

An OEM-Based Goal Modeling

Weida Wang

B.Econ., North-East Normal University, 1990

M.Econ., The University of International Business & Economics, 1998

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE IN BUSINESS ADMINISTRATION

IN

THE FACULTY OF GRADUATE STUDIES

(Faculty of Commerce and Business Administration)

We accept this thesis as conforming to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

February, 2004

©Weida Wang

Library Authorization

In presenting this thesis in partial fulfillment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Weida Wang

Name of Author (*please print*)

24/03/2004

Date (dd/mm/yyyy)

Title of Thesis: An OEM-Based Goal Modeling

Degree: Master of Science

Year:

2004

Department of Management Information Systems

The University of British Columbia

Vancouver, BC Canada

Abstract

Increasing amounts of research have been exploring methods of handling workflow exceptions by introducing goals into the workflow, as clear goals assist in managing the motivation and habits of employees and departments. However, most existing goal-driven mechanisms have adopted the assumption that the task of identifying goals is a straightforward process, without examining the processes whereby goals are determined. In order to overcome the inadequacy of methods for identifying workflow goals, we propose a goal modeling approach based on Object-Oriented Enterprise Modeling (OOEM), incorporating both the Object-Oriented Workflow Model (OOWM) and the Linguistic Negation Interpretation. In the process of constructing this goal modeling method, we first review literature from previous studies and definitions of some common terms and concepts used for this modeling. Subsequently, we present detailed rules and steps of OOEM-based goal modeling. In addition, we illustrate how this method can be applied to identify goals in business processes. Finally, a small-scale empirical study is conducted to demonstrate the usefulness and practicability of this modeling method. The OOEM-based goal modeling method is a comparatively systematic way to identify a complete hierarchy of goals in business processes, structured in four layers from the level of activities to the level of the entire system. Another novelty of OOEM-based goal modeling is its bottom-up approach: higher-level goals are identified by systematically grouping lower-level goals.

Table of Content

Abstract.....	ii
Table of Contents.....	iii
List of Tables	v
List of Figures.....	vi
Acknowledgements	vii
Chapter 1: Introduction.....	1
1.1 Motivation	1
1.2 Thesis Objectives.....	3
1.3 Thesis Outline.....	5
Chapter 2: Goal Identification Methods and Related Research Work	7
2.1 OEM and Related Research Work	7
2.1.1 Object-Oriented Enterprise Modeling (OEM)	8
2.1.1.1 Ontological Principles.....	8
2.1.1.2 Constructs of OEM	10
2.1.1.3 OEM Representation Technique	13
2.1.1.4 Modeling Rules.....	14
2.1.1.5 Advantages and Shortcomings of OEM.....	15
2.1.2 Object Oriented Workflow Modeling (OOWM).....	16
2.1.2.1 New Constructs in OOWM.....	16
2.1.2.2 OAT	17
2.1.2.3 from an OAT to an Activity Diagram	19
2.1.3 MOAP-Wf.....	21
2.2 Definition of Goals and Related Concepts	26
2.2.1 Goals in Requirement Engineering.....	28
2.2.2 Definition of Goals in Business Process Modeling	31
2.3 Concepts Related to Goals.....	33
2.3.1 Owner of a Goal.....	34
2.3.2 Classification of Goals.....	35
2.3.3 Refinement and Abstraction of Goals.....	37
2.3.4 Related Concepts	38
2.4 Existing Goal Identification Methods.....	38
2.4.1 The Goal-Based Requirements Analysis Method (GBRAM).....	39
2.4.2 Rolland's Goal Modeling Using Scenarios.....	45
2.4.3 Other Goal Identification Methods	49
2.5 Summary of Survey	50
Chapter 3: The Rules and Steps of OEM-Based Goal Modeling.....	51
3.1 Modeling Rules.....	51
3.1.1 Rule #1: Object and Service Identification Rule	54
3.1.2 Rule #2: Activity Inclusion Rule	56

3.1.3 Rule #3: The Identification Rule for Goals of Services	60
3.1.4 Rule #4: The Identification Rule for Goals of Objects	66
3.1.5 Rule #5: The Identification Rule for Goals of Systems	71
3.2 A Systematic Approach to Using the Rules	73
3.2.1 Step 1: Using OEM to Identify the Objects and the Services of Each Object and to Draw an OEM Diagram	75
3.2.2 Step 2: Identify the Activities of Each Service Using an Object Activity Template (OAT) and an Activity-Based Diagram	75
3.2.3 Step 3: Identify all the Stable States of Each Service	76
3.2.4 Step 4: Identify the Goal of Each Service	77
3.2.5 Step 5: Identify the Goal of Each Object	77
3.2.6 Step 6: Identify the Goal of the System	78
3.2.7 Step 7: Summarize the Steps in a Table	78
3.3 Summary	81
Chapter 4: Empirical Study	82
4.1 Study Design	84
4.2 Subject Backgrounds	85
4.2.1 Sample Corporations	85
4.2.2 Sample Business Processes	86
4.2.2.1 Office Tools Purchasing Process	86
4.2.2.2 Export Credit Process	87
4.2.2.3 Short Message Service (SMS) Product Process	88
4.2.3 Owners of the Business Process	89
4.3 Analysis of Results	90
4.4 Summary	94
Chapter 5: Conclusions and Future Research	96
5.1 Conclusion	96
5.2 Contributions	98
5.3 Limitation and Future Research	99
References	101
Appendix A: Consent Form	107
Appendix B: Subject Background Questionnaire	108
Appendix C: Corporation Brochure	109
Appendix D: Answer Sheet	110
Appendix E: Identifying Goals of ACME House Case Using OEM-Based Goal Modeling	112

List of Tables

Table 2.1: Mapping ontological concepts into object constructs	10
Table 2.2: An Object Activity Template (OAT) (adapted from Hui, 1997)	18
Table 2.3: Object Activity Template.....	19
Table 2.4: Schema Syntax for Goal Models [Anton, 1996].....	44
Table 2.5: Mapping Roland's concepts into OEM concepts	47
Table 3.1: An Object Activity Template (OAT) (adapted from Hui, 1997)	58
Table 3.2: Relating the steps to the rules	74
Table 3.3: OAT of a Planner.....	76
Table 3.4: Summary of OEM-Based Goal Modeling Steps	80
Table 4.1: Goals comparison table	91
Table 4.2: Identifying the objectives of business processes	94
Table App.1: OAT of an Office Clerk.....	112
Table App.2: OAT of a Warehouse	114
Table App.3: OAT of Truck Driver	116

