CHAPTER 1

INTRODUCTION

This chapter provides an overview of the research leading up to this thesis. In this research, transactional management during coordination of services across organizations is investigated, with a focus on reliable web service composition. Tx-FAITH, a novel transactional framework that aims at failure tolerant coordination of services is proposed in this thesis.

The introductory chapter has five parts. In Section 1.1, a brief overview of the research context, as background knowledge is provided. Section 1.2 introduces a motivating example which will be used throughout the thesis to illustrate the approaches proposed in this thesis. Section 1.3 presents the motivation behind the proposed research. Section 1.4 presents the research goal and decomposes that into a number of objectives. Section 1.5 summarizes the intended contributions of this work. Finally, Section 1.6 outlines the organization of the dissertation.

1.1 BACKGROUND

This chapter provides the background knowledge about the relevant concepts to this research work. In particular, a short overview of service oriented architecture, web services, transaction management and workflows are discussed.
1.1.1 Service Oriented Architecture and Web Services

Service Orientation (SO) is a design paradigm which proposes realizing distributed software solutions as services. Service-Oriented Computing (SOC) (Huhns & Singh 2005) emerged as a new computing paradigm following the design philosophy of SO. The aim of SOC is to allow a rapid and low-cost development and composition of distributed applications where they can be easily assembled from existing services (Papazoglou et al 2003). The key to realize this vision is the Service Oriented Architecture (SOA) (Erl 2005). The SOA prescribes a set of principles that facilitates interaction between services. Services are self-contained, self-describing, and platform-independent computational entities that can be described, published, discovered, invoked, and composed using standard protocols. A service performs some function that could range from a response for a simple request to the realization of a complex business process.

A service offers business functionality in SOA, and a service consumer is a software entity that calls the service and uses its functionality. The functionality of a service is defined through an interface. Service interfaces are useful for exchange of messages and for achieving loose coupling of services that improves robustness, makes systems more resilient to changes, and promotes reuse of services. In order to simplify and automate searching for the appropriate services, services are maintained in a service registry. Figure 1.1 shows the typical roles and interactions in SOA. A service provider publishes a service description into a service registry. A service consumer discovers the service in the service registry and obtains the necessary information to bind the service.
Web services (Alonso et al 2003) are the predominant technology for realizing an SOA. As defined by World Wide Web Consortium (W3C) (Roberto et al 2004), a web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format, specifically WSDL (Roberto 2007) and published into UDDI (2004). On the other hand, a service consumer can find the service through UDDI and use it by sending SOAP (2007) messages.

One of the most important concepts in SOA is service composition. If the functionality cannot be provided by a single service alone, several of them need to be composed to achieve the desired functionality. If a service is implemented by combining the functionality provided by some other services, it is known as composite service. Otherwise, it is known as a basic service or simply a service. A composite service can be further used in other composite services. The services that constitute a composite service are called component services. A component service can be either a basic service or a composite service. By service composition, business applications can be built in a flexible and easy way, and can also be modified quickly with less effort.

For both basic and composite services, the description of their functionalities can be provided in the same way by using WSDL. However,
the implementation of their functionalities is different. Particularly, the functionalities of basic services can be implemented by traditional programming languages, such as Java and C#. However, the composite service is currently implemented by an XML-based language, WS-BPEL (Jordan & Evdemon 2009). Figure 1.2 depicts the architectural view of SOA and the corresponding web services standards that realize the SOA.

