the materials happened in the resource dimension of the workflow model.

The case exception includes everything dealing with the incoming information, e.g., orders. If the client cancels the order or modifies the details of the order, it also generates an exception to the workflow. In the LCM industry, the main clients (the industrial downstream and notebook computer suppliers) provide unclear forecasting information or even overbook products, such that the panel manufacturer must endure the cost caused from the stock and the higher risk caused from the uncertainty.

Any problem which causes task malfunction can be one of the situations leading to exceptions, and that may include machines in the factory broken down, suppliers do not provide this task (service) temporally or forever, or the company supplying products goes into bankruptcy. Once the task is broken down, the workflow stops executing the rest of tasks.

There are several ways to handle exceptions in the past. The first exception handling method is the rule-based approach. It determines which runs workflow instances and which regions are affected and need adjustment. These rules enable the system to monitor certain situations and to react to them automatically in a predefined way.

For example, the active object-oriented database system (AOODBS) can meet requirements by dynamically reacting to frequently changing requirements or unpredictable situations through flexible coordination policies and Event/Condition/Action (ECA) rules to realize these policies. ECA rules enable the system to monitor certain situations and to react to them automatically in a pre-defined way (Grahm, et al., 2001).

The second one is the case-based approach which models how reuse of stored experiences contributes to expertise. A new problem is solved by retrieving stored information about previous
problem solving steps, and then suggesting solutions to the new problems. The results are then added to the case repository for future use.

For example, METEOR system adopts competence-driven exception handling method based on the case-based approach. A component that has a handler specified for a given exception is said to have competence to handle the exception, and the component hierarchy includes a competence hierarchy and exceptions are handled according to the competence hierarchy (Luo, et al., 2002).

The knowledge base contains rules representing standard situations and cases representing exceptions or non-standard situations, but both of them are based on the past-experiences; in the real world, we cannot predict what will happen in the next step and exceptions might not happen before. There is also other way for solving the unpredictable situation. Negotiation is one of the solutions for letting group of agents to make mutual agreement regarding their belief, goal, or plan if there are two opposing goals meeting. The negotiation process consists of a number of decision-making episodes, each of which is characterized by evaluating an offer, determining strategies and generating a counteroffer (Wong, Zhang, and Kara-Ali, 2000). Through negotiation, suppliers and clients make a balanced point and a compromise between supply and demand, but it still doesn’t remedy the shortage and solve the problem indeed. This study proposes a different mechanism based on the resource management to deal with emerging problems occurred during execution as discussed in Section 3.

3. Exception Handling mechanism in the BPEL4WS Business Processes

This study uses the LCM industry as the example to elaborate the exception handling mechanism in BPEL4WS business processes. The upstream suppliers in the LCM supply chain are like satellites orbiting around a LCM supplier, which has the authority to master the workflow. A LCM directs and controls the whole situation to avoid chaotic states of the business process after facing the exception. The centralized control of flows inside the supply chain suits the basic concept of BPEL4WS.

In the design, BPEL4WS Engine is located in the LCM Supplier and it acts like a coordinator to orchestrate information and exception flows. BPEL4WS Engine can parse the BPEL4WS file and follow the specifications, like invoke, receive, reply, etc., to monitor, execute operations, detect the unexpected situations and find alternative path to continue the process.

3.1 Supply Chain Structure of the LCM Industry

This research uses the LCM process of the LCD industry as the example and simplifies the whole processes into four main business processes including the LCM, the Backlight, Cell and Driver IC processes. Its manufacturing process includes many high technology activities which technical requirements are also strict. The LCM process (module assembly process) connects additional components, such as driver integrated circuits and backlight units, to the fabricated glass panel and is responsible for dealing with the downstream customers, such as monitor, NB, LCD-TV, and cell phone manufactures. In this supply chain, each manufacturer produces the same product category, and one finishing a LCM product needs three different types of components. A LCM supplier like an entry to downstream clients acts as a coordinator to handle the regular processes and exceptions.

Once a customer sends an order to the LCM, the LCM has to evaluate the orders to check its validity. If the order is valid, it will be divided and sent to upstream suppliers providing components individually. Before signing the contract, the negotiation process is essential to find the better solution to make an agreement. The contract is based on the bargains, and suppliers and clients need to obey these rules to do the business. Based on this reason, we assume that all sub-orders sent to upstream suppliers have already gone through the negotiation process and also planed the schedule well before
manufacturing. Exceptions happen during manufacturing, that is, one of them breaks the deal after signing the contract.

