ABSTRACT

Patterns are used as building blocks for design and construction in many fields such as architecture, music, literature, etc. Researchers and practitioners in the information systems area have been exploring patterns and using them in system analysis and design. Patterns found in the analysis stage, when analysts create conceptual models to abstractly represent domain reality, are called business patterns or analysis patterns. Although various business patterns were proposed in previous studies, we found that business semantics were missing in these patterns. These business patterns failed to show functionalities that are essential to patterns in general. Most of these patterns were also not capable of describing business activities, the dynamic aspect of business. This study is conducted to address these issues. In this thesis, we provide a brief literature review on business patterns, and discuss the major problems we found in these studies. Then we introduce our research approach and the major outcomes. We propose a new definition of business patterns with business semantics, which enables us to recover the missing functionality in business patterns. We suggest the key elements to represent business patterns, and propose a two-level template (functional and operational) to describe these elements. Based on the R²M approach, we propose a modeling method with graphical notations to describe the operational level of patterns, where business activities can be modeled. Examples and a case study are provided in this thesis to demonstrate how to use the modeling method and how to use business patterns in practice.
# TABLE OF CONTENTS

ABSTRACT ............................................................................................................................................ ii

TABLE OF CONTENTS .......................................................................................................................... iii

LIST OF TABLES ..................................................................................................................................... iv

LIST OF FIGURES ................................................................................................................................. v

ACKNOWLEDGEMENTS ........................................................................................................................ vi

1. INTRODUCTION ............................................................................................................................. 1

2. SEEKING BUSINESS PATTERNS ................................................................................................. 8
   2.1. Business Modeling & Business Patterns .................................................................................... 8
   2.2. Patterns in Literature ................................................................................................................. 12
   2.3. Problems & Research Questions ............................................................................................... 21

3. APPROACHING BUSINESS PATTERNS ..................................................................................... 25
   3.1. The Research Approach & Scope ............................................................................................. 25
   3.2. Definition of Business Pattern .................................................................................................. 27
   3.3. Elements in a Business Pattern ................................................................................................. 29
   3.4. Levels of Business Patterns ...................................................................................................... 32
   3.5. A Methodology to Model Business Patterns ............................................................................ 33

4. MODELING BUSINESS PATTERNS ............................................................................................ 35
   4.1. Brief Review of R²M ................................................................................................................. 35
   4.2. A Template for Describing a Business Pattern ........................................................................ 40
   4.3. Graphic Notations ...................................................................................................................... 42
   4.4. Examples of Business Pattern .................................................................................................. 51

5. USING BUSINESS PATTERNS ...................................................................................................... 78
   5.1. Combining Patterns .................................................................................................................... 78
   5.2. Employing Patterns in a Case Study ......................................................................................... 84

6. CONCLUSIONS .............................................................................................................................. 91

REFERENCES ....................................................................................................................................... 96

APPENDICES ....................................................................................................................................... 100
LIST OF TABLES

Table 2.1 Patterns in IS Literature ................................................................. 20
Table 3.1 Pattern Elements in Literature ....................................................... 31
Table 4.1 R²M Concepts .............................................................................. 36
LIST OF FIGURES

Figure 4.1  Graphic Notations for $R^2M$ ................................................................. 37
Figure 4.2  Graphic Notations for Modeling Business Patterns ................................. 43
Figure 4.3  Functional Model of “Customer Purchase” Pattern .................................... 56
Figure 4.4  Operational Model A of “Customer Purchase” Pattern ............................... 57
Figure 4.5  Operational Model B of “Customer Purchase” Pattern ............................... 59
Figure 4.6  Operational Model C of “Customer Purchase” Pattern ............................... 60
Figure 4.7  Operational Model D of “Customer Purchase” Pattern ............................... 62
Figure 4.8  Operational Model A of “Account Payable” Pattern ................................... 64
Figure 4.9  Operational Model B of “Account Payable” Pattern ................................... 66
Figure 4.10 Operational Model C of “Account Payable” Pattern ................................... 68
Figure 4.11 Functional Model of “Account Payable” Pattern ....................................... 70
Figure 4.12 Functional Model of “Push-Pull” Pattern ................................................ 74
Figure 4.13 Operational Model A of “Push-Pull” Pattern ............................................. 75
Figure 4.14 Operational Model B of “Push-Pull” Pattern ............................................. 75
Figure 4.15 Operational Model C of “Push-Pull” Pattern ............................................. 76
Figure 5.1  Functional Model of “Customer-Business-Supplier” Pattern ....................... 79
Figure 5.2  Operational Model of “Customer-Business-Supplier” Pattern ..................... 80
Figure 5.3  Operational Model A of “Push-Pull” Pattern in Account Payable Process ......... 81
Figure 5.4  Operational Model B of “Push-Pull” Pattern in Account Payable Process ......... 82
Figure 5.5  Operational Model C of “Push-Pull” Pattern in Account Payable Process ......... 83
Figure 5.6  Operational Model of B-Line Overall Operation ......................................... 86
Figure 5.7  “Push-Single” Model of Transferring Order Assembly List ......................... 88
Figure 5.8  “Push-Batch” Model of Transferring Order Assembly List .......................... 88
Figure 5.9  “Pull-Batch” Model of Transferring Order Assembly List ............................ 89
Figure 5.10 Operational Model of Transferring Order Assembly List & Aggregate List .... 90
ACKNOWLEDGEMENTS

I offer my enduring gratitude to my thesis supervisor Dr. Yair Wand, a great mentor, who has enlarged my vision of science and philosophy, and encouraged and supported my research in this field. I own particular thanks to Dr. Carson Woo and Dr. Andrew Burton-Jones, my thesis committee members, for their insightful advice and constructive comments.

Special thanks are owed to my parents and my wife, Wei Song, who have supported me throughout my years of education.
1. INTRODUCTION

Modeling, the creation of abstract representations of certain domain realities, is important to information system development. The two major areas in which IT professionals have been using modeling are system analysis and design. For many years, researchers and practitioners have developed modeling methods and techniques for system design where these methods and techniques are used to create blueprints of new information systems. Since the blueprints are directly related to the success of new systems, the necessary role of modeling in system design has been well recognized for decades. In recent years, rising attention has been given to modeling for system analysis. The system analysis phase of information systems development is concerned with representing a business domain (Evermann & Wand, 2005) and involves looking behind the surface requirements to develop a mental model of what is going on in the problem (Fowler, 1997, p.1). Therefore, modeling is important in the analysis stage of system development when abstract models of the represented system and its organization environment are created. Such models are termed conceptual models (Wand et al, 1995). In this thesis, these conceptual models are also called business models because they are representations of business domains.

There are several motivations for business modeling: to better understand the key mechanisms of a business, to act as the basis for supporting information systems, to improve the business, to radically change the business, to identify new business opportunities, and to identify outsourcing opportunities (Eriksson & Penker, 2000, p.16). A modeler can also choose different modeling methods to develop desirable models which capture specific facets of a business, for example, an enterprise model for the whole organization or a process model for a specific business function.
The notion of a pattern was introduced to the information system area after many IS researchers and practitioners were inspired by Christopher Alexander's architectural patterns. Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem (Alexander, 1977, p.x). The idea of summarizing architectural patterns to provide a guideline to build buildings motivates IS researchers and practitioners to seek patterns in the IS area and use these patterns to guide system development. Various patterns have been identified and proposed for different phases of software development, some of which are more specific for system design while the others are more specific for conceptual design (analysis) (Purao et al., 2003; Yacoub & Ammar, 2003, p.10). In this thesis, we categorize these patterns as “design patterns” and “analysis patterns”, or patterns for system design and patterns for business analysis. Examples for the first category are the design patterns proposed in (Gamma, 1995), the architecture patterns and design patterns in (Buschmann et al., 1996), and the EAI/communication patterns in (Linthicum, 1999; Ruh et al., 2000; Hohpe & Woolf, 2003). Within the second category, different works vary to a great extent in terms of their focus. Some focus on general business concepts, for instance, the analysis patterns proposed in (Coad, 1992; Coad et al., 1995), the analysis patterns in (Fowler, 1997), the business patterns in (Eriksson & Penker, 2000), and the REA based business patterns in (Hruby, 2006), as some focus more on business communication, for instance, the Action Workflow approach in (Medina-Mora et al., 1992) and the transaction pattern based on speech act theory in (Dietz, 1994 & 2006).

The notions of “business pattern” and of “analysis pattern” in the literature refer to the patterns used for business analysis as they both refer to certain recurring phenomena or common constructions in business modeling. While the notion of analysis pattern expresses the usage of the patterns and the notion of business pattern indicates the content of the patterns, we use the notion of “business pattern” in this thesis to be more consistent with the notion of “business modeling”.

2
During the review of the existing works on business patterns, we found two common problems. The first problem exists in the studies such as the business patterns proposed in (Eriksson & Penker, 2000) and (Hruby, 2006). These studies try to identify and define business concepts in business practice and try to propose a formal way to model these concepts in business modeling. Although these patterns may serve well as modeling manuals for business modelers, the problem is that such patterns are more like “patterns of business modeling” than “patterns of business itself”. The second problem exists in the studies such as the transaction pattern in (Dietz, 1994 & 2006), the action workflow pattern in (Medina-Mora et al, 1992), the action types in (Umapathy & Purao, 2007), and the workflow patterns in (van der Aalst et al, 2003; Russel et al, 2004a, 2004b, 2006a, 2006b). Although these patterns describe how people communicate or arrange resources to conduct business activities and help to reveal the detailed dynamics among different parties, the problem is that they are lacking specific business meanings. Therefore, such patterns may be a way to present business patterns, but are not business patterns in themselves. These two problems will be further discussed and explained with examples in the next section.

We argue that the fundamental reason which causes these two common problems is that most of these patterns are a result of analysis of the very basic unit of business domain so that each pattern is dedicated to define or to model the concept of the building block. Despite of the strength of providing generic definition and classification of the building block of business, the common weakness of such level analysis is the lack of business semantics because the concepts are too generic. We argue that a business pattern should possess certain business semantics. In other words, a business pattern should describe certain business functionalities and activities, and should be able to describe business rules.
But what will be a business pattern with semantics? What information should be included? How can we model such business patterns? How can we use these patterns? This thesis has been motivated by these research questions. We try to answer these questions by the following steps: 1) review the existing definitions of patterns and try to propose a new definition of business patterns, which incorporates business semantics, 2) based on the definition and related existing literature, propose what level of abstraction business patterns should be shown of and what kinds of information should be included, 3) propose a methodology to model business patterns at the suggested level of abstraction, 4) provide examples of business patterns to demonstrate how to model the patterns using the proposed modeling methodology, and 5) employ a case study to further explain how these business patterns can be used.

Thus, the objectives are to claim what can be business patterns and to propose a method to model such patterns. The intention is to introduce an idea of “business patterns” and how to represent and use them in the context of system analysis for future research and practice, but not to introduce a detailed set of business patterns or a complete catalogue of patterns that can be used as reference models.

This thesis summarizes our research on business patterns for business modeling. The major outcomes of the research include: a definition of business pattern, a template to represent a business pattern, and a modeling method for business patterns.

Our modeling method has been developed based on $R^2M$ or Role-Request Modeling (formerly called OOEM or Object-Oriented Enterprise Modeling approach), a modeling method designed to model work systems which are combinations of people and resources.
working together to attain certain goals. Because an organization can be viewed as a group of interacting work systems, $R^2M$ can be used to model business organizations or enterprises (Wand & Woo, 2008). Because the original $R^2M$ concepts and the graphic notations (Wand & Woo, 1999; Wand et al, 2000; Wand & Woo, 2008) cannot cover all the elements we expect to fully represent a business pattern, we have modified and extended some of the concepts and the notations to reach the coverage. Although there are many modeling methods used for business modeling, $R^2M$ possesses unique strengths which motivate us to adopt it as our starting base to develop our methodology of modeling business patterns.

First, $R^2M$ is developed based on an ontological theory, Bunge's ontology (Bunge, 1977 & 1979), which is one of the potential theoretical foundations for conceptual modeling in system development (Wand et al, 1995). The ontological background provides $R^2M$ with a set of modeling rules, which can help users to effectively identify and model the key aspect of a business domain using $R^2M$ concepts (Wand & Woo, 1999; Wand & Woo, 2008). For instance, $R^2M$ has strict rules to identify the system scope, the roles and their service to be included in a model.

Second, as a method to model work systems, $R^2M$ is capable of describing dynamics and functionalities in organizations. Its concepts, such as roles, services, and communications, are consistent with the elements we propose for a business pattern.

Third, $R^2M$ directly uses business constructs and business meaning is already part of $R^2M$. On the other hand, many other modeling approaches, such as UML, were formed to model software and require the attachment of meaning to constructs.

Fourth, because $R^2M$ use concepts such as roles, services, and communications, we
suggest that it is understandable to business users. We believe that the $R^2M$-based modeling methods would be easy to learn, and not only system analysts but also general business users can use the method in practice.

The original $R^2M$ method is described in detail in (Wand & Woo, 1999; Wand et al, 1999; Wand et al, 2000). We will introduce the basics of $R^2M$ in a later section of this thesis. Our introduction will cover the $R^2M$ concepts and their relationships, the $R^2M$ rules, the suggested $R^2M$ notation, and an example for demonstrating how to use the $R^2M$ method to model a work system.

In practice, business/system analysts and business users are expected to be the major users of the modeling method and the patterns. Since identifying and collecting patterns is out of the scope of this thesis, the users of patterns are expected to collect patterns by themselves. They can employ the modeling method to summarize and record the patterns they collect. After an analyst accumulates a set of business patterns, s/he actually can use the patterns as building blocks to compose major business functions in organizations or use the patterns as formal representations to describe business requirements for system development. Furthermore, when an analyst or designer finds that the functionalities of a business pattern meets certain requirement, s/he can use available patterns on hand to explore the appropriate solution given the constraints specified in the real case.

