Designing an Efficient Web Service Transaction Protocol Using 2PC and THP

Seung-Kyun Han *, Jonghun Park **, Ki-Seok Choi***

Abstract

Web services provide an effective means to carry out coordinated transactions between multiple, independent business parties. While there are several specific protocols currently being discussed to address the coordination of web services-enabled business transactions, we consider the tentative hold protocol (THP) that allows the placement of tentative holds on business resources prior to actual transactions in order to provide increased flexibility in coordination. In this paper, we present a formal coordination framework for applying THP in conjunction with 2PC to the problem in which service providers independently manage resources and clients seek to acquire the resources from multiple providers as a single atomic transaction. The proposed framework facilitates the performance optimization of THP through effective parameterization with the notion of overhold size and hold duration. The simulation results show that the proposed adaptive approach yields a significant improvement over other non-adaptive policies.

Keywords: *Web services, Business transactions, Tentative hold protocol, Two Phase Commit Protocol, Distributed Coordination*

* Research Assistant, Engineering Research Institute, Seoul National University (Email: jackleg83@gmail.com)

** Assistant Professor, Dept. of Industrial Engineering, Seoul National University (Email: jonghun@ieee.org)

*** Assistant Professor, Dept. of Industrial & Information Systems Engineering, Hankuk University of Foreign Studies (Email: kchoi96@hotmail.com)

1. Introduction

As web services[4] are increasingly becoming the predominant means for business-to-business collaboration, there are needs for automating the coordination of multi-business interactions. THP attempts to provide a building block that can work with other technologies to facilitate the coordination of complex multi-business interactions as well as the creation of new opportunities to leverage the web services to improve business efficiencies. THP is an open, loosely coupled, messaging-based framework for the exchange of tentative commitments between businesses prior to the actual transaction[6]. In particular, it provides a standard means for trading partners to place tentative holds for business resources.

When combined with existing approaches for managing the coordination of multi-business transactions, THP can provide several significant benefits. For instance, we consider a client business application that attempts to coordinate the purchase of an item X from company A and an item Y from company B as a bundle. Without THP, the application would place an order for item X from company A, and then try to purchase item Y. If it is unable to purchase item Y, then it would need to compensate for its earlier action by canceling the order with company A[5]. However, with tentative holds, the client can first place tentative holds on both the items to ensure their availability, and then complete both purchases. Therefore, through the use of THP before the actual transaction, one can expect the reduced number of compensating transactions.

In this paper, we consider a situation in which businesses independently manage and expose their resources through web services and client applications seek to acquire resources from multiple providers as a single atomic web service transaction. In such circumstances, the clients are competing for finitely available resources, and it is not guaranteed that the required resources are always available. Hence, the client should be able to achieve all-or-nothing semantics for the entire end-to-end transaction. This problem is getting increasing attention in many real world application contexts[2, 3].

2. Proposed Transaction Protocol

The distributed system model considered in this paper consists of a set of service provider processes that expose their resources as web services, and a set of client processes each of which seeks to acquire a set of resources from the providers. Service provider manages resources of the same type. It is assumed that client needs to acquire the resources as a single atomic transaction.

We use 2PC protocol to achieve the atomicity. Following the XA standard [1], each service provider implements interfaces for a *prepare* method, an *abort* method, and a *commit* method. A simple application of 2PC protocol to coordinate long-running collaborative business applications may end up with undesirable long lock duration, resulting in lower resource utilization and less number of successful transaction completions. To overcome this problem, we add a tentative hold phase to 2PC before they initiate a 2PC transaction. This augmented protocol will be referred to as

THP+2PC.

We define a *hold* method for clients to request a tentative hold on some resource, and a *cancel* method to request removal of a current active hold. A hold request will be either approved or rejected. If approved, the service provider will reply with yes, otherwise no. On the contrary, when a hold placed on a resource needs to be invalidated, the service provider uses an *expire* method to notify such event to the corresponding clients.

Having defined the necessary control interfaces for implementing THP in conjunction with 2PC, we proceed to describe the proposed parameterization of THP that can facilitate optimization of the protocol's performance. The first parameter proposed in this paper, named overhold size, is defined as the number of holds that can be placed on one unit of available resource.