### 3.2 Exception Handling Mechanism Embedded in the BPEL4WS Business Processes

According to Chiu and his colleagues (2000), exception-handling procedures are classified into three categories: remedy, forward recovery, and backward recovery. This study combines these three procedures to implement the exception handling mechanism. The executed business processes will be rolled back to the original state when detecting exceptions; this part is mapping to “backward recovery” and relies on the fault and compensation handlers of BPEL4WS specification. The second one is to search candidates to replace the original one. This part is maps to the “remedy” and relies on the resource searching mechanism. The third one is to modify the business process to alternate the execution path. This part maps to “forward recovery” and relies on the extensibility and flexibility of BPEL4WS business process.

A business process is basically based on the Web service stack. Each task is implemented as a Web service treated as an independent and plug-and-play component, and information is transferred by SOAP in order to interact with multiple platforms. Entries that let users access services are specified in WSDL and registered in UDDI registry. All activities are linked in one specific sequence by BPEL4WS, and the composed BPEL4WS business process also can be viewed as a reusable Web service component. Figure 1 illustrates the three-phase exception handling process: fault and compensation handling, resource searching, and process reforming.

In BPEL4WS specification, scopes provide the behavior, such as fault handler, event handler, a compensation handler, data variables, and correlation sets to solve that problem. In other words, scopes are sets of activities; an exception handler is executed if an exception of the declared type is generated within the scope. It adopts dead path elimination (DPE) to allow a process engine to decide whether a process instance has terminated by checking its activities’ states.

In a BPEL4WS business process, data flows and control flows are orchestrated by BPEL4WS engine, and driven by cases, such as orders (Hung and Chiu, 2004). Once an exception occurs, BPEL4WS engine will generate fault message to trigger the exception flows. All messages transferred inside flows are based on SOAP. Using the fault and compensation handlers, the BPEL4WS business process can easily detect exceptions. If suppliers cannot afford the workload or offer the goods fitting the clients’ expectation, making a replacement or finding partners is a good way for solving these exceptions. On account of the reason above, we need a mechanism for the effective resource management and efficient resource searching.

In this scenario, this study proposes two crucial parts for searching resource. The first one is BE4WS (Business Explore for Web Services) providing the technical information about where and how we can access the resource. The second one is BCME (Business Criteria Matchmaking Engine) providing the information about the suppliers’ detail business information and convenient for clients to search who is the suitable one from the business respective.

BE4WS is extended from the concept of UDDI, but it provides standard interfaces to perform complex searches using a single query request retrieving businesses, services, and service types (tModels) from one or more UDDI registries. The query is based on the proposed UDDI Search Markup Language (USML) for carrying a search request including multiple queries, keywords, UDDI sources, and aggregation operations.

After finding suppliers providing similar functions, and having similar characteristics, we need to filter them again to find the suppliers who can afford the workload. BCME can receive queries and advertisements and also provide management and auxiliary functions. We can rely on BCME to filter
these suppliers by the business criteria and gain the lowest cost or the shortest fulfillment time. This study only focuses on the time issue in order to react to exceptions immediately, and sets the threshold to limit the maximal time.

There are two strategies to reform BPEL4WS processes. The first strategy is the replacement mode, where if suppliers cannot afford the workload for a specific period of time, to find the substitute matching the business criteria is necessary to replace former one completely and produce total quantities. The second one is the cooperation mode, where if the suppliers malfunction but they could afford part of the job, they will find at least one partner to share the workload. It needs to extend the business process to let more suppliers cooperate with each other in order to fill up the vacancy for goods, and it is like combining more than one BPEL4WS processes together.