To answer the research questions we raised in the beginning, we will further explain our research approach and introduce the outcomes in the following parts of this thesis. Section 2 provides a literature review on the topic of business modeling and business patterns, discusses common problems we found in the previous studies, and raises our research problems. In section 3, we introduce our research approach and define the research scope, following which we propose our definition of business pattern, suggest
the elements to be included in a business pattern, explain why we suggest modeling business patterns in two levels (functional level and operational level), and introduce the R²M method as a starting point to develop our methodology for modeling business patterns. Section 4 reviews more details of the R²M method, based on which we propose a template to model business patterns and several graphic notations, by using which we can extend the R²M method to model the both levels of business patterns. Section 4 also provides three detailed examples of business patterns, for each we explain how to identify the pattern and how to model it in two levels using our proposed methodology. Dedicated to the use of the patterns, section 5 is divided into two parts, where the first part introduces two examples of how to use the patterns in general and the second part demonstrates how to use the patterns in a case study. Finally, section 6 concludes our research on this topic by discussing both strengths and limitations of our current work and suggesting future research and practice.
2. SEEKING BUSINESS PATTERNS

2.1. Business Modeling & Business Patterns

2.1.1. Business Modeling

It is undeniable that information technologies have brought tremendous changes to business. Hammer and Champy emphasize the enable role of information technology in reengineering as such: it is the disruptive power of technology, its ability to break the rules that limit how we conduct our work, that makes it critical to companies looking for competitive advantage (Hammer & Champy, 1993, p.91). At the same time, the continuous evolution of business is posing a challenge to both IT professionals and business managers. As enterprises are getting larger and business processes are getting more complicated than ever, developing information systems for such enterprises and business processes is becoming more difficult.

Information system projects begin by examining and understanding the business and organizational domain in which the information system is to be embedded. The system analysis phase of information systems development is concerned with representing this business domain (Evermann & Wand, 2005). Modeling is especially important in the analysis stage of systems development when abstract models (conceptual models) of the represented system and its organization environment are created (Wand et al, 1995). In other words, analysis involves looking behind the surface requirements to come up with a mental model of what is going on in the problem (Fowler, 1997, p.1). In practice, such conceptual models of business domains are also called business models. A business model is an abstraction of how a business functions (Eriksson & Penker, 2000, p.2).
The benefits from business modeling are not limited to passing clear requirements from the analysis stage to the design stage and facilitating information system development. Business modeling can be used as an independent tool and "the difference is whether we think about conceptual modeling as a process itself or as one aspect of the entire software design process" (Fowler, 1997, p.2). There are several motivations for business modeling: to better understand the key mechanisms of a business, to act as the basis for supporting information systems, to improve the business, to radically change the business, to identify new business opportunities, and to identify outsourcing opportunities (Eriksson & Penker, 2000, p.16).

A business is often a complex system and can be viewed in many different ways. To capture specific facets of a business and to fulfill specific objectives, a modeler can choose different modeling methods to develop desirable models. For an example, an enterprise model is a computational representation of the structure, activities, processes, information, resources, people, behavior, goals, and constraints of a business, government, or other enterprise; the role of an enterprise model is to achieve model-driven enterprise design, analysis, and operation (Fox & Gruninger, 1998). A process model is an abstract description of an actual or proposed process that represents selected process elements that are considered important to the purpose of the model and can be enacted by a human or machine (Curtis et al, 1992). Although these models are different in terms of their focus and objectives, it should be noted that these different models are sometimes closely related. For example, enterprise modeling is a prerequisite for enterprise integration (Vernadat, 1996, p.18) while process modeling is often used to facilitate process reengineering. Vernadat (1996) suggests that the goal of an enterprise modeling approach is not to model the entire enterprise in all its details; the term enterprise means a part of company which needs to be represented and its size and scope is defined by the business users; enterprise modeling and integration is essentially a matter of modeling and
integrating these processes; because business processes represent the flow of control of thing happening in the enterprise, enterprise modeling is driven by business processes modeling (Vernadat, 1996, p.19).

2.1.2. Business Patterns in Business Modeling

In this thesis, we use "business modeling" as a general notion for all the modeling approaches dedicated to the representation of certain business domains or aspects. Because a business model is an abstraction of business domain reality, one of the challenges facing modelers is to identify the most important information in the domain reality, based on which the abstraction is developed. In the IS literature, such important information in a business domain is termed the essence of business or the core business. Wand et al claim that a business model reflects the core business of an organization, and "it captures, at a very high level, how an organization behaves, independent of the actual organizational structure and implementation of business processes" (Wand et al, 1999,p.137). In the book on enterprise ontology, Dietz claims that the lasting reward of such a study (enterprise ontology) is a novel and powerful insight into the essence of the operation of enterprises; by this we mean insight that is fully independent of the realization and implementation" (Dietz, 2007, p.7).

The underlying assumption or belief behind such statements is that some levels or some parts of the business have been much less changed compared to the unceasing evolution of organizational structures and business processes. Because of their recurrence over time in the business domain, they can be called "business patterns", which is the research topic of this thesis. The motivation of studying the business patterns are: 1) to find a better way to understand and analyze business via the patterns and 2) to use these patterns to better capture and represent the essence of business in business modeling.
As Coad claims in his paper introducing Object Oriented analysis and design patterns, many fields have been using patterns in various ways. For instances, in music and literature, a pattern is the coherent structure or design of a song or book; in architecture, a pattern is an architectural design or style; in psychology, a pattern is a thinking mechanism that is basic to the brain’s operation, helping one to perceive things quickly. With each pattern, small piecework is standardized and assembled into a larger chunk or unit; patterns become the building blocks for design and construction and finding and applying them indicates progress in a field of human endeavor (Coad, 1992).

But what exactly are business patterns and how can we model these patterns? To answer these questions, our research starts with a review of previous studies relevant to this topic. During the literature review process, we find some studies are easier to identify since they are directly labeled with “business patterns” while others are more implicit in their connection to business patterns.
2.2. Patterns in Literature

2.2.1. Introducing Patterns in IS

Although the very origin of the notion of “pattern” remains unclear, there seems to be a consensus in information systems research and the software industry that the architectural patterns proposed by Christopher Alexander (1977) was a major inspiration to the researchers and practitioners in these areas. In his books, “A Pattern Language: Towns, Buildings, Construction”, Alexander describes 253 architectural patterns and organizes them into 3 categories. He claims that “each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem”, and these patterns are able to provide a guideline to build a compete building (Alexander, 1977, p. x). Our definition of business patterns, which is proposed in later sections, is based on Alexander’s definition of pattern.

In the context of information system and software engineering, researchers and practitioners have been trying to identify and propose “patterns” in different phases of software development. Among these works, some are more specific for system design while the others focus on analysis patterns for conceptual design (Purao et al, 2003; Yacoub & Ammar, 2003, p.10). In this thesis, we classify most of these works into two categories: 1) patterns for system design and 2) patterns for business analysis.

The most popular works in the first category include: the design patterns proposed in (Gamma, 1995), the architecture patterns and design patterns in (Buschmann et al, 1996), and the EAI/communication patterns in (Linthicum, 1999; Ruh et al, 2000; Hohpe & Woolf, 2003). The most influencing studies in the second category include: the analysis patterns proposed in (Coad, 1992; Coad et al, 1995), the analysis patterns in (Fowler,
1997), the business patterns for UML modeling in (Eriksson & Penker, 2000), and the REA based business patterns in (Hruby, 2006), the Action Workflow approach in (Medina-Mora et al, 1992) and the transaction pattern based on speech act theory in (Dietz, 1994 & 2006). There are also a few works that can be used for both design and analysis. For instance, the workflow patterns summarized in (van der Aalst et al, 2003; Russel et al, 2004a, 2004b, 2006a, 2006b) can be employed for business process analysis and workflow system design. Since the major interest of this thesis is the business modeling, we will briefly introduce the design patterns and discuss business/analysis in more details.

2.2.2. Patterns for System Design

Design patterns, according to Erich Gamma (1995), describe general solutions to specific problems in object-oriented software design. With a purpose of promoting reuse and flexibility in software design, his book collects and summarizes 24 patterns in three categories (creational, structural, and behavioral patterns), where “each design pattern systematically names, explains, and evaluates important and recurring design in object-oriented systems.” The template proposed by Gamma for describing a design pattern includes following sections: pattern name, intent, motivation, applicability, structure, participants, collaborations, consequence, implementation, sample code, known uses, and related patterns. According to Buschmann et al, patterns at the design phase can be differentiated into architecture patterns and design patterns: an architecture pattern expresses a fundamental structural organization schema for software systems; architectural patterns specify the system-wide structural properties of an application, and have an impact on the architecture of its subsystems (Buschmann et al, 1996, p.12); a design pattern provides a schema for refining the subsystems or components of a software system or the relationship between them (Buschmann et al, 1996, p.13). Buschmann et al
propose four categories of architectural pattern (form mud to structure, distributed systems, interactive systems and adaptable systems) and five categories of design patterns (structural decomposition, organization of work, access control, management, and communication.). Examples of the architectural patterns include layer, filters and pipes, and model-view-controller; examples of the design patterns include whole-part pattern, master-slave pattern, proxy pattern, command processor pattern, and forwarder-receiver pattern.

During recent years, there has been a shift from a functional to a process-oriented approach in business practices, as well as in information system development. New demands have been raised for integrating the underlying information systems. Enterprise Application Integration (EAI) has become an area that attracts growing attention from the software industry and the research community (Linthicum, 1999; Wohed et al, 2003). Similar to (Ruh et al, 2000), which identifies a set of communication patterns, where the communication is realized by exchanges of messages between points located in different processes, (Hohpe & Woolf, 2003) proposes a set of messaging patterns for enterprise integration. The messaging patterns are a comprehensive summary of various forms of messages, channels, routings and endpoints, where the messages are delivered to and received from the endpoints of applications by being transmitted through channels according to certain routing.

2.2.3. Patterns for Business Analysis

Peter Coad is one of the pioneers working on analysis patterns (Purao et al, 2003; Yacoub & Ammar, 2003, p.10). He describes seven simple patterns in OO analysis and design (item description, time association, event logging, role played, state across a collection, behavior across a collection, and broadcast) (Coad, 1992). In his later book, Coad et al
summarize five categories of patterns for building object models: fundamental pattern such as “collection-worker”, transaction patterns such as “participant-transaction”, aggregate patterns such as “group-member”, plan patterns such as “plan-step”, and interaction patterns such as “peer-peer” and “sender-lookup-receiver” (Coad et al, 1995). One of the major contributions of Coad’s works is the idea of using patterns for analysis besides of design. Tine between analysis patterns and design patterns has not been clearly drawn. However, Coad’s OO patterns are referred more as analysis patterns than as design patterns by other researchers and practitioners.

After recognizing that as a result of the diversity of the patterns community, it is difficult to come up with a single definition, Martin Fowler in his 1997 book proposes his general definition of a pattern: a pattern is an idea that has been useful in one practical context and will probably be useful in others. Fowler also claims that analysis patterns should reflect conceptual structures of business processes rather than actual software implementations (Fowler, 1997, p.xv). The book proposes two categories of patterns: analysis patterns which are groups of concepts that represent a common construction in business modeling and supporting patterns which describe how to take the analysis patterns and apply them. The analysis patterns include: patterns for describing relationships that define responsibilities between parties such as organization hierarchies, organization structure, and operating scope; observation and measurement patterns for recording facts about the world such as quantity, conversion ratio and range; basic accounting patterns for describing how a network of accounts and posting rules can form an active accounting system such as account and transaction; planning patterns such as proposed and implemented action pattern; and trading patterns such as pattern.

Eriksson and Penker, in their book on UML modeling, claim that “a pattern is a generalized solution that can be implemented and applied in a problem situation, and
thereby eliminate one or more of the inherent problems in order to satisfy one or more objectives” (Eriksson & Penker, 2000, p.169). They also claim that “Patterns are found in all phases of development, from business modeling to the actual coding and testing stages. Patterns found in business models are referred as business patterns; patterns found in system design are known as architectural patterns (high-level system patterns); patterns closer to the programming level are called design patterns” (Eriksson & Penker, 2000, p.170). In their book, 26 business patterns are identified and grouped into three categories (resources and rules, goals, and processes). In the book, these patterns are described in a template, which is similar to Gamma’s template (1995). The template used by Eriksson and Penker includes: pattern name, intent, motivation, applicability, structure, participants, consequences, example and related patterns. Example A1 in Appendix A shows the structure diagram of the “contract” pattern.

Hruby in the preface of his book (Hruby, 2006, p.xii) claims that “REA (resources, events and agent) model specifies the fundamental laws of the business domain while the knowledge of these laws would radically enhances the application designer’s potential to configure business solutions without omissions and ensures consistency of software applications from the business perspectives”. Based on the REA model, he proposes a pattern map consisting of 25 business patterns in three different levels (REA structure at operational level, at policy level, and behavior level). The operation level, called the fundamental skeleton, consists of the REA exchange process, the REA conversion process and the REA value chain, which is actually a combination of the first two. The fundamental business patterns are the exchange (trade) process and the conversion (production) process, where businesses are able to add value to their activities which enable them to make profit (output outweighing input).
In his paper on DEMO (dynamic Essential Modeling of Organizations), a business modeling method (Dietz, 1994), and his later book on enterprise ontology (Dietz, 2006), Dietz proposes the concept of essential action and the pattern of a transaction. According to Dietz, an action is essential if it can only be performed by a human being because it requires the decision of a responsible person. Furthermore, based on the speech act theory by Searle (1969) and the communicative action theory of Habermas (1981), the paper claims that the meaning of every sentence or message consists of a proposition and an illocution. Four illocutionary categories are distinguished in his work: directives, commissives, statutives and acceptives. A conversation consisting of the first two categories is called an actagenic conversation where an action or an agenda is often proposed and committed; a conversation consisting of the last two categories is called a factagenic conversation where a fact or a status is set or recognized. The pattern of a transaction is a sequence of three steps: carrying on an actagenic conversation, executing an essential action, and carrying on a factagenic conversation (Dietz, 1994 & 2006). The diagram in Example A2 in Appendix A describes the Transaction Pattern using plain vocabulary instead of speech act theory terminology.