Another parameter that governs the behavior of THP is hold duration. It represents the time period during which a granted hold is valid, and it is determined independently by individual service provider whenever a new hold is approved. When the lifetime of an active hold reaches the preset time period, the service provider will immediately invalidate the hold, and notify an expiration to the corresponding client. Therefore, it is not allowed for clients to maintain a hold indefinitely during their resource acquisition process, and consequently, with the finite hold duration, THP+2PC can effectively eliminate deadlock and avoid unnecessary long blocking.

Our objective is to find a hold duration that can minimize the average waiting time incurred to clients before they successfully complete their atomic multi-transactions.

3. Experimental Results

Simulation analysis results are given in this section to demonstrate the performance of the proposed THP+2PC framework. We consider an experimentation scenario in which e-tailers sell the commodity products through the web services. Clients purchase the products from multiple e-tailers, and seek to purchase the items as a single atomic transaction. E-tailers, on the other hand, are assumed to start their business with some initial stock, and replenish the products whenever sale is made. We also consider a time delay required for the product replenishment. Since each e-tailer wants to maximize its inventory turnover rate, we consider the objective to minimize the time required for clients to complete their entire transactions. Both clients and service providers follow THP+2PC protocol defined in this paper.

In each simulation, we compare the proposed method for computing hold duration with the fixed-term policies that assign the same length of hold duration to all the holds granted.

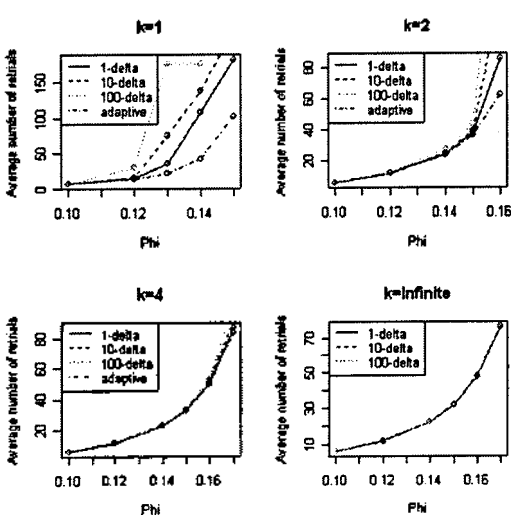
We carried out experiments on 3 cases. In each case, we use minute as a basic time unit. And clients are dynamically created during simulation run, with the exponential distribution with mean 10 minutes as inter-arrival time. Upon creation, each client is assigned with a random value that represents the number of products it needs to purchases as a bundle. For this purpose, we use the binomial distribution defined by 20 identical trials with some pre-specified success probability

π. For each case, the number of independent providers, initial stock size, and the replenishment time for each provider are pre-specified. And the message delivery delay is ignored.

We first consider the cases of finite overhold size, and experimentally compare the performance of the proposed hold duration computation method (to be called as adaptive policy) with the fixed-term policies by examining the effect of varying π. The fixed-term policies considered in the experimentation are denoted by fixed-1, fixed-10, and fixed-100 which assign the hold duration of lengths, 1, 10, and 100 time units, respectively. We use the average number of retrials made by clients as a performance measure. The values of parameters in each case are shown in table 1 where k represents the overhold size.