(a) Architectural view of SOA
(b) Realizing SOA through web services

Figure 1.2 The architecture of SOA and its realization using web services

1.1.2 Transaction Management

Transactions are one of the basic building blocks in all business applications. The flat transaction (Bernstein 1997) concepts were originally proposed in the context of databases to preserve the ACID properties. A nested transaction (Moss 1985) may recursively start a number of subtransactions, forming a transaction tree. The sub-transaction will commit and externalize its results only upon the commit of all its ancestors. A Multilevel transaction (Weikum & Schek 1992, Gray & Reuter 1993) has a transaction tree that has its levels corresponding to the layers of the underlying system architecture. The sub-transactions can commit before the commit of the root transaction (called pre-commit), thus making the rollback of committed sub-transactions impossible. When a parent transaction wants to rollback its sub-transaction, it uses a compensating transaction to semantically undo the effect
of the committed one. When the transaction tree structure allows leaves at
different levels, multilevel transactions evolve into open nested transactions
(Gray & Reuter 1993). Open nested transactions improve performance by
relaxing the isolation property at a global level. Nested transactions are
applicable only in specific environments like federated databases but are not
suitable for applications involving long-running transactions. In such cases,
the idea of chained transactions (Gray & Reuter 1993) and Sagas (Molina &
Salem 1987), were proposed. The saga either completes successfully or in the
case of an abortion, each of the committed sub transactions is undone by
applying the corresponding compensations in reverse commit order. The
ConTract model (Wachter & Reuter 1992, Reuter & Schneider 1997)
generalizes ACID transactions to a control mechanism for large distributed
applications.

The extended and relaxed transaction models are mainly geared
towards database entities with transaction management features. Database
management systems focus only on preserving data consistency and not on
coordinating independent tasks among different entities. Hence, the concept
of transactions was introduced in workflow management systems. A
transactional workflow (Sheth & Rusinkiewicz 1993) involves the
coordinated execution of multiple long-running tasks which can be performed
by distributed and heterogeneous processing entities. From a transactional
point of view, workflows are generalized extended transactions which focus
on the automation of the complex, long-running business processes in
distributed and heterogeneous systems. Similar to the decomposition
mechanism of advanced transaction models, a workflow process can be
modeled by decomposition into some sub-processes in a hierarchical or
sequential way. From this perspective, a workflow process can be viewed as a
complex transaction hierarchically or sequentially consisting of sub-
transactions (Warah & Sheth 1997).
The WIDE transaction model (Grefen et al 1997) has two layers that cater to both the requirements of long lived global transactions and traditional local transactions. To apply the workflow management model in this B2B e-business scenario, it must be extended to support cross-organizational processes with transaction support. The X-transaction model (Vonk & Grefen 2003) is a three-level, compensation based transaction model to support and manage cross-organizational workflows. A limitation of the workflow transaction model is the software / platform dependency. With the development of internet applications, transaction support for workflow processes has evolved into services that provide more attention to communication, distribution, and coordination aspects. For interaction between applications, the web services standard provides a platform independent exchange of messages. Web service transaction models (Kratz 2004) help in guaranteeing the consistency and reliability of WS applications.

In order to cater to the new features of transactions executed by web services, various web service transaction protocols such as Business Transaction Protocol (BTP) (Ceponkus et al 2002), WS-Coordination (WSC) (Carera et al 2005a), and WS-Composite Application Framework (WS-CAF) (Bunting et al 2003) had been released from OASIS (Organization for the Advancement of Structured Information Standards). The applicability and role of these three protocols have been identified and analysed by Kratz (2004), Little & Freund (2003), and Husemann et al (2007). Leymann & Pottinger (2005) identified the shortcomings of WS-C such as its inability to query the coordination state. The authors had mentioned the coordination rules and parameterization of coordinators by clients, but did not relate these concepts to transaction management. Though the problem of long-running transactions with web services was addressed by BTP and WS-C standards, interoperable transactions have not been considered. Kratz (2004) presented a comparison of web service transaction standards. It concluded that a unified
operational standard is essential to realize interoperability, by integrating existing specifications within WS-CAF framework. However, the work on WS-CAF specification has no longer been continued by OASIS.

ACTA (Chrysanthis & Ramamritham 1990, 1992) is a framework designed to unify the existing models. The ACTA Meta model (Chrysanthis & Ramamritham 1994) can be used to develop new transaction models by analyzing the inter-dependencies of the existing models. The Business Transaction Framework (BTF) (Wang et al 2006), supports contract-driven, cross-organizational business processes in a service oriented environment. BTF achieves flexibility by leveraging and abstracting the existing transaction models. It also guarantees business trustworthiness by using contractual agreements to specify transactional qualities of business processes. A taxonomy of different transaction models is presented in Figure 1.3.

In summary, existing WS standards and frameworks that propose to implement web services transactional coordination protocol are not adequate for handling hierarchically composed interoperable services belonging to different organizations.