Similar to Dietz’s transaction pattern, the Action Workflow loop is developed based on the communicative action theory. (Medina-Mora et al, 1992) describe the Action Workflow approach as a design methodology to support workflow management technology in organization. They distinguish three different domains (material processes, information processes, and business processes) in which activities of an organization can be described. They also find that in the domain of business processes, people enter into language actions that have consequences for their future activities. The example in the paper is: when a customer hands a supplier an order form, there is a physical activity (transferring a piece of paper) and an information dimension (communicating a form with information about a particular set of goods, delivery instruction, etc.). They argue that the
true significance is in the business dimension: it is a request for the supplier to perform some particular actions, in return for which the customer is committed to perform other actions such as payment. Based on their observation, they propose the Action Workflow loop, which consists of four phases (proposal, agreement, performance and satisfaction), and claim that the simple workflow loop structure is both general and universal since the loop is like an atomic element and all the complex phenomena of organizations can be generated by combining these loops (Medina-Mora et al, 1992). Example A3 in Appendix A shows the Action Workflow loop.

Believing that requirements for workflow languages are indicated through workflow patterns, Aalst et al propose over 40 workflow control-flow patterns as a basis to compare different workflow languages and commercially available workflow management systems to evaluate their suitability and expressive power (van der Aalst et al, 2003; Russell et al, 2006a). These patterns are expected to form a compressive set of workflow functionality and are categorized into groups such as basic control flow patterns, advanced branching and synchronization patterns, structural patterns, patterns involving multiple instances, state-based patterns, and cancellation patterns, based on their complexity and relevance. Besides the control-flow patterns, Russell et al also identified and proposed workflow resource patterns (Russel et al, 2004a), workflow data patterns (Russel et al, 2004b), and workflow exception handling patterns (Russell et al, 2006b). As (Wohed et al, 2003) discusses, a workflow pattern is an abstracted form of a recurring situation related to the ordering of the activities in a workflow. Workflow patterns are language and domain independent, which is ideal for analysis.
2.2.4. Summarizing Patterns in Literature

As Table 2.1 in the next page summarizes, the above works vary greatly in terms of their focuses and scope. In the scope of system design, the design patterns proposed in (Gamma, 1995) and the architecture patterns and design patterns in (Buschmann et al, 1996) focus on general design techniques while the EAI/communication patterns proposed in (Linthicum, 1999; Ruh et al, 2000; Hohpe & Woolf, 2003) focus on system communication. Due to our specific interest in business patterns, further discussion on design patterns is omitted in this thesis.

In the scope of business analysis, the analysis patterns proposed in (Coad, 1992; Coad et al, 1995), the analysis patterns in (Fowler, 1997), the business patterns for UML modeling in (Eriksson & Penker, 2000), and the REA based business patterns in (Hruby, 2006) focus on general business concepts; the Action Workflow approach in (Medina-Mora et al, 1992) and the transaction pattern based on speech act theory in (Dietz, 1994 & 2006) focus on business communication; the workflow patterns summarized in (van der Aalst et al, 2003; Russel et al, 2004a, 2004b, 2006a, 2006b) are specifically used to model workflow for business process.

The above patterns for business analysis can be classified into two larger categories: patterns used to build conceptual models of multiple facets of business, and patterns used to model specific business aspects such as business communications and processes. The patterns in the first category provide a set of important business concepts and describe how to model these concepts and the relationships between these concepts. However, these patterns usually represent the concepts and their relationships in a static fashion; the dynamic view of business is lacking in these patterns.
### Table 2.1 Patterns in IS Literature

<table>
<thead>
<tr>
<th>Scope</th>
<th>Focus</th>
<th>Patterns &amp; Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Design</td>
<td>General Design</td>
<td>Design patterns</td>
</tr>
<tr>
<td></td>
<td>Techniques</td>
<td>(Gamma, 1995)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Architecture pattern and design pattern</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Buschmann et al, 1996)</td>
</tr>
<tr>
<td>System Communication</td>
<td></td>
<td>EAI communication patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Linthicum, 1999; Ruh et al, 2000; Hohpe &amp; Woof, 2003)</td>
</tr>
<tr>
<td>Business Analysis</td>
<td>General Business</td>
<td>Analysis patterns</td>
</tr>
<tr>
<td></td>
<td>Concepts</td>
<td>(Coad, 1992; Coad et al, 1995)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Analysis patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Fowler, 1997)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Business patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Eriksson &amp; Penker, 2000)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>REA based business patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Hruby, 2006)</td>
</tr>
<tr>
<td>Business</td>
<td>Action Workflow</td>
<td>Action Workflow approach</td>
</tr>
<tr>
<td>Communication</td>
<td>approach</td>
<td>(Medina-Mora et al, 1992)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The transaction pattern</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(Dietz, 1994 &amp; 2006)</td>
</tr>
<tr>
<td>Business Processes</td>
<td>Workflow patterns</td>
<td>Workflow patterns</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(van der Aalst et al, 2003; Russel et al, 2004a, 2004b,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

On the other hand, the patterns in the second category are effective in describing the dynamics such as business communication or business processes but they are not efficient tools to represent a larger picture of an organization. In the next sections, we will discuss more problems with the above patterns proposed for business analysis, formalize our research questions, and propose our approach to seeking business patterns.
2.3. Problems & Research Questions

The notions of “business pattern” and of “analysis pattern” in the literature are both referred to the patterns used for business analysis but there seems to be no common definition for both notions in previous studies. Perhaps, the most detailed description of business patterns was provided by (Eriksson & Penker, 2000, p170): "business patterns address problems within the business domain, typically analysis situations such as how to model and structure business resources that include invoices, organization, information, and so on; business patterns also address how to organize and relate business processes, business rules, corporate visions, and goals" (Eriksson & Penker, 2000, p170-171). On the other hand, the most explicit definition of analysis pattern may be Fowler’s statement: analysis patterns which are groups of concepts that represent a common construction in business modeling (Fowler, 1997, p.8).

We argue that “business patterns” and “analysis patterns” can be used interchangeably since they both refer to certain recurring phenomenon or common construction in business modeling. The notion of analysis pattern expresses the usage of the patterns and the notion of business pattern indicates the content of the patterns. In this thesis, we use the notion “business pattern” to be more consistent with the notion of “business modeling”. We also argue that the lack of more precise definitions has led to the difficulty to formalize business patterns in practice because a critical question remains unanswered: what kinds of information should be included in a business pattern?

As we proposed in the previous section, business patterns can be classified into two large categories based on their focuses. In the first category, business concepts are identified from business practice, then classified and defined in the business modeling context. For instance, most business patterns in (Eriksson & Penker, 2000) and (Hruby, 2006) describe
important business concepts and how to model these concepts using particular modeling languages. These patterns may serve as modeling manuals for business modelers; however, despite the label of "business patterns", such patterns are more like "patterns of business modeling" than "patterns of business itself". For instance, the "document" pattern in (Eriksson & Penker, 2000, p.223) defines the concept of a document and shows how to represent its relationships with other concepts such as an index, author, location, version and distribution. The distribution of a document has been modeled as a simple static concept and the pattern does not capture how a document can be distributed. This pattern captures several concepts in business domains and provides modeling techniques for these concepts. Referring to Alexander's definition of patterns (1977), the concepts are recurring "problems" and the modeling techniques are common solutions. However, we argue that a business is not just a set of concepts and the essence of business lies in its functionalities and dynamics. Therefore, a pattern which only introduces how to model business concepts can be a pattern of business modeling, but not a business pattern. A business pattern should not only cover the concepts but also business functionalities and activities. Hence, a real business pattern of a document should reveal recurring activities of how a document is created, stored, and distributed in organizations.

In the second category, the transaction pattern in (Dietz, 1994 & 2006), the action workflow pattern in (Medina-Mora et al, 1992) and the action types in (Umapathy & Purao, 2007), are dedicated to identify a recurring pattern which is found in the communications among different parties in enterprises or business transactions. The workflow patterns in (van der Aalst et al, 2003; Russel et al, 2004a, 2004b, 2006a, 2006b) are used to represent various control-flows and data/resources usage in business processes. These patterns describe how people communicate or arrange resources to conduct business activities. Although these patterns help to reveal the detailed dynamics among different parties or even the nature of such dynamics, they are lacking specific business
meanings. Therefore, these patterns may be a way to present business patterns, but are not business patterns in themselves. For example, the action workflow loop proposed in (Medina-Mora et al, 1992) consists of four phases: proposal, agreement, performance, and satisfaction. The loop has been claimed as an atom block for building or decomposing business processes. In the example (Medina-Mora et al, 1992, p.283), the researchers created a business process map of conducting pilot projects in certain organization. The process map was composed using connected action workflow loops. While a higher level loop were decomposed into several lower level loop, each loop described a specific business process, for instance, the higher level loop could be “conduct pilot project” and two of the lower level loops could be “assess applications” and “conduct analysis”. But because an action workflow loop can be used to describe any business process, without any context, a workflow loop alone cannot indicate specific business functionality and activities.

Most of the patterns in both categories are a result of analysis of the very basic unit of business domain so that each pattern is dedicated to define or to model the concept of the building block, for instance, what is a resource and how to model it. The strength of such level of analysis is to provide a very generic definition and classification of the building block of business. But as the above examples showed, patterns resulted from such analysis are too generic and lacking of business functionalities and business activities. We argue that functionalities are the most important aspect of business semantics. A business exists because it can provide certain functionalities that satisfy customers’ needs. Because these business functionalities are always realized by certain activities in organizations, business semantics should also include activities. Because there can be business rules that constrain these activities, business semantics may further include business rules (or policies). We propose that a business pattern should possess such business semantics. In other words, a business pattern should describe certain business functionalities (expected
results) and activities (dynamics to reach the results), and should be able to describe policies (rules to constrain the dynamics).

Functionality is important for patterns in general independent of fields and areas. The argument of the importance of semantics is shared by Alexander in his architectural patterns. Most of the architectural patterns have their own functionalities. The examples, from macro-level to micro-level, are country towns, ring roads, shopping street, health center, small public squares, still water, street café, small parking lots, main entrance, outdoor room, and raised flowers. But what if one continues to identify very basic building blocks? We believe that at least one can dig into the level of wood, bricks and plants and so on. But as we go into more basic components, these building blocks or so called patterns lose specific functionality. One may argue that one piece of wood can be used in many ways; but before the wood is turned into a chair or a part of the door, it has no specific functionality which people could identify and take advantage of. For this reason, the “wood” may never be included in the architectural patterns but a chair or a door could be. We claim that a necessary condition for identifying a building component as a useful pattern is: there should be some functionality attached to the component.

The above literature review and analysis lead to the following research questions:

1. What is or can be a business pattern with semantics?
2. What is the “right” level of abstraction? What information would be included in modeling a business pattern?
3. How can a business pattern be modeled?
4. How can such business patterns be used?

These questions will be explored and answered, hopefully, in this thesis.
3. APPROACHING BUSINESS PATTERNS

3.1. The Research Approach & Scope

In this thesis, we try to answer our research questions step by step with following research approach.

1) We will review the existing definitions of patterns and try to propose our own definition of business patterns, to which we expect to incorporate business semantics.

2) Based on the definition proposed, we will discuss the level of abstraction we suggest for business patterns and what kinds of information should be included. The discussion will also be based on existing literature.

3) We propose a methodology to model business patterns at our suggested level of abstraction. Instead of developing a modeling methodology from scratch, we find an existing modeling approach that we believe to be appropriate for modeling business patterns. If the modeling approach cannot completely meet our requirement for modeling business patterns, we will try to propose our modification and extension to the original method so that it can cover the missing information.

4) We will provide examples of business patterns we have identified and demonstrate how to model these patterns using our proposed modeling methodology.

5) A small case study will be used to explain how these business patterns can be used.
It should be noted that the main objectives of this thesis are to claim what can be a business pattern and to propose a methodology to model such a pattern. The intention is not to introduce a thorough set of business patterns or a complete catalogue of patterns that can be used as reference models, but to introduce an idea of "what are business patterns" and "how to represent and use them" in the context of system analysis for future research and practice. The collecting and summarizing of the patterns is out of the scope of this thesis.
3.2. Definition of Business Pattern

As the importance of semantics is shared by Alexander, we use Alexander’s definition of architectural patterns as a starting point to propose a definition of business patterns, which consists of business semantics. In (Alexander, 1977), “each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem.” Therefore, before reaching a complete definition of business patterns, one has to understand what could be problems, the environment and the solutions in the business context.

In the business context, the environment could be an organization, or part of an organization, or its interactions with other organizations or individuals. A recurring problem could be that someone or some unit expresses a similar “wish” for some change to be executed by others in the environment. A wish is an intention to reach or remain in a desirable state. In order to fulfill a wish, certain actions have to be taken to change the current state or to prevent any change to the current state. The “problem” in a business pattern is how to accomplish a certain wish under a set of constraints. The “solution” in a business pattern is a set of actions, which together are capable of fulfilling the wish.

The proposed general definition of business pattern is therefore: a description of a recurring business phenomenon where a similar wish in organizations can be fulfilled by similar activities.