List of Figures

Figure 2.1: OEM Graphical Constructs.....	13
Figure 2.2: Student Registration Process in OEM Model (example case of BAIT 506, Woo, 2001).....	14
Figure 2.3: Activity-Based Diagram and an OAT	20
Figure 2.4: Modified Micro-Organization Activity Processor [Guo, 2002]	22
Figure 2.5: Goal Tree Diagram [Guo, 2002]	23
Figure 2.6: MOAP-Wf Execution/Exception Handling Diagram [Guo, 2002]	25
Figure 2.7: Goal Structure [Rolland et al., 1998].....	30
Figure 2.8: Overview of GBRAM Activities [Anton, 1996]	40
Figure 2.9: The Scenario Structure	46
Figure 2.10: Overview of the Discovery Process [Rolland, 1998].....	48
Figure 3.1: OEM Diagram of ACME Warehouse example case	75
Figure 3.2: Activity-Based Diagram of "Plan Transportation and Truck Assignment"	76
Figure 4.1: the Relationship between Proportion of Problems Found and the Quantity of Evaluators [Nielsen, 1999]	83
Figure App.1: Activity-Based Diagram of "Process Withdrawal Request".....	113
Figure App.2: Activity-Based Diagram of the Service "Check Item Availability"	115
Figure App.3: Activity-Based Diagram of the Service "Process Withdrawal Request" of object Warehouse	115
Figure App 4: Activity-Based Diagram of the Service "Do Transport"	116

Acknowledgements

First of all, I am very grateful to my thesis supervisor, Dr. Carson Woo. Without his research guidance, encouragement and support, this thesis would not have been possible. Dr. Woo gave me the initial idea and he has guided me through the final version of my thesis, with 233 emails. I also wish to express my gratitude to Dr. Yair Wand and Dr. Jacob Steif for being my thesis committee members and for their insightful input and advice.

I also want to say thanks to Huijin Guo, because my thesis is based on his research, and he has given me many comments and ideas throughout the entire process.

I thank Steve Doak for his excellent editing, and Leon Qiu, Weiquan Wang and Jack Jiang for fruitful discussions on empirical study methods. I also thank all the experiment subjects who have helped me to finish my empirical study and who have given me many useful suggestions and comments.

I also want to say thank you to my dear wife, Lei Xia. She has been trying her best to understand my abstract thesis with great patience and emotion, and she has given me countless suggestions. Last but not least, I would like to thank all my fellow Masters students for their support.

Chapter1: Introduction

The present research endeavors to enhance the early stages of modeling business processes, by improving the identification of the goals of an enterprise, and the goals of particular systems used for business processes. The goals identified in this paper can be further used to construct additional goal-oriented methods, for approaching a variety of issues. The value of identifying goals in business processes, and consequently the value of this study has already been established by many observers, yet there have been few substantial advances in this arena.

This study focuses primarily on identifying different layers of goals that business systems must achieve. Specifically, the method detailed in this thesis provides procedural support for correlating goals from lower levels in managerial hierarchies to goals at the highest levels. The objective of this chapter is to establish the context for this research.

1.1 Motivation

Most of the research and practices associated with business process management have shared a common aim, to render business processes more efficient by reducing costs and time, by upgrading product or service quality, and by using other similar strategies [Bider, 2002]. Business Process Reengineering (BPR) is a prominent demonstration of this tendency, as a concept whereby organizations strive to radically change their methods of performing business activities with the help of information systems, to increase

profitability [Nishit, 2002]. Researchers and professionals have increasingly focused on business practice reengineering to streamline business process management systems.

Because it is pragmatically difficult for system designers to specify all the possible outcomes and alternatives of a given system, particularly regarding various special cases and unanticipated possibilities, exceptions to the usual operations of any business system should be expected to occur frequently [Chiu, 2000]. A comprehensive workflow management system (WfMS) should be able to automate exception-handling by supporting efforts by users to reallocate resources or to amend workflows; however, methods to uncover and resolve exceptions are not adequately addressed by current commercial products or by previous research.

Although this issue has not been extensively investigated, increasing amounts of research have begun to explore methods to handle workflow exceptions [Casati et al., 1999; Chiu, 2000; Eder and Liebhart, 1995; Hagen and Alonso, 1998; Klein and Dellarocas, 2000]. Several researchers have implemented the promising technique of introducing goals into the workflow, as goals help to understand the motivations of people and departments within an organization [Simon, 1964]. Instead of focusing on what a system needs to accomplish, goal-driven modeling methods involve an analysis of the reasons that certain functionality is needed, and how that functionality can be developed. Goal-driven methods lay out a rationale for system functionality, while also tracking different implementation alternatives and criteria for selecting from these alternatives.

At the same time, several empirical studies have also supported the view that modeling goals is a critical step in the creation of useful workflow modeling [Kueng et al. 96, Warboys 96]; however, little attention has been paid to explicitly analyze the processes by which goals are identified.

Practical experience indicates that goals are not always readily apparent. Therefore the question of their origins warrants some investigation. Goal discovery is rarely an easy task [Anton, 1996; ELEKTRA consortium, 1997].

A few methods for identifying goals have been proposed in the field of requirements engineering [Anton, 1996; Rolland, 1998]. However, in the field of business processes, this topic has not been given much attention.

1.2 Thesis Objectives

The limitations of existing goal-driven methods are associated with the assumption that the task of identifying goals is a straightforward process [Anne, 1997]. In contrast, we assume that goals are not always explicitly stated, and the process of identifying the goals requires careful coordination before it can be deemed 'straightforward'.

To bridge the gap between the widespread use of workflow goals and the limited methods for identifying those goals, we first propose a goal modeling approach based on Object-Oriented Enterprise Modeling (OOEM) [Yair and Woo, 1999], supported by the

Object-Oriented Workflow Model (OOWM) [Hui, 1997] and incorporating “linguistic negation interpretation” [Pacholczyk, 1998]. Our goal modeling method is designed to be a comparatively systematic approach to identify goals in business processes. Then, we use a real case to test whether the method can accurately identify particular goals step by step. Finally, an empirical study is conducted to show that the goals identified by this approach are the real objectives pursued by business owners.

In this goal modeling approach, OOEM has been adopted to model goals primarily because it is based on ontology, a formal model of the real world. This approach provides OOEM constructs specific and well-defined semantic systems and rules when they are used to model real world business enterprises. The OOEM object and system representations contain all of the information necessary for an analyst to understand organizational activities [Zhao, 1995]. Supplementing OOEM, OOWM is used in the present study because it addresses the inability of OOEM to capture comprehend task structures at an object level, and it instead proposes workflow constructs to supplement OOEM. As a result, a more complete model can be formulated, particularly when OOWM decomposes the service of OOEM into further detailed layers, based on specific activities conducted in an organization [Hui, 1997]. *Linguistic negation interpretation, which is proposed by Pacholczyk [1998], is then used to interpret all of the negative statements in a description of business processes into positive ones. This theory is instrumental in generalizing goals when we use OOEM-based goal modeling rules.*

In view of recent progress within academic studies and business practices involving business process management, this thesis sets out to achieve the following objectives:

- to define the domain and purpose of OOEM-based goal modeling;
- to define OOEM-based goal modeling rules and systematic steps;
- to demonstrate how an OOEM-based goal modeling method can be applied to a business process management case; and
- to conduct an empirical study to test whether the goals identified by OOEM-based goal modeling are the goals pursued by business process owners.

