1. Introduction

Nowadays in order to stay competitive, business needs to collaborate with each other to offer value added services. Collaborative business process development require loose coupling between individual business processes so that they can interoperate freely across the Internet and that the collaboration can be established in a highly dynamic fashion and on-demand basis. Service Oriented Computing (SOC) is able to meet such request by establishing Business Collaboration (BC) to utilize Business Processes (BPs) crossing organizational boundaries. BP is organizational based, which is about one organization composing services; while BC is choreographically based, which is about cooperation between organizations by linking their BPs.

A successful BC development needs to follow the choreographical requirement to ensure the consistency among individual BPs across organizations. It also needs to meet the organizational requirements to ensure each organization’s BP performs according to its business rules and conforms to what is specified from collaboration point of view. The current Web Services (WS) technologies have provided specifications and tools for BC development. However methodologies, and real case study and experience report in BC development is lacking in the literature. In this paper a Business Collaboration Framework (BCF) is proposed which breaks the BC requirements into 3 layers, each of which models different aspect in the BC development. Practical experience and issues on how to use the BCF is presented and discussed in a case study - developing BC in Capital Marketplace (CM), a project sponsored by the Capital Markets Cooperative Research Centre (CMCRC).

2. Related Work

In this section, we shall discuss the previous work in BC development, particularly based on two approaches: organizational based approach and choreographical based approach. The work in organizational based approach can be further classified into two types depending on whether the starting viewpoint is from service provider or service requester.

For the organization as a service provider, Zhao, Liu and Yang [1] have developed a concept called relative workflow. They defined the organization as a list of local processes. Each task in the process has a visibility constraint as invisible, track-able or contactable. As a result, perception of each local process for external organizations in the collaboration can be determined. So the single-organization-oriented relative workflow to can be developed based on this perception, and the BC can be represented as a list of relative workflow processes. Benatallah et al [2] have developed an adaptor mechanism to overcome the differences between public behaviour of two processes, e.g., the mismatch of message ordering, extra message, missing message, split message and merge message. The adaptor enables the service provider to interact with different service requesters who have different interfaces and protocols.

All the BC development approaches mentioned above are based on individual organizational views either as requesters or as providers. Alistair, Marlon and Phillipa [3] started BC development from chorographical (‘global’) point of view. The choreography describes BC at global level by capturing all interactions between organizations. The participating organizations can then derive their public behaviours to support the interactions specified in the global choreograph. However, details on how to apply such approach for BC development in practise was not discussed.

In this paper we will follow the global choreograph oriented approach for BC development by presenting a three level framework. This framework captures the requirements for BC development at three different layers, each of which models the different aspects of BC development. A real business application is developed based on this framework. An evaluation of this top-down approach for BC development is provided at the end of that paper.


We now introduce the three development phases corresponding to the three aspects of the BCF.

1. Global Behavior Determination. It firstly produces an organization list that includes all organizations
involved in the BC. Secondly it determines the sequence of interactions to archive the BC goal.

2. Exposed Behavior Identification. It derives the public behavior of each organization involved in the BC from the Global Behavior. Firstly, in each interaction described in the Global Behavior, for the organization identified as message retriever, it is required to add a receive activity to its Exposed Behavior in order to retrieve the message. For the organization identified as a message invoker, it is required to add an invoke or reply activity to its Exposed Behavior in order to send the message. Secondly, for each organization, organize all interacting activities by flow activities based on the same order of the corresponding interactions. The list of activities in the Exposed Behavior of each organization can be viewed as an abstract process provided by that organization, which describe the sequence of the business logic that has to be performed by the internal BPs.

3. Business Process Implementation. As the abstract process for each organization is described by the Exposed Behavior, we can then select the necessary Internal Services from the Internal Behavior to turn the abstract process into an executable BPs.

4. A Case Study

Capital Markets (CM) is a virtual institution that supports money exchange cross multi-organizations such as investors, companies and banks. The CM trading cycle is characterized by a large number of heterogeneous systems including information distribution systems (e.g. real-time quotes, historic trade data), order processing systems (e.g. order routing, presentation, and execution), clearing and settlement systems, and research and analysis systems [4]. Due to the globalization of world economy in recent years, the needs of collaboration crossing heterogeneous system in CM grow tremendously. For example: investor wants to be able to invest money in multiple exchange services in other countries through one account. This involves activities such as setting up connections between the account and the heterogeneous systems used in the exchange services and related banks.

As showed in Figure 1, Brokers constitute a necessary mid-layer in CM between Investor and Stock Trading. They also act as advisors providing suggestions to their clients on what stocks to buy and sell. Accordingly, the main functionality of the software application - Broker System is Client Profit Management that formulates a trading strategy plan that advises customer on how or when to place an order by identifying some relevant market conditions to maximize profit. Currently trading strategy plan is implemented by a software package called Simulation Conductor as part of Broker System.