Because there are always objectives behind these wishes, a business pattern should reveal these objectives which belong to the business semantics of the pattern. For example, a customer’s wish to purchase a certain product is a common business phenomenon. Such a wish indicates that the customer does not possess the product in the “original state” and
has an intention to reach a new "desirable state", in which the customer is able to possess the product. To fulfill such a wish, the customer has to communicate and interact with the vendor; the very common communication and interaction will include information query and providing, ordering, payment and delivery.
3.3. **Elements in a Business Pattern**

According to the general definition we proposed for business patterns, the idea of business patterns is not just to use certain notations syntactically to model business, but to reveal the nature of the business in a semantic level. To reveal such a nature, we need to decide what kinds of information we would include in modeling a business pattern as well as how to represent such information in a model.

Different kinds of information have been used in business modeling as our literature review suggests. The existing studies in our first category of business patterns describe a wide range of information. For instances, Eriksson & Penker’s business patterns cover three main categories of information which are *resources* and *rules, goals*, and *business processes* (Eriksson & Penker, 2000, p.173); Hruby’s business patterns are based on *resources, events* and *agents* (Hruby, 2006, p.4). The ones in the second category focus more on specific business elements. For examples, the key elements in both Action Workflow in (Medina-Mora et al, 1992) and the transaction pattern in (Dietz, 1994 & 2006) are *roles* and *communication*; the key elements in the workflow patterns in (van der Aalst et al, 2003; Russel et al, 2004a, 2004b, 2006a, 2006b) are *tasks, sequences/flows, data* and *resources*. As we already discussed in the previous sections, the common limitation of the approaches in the former category is the dynamics, and the common weakness of the latter category is the inability to describe a complete set of business semantics.

The level of abstraction and the key elements of patterns should be decided by the usefulness of the patterns in particular domains (Fowler, 1997, p.7). Wand et al (1999) argue that a business model should reflect the core business of an organization from the perspective of its main mission – the products and services that it provides to its
customers; the component of a business model shows the stable characteristics of a business and hides business implementation details. According to Wand et al, business objects together with their services and interactions can be modeled to show the stable characteristics of a business. These elements are refined as roles, services and requests/responses in the R²M approach proposed in (Wand et al, 1999 & 2000; Wand & Woo, 2008). Since the idea of business patterns is to identify and summarize recurring business phenomenon and a recurring phenomenon indeed reflects certain stable characteristics of a business, we argue that roles, services and communication should be included when modeling business patterns. In addition, we propose that the sequences and conditions of the services and communication can be included and the benefits are discussed below.

a. Business roles
b. Communications between the roles
c. Services provided by the roles
d. Sequences of the communications and services
e. Conditions for the communications and services to occur

Among the above elements, the concepts of role and communication can be directly referred to (Medina-Mora et al, 1992) and (Dietz, 1994 & 2006), which provide deep insight of the pattern of business communication or transaction. However, the activities in a business pattern are not limited to the communication between roles. Each role should have capability to perform particular tasks that fulfill the request from other roles. The concept of service we propose is similar to the concept of task in workflow and process modeling but a certain service is specifically executed by a certain role while a task in a workflow is a general description of a step in business processes and who performs a particular task is usually not specified.
We introduce the concept of sequences into our business patterns because a representation of the sequences of a set of communication and services is indeed a description of a business process. In addition, the concept of condition is expected to enrich the expressiveness of a model by adding a logical view similar to the control-flow in workflow.

Table 3.1 summarizes the proposed elements in a business pattern and compares the use of these elements in previous studies. It should be noted that though a few elements used in other studies are not employed directly in our business patterns, most of them will actually be implicitly included in our models. For examples, data and resources exchanged could be revealed in communication while data and resources processed could be revealed in services; business rules will be modeled as policies which is a special type of role in our modeling method and will be explained in details in the next section; the concept of agent could be largely represented by the concept of role; events can be identified when and where communication and services are imitated and completed; our semantics of a business pattern always indicates a goal (to fulfill a wish) and to fulfill each request sent to them is indeed the goal of related services.

<table>
<thead>
<tr>
<th>Elements in Patterns</th>
<th>Use in Other Literature</th>
<th>R²M-based Patterns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Role</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Service</td>
<td></td>
<td>√</td>
</tr>
<tr>
<td>Communication</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>Sequence</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Condition</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.4. Levels of Business Patterns

Based on the above discussion, we claim that our proposed key elements are appropriate for modeling a business pattern we defined. However, we also argue that to include all the above information in one model is challenging. Even if one did manage to model all the information in a single model, the model would inevitably become complicated because of the coverage of information. Such complexity may lead to greater effort to draw a model and larger difficulty to understand the model. In other words, the complexity would diminish the benefits of the modeling practice since the models are expected to facilitate documentation and communication.

Moreover, the examples provided in the next sections will also indicate that the degree of generalization of these information vary in different situations, where some of the information tend to be more stable than others. For instance, the set of communication and services can be quite stable for a particular business pattern but their sequence can be different in various implementations. Our explanation for such observations is that the elements as roles, services and communications show the stable characteristics of a business (Wand et al, 1999) while the elements as sequences and conditions are less stable and vary in different implementation.

To cover all the elements we propose and based on the importance of the information and the capability to generalize, we suggest that business patterns can be modeled in two levels: functional level and operational level. The functional level, also the top-level of business patterns, should include the information of roles and their communications and services; while the operation level, which can be extended from the functional level, should consist of other information such as the sequence and conditions. In the next section, we propose a methodology to describe these different levels of business patterns.
3.5. A Methodology to Model Business Patterns

The methodology to model business patterns should consist of a modeling language that defines both syntactic notations and semantics for these notations, and modeling techniques which provide rules and guidelines to use the modeling language. Because the business patterns can be described at functional level and operational level, the notations and their semantics of the modeling language should cover all the information proposed above: roles, communications and services at the functional level, and sequence and conditions at the operational level.

Based on our literature review, we claim that no single existing modeling approach is able to cover all the information that proposed for describing business patterns. Therefore, following our research approach which is introduced earlier, we adopt an existing methodology, which fits the most part of our expectation, and extend the methodology to fulfill our needs. We claim that the R²M (formerly called OOEM) modeling approach proposed by (Wand et al, 1999 & 2000; Wand & Woo, 2008) will be an appropriate starting method for modeling business patterns. As introduced in the previous section, the major elements used in the original R²M approach include roles, services and requests/responses, which cover the functional level of business pattern.

R²M is a modeling method for conceptual modeling based on ontological theory. The modeling method is designed to describe the essentials of a particular business domain. In R²M, these essentials include business roles, attributes and services of the roles, and requests and responses between different roles. Requests and responses are the interactions between different roles; services of roles are business activities which are triggered by request and, deal with the request and are responsible of the responses.
Therefore, $R^2M$ is capable of covering the information of business patterns at the functional level and will be adequate to model this level of business patterns. Using $R^2M$ concepts, a business pattern at the functional level is *a description of a set of roles, communications (namely request between roles and responses to request) and services while this set of roles, communications and services are recurring in the business domain to realize certain business functionality*.

It should be noted that $R^2M$ does not cover the sequence and conditions which are desirable at the operation level of business patterns. Therefore, other modeling notations and techniques have to be proposed in order to complete a business pattern. In the examples in the next sections, a few extended notations developed in this work will be introduced. We would like to point out here that the syntax of the notations is far less important than the semantics. One modeler could use different notations to represent the same set of objects. The notations we use are just examples since proposing a set of standard notations is not an attempt of this thesis. In other words, what is important are the modeling constructs and their relationships, not the symbols used to represent them. In the next few sections, we will describe the $R^2M$ method and provide examples of business patterns as well as how to use the extended notations to model the operational level patterns.
4. MODELING BUSINESS PATTERNS

4.1. Brief Review of R²M

R²M or Role-Request Modeling (formerly called OOEM or Object-Oriented Enterprise Modeling approach) is a modeling method designed to model work systems, which are combinations of people and resources working together to attain certain goals. Since an organization can be viewed as a group of interacting work systems, R²M can be used to model business organizations or enterprises (Wand & Woo, 2008). The R²M method is described in detail in (Wand & Woo, 1999; Wand et al, 1999; Wand et al, 2000).

Adopting an object-oriented approach, the R²M method is developed based on an ontological theory and, which is claimed to be one of the potential theoretical foundations for conceptual modeling in formation system development in (Wand et al, 1995). The ontological theory chosen for designing R²M is Bunge’s ontology (Bunge, 1977 & 1979). The fundamental concepts of ontology based object-oriented modeling are reviewed in (Wand & Woo, 1999; Evermann & Wand, 2005). Table 4.1 summarizes important R²M concepts and their relationships.

The fundamental R²M concepts are roles and communications. Roles possess attributes and services, which define a role and distinguish it from other roles. Communications are interactions between roles. Requests and responses are two types of communications. A role sends/receives requests/responses to/from another role for communications: the requests received from other roles activate its services which later may send responses back; an activated service may also send requests to other roles and later receive responses from others. While the concepts of roles and their attributes describe a static view of a domain, the concepts of services and communications represent the dynamics.
Table 4.1 R²M Concepts

<table>
<thead>
<tr>
<th>R²M Concepts</th>
<th>Explanation</th>
<th>View</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actor</td>
<td>An actor is somebody or something that can initiate and action or respond to a request for an action by another actor. Actor represents individuals.</td>
<td></td>
</tr>
<tr>
<td>Role</td>
<td>A role represents a view of one or more individuals which reflects responsibilities and capabilities of an actor. Roles have attributes and services.</td>
<td></td>
</tr>
<tr>
<td>Attribute</td>
<td>An attributes of a role is an attribute function where the arguments are the role and an observation point and the output is a value. A set of attribute functions defined over a certain domain is a functional schema which represents a class of similar things or a defined role. A set of values of the functions in the functional schema defines a state of the role.</td>
<td>Static</td>
</tr>
<tr>
<td>Service</td>
<td>Services of a role are internal transformations which change the state of the role. A service can receive/send requests/responses, but services are activated by requests.</td>
<td></td>
</tr>
<tr>
<td>Communication</td>
<td>Communications are interactions between roles. Communications can be either requests or responses.</td>
<td>Dynamic</td>
</tr>
<tr>
<td>Request</td>
<td>A request to a role is an external event for the role. Requests are a type of communications and they initiate communications.</td>
<td></td>
</tr>
<tr>
<td>Response</td>
<td>Responses are another type of communications and they are feedbacks to the initiating requests.</td>
<td></td>
</tr>
</tbody>
</table>
Figure 4.1 below shows the original $R^2M$ graphic notations proposed by Wand et al (Wand & Woo, 1999; Wand et al, 2000; Wand & Woo, 2008).

Figure 4.1   Graphic Notations for $R^2M$

Role and its Properties

Role Name

Attributes (Information)

*Interface attributes:* information shared with other roles (via communication)

*Internal attributes:* information only available to the role

Services
capabilities, actions a role can perform

Communication between Roles

![Diagram showing communication between roles](image)

Note:
The arrow shows who is requesting; information may flow bi-directionally.
Communication must involve sending information from requestor.
Communication may involve returning information from the receiver (of the request).
Besides the concrete theoretical background, the R²M method includes a set of modeling rules which help users to effectively identify and model the key aspect of a business domain using R²M concepts (Wand & Woo, 1999; Wand & Woo, 2008):

1. The scope identification rule: all aspects of the system are involved in responding to a set of specified requests.

2. The role identification rule: a role will be included if and only if it provides or requests a service as part of system activities included in the scope.

3. The service inclusion rule: a service will be included in a role if and only if it is invoked directly by at least one request in the scope.

4. The attribute inclusion rule: every attribute included in a model must be either used or affected by at least one service of the role or "shared" with at least one other role.

5. The attribute ownership rule: every attribute can be modified by the services of one role only; an internal attribute can only be accessed by the owner role.

6. The composite role rule: a composite role must have emergent properties which are not possessed by its component roles.

7. The sub-classification rule: a sub-role should be formed and showed only if it includes additional properties (attributes and/or services) with respect to its super-role; a super-role should be formed and showed only if the sub-roles have some common properties.
Example A4 in Appendix A demonstrates how to use the $R^2M$ method to model a work system. The example shows how to use $R^2M$ concepts to abstract certain domain and how to use the $R^2M$ modeling rules to construct a model with the $R^2M$ graphic notations.

In this thesis, the $R^2M$ method is adopted and extended for the purpose of modeling business patterns. We will introduce our extension for $R^2M$ in the next section. Since our focus is on how to use the extended $R^2M$ to model business patterns, further details of the above example including the steps to develop a $R^2M$ model will not be reviewed here. It should be noted that we use the proposed $R^2M$ notations in the example, but as we emphasized in the previous section, the syntactical notations are not important as the semantics of the models.
4.2. **A Template for Describing a Business Pattern**

We claim that the modeling of business patterns largely depend on three fundamental questions. 1) What is the right level of the abstraction? 2) What information should be included? 3) How should we separate the different levels?

The answers could be different depending on the objectives of modeling a pattern and the situations where special information has to be included or even emphasized. The level of abstraction and the separation of levels proposed in this thesis are not expected to become standards but will hopefully serve as reference for modeling practice.

As discussed in the previous section, we suggest that a business pattern be modeled at such a level of abstraction that it covers the following information: roles, communications between roles, services of roles, attributes of roles, sequences of communications and services, and conditions of communications and services to occur. We also argued that it is challenging and inefficient to use a single model or diagram to cover all the information we proposed for describing business patterns. In addition, the importance of the information and their capability to generalize vary. Therefore, we suggest modeling business patterns at two levels: functional level and operational level, the former covers the information of roles and their communications and services and the latter covers other information such as the sequence and conditions. The rationales to include such information and to separate them into different levels were already explained in the previous section.

In this section, we propose a template for describing these different levels of business patterns. In this template, a business pattern is described at both levels. In the functional level, a business patterns is described using plain texts in certain format, which is called
“functionality form” of the pattern. In the operational level, a business pattern is represented using R²M-based models, which is called “operational models”. Because the original R²M method does not include the sequence and conditions, a few extended notations will be introduced. As mention in the previous section, by adding sequence and conditions, a pattern can have several instances at the operational level. Therefore, a complete description of a business pattern in this thesis will include 1) a functional form for the functional level information and 2) multiple R²M-based diagrams describing the operational level patterns.