1.3 Thesis Outline

The next chapter of this thesis provides an overview of existing goal modeling methods and definitions of some common terms and concepts that are used throughout the thesis. In this chapter, we also review the basic theories used in the thesis, including Object-Oriented Enterprise Modeling (OOEM), the Object-Oriented Workflow Model (OOWM) and MOAP-Wf.

In Chapter 3, we present the rules and steps of OOEM-based goal modeling. In addition, we illustrate how this method is applied, to identify goals in business processes. This chapter is the core of the theory and contribution of the thesis.

Chapter 4 deals with the usefulness and practicability of the methodology proposed in Chapter 3 by conducting a small-scale empirical study. We will attempt to discover any

potential flaws in the method, and suggest solutions to these problems.

Chapter 5 concludes the thesis by reviewing the contributions made by the study, acknowledging its limitations, and suggesting future research directions.

Chapter 2: Goal Identification Methods and Related Research Work

Before presenting our OEM-based goal modeling, we would like to provide a broad overview of theoretical and practical aspects of OEM and goal modeling. In this chapter, we review major theories used in the present approach, a few prominent goal modeling methods, and the principle concepts of goal modeling. The major theories include Object-Oriented Enterprise Modeling (OEM) [Wand and Woo, 1999], Object Oriented Workflow Modeling (OOM) [Hui, 1997], and MOAP-based Workflow Models (MOAP-Wf) [Guo and Woo, 2002]. The prominent goal modeling methods include Anton's Goal-Based Requirements Analysis Method (GBRAM) [Anton, 1998], and Rolland's Goal Modeling Using Scenarios [Rolland, 1998]. The main concepts in relation to these methods and theories include goals, the owners of goals, the classification of goals, and refinement/abstraction of goals.

OEM is the theoretical foundation relied on most extensively for our OEM-based goal modeling, and therefore before we review existing goal modeling methods and related concepts, we will begin with a close analysis of OEM.

2.1 OEM and Related Research Work

As noted above, OEM is the theoretical foundation of the goal identification method developed in this study. In this chapter, OEM and its foundation, ontology, will be

briefly introduced, followed by a discussion of OOWM, an extension of OOEM. Finally, we introduce Guo's MOAP-Wf [2002] to illustrate how goals are used in a workflow.

2.1.1 Object-Oriented Enterprise Modeling (OOEM)

The OOEM methodology, derived from Bunge's ontological principles [1977], was first proposed by Wand and Woo [1999], and then further developed by Zhao [1995]. Bunge's ontology deals with methods of modeling the world, and it was later adapted for modeling processes in information systems by Wand and Weber [1990]. The ontology has further been used to develop a theoretical foundation specifically for object-oriented modeling, OOEM, by later research conducted by Wand and Woo [1999]. As the name of OOEM suggests, it involves a set of constructs and modeling rules to construct an object-oriented model of an organization.

OOEM is used in the present thesis to develop a goal identification method, because compared with other object oriented (OO) approaches, OOEM includes a more extensive understanding of how human beings perceive an organizational process. While many other OO approaches have been used for software development, OOEM is designed to provide a high level of abstraction to describe essential business activities. In this thesis, we model goals in business processes, and therefore it is more desirable to develop our modeling method from OOEM, rather than other OO approaches.

2.1.1.1 Ontological Principles

Ontology is the theoretical foundation of OOEM, and some ontological concepts are also directly used by our present OOEM-based goal modeling method. Therefore, a brief

summary of Bunge's ontology is necessary here.

Ontology is a philosophical approach to comprehending the basic traits of the world [Bunge, 1977]. He has used the following concepts and premises to construct his theory:

- The world is made of things that possess properties.
- Properties can be intrinsic or mutual to several things.
- Things can associate to form composite things.
- A composite thing possesses emergent properties. An emergent property means a property of a composite thing not possessed by any of its components.
- Attributes are characteristics that humans assign to things.
- Every property can be modeled as an attribute.
- Similar things are modeled using the same set of attribute functions: a functional schema.
- A set of attribute functions form the state of a thing. State is the value of all functions at a given time.
- Every change is tied to things and every thing changes.
- A change is modeled as an event – a state transition.

According to these basic ontological concepts, every *thing* changes and every change is a *change of things*. In Bunge's ontology, several concepts are related to these changes of things:

- Behavior: the states through which a thing traverses in time.
- Events: changes can be modeled as changes of states, in other words as state transitions.

- External event: a change of state due to the actions of other things.
- Internal event: a change of state due to the transition laws of a thing.
- Stable state: when a thing can change state only due to an external stimulus.
- Unstable state: when a thing must change state.

Bunge's ontological model has been adopted as the foundation of OEM, because it deals with systems in a comprehensive and formalized manner. Ontology offers a foundation for analyzing fashionable but obscure notions related to systems, hierarchies, structures, events, and information [Bunge, 1977].

2.1.1.2 Constructs of OEM

The fundamental constructs of OEM are *objects*, *services*, *attributes*, and *requests*. As OEM is derived from ontology, all the constructs of OEM are also borrowed from the mapping of ontological constructs. The following table briefly outlines the mapping.

Ontological Concepts	Object Concept
System	System
Thing	Object
Thing (external to the system)	External object
State	Attribute values
Transition law	Service
Interaction	Request
Functional schema	Object class

Table 2. ■: Mapping ontological concepts into object constructs

1. Objects: An *object* is a model of a substantial thing, within a problem domain, that interacts with other things. A class of objects is a set of objects that have common

properties [Zhao, 1995]. First, an object is defined as a model of a substantial thing. This reflects the underlying ontological principle that the world is composed of things. Second, the conceptualization of an object requires that a substantial thing must interact with other things in the modeling domain. Interaction with another object can be modeled as either submitting a request, or providing a service to it. These two concepts will be explained below.

Objects can be further classified into two categories: *internal objects* and *external objects*. In this study's proposed goal identification method for business processes, internal objects are different actors (e.g., departments) within the modeling domain, and external objects are customers or organizations related to the business process, such as banks and vendors.

2. Attributes: Attributes are assigned by humans as model properties of objects. An object must possess properties whether humans are aware of them or not. Attributes are the names assigned to these properties [Wand and Woo, 1999]. An attribute can be represented in the form of "attribute name + value", in which the value is the measurement of the attribute at some particular point of time. Wand and Woo have provided a simple example: a person always possesses the attribute of height, while the value of height may be 4 feet when the person is 7 years old.

A set of attribute functions is called a *functional schema* in Bunge's ontology. A thing can possess more than one attribute, thus enabling us to use a set of attributes and a set of values to model the object. Things similar in attributes can be modeled using the same functional schema.

Attributes can be further categorized into two types: *internal attributes* and *interface attributes*. Internal attributes model the intrinsic properties of a thing, while interface attributes model the mutual properties of things.

3. Request: a request is the interaction between two objects. An object communicates with another object by making a request to it. This request is executed through changing the interface attribute of the recipient, and a service is triggered.