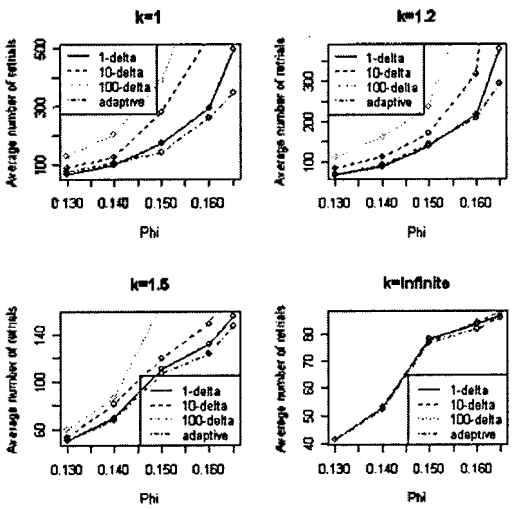
<Table 1> Values of parameters

	# of service providers	Initial stock size
Case 1	20	5
Case 2	5	7
Case 3	20	20

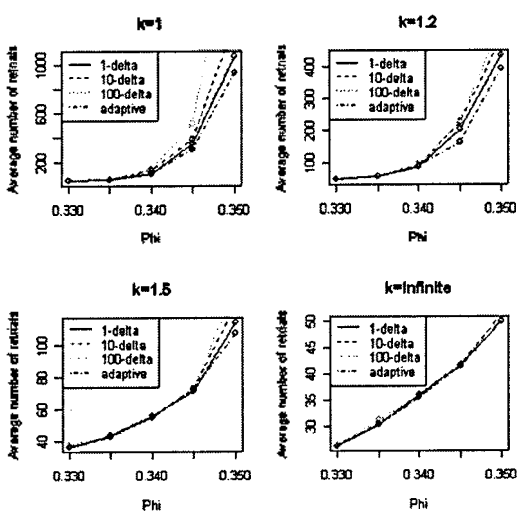


<Figure 1> result of case 1

Figure 1, figure 2, and figure 3 show the results of the experiments.



<Figure 2> result of case 2



<Figure 3> result of case 3

Four plots in each figure show the average performance comparison results of each overhold size(k). In particular, the comparison of the plots when the overhold size is finite indicates that

performance differences become significant as the probability π becomes larger. Hence, it can be observed that the proposed adaptive policy achieves a significant improvement over the fixed-term policies as the overall resource contention is increased. It is also interesting to note that the performance of the fixed-term policies blows up more quickly than the proposed adaptive policy as π increases. Furthermore, from the plots for finite overhold cases, it can be observed that a better performance can be achieved by increasing the size of overhold. This performance improvement is the result of reduced blocking and livelock during the hold acquisition process when the service providers increase the overhold size. Nevertheless, the higher value of overhold size will incur more frequent unexpected hold expirations to clients, resulting in less satisfactory quality of services. Therefore, the actual determination of the overhold size will not be solely based on the performance concern.

4. Conclusion

The recent advent of web services has opened up the possibility of automating multi-business transactions that span across the organizational boundaries. While there are several specific protocols currently being discussed to address the coordination of web services-enabled business transactions, this paper considered the tentative hold protocol that allows the placement of tentative holds on business resources prior to actual transactions in an attempt to provide increased flexibility in coordination.

We proposed a framework for applying the THP in conjunction with 2PC (named THP+2PC) to the problem in which service providers independently manage and expose their resources through web services and clients seek to acquire the resources from multiple providers as a single atomic transaction. In particular, by refining THP through the notion of overhold size and hold duration, our approach provides an effective framework to facilitate optimizing the performance of THP+2PC when it is used for automated coordination of multi-business transactions.

Finally, simulation results confirmed that the proposed approach performs significantly better than the other possible non-adaptive schemes considered.

Acknowledgement

This work was supported by the Korea Research Foundation Grant. (KRF-2004-003-D00482)

References

[1] P. A. Bernstein and E. Newcomer. Principles of Transaction Processing. Morgan Kaufmann, 1997.

[2] F. Casati and M-C. Shan. Dynamic and adaptive composition of e-services. Information Systems, 26:143–163, 2001.

[3] M. Hansen, S. Madnick, and M. Siegel. Process aggregation using web services. In Proceedings of the Workshop on Web Services, e-Business, and the Semantic Web (WES), 2002.

[4] E. Newcomer. Understanding Web Services. Addison-Wesley, 2002.

[5] M. P. Papazoglou. The world of e-business: Web-services, workflows, and business

transactions. In Proceedings of the Workshop on Web Services, e-Business, and the Semantic Web, 2002.

[6] J. Roberts and K. Srinivasan. Tentative hold protocol part 1: White paper. http://www.w3.org/TR/tenthold-1/, 2001.