**Functionality Form (a template)**

<table>
<thead>
<tr>
<th>Pattern Name:</th>
<th>Name...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality:</td>
<td>Description of the functionality...</td>
</tr>
<tr>
<td>Roles &amp; Services:</td>
<td></td>
</tr>
<tr>
<td>role X</td>
<td>- service SX1</td>
</tr>
<tr>
<td></td>
<td>- service SX2</td>
</tr>
<tr>
<td>role Y</td>
<td>- service SY1</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>Communications:</td>
<td></td>
</tr>
<tr>
<td>e.g. role X send request R1 to role Y...</td>
<td></td>
</tr>
</tbody>
</table>

*Note: A functionality form can be represented graphically as a functional model. Examples of functional models will be illustrated in the next section along with the examples of business patterns.*

**Operational Models**

There are no generic templates for operational models because the models can be totally different in responding to different business situations. But in this thesis, operational models of a business are R²M-based diagrams, which can be seen as extensions and instantiations of its functionality form. The number of the operational models will depend on the instantiation of the pattern in different scenarios.
4.3.  Graphic Notations

Figure 4.1 shows the original graphic notations for R²M, which includes constructs such as roles, attributes, services and request/responses, but does not include the sequence of the requests/responses and conditions for the services. Therefore, we propose a few extended notations to represent such sequence and conditions. We also make some modifications to the original notations for attributes and services to better incorporate the extended notations to the models.

The following Figure 4.2 shows the modified and extended notations for modeling business patterns. The modification and extension are introduced here, as well as the motivation and benefits of such modification and extension. The examples in the next section are also expected to help understand the use of these notations.

**Modified Notations**

**Services**

Compared to the original notations, where services are listed below attributes in a role, each service is now represented by a rectangle in a role in the new notations. The use of the rectangle graphically separates services from each other so that we can later connect requests/responses directly to services, and add conditions or other connections for the services.

**Attributes**

Compared to the originals notations, where attributes differentiated as interface attributes or internal attributes are listed above services in a role, in the new notations, they are listed in the bracket right after a service name. Such notation is similar to a function with
parameters, which indicates that the information is required by the service. In many cases, a service prepares certain information which later will be used by another service in the same role. In such cases, the information has to be "stored" after it is generated and before it is used by other services. We suggest using a unique notation for this type of information to explicitly describe the information stock and flow. As Figure 4.2 shows, we suggest using a sold arrow with dot-line to connect the services where the arrow shows the information flow; these information stock can annotated in a pair of square brackets along the dot-line.

Figure 4.2 Graphic Notations for Modeling Business Patterns

Part 1: Role, Service & Attribute

A role has services and attributes.

Services are capabilities, actions a role can perform.

Attributes are information used by the role for its services and communication with other roles.

Attributes can be represented as parameters of the services. (For simplicity, some models may omit attributes.)

The sold arrow with dot-line shows the information flow between the two services; the information stock can be annotated in a pair of square brackets along the dot-line.

Policies, which include laws, industry regulations and corporate policies, are a type of special roles. Policies may not be as explicit as other roles, but as roles they are able to trigger services in other roles. E.g. a policy can be certain timing mechanism requiring a role to activate specific services (refer to Part 3 of the notations).
Part 2: Communication & Sequence

Communication between roles includes requests and response. The arrow with solid line shows who is requesting; information may flow bi-directionally. Communication must involve sending information (requests) from requestor and may involve returning information (response) from the receiver of the request.

The sequence of sending requests and responses can be represented by the sequence numbers. A request/response with a larger sequence number is always sent after a request/response with a smaller number; requests/responses with the same number can be sent in the same time. (In simple models, the sequence numbers are not necessary and can be omitted.)

Part 3: Condition for Service & Policy

A dot-line connecting two services in a role with one-way arrow and a brief annotation starting with "if" indicates a Conditional Follow logic between the two services. The service pointed by the arrow is expected to be triggered after the other service is executed while the actual triggering of the service and the timing stills depend on a specific request from other roles.

An arrow from a policy to a service means the policy triggers the service just like a role sending a request to activate the service (e.g. policy 1 and service 1 in the figure).

An arrow from a policy to an dot-line arrow, which connects two services, means the policy poses a Necessary Follow logic between the two services by sending a "request" to trigger the service pointed by the arrow after the other service is executed or activated (e.g. policy 2, service 2 and service 3).
Extended Notations

Sequence Numbers (#) of Requests / Responses

In Figure 4.2, a pair of square brackets can be found right after a request or a response. The square brackets and the number in the brackets is the new graphic notation to represent the sequence of sending requests/responses. In a particular diagram, a larger sequence number indicates the request or response will be sent after a request or a response with a smaller number. Every request will have its sequence number while the response's sequence number can be omitted if the response is sent right after the request and before other requests are sent. The examples in this thesis also use zero as a sequence number in the cases where there is no rigid sequence for a request/response regarding to other requests/responses. In other words, a request with sequence number zero means the request may be sent before or/and after any request with non-zero sequence number. For example, there are four communications (A, B, C, and D) in a model. Among these communications, B always precedes C; C always precedes D; A can happen anytime, either before or after other communications. In this example, a modeler can use sequence number 0 for A, 1 for B, 2 for B, and 3 for D.

There are other methods to describe the sequence of communications. Instead of using a number for each request and response, a modeler can describe sequence by defining the ordering relationships of the communications. In the above example, an alternative sequence notation can be: <B, C>, <C, D> or <B, C, D>. Here, <B, C> means B precedes C; <B, C, D> means B precedes C and C precedes D. Because A can happen anytime, there is no need to define its relationship with other communications. Compared to our proposed graphical notation for sequence, the alternative notation can be more flexible for a modeler to define and change the orders, especially for larger models. Despite the
limitation, we still propose the numbering method because numbers can be incorporated into diagrams and are easy to interpret in simple models.

**Conditions for Service**

In R²M, services are triggered by requests and responsible for sending requests and responses. Since we already introduce sequence notation for requests/responses, there is no need to model the sequence of the services triggered. However, some services in the same role may still have logic connections in which a service will always be triggered after another service is executed. We propose two types of logical connections: *conditional follow* and *necessary follow*. *Conditional follow* means a service, under some condition, is expected to be triggered after another service is executed while the actual triggering of the service and the timing stills depend on a specific request from other roles. *Necessary follow* means a service will always be triggered after another service is executed. The logic of necessary follow will be further discussed when we propose another extended notation. The notation of conditions for services is proposed here to describe the logic of conditional follow in business patterns.

In Figure 4.2, the notation of conditions for services is a dot-line with one-way arrow which connects two services in a role. The service connected with the arrow is the service which under some condition is expected to be triggered after the other service is executed. The condition can be briefly described using text annotation along the line.

The benefits of using this notation include:

1) Instead of referring to the sequence numbers of the requests/responses, one can easily tell the triggering sequence of the services if they are connected via this notation.

2) More importantly, for any service S pointed by an arrow, one can easily trace back and
identify the services which are directly or indirectly followed by S. These services are exactly the “necessary conditions” for S since all of them, according to the reverse logic of conditional follow, must have been executed if S is triggered.

Policy

Organizations usually operate under some policies and rules. Policies and rules can emerge from external sources (e.g. government regulation) or be internal, reflecting how the organizations prefer to operate. Policies and business rules constrain how functionalities can be realized. At the operational level, policies can determine the sequence and conditions of the communications and services. However, policies can be part of the overall business context and they predetermine how the business can operate. Hence, they might appear at the functional level and the appearance will limit the operational possibilities. In such cases, where certain policies have already been included explicitly in the overall business context, a modeler can choose a three-layer approach, which adopts two functional levels and one operational level. The top functional level may or may not include policies (depending on the context). The second functional level may include certain policies when there are higher level commonalities which are not subject to policies. In this thesis, we only use the two-layer approach, functional and operational, because the three-layer approach will still be based on the two-layer one.

In the two types of logic connections mentioned above, Necessary follow can only exist where the following service is not responsible for receiving any requests but for sending out request, because in the R²M context the triggering of a service will depend on a request if a service can receive any request. Then the logic of necessary follow seems to break R²M rules since the service is not triggered by any request but actually by another service while the service inclusion rule of R²M indicates that “a service will be included
in a role if and only if it is invoked directly by at least one request in the scope (Wand & Woo, 2008). However, an example here will reveal that the logic of necessary follow is not contradicting the rule but actually provides more insight into identifying roles.

Consider an example. Unit A, B and C together are responsible for the whole process of application approval. Each unit is responsible for evaluating one part of an application in A-B-C sequence. Whenever A receives an application, A will finish evaluating the first part and immediately passes it to B; when B finishes the second part, B will passes it to C to complete the evaluation.

In this example, every time A passes an application to B, A is sending a request to B to evaluate the second part of the application. If B immediately passes the application to C after finishing the second part, we can claim that the role of B has one service that is triggered by A’s request (receiving an application from B), takes action (evaluating the second part of the application), and finally sends a request to C (passes the application to C). But what if B does not immediately passes the application to C after finishing it because B has to keep all the applications untill the end of the day when B categorizes the applications to different piles and B, then passes them together to C?

Considering the task frequency, B now has two different tasks: one is evaluating application, which is executed every time B sends a request; the other is categorizing and sending applications, which is executed once a day. If we include them into a single service, the service will be triggered every time B receives an application from A and B will have to do categorizing and pass application to C each time. Therefore, we claim that these two tasks can be identified as two services of the role of B because they involve different actions and are executed at different frequency. In this case, the service responsible for batch work (categorizing and sending applications) fits the logic of
necessary follow we proposed. But this service is not triggered by any request from A because A only triggers the service of evaluating every time A sends a request with an application. Now we seem to have a real case of service which is not triggered by requests but other services. However, by questioning why B has to provide the batch work service, we find the explanation for the problem: B has to provide the service because some policy (here possibly corporate policy) defines the working process and the service is actually triggered by a policy which asks B to do the batch work in the end of every day.

We propose that such polices are a special type of roles, which are not responsible for receiving requests but are capable of sending requests to trigger services in other roles. In Figure 4.2, the notation for policies is an ellipse with the name/description of policy in the ellipse.

The notation can be used in two different ways. A policy, such as policy 2 in the diagram, may send a request to trigger a service when another service is activated. In such a necessary follow case, a dot-line arrow is used to link two services (service 3 and 4 in the diagram) and the arrow points to the following one (service 4); but unlike conditional follow, there is no description of conditions along the line. Instead, a request from a policy will point an arrow to the link between these two services and indicates that there is a necessary follow. Different from policy 2, policy 1 in the diagram sent a request directly to a service 2. In such case, the policy acts just as an external role which sends requests to triggered services in internal roles. An example is telemarketing, where marketing representatives actively call (send requests to) potential customers and ask them to try a product or a service. In this example, these representatives are driven by some corporate policy which demands active calls to customers.
Benefits of introducing the role of policies and the notation are: 1) This notation, together with the notation of conditions for sequence, enables us to model both types of logic connections for services. 2) The role of policies helps us to identify the "hidden" requests sent from these roles to trigger services in other roles. As result, introducing this type of role not only fulfills but also reinforces the R²M rules.
4.4. **Examples of Business Pattern**

4.4.1. **Purpose of the Examples**

In this section, we provide examples of business patterns and show how to describe each using our template. Besides the functionality forms/models and operational models, we also provide relevant notes in the examples to help explain how we identify information that is included in the functionality forms and more details of the corresponding operational models. Each business pattern is described using a single functionality form/model and multiple operational models. For illustration purposes, we included three or four operational models for each pattern. However, it should be noted that neither set of these models is expected to cover exhaustive solutions for each pattern.

The general purposes of including the examples are: 1) to show what are business patterns according to the proposed definition; 2) to demonstrate the usefulness and the usability of the proposed modeling method and its ability to deal with varied situations. Different examples are chosen to especially serve the second purposes because they vary in detail levels, development logic, notations used, and techniques involved. The first example shows how we instantiate an abstract pattern (a functional model) to develop different operational models. The second model shows how we summarize different operational models to create a generic functional model. The third example shows how we elaborate operational models in more detail.

Example 1, Customer Purchase pattern, describes a common and major business function of many enterprises: to fulfill customers purchase demand. The pattern looks at a big picture where the enterprise as a single role interacts with the customer. It is developed with deductive logic. We first identify a potential business pattern from business practices and define the functionality form by abstracting key information for the pattern. Then we
develop operational models based on the functionality form. Example 1 introduces the role of policy and demonstrates the use of the notations of policy, the logic of conditional follow and necessary follow between services, the composition and decomposition of services. Example 2 describes another common business function: Account Payable Process. However, it looks into an enterprise and reveals the interactions between different internal roles. It is developed in an inductive fashion. We first observe existing business activities, which operate differently but have the same function, and develop the operational models according to existing scenarios. Then we try to abstract the common aspect from these models to define the functionality form for the business pattern. Compared to Example 1, Example 2 shows more variations of the operational models, where some requests/responses can be redirected, or even removed; it also implies that the removal of a pair of request and response could change the workload or complexity of several services. Example 3 is a complement to the first two examples. It is developed using deductive logic, which is similar to Example 1, and it describes operation details inside an organization, which is similar to Example 2. Because the pattern in Example 3 describes a lower level of operation compared to other examples, it represents a smaller building block for the business domain.