4. Experimental Design and Experiments

In order to evaluate effectiveness of the proposed exception handling mechanism, this study designs an experimental environment similar to the real world supply chain environment. We use BPWS4J Engine, BPWS4J Editor, and UML Design tool to implement supply chain system embedded the exception handling mechanism.
4.1 Experimental Settings

We build and execute the business process using BPEL4WS specification and BPWS4J engine for an experimental LCM Supply chain by referencing Taiwan’s LCD industry. Because it is a highly trusted, inter-dependent partner relationship, the main issue of the order fulfillment process is to prove the effectiveness of the exception handling process. In the system design, a LCM company receives three components, DriverIC Provider, BackLight Provider, and Cell Provider, from three suppliers respectively to assemble a LCD. In order to create variations to simulate real world supply chain partners, each partner may accept orders from another supply chains which reduce its capability. Each supplier has different capabilities for producing products, and the time that a supplier has been occupied by other supply chain partners is \( \frac{\text{Quantity GeneratedFrom}}{\text{ManufacturingDays For50products}} \times 50 \). The experiment also randomly generates the malfunction time in 5 to 35 days to simulate capacity unavailability. Table 1 shows the experimental settings.

A product order is generated by the normal distribution with the order arrival date, due date, order inter-arrival time, product quantity, and expected receiving days. The order inter-arrival time controls the frequency of order arrival and is generated by the exponential distribution, \( f(x) = \frac{1}{15} e^{-\frac{x}{15}} , x \geq 0 \). Product quantity is generated by the normal distribution, \( N(\mu=1000, \sigma^2=50^2) \). The expected manufacturing days for 50 products is generated by the normal distribution, \( N(\mu=5, \sigma^2=2^2) \). Thus, a client’s expected receiving days is \( \left( \frac{\text{Quantity}}{50} \right) \times \text{ManufacturingDays} \).

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main issue</td>
<td>Time</td>
</tr>
<tr>
<td>System time unit</td>
<td>Day</td>
</tr>
<tr>
<td>Receiving order from other supply chains</td>
<td>10% suppliers would be occupied</td>
</tr>
<tr>
<td>Number of main companies</td>
<td>4 (LCM Provider, DriverIC Provider, BackLight Provider, Cell Provider)</td>
</tr>
<tr>
<td>Number of UDDIs</td>
<td>2 (Microsoft, IBM)</td>
</tr>
<tr>
<td>Number of matchmaking engine</td>
<td>1 (BCME)</td>
</tr>
<tr>
<td>Number of candidate companies for each product</td>
<td>30</td>
</tr>
</tbody>
</table>

A product order is generated by the normal distribution with the order arrival date, due date, order inter-arrival time, product quantity, and expected receiving days. The order inter-arrival time controls the frequency of order arrival and is generated by the exponential distribution, \( f(x) = \frac{1}{15} e^{-\frac{x}{15}} , x \geq 0 \). Product quantity is generated by the normal distribution, \( N(\mu=1000, \sigma^2=50^2) \). The expected manufacturing days for 50 products is generated by the normal distribution, \( N(\mu=5, \sigma^2=2^2) \). Thus, a client’s expected receiving days is \( \left( \frac{\text{Quantity}}{50} \right) \times \text{ManufacturingDays} \).
4.2 Experiments

This study conducts experiments by changing different percentages of unexpected orders, numbers of candidates, and adopts two strategies with different informing times (before or after the due date) to test the exception handling performance. The first one is “replacement mode” for finding a substitute and replacing the original one completely, and the other one is “cooperation mode” for finding partners to work together. The experimental results are evaluated in four criteria: order fulfillment rate, overdue-date processing time, goods in stock, and work-in-process inventory.

5. Experimental Results and Discussion

5.1 The Effects of Exception Handling Mechanisms

Experiment A shows that WIP Inventory could be influenced by the extra processing time by the Pearson correlation. The level of WIP Inventory by the strategies that inform exception between before and after the due date is significantly different by t-test ($p<0.05$). The extra processing time is affected by the timing of generating the exceptional messages. It is an important issue for the supply chain to monitor and detect the probable and unexpected situations in the distributed and heterogeneous environment during execution. BPEL4WS provides such mechanisms as fault and compensation handlers for the long-duration business processes composed of many transactions in reducing the time for reacting to exceptions and undo these failed partial works. From other perspectives, to advance the time of generating the exception messages can lose the constraint for filtering candidates and search more suitable partners. On account of this reason, it can discover better choices to make up the shortage and handle exceptions well.

5.2 The Effects of Resource Search

The success of the order for requesting LCM is influenced by the availability of three categories of materials, Cell, Back Light, and Driver IC. By Pearson correlation analysis, the correlation between the total failure times and number of suppliers as partners or substitutes is significant; it shows that the more exceptions happen, the more substitutes we need. As the percentage of unexpected situation is high, the substitution of suppliers is getting popular.