1.2 ILLUSTRATIVE EXAMPLE

A B2B application of Vehicle Purchase and Registration System (VPRS) is considered to bring out the requirement of reliable transactions and to motivate the need for a transactional framework. Whenever a common citizen wants to buy a new vehicle, two-wheeler or four-wheeler, multiple websites must be visited to search for a suitable model. Also, the vehicle must be insured with an insurance agency and registered with a government transport authority before being delivered to the requester. The details pertaining to insurance must be gathered from various websites. In order to avoid multiple lookups, a unified entry point in the form of a web
Figure 1.3 A taxonomy of different transaction models

portal can be provided to a customer for easier access to information and reducing process delays. When each of these processes is provided as a web service, several of them can be composed to fulfill the user requirements.

The vehicle details such as type, make, model, and colour along with the personal details of the customer such as Social Security Number (SSN), name, address, and Permanent Account Number (PAN) are gathered through a portal interface. SSN and PAN are mandatory for valid citizenship and tax payment for purchases. Existing vehicle dealers for the specified type (two-wheeler or four-wheeler), the make, and the model are searched and cost details are gathered. The enquiry services of different dealers can be invoked in parallel. The customer chooses a particular dealer based on budget, colour, and other preferences. The customer details are verified with the Birth and Death Registration (BDR) department of the government. Similarly, PAN is verified with the Income Tax (IT) department of the government.
In general, services offered by different organizations are used for verification processes. Hence, they can be carried out simultaneously. If the customer details are found to be valid and the requested vehicle can be delivered by a specific dealer, payment is received from the customer. On successful payment, a purchase order is placed with the vehicle dealer. After the dealer delivers the requested vehicle, the vehicle is insured with an Insurance Agency (IA) with a suitable scheme matching the credit status of the customer. The insured vehicle is then registered with the Transport Authority (TA), a governmental body for performing vehicle registration. The registered vehicle is delivered to the customer with a new registration number.

When an order for a vehicle is received by the dealer, the order processing commences with generating a unique order id and receiving payment for the order. The stock is then checked for availability of the ordered vehicle. If the required vehicle is not available in stock, it needs to be obtained from the manufacturer. The orders for the vehicles which are not available in stock are accumulated for a day or a week before submitting them to the respective manufacturer. On receiving the request, manufacturer assembles and inspects the vehicle according to the order. The stock is updated and the shipping details are generated to deliver the vehicle to the dealer. After receiving the vehicle, the dealer updates the stock and generates the corresponding Delivery Challans (DC). The vehicle is then delivered by the dealer to the one who had placed the order.

The vehicle registration process involves verifying the owner details. If the vehicle is registered for personal usage, the payment for Life Tax (LT) is received. Alternatively, in case of a commercial vehicle, Weight Certificate (WC) is produced to obtain permission for the vehicle. After receiving the payment, Registration Certificate (RC) is issued for the vehicle. The control flow of the VPRS is depicted in Figure 1.4.
Figure 1.4  Vehicle purchase and registration system (VPRS)
It is clear that, a single service in itself cannot implement the entire functionality of the VPRS, since it involves interaction among multiple organizations and integrating the services offered by them. Hence, the VPRS application involves composition of services provided by BDR department, IT department, vehicle dealer, bank, vehicle manufacturer, insurance agency, and transport authority. The vehicle dealer provides a non-composite service for matching the requirements with the available features and a composite service for order processing. The transport authority offers a composite service for vehicle registration. Similarly, other participant organizations offer non-composite or composite services. Such composite services may involve a component which itself is another composite service. For example, the composite service for order processing involves a nested composite service for placing an order with the vehicle manufacturer. Some of the long-running transactions in the VPRS include dealer-side order processing, manufacturer-side order processing, order accumulation, vehicle insurance and registration. Since this vehicle purchase process is a complex and long-running service-oriented business process, and is full of exceptions and failures, reliability (of both the execution and the result) is highly significant.