When a request is made, the state of a recipient might necessitate changes in response to the request. For instance, when Object A sends a request to Object B, the three possible consequences of this request are as follows:

i). Object A is in an unstable state if it waits for the response to the request from Object B. Object B is in a stable state because it does not need to do anything about the request.

ii). Object A is in a stable state, while Object B is in an unstable state. In this scenario, the request sender is usually not concerned about the response. In other words, Object A simply delivers a piece of information to Object B, which Object B requires to perform a service. For example, in a corporation, the cashier department may need to send daily cash data to the accounting department. At the end of each month, the accounting department may produce a monthly report based on the daily cash data. In this case, the cashier department does not expect the accounting department to respond. Instead, the cashier department simply sends information to the accounting department and its work is completed.

iii). Both Object A and Object B are in unstable states. This is a more complicated consequence than the previous two. In this situation, when Object A sends a request to Object B, Object A must wait for the response from Object B; therefore, Object A is unstable. Simultaneously, Object B must act upon the request and provide a response to the request, which also means that Object B is unstable.

4. Service: Ontologically, a service models the state of transformation of an object. A service is invoked as a result of a change of state, and results in a different change of state. A service is invoked when a request is sent to an object. Then, the object needs to take a series of actions in order to satisfy this request. In this process, the object may send requests to other objects, including internal or external objects, to get more information to respond to the request.

2.1.1.3 OEM Representation Technique

In OEM, both incoming requests and requests sent out are represented by arrows. The objects, represented with a round-corner rectangle, are divided into the object name, the interface attribute and internal attribute, and the service name. All these constructs are clearly shown in Figure 2.1.

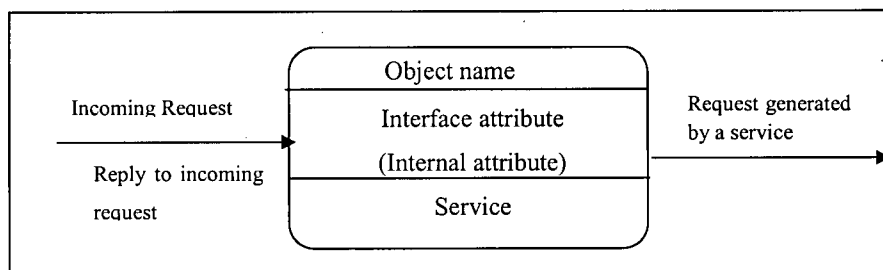


Figure 2. 1: OEM Graphical Constructs

2.1.1.4 Modeling Rules

As a necessary part of the modeling method, modeling rules provide fundamental guidelines for the model construction method. OEM modeling follows seven rules, which Wand and Woo have proposed based on particular modeling principles and ontological principles [1999]. The seven rules are:

Rule #1: The scope identification rule.

Rule #2: The object identification rule.

Rule #3: The service inclusion rule.

Rule #4: The attribute inclusion rule.

Rule #5: The attribute ownership rule.

Rule #6: The composite object rule.

Rule #7: The sub-classification rule.

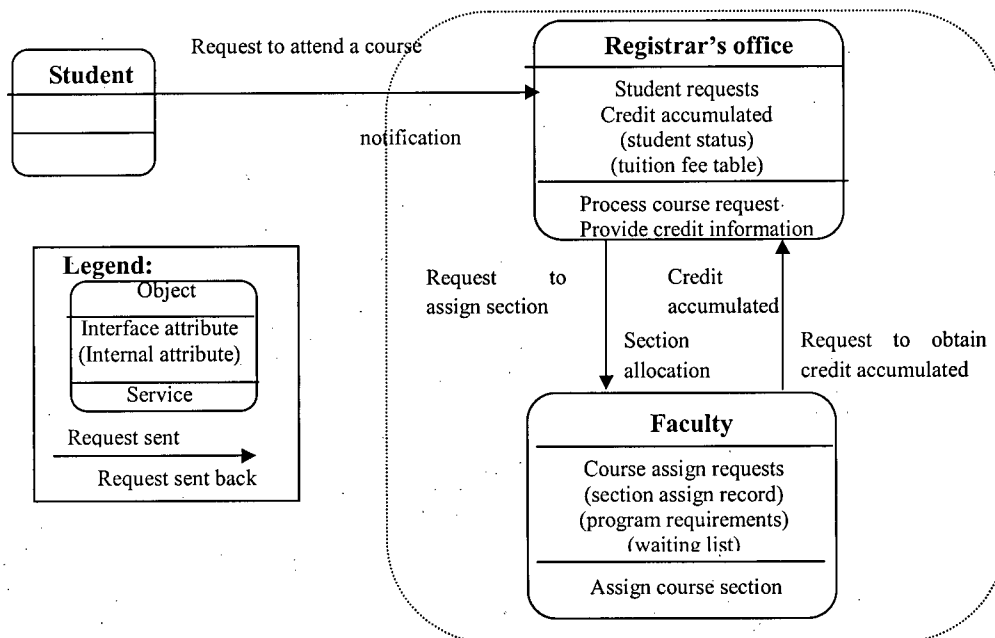


Figure 2. 2: Student Registration Process in OEM Model (example case of BAIT 506, Woo, 2001)

These seven rules are useful to identify the scope of an enterprise model, and to identify objects, services, and attributes to be included in the model. One focus of this thesis is to develop a model for identifying objects and services. Therefore, the first three rules will be the most relevant.

Figure 2.2 illustrates an example of a student registration process, using the OEM model after the first five rules have been applied.

2.1.1.5 Advantages and Shortcomings of OEM

OEM modeling constructs are based on ontology, a formal model of the real world. OEM constructs also involve specific and well-defined semantics when they are used to model real world business enterprises. Therefore, the OEM modeling method can give us all the information necessary to understand organizational activities.

The objective of this thesis is to identify goals in business processes. To do so, we need to know what things (Objects) are involved in business processes and we need to understand how objects interact by means of requests and services. We can then model business processes easily and scientifically using the OEM modeling method.

Despite its advantages in modeling organizational activities, OEM does not capture all the behavioral aspects of an organizational process. Specifically, OEM does not capture the execution order of work [Zhao, 1995]. In order to address this limitation, Zhao has suggested that conditions before and after changes may be specified for services in an Internal Object Template (IOT). Similarly, Hui [1997] has proposed Object-Oriented Workflow Modeling (OOWM), in which additional constructs, which are activities and

business rules, are introduced to extend OEM to the field of business process modeling. Furthermore, in OEM, all the actions responding to an external request are organized under one service name. This does not reflect the complexity involved in handling a request. As a result, in OEM, it is not unusual for two rather different services to have similar or even identical names. In our goal modeling method, we need to capture detailed activity of each service in order to identify goals. Hui's OOWM [1997] has addressed this limitation of OEM, therefore it is used as another theoretical foundation of this thesis.