The extra processing time is significantly different between 30 and 60 candidates, and the slope of the rate is increasing gradually. By adopting 60 candidates, we also can get 100% rate of exceptions successfully handled, lower price for each piece. We can make a conclusion that the resource searching mechanism is essential for the exception handling, and it can be used to solve problems by getting resources (other Web services) from the resource pool (UDDIs and BCME). Especially, the mechanism we provide can break down the barriers between companies, systems or platforms, so that it is richer in resources and provides various choices. The benefit mentioned above also complements with the extensibility, flexibility and scalability of the BPEL4WS business process and the reusability of Web services because it can be modified when meeting difficulties.

Even though it is beneficial to find suppliers to make up the failed part, it also has to pay the extra cost and days for trust building, negotiating, and communicating. It is kind of trade-off between the effectiveness for exception handling and minimization of cost/time.

5.3 Discussion of Cooperation and Replacement Modes

For the replacement mode, once the suppliers evaluates the whole capacities and conditions and figure out that they cannot provide satisfiable job, they would throw the exceptional message to inform the coordinator that they cannot afford the workload. The coordinator, a LCM supplier, will assign the
whole job to another substitute. For the cooperation mode, the original suppliers evaluate how many quantities they could afford and only inform the LCM supplier that they cannot afford part of the job.

In cooperation mode, once the exception happens, the quantity will be divided into at least two parts and assigned each part to different partners discovered from the UDDIs and BCME. The advantage of lowering down the threshold for filtering suppliers gains more choices. However, in the real world, the mass production can reduce the cost and bring more profits; many suppliers set the constraint that clients have to request quantities higher than one specific standard. In some situations, the original supplier can produce most part of products, so the quantities for the partners are very small; it is quiet difficult to pass the threshold candidates set beforehand. It is shown in Figure 2.

Another reason is that once the percentage of exceptions is getting higher, the choices for searching these suppliers who do not set the minimal constraint of quantity are also decreasing; therefore it is getting more difficult to find suitable suppliers to makeup these shortages and the performance becomes worse than the replacement mode in the higher percentage of exceptional orders.

There is a significant difference by *t*-test in the over-due-date processing time for these two strategies, replacement and cooperation modes, in lower percentage of exceptional orders (5%) and more candidates for discovery (60 members). The main reason affecting the performance is insufficient selectivity to match clients’ requirements and suppliers’ constraints, and using 60 candidates, representing richer resources, can solve the problem. In this research, the proposed mechanism provides an interface to access most of resources existing in this open environment, and it is easier to solve exceptions in the real world and to react to exceptions immediately. But cooperating with several members will increase cost and time for trust building, negotiating and waiting for producing, which is a kind of trade-off for exceptions handling.

![Figure 2. The rate of exceptions successfully handled for 30 candidates and different percentages of exceptional orders](image)

6. Conclusions and Future Work

By facing the uncertainties which may lead to unexpected results, we have spent efforts on developing exception handling mechanisms to handle these situations. First, we make use of BPEL4WS specification to detect the unexpected situation; that is, to catch exceptional messages thrown from upstream suppliers. Second, after detecting these unexpected situations, we make use of the dynamic resource management for discovering suitable substitutes or partners to replace the original supplier who cannot fulfill customers’ needs. Because each Web service, even the BPEL4WS business process, can be treated as an independent and plug-and-ply component, it is reusable. By adopting this approach, we do not only shorten the time for development, but also change partners dynamically. Third,
we also utilize the extensibility and flexibility of BPEL4WS business process to generate alternative path to replace the problematic one.

The major contribution of this research is the initiative efforts on developing exception handling mechanisms for solving unexpected situations based on BPEL4WS. In the meanwhile, it can also reduce the inventory for materials, and in turn, reduce the cost for storage. In order to speed the response time for handling exceptions, we only take “time issue” into consideration, which means that the cost for materials or for other things might not be lower enough. The “cost issue” can be taken into account in future study. In this research, the coordinator is passive to receive the exceptional messages thrown from the upstream suppliers. In the future, an automatic tracking system can be used for detecting unexpected situations. Therefore, the coordinator and a LCM supplier can react to exceptions earlier, and find alternative way to execute the processes.

References