With the current trend of globalization of economics, the trading strategy plan in CM is affected by many factors from all over the world. For example, the raise of petrol price in Mid-East could affect the stock price of the car manufactures in Japan such as Toyota. As a result, the trading strategy plan becomes far more complex in order to response to these new factors, and more dynamic to adapt to the rapid changing environment. The original Simulation Conductor provided by the existing Broker System is no longer up to support these requirements. Rather spending considerable amount of money on both IT and financial expertise to redevelop Simulation Conductor inside Broker System, it will be much more economical and efficient to outsource Simulation Conductor to third party who is expert in this field. As showed in Figure 2, in order to address the requirements mentioned above, the proposing system is designed to be different from existing system as follows:

1. The Simulation Conductor is outsourced to third party.
2. Application-to-application based interactions are shifted to Web services based business collaboration.
3. The Broker System can interact with other business partners to find the suitable trading strategy plan for its customers.
In this case study, the aim is to implement a new BC – Trading Strategy Simulation (TSS). TSS combines the functionalities of Simulation Conductor with other two organizations. TSS formulates trading strategy plan to help maximize clients’ profit. Best trading strategy plan which generate maximum profit is calculated by re-running different trading strategy parameters over historical data. The design and implementation decision of TSS will be discussed in the following sub-sections based on the BCF discussed in the previous section.

4.1. Global Behavior Determination

As introduced in section 3, in this phase we firstly identify necessary organizations involved in the TSS BC. We represent each organization as a Provider that provides essential functionalities involved in the TSS BC. Three providers are identified based on their provided functionalities:

1. **Trading Engine Provider (TEP)**: simulates the behavior of Exchange Services (E.g. Australian Stock Exchange) that allows simulation over historical data.
2. **Trading Data Provider (TDP)**: by reading the input and output message format of TEP, TDP provides necessary historical data for simulation to calculate best trading strategy in a Trading Engine specific format.
3. **Trading Strategy Provider (TSP)**: this service is developed by third party vendor which provide the functionalities of Simulation Conductor. It provides all the necessary Trading Strategy plans based on simulation over historical data. In the simulation, each plan decides to generate different new buy/sell
orders that are submitted to TEP together with historical data. The decision is based on inspecting the buying/selling information in the updated order book. Comparison can be made between the profits generated by each plan based on the outputs of TEP so that the optimized plan can be generated.

4.2. Exposed Behavior Identification

As the Global Behavior has already been described in last section, the task in this phase is to turn the interactions into organizational based abstract BPs, which can be done through the two steps. Firstly, we identify interacting activities for each interaction. Take WS-CDL as an example, each interaction involves the “fromRole” and “toRole”. The “fromRole” corresponds to the organization role as message requester. The “toRole” corresponds to the organization role as message retriever. An invoke or reply activity needed to be add to the message requester. A receive activity needs to be added to the message retriever. Recall that each invoke activity is paired with receive or reply activity in another organization. Secondly, by arranging all interacting activities with flow activities, we are able to describe BP skeleton as an abstract process in BPEL.

4.3. Business Process Implementation

The aim in this phase to turn the abstract process described by the Exposed Behavior into the executable process.

In this case study, BPs need to be implemented for the TSP, TEP and TDP respectively. To save spaces and provide a more illustrative view, instead of providing BPEL executable process code, we choose to use the Business Process Modeling Notation (BPMN) version 1.0 standard to draw Business Process Diagram (BPD) which shows the functionalities involved in BPs and the relationships between them. Note that a BPD conveys in pictures what BPEL encode in XML.

In the BPD, the services belongs to the Exposed Behavior are described in bold font; the rest internal services are described in normal font. The starting and finishing services of the BC are underlined. Due to the space limits, not all the functionalities or activities are sketched in the BPD.

When there is one trading strategies requirement coming to TSP, the TSS BC is initialized by the activity “Receive Trading Strategy Request”. TSP takes required trading strategy type and related historical data period as input, and then returns best trading strategy parameter maximize profit as output of TSS. We can see how TSP is performed by interactions among the TSP, TDP and TEP in Figure 3. Note that the interactions between TDP and TEP are invisible from TSP.

5. Discussion And Conclusion

There are several major concerns during the design and implementation of this case study:

Modeling multiparty interactions in Global Behavior. Currently, WS-CDL only supports describing the interaction between a pair of roles, e.g. one “fromRole” with one “toRole”. It does not explicitly support multiparty interactions, e.g. one “fromRole” with several “toRole”, or several “fromRole” with one “toRole”. We can break the “one to many” interaction into several “one to one” interactions. However, it will increase the complexity of the WS-CDL code if we do so, and it will also make the specification even harder for user to understand.

Mapping between Global Behavior and Exposed Behavior. BPEL came out as a WS orchestration language that does not explicitly support modeling the interaction abstractly. As a result, the mapping between WS-CDL and BPEL is not always straightforward. There are two main problems we encountered in the case study.

Firstly, the syntax mapping is not straight forward. For example, the mapping of “Workunit” in WS-CDL to BPEL remains open. How to map “control flow” from WS-CDL to BPEL is not clear. Secondly, important information is missing for mapping. For example, an abstract BPEL process is consisted of activities such as “invoke”, “receive” and “reply”. BPEL does not explicitly express which of these activities belong to which interaction. Thus, if we modify the Exposed Behavior, it is hard to find the impact on which interaction of the Global Behavior. As a result, missing the explicit expression of the relationship between a message based interaction and activities in abstract process will make it hard for organizations to maintain their BPs in a business collaboration. To solve these problems needs further improvement of BPEL itself and its cooperation with WS-CDL.

6. References