2.1.2 Object Oriented Workflow Modeling (OOWM)

As mentioned above, OEM has its own limitations. For instance, OEM does not provide workflow specification, that is, it does not capture how organizational policies govern the activities of a process [Hui, 1997]. In order to address this limitation, Hui has introduced a workflow model that adds workflow constructs to OEM, which will be covered in greater detail in the following section.

2.1.2.1 New Constructs in OOWM

OEM deliberately excludes certain low-level details. As a result, services are not further divided into more specific details no matter how complicated the services are. Through an analysis of the relationships between services and activities, Hui has demonstrated that business rules can be spontaneously introduced in the OOWM to determine when activities should begin and end if certain conditions are met [Hui, 1997].

- Activity

An *activity* is a unit of work that forms a part of a business process [WfMC, 1997].

Activities are the basic units of operations taken by an object. A service contains an ordered set of activities $\{A_1, \dots, A_n\}$, and the mechanism of the activities is encapsulated within an object; they can access interface and internal attributes and generate requests to other objects [Hui, 1997].

In OOWM, an activity begins with an incoming request or the response to a request from another object. It terminates when one of the following conditions are met:

1. The activity generates a request to another object.
2. The activity has completed all it needs to do.

- Business rules

A *business rule* is an organizational policy that governs activities within a process. A business rule can be interpreted into pre-conditions and termination conditions for an activity. The pre-conditions are the entry criteria for an activity, and the termination conditions are the completion criteria for an activity [WfMC, 1997]. It should be noted that pre-conditions are generally referred to as *stimuli* in this thesis. As we mentioned above, a pre-condition, i.e., the condition by which an activity can be triggered, is either an incoming request or the response to a request from another object. Similarly, a termination condition, on which an activity can be terminated, is either the case where the activity generates a request to another object or where an activity has completed all it needs to do.

2.1.2.2 OAT

An Object Activity Template (OAT) is used to specify the behavior of objects. It is an extension of Zhao's Internal Object Template (IOT), as discussed by Hui [1997]. The

purpose of OAT is to decompose services into particular activities, and to address how each activity is triggered under certain conditions (pre-conditions) and how each activity is terminated under defined post-conditions. OAT is composed of three principle factors: interface attributes, internal attributes, and services. Services are classified further into pre-conditions, activity, termination conditions, requests generated, and receivers. Table 2.2 gives an example of OAT.

Object Name – Object Code							
Interface Attributes	Internal Attribute		Service				
Incoming interface attributes	Internal attribute to support service	Access code	Pre-condition	Activity	Termination Condition	Request generated	Receiver
			Pre-condition 1	Activity 1	Termination condition 1	Request generated from Activity 1	Object receiving a request generated from Activity 1
			Pre-condition 2	Activity 2	Termination condition 2	Request generated from Activity 2	Object receiving a request generated from Activity 2

Table 2. 2: An Object Activity Template (OAT) (adapted from Hui, 1997)

Usually, a single service consists of several activities, and all of its associated activities and conditions can be listed in a table in the order of activities. If a precondition of one activity arises, then that activity will be performed by an object. Similarly, if a termination condition of an activity which is being performed is met, the activity will stop.

2.1.2.3 from an OAT to an Activity Diagram

When activities and their associated conditions are represented in an Object Activity Template (OAT), it becomes possible to illustrate all of the activities constituting a service in an activity-based diagram, extended from the traditional activity-based model presented by Hui [1997]. In Hui's activity-based diagram, all interactions among objects are illustrated at the level of activity, rather than the level of service in OEM. Therefore, the Activity-based diagram can provide more detailed information about business processes.

We will use the Student Registration case introduced above to illustrate the conversion of an OAT table to an Activity-based Diagram (refer to Table 2.3 and Figure 2.3). A new legend has been added to this diagram, labeled as the "activity of other objects," to make objects that generate requests and the objects which receive requests more clear and apparent in the diagram.

Pre-Condition	Activity	Termination Condition	Request Generated	Receiver
Service 1: Process Course Request				
C1: Student request to attend a course	Activity 1: Check student's right to enroll	T1: Rejection sent back to student OR Request to assign section	Student	Student or Faculty
C 2: Response from faculty	Activity 2 : Notify students of the status of their request	T2: Student notified	Faculty	Student

Table 2. 3: Object Activity Template

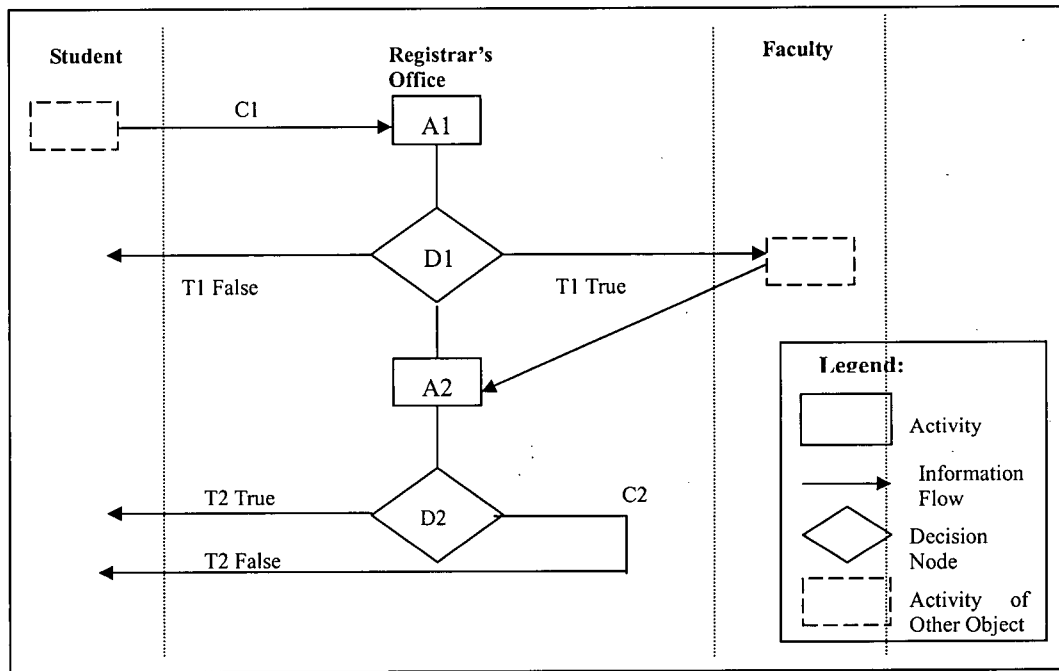


Figure 2. 3: Activity-Based Diagram and an OAT

In Figure 2.3, “T1 False” means that C1 is not satisfied (i.e. C1 is evaluated to be false at the decision node D1), and “T1 True” means that C1 is satisfied. Similarly, we use “T2 True” or “T2 False” when C2 is either satisfied or not.