In short, the similarities and differences in these examples help to demonstrate the usefulness and usability of the pattern modeling method and its ability to deal with varied situations. Furthermore, the business patterns in these examples will be used in later sections where we illustrate how to combine patterns and how to employ business patterns in a case study. It should be noted that in the case study, the functional form/model are omitted because they have already been included in the examples of this section.
4.4.2. Example 1: Pattern of Customer Purchase (Order to Delivery)

Functionality Form

<table>
<thead>
<tr>
<th>Pattern Name:</th>
<th>Customer Purchase (Order to Delivery)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality:</td>
<td>The patterns fulfill the customer’s purchase request by handling query, process order and payment, and realize delivery. This pattern describes exchanging for products/services in return for money.</td>
</tr>
</tbody>
</table>

Roles & Services:
- Customer (C)
  - external role, no need to model its service
- Vendor (V)
  - V provides information, such as product info. and quote
  - V provides order confirmation & fulfillment
  - V provides service to process payment
  - V provides delivery

Communications:
- C’s requests for information & V’s responses
- C’s request to buy (order) & V’s response
- V’s request to pay & C’s response
- C’s request to deliver & V’s response

Note: the listing of the above services and communications does not indicate their sequences; the actual sequences vary in different scenarios and will be represented in the operational models.

Note 1: Commercial enterprises must have some product or service that they can provide to its customers for exchange of income. Dealing with customer purchases, either products or services, is therefore a major function of any business.

Note 2: In R2M context, the customer is an external role and the enterprise can be modeled as a single internal role since in this pattern we are mostly concerned with how an enterprise as a whole fulfills the functionality.
Note 3: Any customer, who wants to purchase something, has to place an order so that the enterprise would know which product(s) the customer want to buy. The notion of product here can refer to physical goods or non-physical ones such as services. Before actually placing an order, a customer has to know some important information such as the price of a product and may include the detail description of the product, payment options, and policies for warranty. We can identify at least two pairs of request and response: 1) customer’s request for information and vendor’s response; and 2) customer’s request to order and vendor’s response which usually is the confirmation of the order. For any verified order, the enterprise will naturally ask the customer to pay for the product or service and the customer will pay if s/he wants to complete the transaction. Now we have the third pair of request and response: 3) vendor’s request for payment and customer’s response which is the payment. Moreover, for many traditional brick and mortar stores and most of the e-businesses, delivery is an important service. A customer usually has the option to choose in store pick-up or delivery by the vendor. If a customer wants delivery, we will have our fourth pair of request and response: 4) customer’s request to deliver and vendor’s response which may include confirmation and actual delivery.

It should also be noted that at the functional level, we only identify and describe the essential communications required in a generic business situation, not their sequence. In addition, there are up to eight possible communications (either request or response) some are necessary while others are not in particular business scenarios. For each set of communications we identified above, either side of the communication can be initiator in different scenarios. What communications should be included and their sequence depend on specific business scenarios and will be represented in the operational models in responding to these scenarios.
Note 4: Services can be identified after the requests and responses. Because the vendor is the internal role in which we are interested, we only need to identify the services of the vendor. According to the identified requests and responses, the vendor is responsible for three responses and one request. For each request or each response, there will always be a service. Therefore, we can identify four candidate services, each of which is expected to deal with one response or one request. But since one service can receive requests, send responses and send requests at the same time, we have to analyze the four candidate services in order to find out whether some of them can be combined into one service. In this case, we found that none of the four services can be perfectly combined into another service since each of them may be triggered at different times and deal with different issues. Although we separate these four services in this functional level, we may choose to compose or decompose some of the services in the operational level. Some of the operational models below will show such composition and decomposition with further explanation. Again here, the sequence of these services will be addressed later in the operational models.

Note 5: In this example, four operational models are introduced next. Each operational model is an instance of the functional level of the pattern of Customer Purchase and can be understood as a sub-pattern, which not only inherits the business functionality from the upper level but also is able to reveal more details of the operating aspect. By describing the sequence of the requests/responses between roles and the conditions for the services of each role, an operational model actually represents a possible way for an enterprise to realize the business functionality. In other words, an operational model could represent a scenario in business practice.
The generic functional model below is a graphical representation of the functional form. The four services identified are showed inside the role of vendor. The four sets of communications are summarized into four arrows connecting the customer and the vendor: the two-way information arrow stands for the information inquiry and feedback between the two roles; the two-way commitment arrow stands for the ordering process where the two role may negotiate back and forth; the one-way payment arrow indicates that the payment flow is from the customer to the vendor; the one-way delivery arrow suggests the delivery direction is from the vendor to the customer.

Functional Model

Figure 4.3 Functional Model of “Customer Purchase” Pattern

Just as the functional form, the functional model only covers the functional level information of a business pattern (roles, services and communications); the sequence of communication and the conditions of services are described only in the following operational models.
Operational Models

Model A

Figure 4.4 Operational Model A of “Customer Purchase” Pattern

Note: The operational model A represents a common scenario in which a vendor responds to a customer’s purchase request and completes the purchase transaction. In this scenario, the customer conducts queries for product information and places an order. In practice, the query and the order can be realized by different means, such as by visiting the store, by phone calls, or by a shopping website setup by the vendor. The basic idea is the customer actively seek information and make decisions to purchase while the vendor is responsible for provide information and channels for purchase. A customer will always need to know some information of a product, at least the price, before s/he put an order, but even after getting enough information, a customer may not decide to purchase. Therefore, as the model shows, there is a conditional follow of the service of order fulfillment, which deals with the order request, after the service of information providing.
Similarly, there is a conditional follow of the service of delivery after the service of order fulfillment because a customer wants to get the product after he put an order, but s/he may have the options of picking it up by herself/himself or asking the vendor to deliver. After fulfilling these requests from a customer, the vendor will request the customer to pay for the product, upon delivery in this scenario. Here the service of payment processing, which sends the request and collects the payment, is not triggered by customer but by a policy. More importantly, the policy requires the vendor to initiate the service right after the delivery service. In other word, the policy poses the necessary follow logic on the payment service after the delivery, as the notations shows in the model.

Since an information query, such as query for customer services, may happen after a product is ordered or delivered, the sequence number of the request is zero in the model. But other requests are given non-zero numbers which indicate their time sequence. Because every response is sent right after the corresponding requests and before other later requests, the sequence numbers for these responses are omitted in the model.
Pattern B

Figure 4.5 Operational Model B of "Customer Purchase" Pattern

Note: The operational model B describes a similar scenario as the one in model A. The only difference is the sequence of the payment and delivery. In this scenario, the policy, instead of requiring the vendor to process payment after delivery, requests the vendor to process the payment right after the order fulfillment service before executing the delivery service. Comparing these two models, the model B provides more security for the vendor and B may provide more convenience to the customer. This may be the reason that the model A is more often found in business selling low value products such as grocery and the model B is more often used for selling high value products such as computers.
Pattern C

Figure 4.6 Operational Model C of “Customer Purchase” Pattern

Note: The operational model C above is another possible scenario for the customer purchase pattern. Unlike previous scenarios, in which a customer is the role who initiates the whole transaction by sending requests to a vendor, this scenario requires a vendor to be more active. In this example, policy 2 acts similarly to the policies in model A and B. But policy 1 acts as the key that initiates the whole transaction by sending a request to the vendor and requires the vendor actively deliver products to a customer. A customer, who has not ordered the product, will be allowed to examine the product or may be allowed to try the product before the actual purchase. If a customer is satisfied by the product and decides to buy the product, s/he can send a request to the vendor for the actual purchase and the vendor will then request for the payment.
An example for this scenario will be: a sale person knocks your door and recommends some product to you; if you are interested, you may take a close look or further examine the product; if you are satisfied with the product and the price, you may choose to pay in order to keep it.

One may notice that there are only three services in the role of vendor in this model. The reason is that in this scenario a vendor will always attach product information such as functions and price to the delivery of the product so that a customer receiving the product will have adequate information upon which s/he can make the purchase decision. Therefore, the service of providing information, which is showed in the previous models, is combined into the service of delivery in this model. This is an example of composition of services. The next example will demonstrate the decomposition of services.

An organization who wants to adopt this operating model for selling products will also notice that the actual transaction is started only after a customer accepts the products the vendor delivers, which is indicates by the conditional follow logic in the model, Therefore, to complete more transactions with lower cost, the organization should deliver the products to those customers who have the highest probability to purchase. The organization could adopt marketing techniques such as data mining to find targets who are highly potential customers.
Pattern D

Figure 4.7  Operational Model D of “Customer Purchase” Pattern

Note: The operational model D, whose basic operations is based on the model B, adopts the idea of providing a free trial to potential customers. In this scenario, before the actual purchase, a customer is provided opportunity to try the samples of a product by sending a request to the vendor. If the customer is satisfied with the samples and decides to buy the product, s/he can later send a request of order, which initiates the transaction similar to the one in the mode B. An example for this scenario is the subscription of some magazines such as the Economist. A potential customer is allowed to use the phone call or the website to request the publisher to send them free magazines for a certain period of time. If the customer wants to officially subscribe the magazine, s/he should contact the
publisher later and complete the order with payment. Since the samples here can be regarded as part of the product, the delivery service now can be decomposed to two services. The first service of delivery is responsible for the samples and the second handles the rest part of the product.

4.4.3. Example 2: Pattern of Account Payable (Purchasing Order to Payment)

This pattern is developed based on the case described in (Singhvi, 1995). The article describes and compares three different practices for the payables process to demonstrate the importance and significance of process reengineering. We find that 1) three types of payable processes, which are named 3-way match, 2-way match and 1-way match, adopt different operations but fulfill the same business functionality; and 2) the two diagrams, which are included in the articles to illustrate the processes, are not formal in terms of the constructs and notations. Therefore, we propose that 1) the payable process in the case can be abstracted as a business pattern as we define in this paper; and 2) we can use our template to describe such a pattern and the functionality form and operational models together will provide a complete formal representation.

The payable process described in the case is a set of business activities covering corporate purchasing, receiving goods, and payment to the vendor. Opposite to the first example we provided in this thesis, the enterprise we concerned in the payable process pattern is not the vendor but the customer. In this business pattern, we not only describe the interactions between the vendor and the enterprise, but also reveal the interactions between different departments within the enterprise. Therefore, we do not model the enterprise as an individual role; instead we only identify the three departments mentioned in the case as internal roles: the purchasing department (P), the receiving department (R) and the accounting department (A).
Followings are the operational models we develop corresponding to the three different practices in the case:

Operational Models

Model A

Figure 4.8 Operational Model A of “Account Payable” Pattern

Functional Level Pattern: Account Payable Process
Operational Pattern: 3-way match
The operational model A describes a traditional business practice of payable process. In this scenario, the purchasing department sends a purchasing order as a request to the vendor; the vendor will fulfill the order and deliver goods to the receiving department as the response to the order; the receiving department will request the purchasing department for purchasing information (copies of the P. O.) to match the goods received; after delivering the goods, the vendor will send request to the accounting department for payment; the accounting department will request the purchasing department for purchasing information and the receiving department for receiving information (copies of the receivers) to match the invoice which is sent from the vendor for payment; if everything matches, the accounting department will process the payment to the vendor. Because the actual payment (the account payable or A/C in the case) rely on its matching to the invoice, the P. O. and the receiver, such scenario is called three-way or 3-way matching in payable process.
Model B

Figure 4.9 Operational Model B of "Account Payable" Pattern

Functional Level Pattern: Account Payable Process
Operational Pattern: 2-way match

The operational model B describes a similar scenario as the one in model A. The only difference is that when the accounting department receiving the invoice and payment request from the vendor, it does only request receiving information from the receiving
department and process the payment based on the match between the invoice and the receiver. Therefore, the request from the accounting department to the purchasing department for P. O. information is not included in model B. Apparently, the workload of both departments is reduced once a request and response is removed. The motivation to switch 3-way matching to 2-way matching, as the case indicates, is to eliminate redundancy from 3-way matching process where the P. O. has to be matched to the receiver twice, both in the receiving department and the accounting department. In 2-way matching, the accounting department is only required to match the payment or the A/C to the invoice and the receiver. However, compared to the 3-way matching where the redundancy also serves as a double check between the P. O. and the receiver, the accuracy rate resulted from the 2-way matching is challenged because even then the invoice matches the receiver, there is still possibility of incorrectness if both mismatch the P. O. In such case, the payment requested by the vendor is consistent with the goods received by the receiving department, but the goods vendor delivered may not be the right goods as the purchasing department request in the P. O. Therefore, as the case points out, in order to ensure the accuracy, the controls in receiving must be strengthened in matching the P. O. and the receiver so that the accounting department can reply on the receiver to employ 2-way matching.

To summarize the differences between model B and model A, a pair of request and response is removed: The service of providing P. O. information of the purchasing department is lightened since it currently sends response to only one other role. The service of payment process of the accounting department is also lightened thanks to the simpler matching. The service of receiving of the receiving department, however, is expected to be more complicated since the control of matching has to be strengthened.
The article proposes that instead of 3-way matching or 2-way matching, the enterprise can employ 1-way matching process, which is described in the operational model C. In the new 1-way matching, where more information technologies are employed, business processes are further modified to reduce redundancies and increase efficiency. In this
scenario, the vendor similarly delivers the goods to the receiving department, which serves as the response to the request of purchase from the purchasing department. The differences from the other two models are: 1) Not having to send a request to the accounting department for the payment in the end, the vendor could request for the payment as it deliver the goods to the receiving department. As the model shows, the request for payment is sent to the receiving department and the payment from the accounting department serves as the response to the request. 2) The receiving department will use Goods Received Notes (GRNs), instead of the original receiver, to record the receiving information. The receiving department will also requests the purchasing department to close P. O. (a P.O. may be complete fulfilled or partially closed) according to the GRNs. Once the P. O. is closed, the receiving department will send a request to the accounting department for the actual payment. 3) The accounting department will then process the payment only based on the P. O. which has been closed. This scenario is called 1-way matching because the accounting department will not be required to match multiple sources of information but only to match the payment or the A/C to the closed P. O. The key to enable such processes is the adoption of an information system with an integrated database across all the departments. More specifically, by utilizing such an information system, the enterprise is able to share information (such as P. O. opened and P. O. closed) among different units, eliminate some paper-based documents such as the invoices and copies of the P. O. and the receivers while keep the integrity of these information.