1.3 MOTIVATION

Within the context of service-oriented business processes, there have been many research efforts in service composition (e.g. BPEL). However, it is noticed that some necessary properties such as reliability, security, and robustness of such processes are not adequately addressed. The concept of transaction is perceived differently by various worlds. In the IT world, transaction management is originally a database mechanism, while in the business world, a transaction is a trading term for expressing the exchange of values. These different interpretations of the same concept very often result
in technology solutions which are inadequate for business requirements. Existing transaction management mechanisms are widely adopted in information systems to handle exceptions and errors. However, they do not guarantee reliable business process execution, despite the failures. Therefore, the primary motivation behind the current research work is to address the issue of reliability within the context of service-oriented business processes.

In addition, the ability to cancel a service allows more flexibility in varying the business policies and user requirements as they are frequently changing. It is necessary to interrupt the execution of long-running transactions, since execution of the corresponding service to completion may no longer be meaningful. Moreover, it is necessary to ensure that if the service execution is not useful, it should not reflect any change in the system. The overhead in executing a service is higher when it is not possible to cancel its execution in midway and undo the effect of its completed portions. A cancelable long-running service minimizes the overhead involved in using it, since at any point of time, it can be interrupted to terminate its execution and the effect of its completed portion of work can be undone.

None of the mature transaction mechanisms providing reliability are addressing external interruption of a business process. Moreover, the existing transactional frameworks achieve a reliable execution of composite services by either compensating the completed services or discovering an alternate service in place of a failed service. However, services have different behavioural capabilities and based on their behaviour, they need to be recovered. Therefore, it becomes rather essential to have a transactional framework with appropriate cancellation recovery mechanism so that the execution of a composite service can be restored back to a consistent state upon external interruption of a long-running business process.
1.4 OBJECTIVES

Motivated by the shortcomings of the prevailing transactional frameworks for service oriented processes, the overall goal has been identified as:

*The design and development of a transactional framework Tx-FAITH that achieves reliable execution of hierarchically composed service-oriented business processes in the presence of failures and interruptions of the processes.*

The following objectives need to be met in order to achieve the above goal:

- Definition of cancelable web services and cancellation recovery mechanism
- Extraction of transactional requirements from business policies
- Specification of transactional capabilities designed by the service providers, in WSDL descriptions of services
- Selection of services by matching the transactional requirements and capabilities
- Generation of transactional contracts to ensure trustworthiness of the service providers
- Design of a transactional coordination mechanism for hierarchically composed workflows across organizations
- Design of a transactional recovery protocol that includes cancellation recovery of long running processes
- Validation of the usability and effectiveness of Tx-FAITH through a prototype business application
1.5 CONTRIBUTIONS

The contributions of this thesis that achieve the goals and objectives specified above are the following:

- An abstraction based on the recoverability of services, to enable the business analyst to specify the transactional properties of services in a simple manner
- Dynamic transaction aware web services selection that considers cancelable web services
  - Transactional service descriptions by integrating WSDL with transactional capabilities expressed as policies
  - Dynamic contract generation with transactional agreements
- Failure tolerant transactional coordination of hierarchically composed workflows
  - Transactional recovery based on the transactional capability of the failed or interrupted service and the type of the failure
  - Cancellation recovery mechanism for interrupted long running processes

1.6 ORGANIZATION OF THESIS

The remainder of this thesis is organized as follows:

Chapter 2 reviews the related research effort classified according to each research issue.
Chapter 3 defines cancelable property of services and the possible transactional properties of a valid composition. It also introduces cancellation recovery.

Chapter 4 discusses the proposed abstraction based on recoverability for specifying the transactional properties. This chapter also illustrates the gradation of recoverability levels based on the recovery cost and its justification.

Chapter 5 presents the dynamic approach for transaction aware service selection. This chapter also illustrates approaches for including transactional properties in the service description and for including transactional guarantees in contracts as well as dynamic generation of contracts.

Chapter 6 introduces different types of failures handled by Tx-FAITH framework, and then proposes the architecture for Tx-FAITH. This chapter covers the transactional coordination and recovery approaches.

Chapter 7 presents the experimental test bed for the VPRS prototype and shows the validity of all the proposed approaches in this thesis, by adopting them in the prototype and testing them. It also presents a comparative study of Tx-FAITH against the existing transactional frameworks.

Chapter 8 summarizes the results and contributions of this research work. In addition, it outlines a few possible directions for future work.