2.1.2.4 The Advantages and Disadvantages of OOWM

Generally, OOWM has a few advantageous features. First, by extending OEM methodology, OOWM reflects a ‘natural view’ of organizational processes in an object-oriented context [Hui, 1997]. Second, OOWM decomposes the service in OEM into an orderly set of activities. This helps analysts understand business processes in detail. Last, OOWM separates business rules from activity definitions. This approach can improve flexibility and capability for handling exceptions, in that a change of business

rules does not require a change of tasks and activities [Guo, 2002].

On the other hand, OOWM does not differentiate between different kinds of activities based on whether the activity can lead to a stable state for objects. This differentiation is critical when people attempt to determine goals in business processes, because goals are defined as pursuing certain stable states. Thus, we need to understand more clearly about objects that generate requests and objects which receive requests in this thesis. As a result, the legend, "Activity of Other Objects," has been added to the diagram.

2.1.3 MOAP-Wf

Currently available commercial workflow systems are very rigid, in that they cannot easily meet the requirements of dynamic and fast-changing business contexts. Exception handling capabilities of these systems are very limited [Guo & Woo, 2002]. In order to address the limitations, Guo and Woo have proposed a dynamic and flexible workflow model, their MOAP-based Workflow Model (MOAP-Wf).

MOAP-Wf is constructed based on the Micro-Organization Activity Processor (MOAP), which is mobile agent architecture for modeling intelligent information systems. A MOAP represents an organizational unit, which can be a department or an employee, either within an organization or across multiple organizations. A MOAP structure consists of six different types of entities, including a Data Repository, Procedural Knowledge, Problem-Solving Knowledge, an Activity Coordinator, an Activity Agent, and Workspace. Each of the components plays a different role in knowledge representation and communication. For example, the Data Repository stores persistent information, the

workspace is the communication medium for all of the active entities, and the Activity Coordinator controls the execution of tasks. MOAPs interact with each other via requests for services. MOAP is a good candidate for modeling and building workflow systems because of its intelligent and autonomous characteristics. In addition, MOAP has problem-solving ability to handle exceptions in workflow [Guo & Woo, 2002].

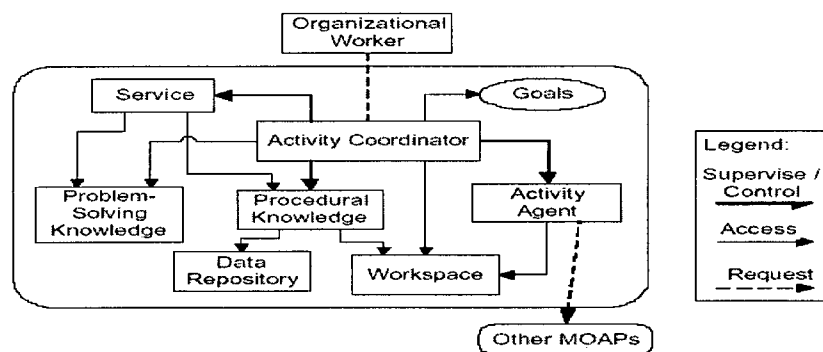


Figure 2. 4: Modified Micro-Organization Activity Processor [Guo, 2002]

To fit the fully distributed MOAP into a workflow context and to integrate it with the OEM Object concept, MOAP-Wf first proposes two new constructs, *goal* and *service*, to its unit, the MOAP. The modified MOAP architecture is illustrated in Figure 2.4.

The definition of a goal will be extensively discussed in Section 2.2 of this thesis. In a MOAP context, a goal is informally defined as a set of future states of business objects that a MOAP may achieve by executing a set of tasks or requests from other MOAPs [Guo, 2002]. A goal can be treated as the MOAP's intention and the 'reasons' underlying a work activity. Introducing goals into the MOAP helps to guide the design of workflow activities, to develop criteria for handling other MOAPs' requests, and more importantly, to work out the criteria to handle exceptions.

In MOAP-Wf, a goal is decomposed and organized in a goal-tree as illustrated by Figure 2.5. The goals at a higher level can be achieved by fulfilling sub-goals at a lower level. The sub-goals at the lowest level, which cannot be further decomposed, are called *leaf-goals*. There are only two types of relationships between sub-goals in Guo's goal tree, which are AND-relationships and OR-relationships.

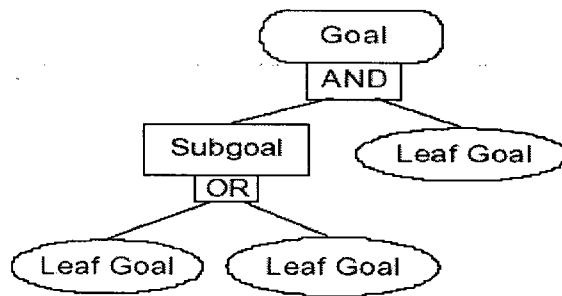


Figure 2. 5: Goal Tree Diagram [Guo, 2002]

Service is another construct introduced in the MOAP-Wf model. A service is the state of transformation of an object, from an ontological point of view. A service comprises a series of actions performed by an object with the purpose of satisfying a request [Hui, 1997; Wand and Woo, 1999]. Externally, the service of a MOAP models one request from another MOAP and is the channel for inter-MOAP communications. Internally, the service is guided by the goal and requires particular given tasks and rules to reach the goal.

The third new construct in MOAP-Wf is *Controller MOAP*. This concept is borrowed from the “controller object” introduced in conjunction with OOWM [Hui, 1997]. The controller MOAP oversees and controls an entire workflow process, to clearly identify what does and does not need to be controlled in the process. Therefore its goal reflects the

goals of the entire process. As MOAP is a distributed system, controller MOAP acts to bridge the gap between MOAP's distribution and the overall business process objectives.

A defining characteristic of the MOAP-Wf is its multi-tiered, hierarchical structure. In this hierarchy, a Controller MOAP only communicates with those MOAPs at the same level, i.e. the Controller MOAP does not send requests to or control MOAPs at other levels, whose independence and autonomy are maintained [Guo, 2002]. The hierarchy also enables multiple levels of abstraction, with higher level MOAPs remaining unaffected by the execution details of lower level MOAPs from which they request services [Guo, 2002].

Another important feature of this leveled hierarchy is that it enables multiple levels of exception handling. In MOAP-Wf, tasks and exceptions are first executed at the level where a request is initially generated. As mentioned previously, all the task executions by one MOAP are encapsulated and hidden from other MOAPs and from higher level Controller MOAPs. Exceptions are first handled at the level of the MOAP where the exceptions have occurred with one MOAP; thus they do not affect the operations of other MOAPs. If the exceptions cannot be solved, the correspondent MOAP generates an error report and replies to the immediate upper-level MOAP. The immediate upper-level MOAP then attempts to solve the problem using its own rules. This process continues until a solution is found or until the topmost level Controller MOAP is reached [Guo and Woo, 2002].