Based on the three operational models, we summarize the account payable process into a functional level pattern, referring to the Functional Model and the Functionality Form below.
Generic Functional Model: Account Payable Process
### Functionality Form

<table>
<thead>
<tr>
<th>Pattern Name:</th>
<th>Account Payable (Purchasing Order to Payment)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality:</td>
<td>The patterns fulfill the vendor’s request for payment.</td>
</tr>
</tbody>
</table>

#### Roles & Services:

- **Vendor (V)**
  - P executes purchasing
  - P provides purchasing information to R and A

- **Purchasing Department (P)**
  - P executes purchasing
  - P provides purchasing information to R and A

- **Receiving Department (R)**
  - R receives delivery
  - R provides receiving information to A

- **Accounting Department (A)**
  - A processes payment

#### Communications:

- P’s request for purchasing & V’s response (delivery)
- V’s request for payment & A’s response (payment)
- R’s request for purchasing information & P’s response
- A’s request for purchasing information & P’s response
- A’s request for receiving information & R’s response

---

**Note 1:** We identify following communication and services in the functional level of the pattern: The role of P is always the initiator, who sends a purchasing request to the vendor (V). V is responsible to deliver the goods to the enterprise and the role of R is responsible for receiving the goods ordered by V. The payment always occurs after delivery. A different role could be requested by the vendor to pay but the role of A is always
responsible for processing the payment. To reduce errors, R may require purchasing order (P. O.) information from P, who is actually the owner of P. O.; while A may require P. O. information from P and receiving information (receiver) from R.

Note 2: Since a service (service of purchasing) in an internal role (purchasing department) sends the initial request while no requests from external roles trigger the service, there must be another internal role (X) or a policy role (P) sending a request to the purchasing department and trigger the purchasing service. However, the pattern in this example is only concerned about the activities from purchasing order to payment process and X or P is omitted in the models.
4.4.4. Example 3: Case Forwarding Pattern (Push-Pull Patterns)

The Case Forwarding Pattern describes a common business practice in an organization where a role, which can be a person, a unit or a department, forwards cases to another role. The cases refer to information required by the next performing role for completing the task. They can be in different forms such as forms, textual files or lists; they can also be hardcopy documents or digitalized data. As the below functionality form and functional model shows, the pattern includes two roles, the forwarder and the receiver; the forwarder will pass the cases to the receiver after the forwarder completes its own task while the receiver, in order to complete its task, has to get the cases from the forwarder; passing the cases is the essential communication between the two roles and the actual direction of passing the cases is always from the forwarder to the receiver.

Functionality Form

<table>
<thead>
<tr>
<th>Pattern Name:</th>
<th>Case Forwarding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Functionality:</td>
<td>This pattern describes a generic business process where an internal role forwards tasks (cases) to another role. The pattern fulfills an implicit request from certain policy or business rule which requires the case forwarding process.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Roles &amp; Services:</th>
<th>Forwarder (F)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Completing its own portion of the task</td>
</tr>
<tr>
<td></td>
<td>- Passing the cases to R</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Receiver (R)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Getting the cases from F</td>
</tr>
<tr>
<td></td>
<td>- Completing its own portion of the task</td>
</tr>
</tbody>
</table>

| Communication:         | F passes cases to R               |
Although the functional level of the pattern indicates that the flow of cases is mono-directional, three operational models below illustrate different mechanisms of how to pass the information. Since these operational models are differentiated based on their “push” or “pull” mechanism, the case forwarding pattern is also titled as “push-pull patterns”
Operational Models

Model A

Figure 4.13 Operational Model A of “Push-Pull” Pattern

Operational Model: Push Pattern - Single Case

Model B

Figure 4.14 Operational Model B of “Push-Pull” Pattern

Operational Model: Push Pattern - Batch

Both the operational model A and B describe the push mechanism of forwarding information. The difference between them is the timing of push and the volume of information pushed each time. In the model A, once a single case is completed by the forwarder and is ready for the next process, the case will be pushed forward to the receiver. In the model B, each case will be stored, either piled up or well organized, after
it is completed; the forwarder will pass the cases in batch to the receiver according to some business policy. For example, the policy may require the forwarder to push a batch when the number of cases stored reaches ten; the policy may require the forwarder to push the cases only in the end of a day regardless how many cases are stored.

**Model C**

**Figure 4.15 Operational Model C of “Push-Pull” Pattern**

The model C adopts a push mechanism of passing the information. Similar to the model B, the forwarder will store cases after completing individual cases. However, the forwarder only stores the cases instead of pushing them; the receiver will send a request to pull the cases in batch according to some policy. For example, the policy may require the receiver to pull the batch of cases in the end of a day or every 2 hours.

It should also be noted that there is no “pull-single case” pattern. The receiver can not pull a single case instead of a batch because the receiver does not know when a single case is completed by the forwarder and ready for pull. If the forwarder sends a short message to notify the receiver whenever a single case is done and asks the receiver to pick up, it becomes a push single case scenario because the forwarder initiates the process.
So the key to differentiate push and pull is not who actually sends or picks up information, but who initiates the process of passing information.

The push-pull patterns described here are generic but can be used to analyze or build most information transforming processes in organizations. Whenever certain information is required to be transferred between different roles, the patterns can employed as alternative mechanisms. Moreover, the different characterizes between physical, hardcopies of documents and digitalized data may affect the efficiency the push-push mechanisms. For the hardcopies of documents, the efficiency of push and push mechanisms are close since the documents have to be physically transferred by someone; however, both batch mechanisms will reduce transferring cost compared to the push-single case mechanism while the latter reduces the average lead time at a sacrifice of cost and time in transferring individual case. On the other hand, digitalized data provides more flexibility for choosing the push-pull mechanisms since there is no need to physically transfer the documents and the data can be sent and received in electronic forms or can be shared in a database.

In this section, we have introduced our template of business pattern, described the modeling method, proposed and explained our graphic notations, and illustrated the use of the template and the modeling method with detailed examples. Next section of the thesis will look beyond the template and modeling techniques and further into the use of business patterns themselves. In the next section, we will briefly introduce a few more business patterns and discuss potential use of business patterns when a brief case study is used as demonstration.
5. USING BUSINESS PATTERNS

5.1. Combining Patterns

In the previous section, we provide three examples of business patterns and these patterns can be used individually to model some aspect of a business. In this section, we will demonstrate how to combine business patterns in order to cover more aspects of business. There are at least two ways of combining patterns: 1) composing several patterns into one model for a wider range of business functionalities, and 2) embedding one pattern in another pattern for more details of operations.

The two examples in this section demonstrate these two ways of combining business patterns. In the first example, the "customer purchase" pattern and the "account payable process" pattern are composed into one model. In the second example, the "push-pull" patterns are used to instantiate part of the "account payable process" pattern, which suggests we can embed the "push-pull" patterns into other patterns.
**Example 1: Customer-Business-Supplier Pattern**

The "customer purchase" pattern and the "account payable process" pattern introduced in the previous section describe how an enterprise or business organization interacts with its customers and suppliers. In this example, these two patterns are composed into a larger pattern named "customer-business-supplier" pattern.

**Functional Model**

![Functional Model of "Customer-Business-Supplier" Pattern](image)

We describe interactions between several internal roles within an organization in the
"account payable process" pattern, however, we model the business as a single role in the new pattern to emphasize how the business as a whole interacts with its customer and supplier. The functional model of the new pattern "customer-business-supplier" is showed above. Similar to other business patterns, the new pattern would have multiple operational models. For the illustration, we illustrate only one operational model where payment collecting occurs after delivery on the customer side and the supplier requires payment before delivery.

Operational Model

Figure 5.2 Operational Model of "Customer-Business-Supplier" Pattern
Example 2: Push-Pull Patterns in Account Payable Process

The “push-pull” patterns introduced in the previous section describe different mechanisms for information forwarding between internal roles. The patterns are generic in the sense that the information passed and the forms of information are not limited. Hence the patterns can be used to model most business processes where information is required to be passed between internal roles. In this example, the “push-pull” patterns are used to describe the information forwarding process between the receiving department and accounting department in the “account payable process” pattern.

According to the original case (Figure 2 in (Singhvi, 1995)), the receiving department in a 1-way match process requests the accounting department to make payment to the vendor when the goods are accepted; the key information, based on which the accounting department makes payment, is the closed P.O. (Purchase Order). However, the case does not provide further details of how this information is passed from the receiving department to the accounting department. Using the idea of combining business patterns, we can try employing “push-pull” patterns to model alternative solutions for this part of the “account payable process” pattern.

Alternative Model A

Figure 5.3 Operational Model A of “Push-Pull” Pattern in Account Payable Process
The model A adopts the “push-single case” pattern, where the receiving department receives goods from the vendor; if the goods match certain opened P.O., the receiving department will close P.O. according to the goods received. Each time the receiving department close a P.O., it sends the information of the closed P.O. to the accounting department. The accounting department will make payment based on the information it received.

**Alternative Model B**

**Figure 5.4 Operational Model B of “Push-Pull” Pattern in Account Payable Process**

The model B adopts the “push-batch” pattern, where the receiving department receives goods from the vendor and closes P.O. according to the goods received. Following an organizational policy, the receiving department sends the information to the accounting department in one batch; and the accounting department will make payment based on the information it received. The policy in such a scenario could be a business rule requires the receiving department to keep all the information of closed P.O. until the end of a day when it sends all the information to accounting department so that the accounting department can make payment for these closed P.O. the next day.
Alternative Model C

Figure 5.5 Operational Model C of "Push-Pull" Pattern in Account Payable Process

The model C adopts the "pull-batch" pattern, where the receiving department receives goods from the vendor and closes P.O. according to the goods received. Following certain policy, the accounting department will send a request to the receiving department for the information in batch. The accounting department will then make payment based on the information it retrieved. The policy in such scenario could be a business rule requires the receiving department to keep all the information of closed P.O. and the accounting department should retrieve all the information for a day in the end of the day so that it can make payment the next day.
5.2. Employing Patterns in a Case Study

So far we have introduced three examples of business patterns and demonstrated how to combine patterns. In this section, we will employ some of these patterns in a case study to illustrate the use of the patterns as well as the adaptability of the graphic notations we have proposed.

The case we choose is the B-Line Grocer Case (referring to the original case description in Appendix B). The major reasons of choosing the B-Line case are: 1) it is concisely described and simple to understand; 2) it covers major business functionalities for a particular online retailer, from supply replenishment to customer order processing and fulfillment.

For the B-Line case, we can employ business patterns in two ways: 1) using patterns to model business aspects which are described clearly and precisely; 2) using patterns to explore possible solution for some parts of the business that are not well described in the case or ambiguous. For the illustration purpose, instead of modeling the whole business of B-Line, we provide two examples to show how we employ the patterns in both ways.

The first example is about using the combined pattern "customer-business-supplier" pattern to model B-Line's interaction with its customer and its supplier. The second example is about exploring possible scenarios for the transferring of the Order Assembly lists and the Aggregate List between internal roles. According to the case, for each order received and confirmed by B-Line, the Order Processing Unit generates an Order Assembly list which will be used by the Fulfillment Unit to assemble and pack the order. For each day, the Inventory Unit uses an Aggregate List of all required items for the next day to check inventory availability to replenish stock. Because the case does not provide
description of how the Aggregate List is generated and transferred, we will provide our assumptions in the following examples.

**Employing Customer-Business-Supplier Pattern to B-Line**

Since the B-Line case covers B-Line’s interaction with its customers and supplier, we can first try directly employing our “Customer-Business-Supplier” (CBS) pattern. We would find: 1) the pattern overall fits the case since all the business functionality and processes, which are required for the interactions, are covered in the pattern; 2) none of our operational model provided in the previous section perfectly fits B-Line’s operations because B-Line offers more flexibility in some processes such as how a customer makes payment and how the supplier deliver supply items; 3) compared to our pattern, the case is missing the description of how B-Line makes payment to its supplier.

Therefore, we can, based on the CBS pattern, build an operational model which reflects major aspects of B-Line processes; then we can make adaptive modifications to the model to fit other aspects by removing those missing in the case description and adding elements to realize the flexibility required by B-Line. The resulting operational model is showed below with the details of the developing process omitted.

First of all, one may find most parts of the operational model are similar to the example we provided in the previous section on combining patterns, which indicates that the CBS pattern does largely catch business essentials and can be used repeatedly in business modeling.

Secondly, the model does not include the service of payment processing (to supplier) and related communications because of missing description in the case. But if considering the
case description is a result of requirement exploration, by comparing the model to our "full" pattern, one may realize that the information (for the case, payment to supplier) could be important and needs to be supplemented.

Lastly, the most significance of the B-Line’s operational model is that it includes two optional routings which are represented via "XOR" notations and are not covered in our previous examples. The first one is that the customer can make payment after delivery or choose to pay before delivery. This optional routing is actually the combination of
operational model A and B in our example of "customer purchase" pattern. The second optional routing is that some suppliers will be required to deliver the supply while B-Line will pick up the items itself from other suppliers. This example indicates that we can combine different operational models of the same business patterns to achieve operation flexibility with optional routings which is supported by our proposed notations.

Employing Push-Pull Patterns in the Case

As we already show in the example of combining "account payable process" and "push-pull" patterns, the push-pull patterns can be either used to directly model precisely define business scenarios or used to explore possible solutions for ambiguous situations. In this example, we use the push-pull patterns for the latter purpose in the B-Line case.