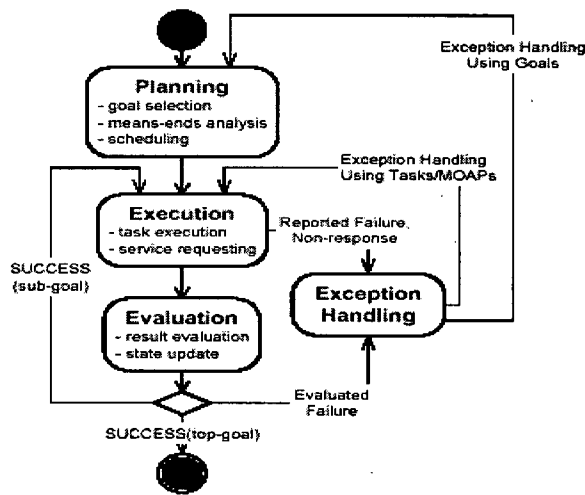


Figure 2. 6: MOAP-Wf Execution/Exception Handling Diagram [Guo, 2002]

The logical execution and exception handling consists of four stages, including *planning*, *execution*, *evaluation*, and *exception handling*, as illustrated in Figure 2.6. At different stages, different types of exceptions can be handled.

1. **Change of Workflow Sequence:** This type of change or exception is usually caused by changes in the preconditions and constraints of a workflow process, which in turn change the sequence of goal accomplishments.
2. **Evaluated Goal Failure:** An evaluated failure is one in which a goal is not achieved because one or more sub-goals fail. If the sub-goals are connected in 'AND' relationships, failure of any of the sub-goals can result in the failure of a parent goal. If the sub-goals are linked in 'OR' relationships, the parent goal will 'fail' if and only if all of the sub-goals fail.
3. **Non-Response / Unavailable MOAP Node:** This type of exception occurs when a MOAP does not respond to a request in a timely fashion or when a MOAP is not

available. In both cases, the requesting MOAP will try to find other capable MOAPs to provide services.

4. **Reported Task/Service Failure:** This type of failure occurs at the execution stage, and can be further divided into two subtypes: *service failure* and *task failure*. A service failure occurs when a MOAP successfully responds to service requests, but fails to meet the requirements of the service. To solve such an exception, the requesting MOAP will try to find other capable MOAPs, if any, or execute the tasks itself if it can. In contrast, a task failure arises within a MOAP when a task execution fails to achieve the desired goal. In this case, the MOAP will try other tasks or means of which it is capable.

MOAP-Wf is more powerful than other workflow management systems for handling exceptions within business processes. The introduction of goals makes MOAP-Wf more flexible and adaptive when exceptions happen. However, the introduction of goals is also where some of the limitations of MOAP-Wf arise. The MOAP-Wf model advocated in this thesis is based on the assumption that business goals can be readily captured and modeled [Guo, 2002]. However, the identification of initial goals in business processes is not always that simple or straightforward.

2.2 Definition of Goals and Related Concepts

Human action is primarily driven by goals [Scherer/Zolch, 1995]. In other words, humans have targets, wishes, desires and purposes, which they try to achieve. In the case of business processes, objectives motivate particular activities, and help to identify other

activities that should be avoided. Clear definitions of the goals to be accomplished by a modeling process can be advantageous in a number of ways:

- they help to manage the process of selecting from various design alternatives;
- they help to evaluate the operating quality of a business process; and
- they make it easier to comprehend the organizational changes that accompany a business process redesign (BPR).

The concept of goals is prevalent in the field of Information Technology (IT), Information Systems (IS), and business processes, particularly as it is used by various goal-driven modeling methods [Kavakli, and Loucopoulos, 1998]. Goal-directed, goal-oriented, and goal-based strategies are also frequently used by many researchers to develop modeling methods [Checkland and Holwell 1998; Nishit, 2002; Mylopoulos et al., 1999; Kueng, 1996].

At the beginning of the process of modeling business processes, it is necessary to define what a *goal* is. According to the *Longman Dictionary of Contemporary English*, a goal is an aim or purpose, a position or object one wishes to reach or obtain [Longman, 1995]. More specifically in relation to technology, a goal is a collection of future states of affairs, where a state is a set of variables and values that those variables have at a given instant [Weir, 1984]. More commonly, definitions of a goal can be found in requirement engineering and business process modeling. We first review definitions of goals used in

previous studies, and then present our own definition based on this research. We will also touch on other expressions of the word 'goal'.

2.2.1 Goals in Requirement Engineering

The concept of goals were used in the area of requirement engineering earlier than in the area of business processes, and therefore the definition of *goals* that have arisen in requirement engineering can assist in formulating a definition of *goals* in business process.

Requirement engineering (RE) is primarily concerned with producing a set of specifications for software systems that satisfy the criteria set by their designers and users, and that can be effectively implemented, deployed and maintained. As noted above, goals are used quite commonly in requirement engineering (RE). A lot of researchers have therefore developed definitions of goals in so called *goal driven requirements engineering approaches*.

According to Dardenne et al. [1993], in their introduction of the Knowledge Acquisition in automated Specification (KAOS) approach to requirement engineering, a goal is defined as “a non-operational objective to be achieved by the composite system.” A *non-operational objective* is an objective that is not formulated in terms of objects and actions available to some agent in the system. In other words, a goal as it is formulated

cannot be established through appropriate state transitions under control of one of the agents. *Objectives* in KAOS are not defined in further detail.

A contrasting definition for goals has been provided by Anton, in her Goal Based Requirements Analysis Method (GBRAM) [1996], where goals are defined as targets for achievement, which provide a framework for a desired system. According to this research, goals are high level objectives of businesses, organizations and systems, and they express the rationale for proposed systems and guide decisions at various levels within the enterprise. The maximization of corporate profits is a notable example of high-level enterprise goals.

According to Rolland et al. [1998], a goal is defined as something that some stakeholder hopes to achieve in the future. The structure of a goal according to this model is displayed in Figure 2.7. Clearly, a goal is associated with a verb and involves one or more parameters (multiplicity is shown by a black dot). It is expressed as a clause with a main verb and several parameters, where each parameter plays a different role with respect to the verb. There are four types of parameters (shown in the grey boxes). Whereas only a single beneficiary is included in this model, *target*, *direction* and *way* all are separated into subtypes. The following sentence is an example of the structure of a goal.

'Provide (cash) Object (to our bank customers) Destination (with a finger print based

ATM) Means'.

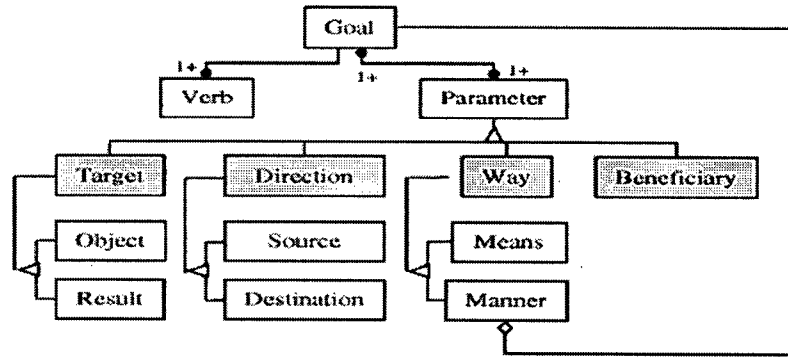


Figure 2. 7: Goal Structure [Rolland et al., 1998].