According to the case description, a clerk in Order Processing Unit will generate an Order Assembly List for each order and the list will be used by the packaging staff at the fulfillment unit to assemble and pack the order. It is clear that the Order Assembly List(s) is important information which is required to be transferred from the former to the latter and the latter has to rely on this information to complete its task. However, the case does not provide details of how the information should be transferred between the two roles. Therefore, we use our push-pull patterns to explore the possible solutions for the transformation of the Order Assembly List(s). The resulting solutions are the three operational models below.
In the push-single case model, the Order Processing Unit will send an Order Assembly List for each order it has processed; the Package Group in the Fulfillment Unit will receive the individual list and store them until next morning when the Package Group start packaging based on the lists.
In the push-batch model and pull-batch model, the Packaging Group does not need to store the Order Assembly Lists if it can get the list in a batch every morning when it start packaging. The difference is that in the push model, the Order Process Unit will send the lists to the Packaging Group while in the pull model, the Packaging Group will actively retrieve the lists from the Order Process Unit.

The B-Line case also mentions that the Inventory Availability Group at the Inventory Unit will receive an aggregate list of all required items for the next business day and replenish stock based on the list. However, the case does not describe who will generate the aggregate list and how the aggregate list is sent to the Inventory Availability Group. In such case, we can still use our push-pull patterns to explore viable solutions under reasonable assumptions. Below is one of the solutions for the transformation of the Aggregate List and the Order Assembly Lists. The assumptions we made are listed below the model.
Assumption 1: The Order Processing Unit sends an Order Assembly List for each order it has processed; the Packaging Group receives the individual lists and keep the lists until the end of a day when the Packaging Group will generate an aggregate list and send it to the Inventory Availability Group.

Assumption 2: The Inventory Availability Group receives the aggregate list at the end of each day, then replenishes and prepares the necessary items for the next morning. The Packaging Group will start packaging next morning using the items prepared by the Inventory Availability Group.
6. CONCLUSIONS

Our objectives of this research were to propose an idea of using business patterns in business analysis and a method for representing these patterns in the context of business modeling. While collecting and summarizing the patterns into a complete catalogue of business patterns is out of the scope of this thesis, we have focused on more fundamental questions such as what is a business pattern, what information/elements should be included in a business pattern, how should we represent/model these patterns, and how could we use these patterns.

Based on the review and analysis of previous studies, we have proposed our definition of business pattern: a description of a recurring business phenomenon where a similar wish in organizations can be fulfilled by similar activities. Following the definition, we suggest that five elements could be included in a business pattern: business roles, communications between the roles, services provided by the roles, sequences of the communications and services, and conditions for the communications and services to occur.

We also suggest that a business pattern could be modeled at two different levels, which can be represented by the template we proposed. First, the functional level (the higher level) of a business pattern can be described by a simple diagram or textual explanation. Second, the operational level (the lower level) of a business pattern can be much more detailed and represented as operational models. We have developed a modeling method, which is based on the R²M method, to represent these operation models. Our proposed method covers basic concepts and their relationships, and graphic notations with semantics. We provide examples to demonstrate how to use these concepts and notations.
in modeling business patterns. In this thesis we first include a few examples of individual business patterns we have identified. Then we illustrate how to combine business patterns into larger patterns and use business patterns in a case study.

The general benefit of exploring patterns is straightforward. Alexander claimed that "you can use this solution (the pattern) a million times over, without ever doing it the same way twice" (Alexander, 1977, p.x). In business context, "patterns make it possible to capture and describe these business-modeling problems and their corresponding solutions so that the solutions can be reused; patterns can be considered prototypes for production" (Erikson & Penker, 2000, p.169). More specifically, based on our research, we summarize the following potential uses of business patterns in conceptual modeling of business activities.

First of all, modeling business patterns helps understanding existing situation. For instance, in the example of account payable pattern, the process of summarizing different operational models into a general business pattern indicates that these different models are just different execution processes of the same business function. More importantly, by comparing the operational models, we can see clearly how the responsibilities of each business role and their interaction vary in the different processes.

Secondly, modeling business patterns can help exploring potential solutions for specific business problems, such as "how can we realize a certain function with minor resources". The process of identifying a general business pattern and instantiating it from functional level into operational level can provide guidance on how to perform certain business functionality. For instance, in both the example of customer purchase pattern and the example of push-pull pattern, we develop several operational models of the same pattern. While most of the operational models we introduced can be related to real case scenarios,
there is possibility that if we keep exploring, we may develop some creative models which are totally new to practice but maybe more effective or more efficient than the existing ones. Especially combining the effect of new technologies, such exploration is expected to be very helpful in business process reengineering.

Moreover, our B-Line case study has also implied another benefit of exploring business patterns. While the B-Line case can be seen as a requirement document, we have found some information is missing in the case description such as the passing of the Order Assembly List. Therefore, analyzing business patterns in requirement documents or descriptions can help find what is missing in business requirement.

It should be noted that for the above potential uses of business patterns, our proposed definition and representation of business patterns and the modeling methodology are the foundation. As our literature review reveals, the notion of business pattern has been used by many IS researchers and practitioners. Previous works on business patterns have their own strengths and weakness. We claim that our research is different from the others and has its own unique strengths. The uniqueness of our study lies in both of our view of business pattern and our modeling approach.

First, different from most of the existing literature on business patterns, we suggest that any individual business concept without business functionality is not the desirable level of abstraction of business patterns. Functionality is important for patterns in general independent of the fields and areas. However, functionality has been missing in business patterns in previous IS studies. One major contribution of this thesis is to recover this critical aspect of patterns with the new definition. We propose the business semantics or the business functionality as a critical aspect of a business pattern and the business semantics is expected to guide the right abstraction level in modeling practice.
Second, we also propose a unique solution of how to model a business pattern with these elements: a two-level template (functional level and operational level. Thanks to the two-level hierarchy, we have been able to identify and describe business patterns in a natural manner: general functionality with less detail versus specific operations with more information. Using such template, modelers would be able to identify and summarize patterns using either deduction or induction logic according to the situation.

Third, comparing to the original R²M method, which is lacking of sequence and conditions, and business process modeling approaches, which usually describe too much detail of business, our modeling approach enables us to take a process view of the essence of business.

As we conclude the uniqueness and strengths of our research, we realize that there are a few limitations of our current work. 1) We have not developed a set of strict rules to guide the whole modeling process of business patterns. 2) Our current research is overall a theoretical work aimed at developing the concepts and methodology. Although we have used examples and a case study to demonstrate and justify our approach, we have not employed rigorous empirical test to reinforce our research outcomes. 3) We have only employed our suggested graphic notations in simple examples which are relatively easy to read. As the models get larger, the diagrams may become much more complicated and the notations may need to be further refined. 4) Because we suggest using five elements in modeling business patterns, we might miss other types of information which may also be important in particular scenarios.

The limitations and the strengths of our work also provide future opportunities for both research and practice. The most directly related research in the future may be: setting up
modeling rules for our methodology and conducting empirical research to test the effectiveness and efficiency of the methodology. Also, as we claimed earlier that our modeling methodology enables a process view of business, we believe that an operational model of a business pattern can be directly translated to a business process model if we can in the future develop a set of translation rules between the two types of models, for instance, our operational models and BPMN models.

In this thesis, we have proposed concepts and methodology for representing and using business patterns. Similar to the architectural patterns which are expected to guide building architectures, the accumulated set of the patterns can help to guide building business. Business/system analysts and business users are expected to be the users of the modeling method and the patterns. They can employ the modeling method to summarize and record business patterns, then use the patterns to compose business functions or to describe business requirements. Therefore, the future work of collecting business patterns in practice is critical. During the iterative processes of identifying, summarizing, recording, and using business patterns, the method itself can also be further refined and improved.
REFERENCES


APPENDICES

Appendix A  Pattern Examples in Literature

Example A1: "Contract" Pattern

Note: The above diagram is adapted from Figure 7.10 in Hans-Erik Eriksson & Magnus Penker, 2000, Business Modeling with UML: Business Patterns at Work, OMG Press, John Wiley & Sons, p.216.
Example A2: The Transaction Pattern

Note: The above diagram is developed based on Figure 3.5 in Jan L.G. Dietz, 2006, Enterprise Ontology: Theory and Methodology, Springer, p20.
Example A3: Action Workflow Loop

Example A4: Using R²M Method

This example is cited from "Role-and-Request Modeling of Work Systems", Lecture Notes for course BAIT506: Business Modeling at Sauder School of Business, University of British Columbia, Yair Wand & Carson Woo, 2008. (Permitted Use)

Part A. The description of a student registration system

Students submit a request to attend a course to the registrar’s office. The registrar’s office checks if the student has the right to enroll (i.e., proper academic standing, no default on tuition) and then submits the approved requests to the faculties. If a request is rejected, the student is notified. In the faculty, students are assigned to course sections, pending availability. If space is limited, priority is given based on program requirements and credit accumulated so far. The registrar’s office is then notified as to status of requests and section allocation. Students requests not approved due to space limitations are placed on a waiting list. The registrar’s office notifies students whether their requests have been approved or not by the faculty and on their fee payable based on student’s status and courses approved.

Part B. The R2M model
Appendix B  B-Line Grocer: A Case Study

Prepared by Dr. Yair Wand and Jennifer Yue Pan
Sauder School of Business, University of British Columbia
(Permitted Use)

1. General Description

*B-Line Grocer* is a fully customized, computerized, home delivery grocery store, operating throughout the Lower Mainland area. The store offers almost everything your local supermarket offers - from fresh items like meat, seafood, produce and baked goods to non-perishable grocery staples. Its selection also includes household necessities, pet supplies, and health and beauty care items.

2. How It Work – Customer Side

- **Ordering**

Customers can place an order either by telephone or via the Web. For the *phone* option, a *customer representative* will assist shopper in identifying the right products that fit certain preferences, such as price range, brand name, size, and/or weight. When shopping via the *Web* option, customers can browse B-Line Grocer’s product list or use a search facility to find items according to their criteria. They can then click for detailed product information, select items to place in their virtual shopping cart, and check out.

Orders can be placed any time before the required delivery date. A customer can even define an “automatically repeating” order (which can be cancelled or skipped at any time).

- **Membership**

While everybody can browse B-Line Grocer’s selection of products, the store sells only to *members*. A first-time shopper, before checking out, will need to register for a membership account. With an account, shoppers can keep track the orders they have placed, change them before the due date, save past orders, place repeating orders, and obtain summaries of items purchased over a period. Furthermore, the *ordering system* can generate a *shopper’s profile*, including demographic information and information about purchase patterns, frequently purchased items and selection preferences. This information is stored in B-Line Grocer’s *customer database* and enables B-Line Grocer to provide customized services, such as a personalized web page and targeted marketing, and automatic shopping list generation.
Besides personal information, such as name, address, contact phone number, to establish an account, B-Line Grocer requires payment information via a credit card or a bank account.

- **Delivery**

“B-Line Grocer’s friendly drivers deliver your order right to your doorstep.” For both ordering options (phone and Web), customers specify delivery address and required delivery day and time. Delivery slots are filled on first-come-first-serve base. For same day delivery, the order needs to be in before 8:00 AM, but the store cannot guarantee that the order will be delivered in full.

- **Payment**

Shoppers may pay by Interac, Visa Card, check, or Cash when receiving their delivery. Alternatively, they can indicate that payment will be done automatically from their credit card or bank account.

3. The Store’s Side: Order Fulfillment

B-Line Grocer maintains its own warehouse/order assembly facility. It procures quality products from both manufacturers and wholesalers with whom they have pre-negotiated agreements.

- **Order Processing**

When an order comes in, either by telephone or via the Web, first, a clerk of the Order Processing Unit screens the order to check if it is complete and accurate. If not, a customer representative will contact the shopper to correct the order. Next, the clerk will generate an Order Assembly List for each order. The Order Assembly List will be used by the packaging staff at the Fulfillment Unit to assemble and pack the order.

- **Stock Replenishment**

At the end of each business day, the inventory availability group at the Inventory Unit will receive an aggregate list of all required items for the next business day. They will compare this list against available inventory. If the quantity available is short of the quantity needed for next day deliveries, or if the item is not of the items kept in stock (e.g. to guarantee freshness), the inventory replenishment staff will send an electronic Purchase Request to a B-Line Grocer’s partnered supplier who can provide the order. Suppliers are expected to respond within two hours as to whether they can provide the necessary items on the morning of the next day. If a supplier
cannot supply the necessary items, an alternative supplier will be contacted to assure the full needed quantity is available the next day.

In addition to ordering items necessary for next-day fulfillment of orders, the inventory replenishment group in the inventory unit issues orders for stocked items based on forecasted demand.

- Receiving Supplies

Some suppliers will deliver the required items the morning after an order was placed using their own delivery trucks. For other suppliers, the B-Line Grocer’s Logistics Unit will send a vehicle to pick up the items.

- Order Assembly

Each morning, the packaging group at the Fulfillment Unit assembles each order into three types of crush-proof containers: temperature-controlled for frozen and refrigerated items, perishable items containers, and a general use container for the rest.

After assembly, a Quality Control Specialist from the Fulfillment Unit will ensure each order is complete and items are expertly packed. If the order is found unsatisfactory, it will be returned to the packaging group. At the end of the day a Quality Control Report will be sent to the VP-Operations. A satisfactory order is passed to the loading area and a note will be sent to the Accounting Unit to prepare the invoice, including a list of delivered items, namely Fulfillment List. These documents will be available to the delivery truck driver.

- Delivery

Based on the geographical distribution of orders, the Delivery Unit will plan routes and assign shipments to truck drivers. The route plan with list of deliveries will be also sent to the Loading Group of the Fulfillment Unit. The drivers will receive each a list of orders to be delivered, in order of delivery. They will collect the shipments at the warehouse where the Loading Group staff will load the truck according to the delivery order. Finally, the delivery van is out, and you will see many happy faces.

*** The end of B-Line Case ***