Rolland's definition is defined for software engineering, so the goal is defined in very specific detail. This definition might be appropriate for software engineering, but obviously it is not ideally suitable to be used in our goal modeling method. Otherwise, the goals we model would be too strict and too specific for possible exception handling.

Kaindl [2000] has considered goals as basic objectives that can be achieved through the execution of a scenario. Goals are viewed as partially specified states of the world that a user participating in such a scenario considers as desirable.

Generally speaking, the concept of goal is clearly defined in the field of requirement engineering; however, this approach involves a few drawbacks for the current study. First, many studies have considered the hierarchical standing of goals to be an absolute value [Dardenne et al, 1993]. Goals are usually defined as system goals (high-level goals)

without considering that goals can be situated in hierarchical structures, in which one high-level goal could be another level's detail. Second, goals are defined as either system goals or private/individual goals, and there is no mention of the goals of communities or divisions [Dardenne et al, 1993; Anton, 1996; Rolland et al, 1998]. In business processes, however, the division of goals is necessary to successfully complete some workflow processes. Therefore the primary task of the present study is to divide goals into the goals of various objects. Inasmuch as Kaindl's definition accords with Bunge's ontology, we have borrowed some ideas for our definition, to assist in constructing a hierarchy of goals, i.e., to achieve high level goals through the execution of lower level goals [2000].

2.2.2 Definition of Goals in Business Process Modeling

The concept of goals appeared in the field of business processes later than it became prominent in requirement engineering. As these two fields are highly related, several researchers have borrowed definitions of goals directly from studies in requirement engineering [Kueng and Kawalek, 1996]. However, some researchers have derived definitions of goals more directly from the perspective of business processes.

According to Narendra's goal-based workflow approach [2000], a goal is defined as an end that needs to be achieved, for the sake of meeting some customer's requirements. Similarly, a goal can also be specified as 'the desired state of the process domain' [Craven and Mahling, 1995].

Goals are targets that either an overall process or individuals engaged in a process strive to achieve [Gary and Lerch, 2000]. Therefore, goals can be further classified into process-level goals and individual-level goals according to this definition.

In Kavakli/Loucopoulos's goal-driven business process modeling approach [1998], goals in business processes imply a hierarchical structure, whereby goals with specific roles constitute refinements of higher-level goals that ultimately make up the more general business goal fulfilled by a given business process.

The various ways of defining goals discussed above are either too general or lack a hierarchical structure that satisfies the requirements of the current study. Kavakli/Loucopoulos's definition has made some progress by introducing the concepts of individual role goals and high-level goals, but it still fails to clearly address each goal level within a hierarchical structure.

In the present paper, goals are defined as the stable states of object, which includes the stable state of the related component(s) and laws. The laws govern which component(s) will be included by executing a set of tasks. Objects abide by the laws that limit the possible states and state transitions (state laws, transition laws). Specifically, a goal involves a hierarchical structure whereby the component goals at lower levels constitute refinements of goals at higher levels, which ultimately constitute the goals of entire systems, to be fulfilled by particular business processes. The goals of a department

will be added to the goal hierarchy, to account for the drawbacks inherent in the definitions of goals found in requirement engineering studies. More importantly, every layer of a goal is clearly addressed, such that the drawback of Kavakli/Loucopoulos's definition is overcome. Each layer of goals will be explained in detail in the next chapter.

We have noticed so far that some words have frequently been used to explain the word *goal* in various sources, like purpose, aim, target, objective, destination, etc. In fact, these words are also used as synonyms of goals in various goal-based modeling approaches. For instance, RM-ODP Enterprise Language uses *purpose* and *objective* instead of *goal* [FDIS 15414, 2002].

In sum, scholars in the field of requirement engineering and business processes have both exerted extensive efforts to define goals. However, they all share various limitations. We have therefore refined our own definition to overcome these limitations. In this thesis, we define goals as stable states that business units attempt to achieve, through the coordinated operations of lower level units of the business. The business units may be different layers of business processes. More specifically, business units are activities, services, objects, and systems, from the lowest level unit to the highest one.

2.3 Concepts Related to Goals

This section introduces several concepts closely related to goals, including the *owner* of a goal, the *classification* of goals, refinement and abstraction of goals, and other related

concepts. These concepts are all basic constructs in goal modeling, and are also used in our OEM-based goal modeling.

2.3.1 Owner of a Goal

An important aspect of a goal is to know whose goal it is. Different methods can be used to identify the different kinds of owners in relation to particular goals.

The theory of KAOS involves distinguishing between *private* and *system* goals [Dardenne et al 1993]. In KAOS, private goals are goals of individual agents, whereas system goals are the goals of the system as a whole.

Alternatively, Cooper [1996] has distinguished between personal, corporate, practical, and false goals. According to him:

- personal goals are goals of individual users, connected with their self-esteem and their interests in the system they use;
- corporate goals are goals of an organization as a whole;
- practical goals are goals that individual users must achieve to satisfy corporate goals; and
- false goals are goals that are imposed on users by a system itself, and that are components of goals that the developers of the system hope to satisfy.

In accordance with the goal modeling method developed in this thesis, based on OEM and OOWM, business processes can be separated into four owners: activity, service, object, and system, ordered from the lowest level to the highest. The goals arising at each of these four owners must be identified, but notions of goal owners discussed in

previous studies have generally distinguished only between personal/individual goals and business/corporate goals.

2.3.2 Classification of Goals

In the literature from previous studies, scholars have constructed a variety of classification systems for goals.

Dardenne et al. [1993] have presented a classification schema for goals in conjunction with the system of Knowledge Acquisition in automated Specification (KAOS). KAOS involves classifying goals according to the conditions that are the targets of these goals. In this classification system, goals fit into three types: *achievement*, *maintenance* and *avoidance* goals. An achievement goal is satisfied when a target condition is attained. A maintenance goal arises when the target condition is already established, and it is satisfied as long as the condition remains stable. An avoidance goal is satisfied if its target condition remains false or absent. Similarly, the Goal-Based Requirements Analysis Method (GBRAM) also differentiates between achievement goals and maintenance goals. According to this method, achievement goals are objectives of an enterprise or system. Maintenance goals, on the other hand, are those goals which are satisfied while their target condition remains constant or true. This classification is useful when it comes to operationalizing a goal as actions a system must perform.

A method of differentiating goals that is more commonly used by researchers in business processes involves identifying whether goals are *functional* or *non-functional*.

Kueng and Kawalek [1997], for example, have not explicitly defined *functional goals* and *non-functional goals*, but rather they have illustrated the distinction using an insurance example. A functional goal may involve “selling insurance”, while a non-functional goal may be ‘with politeness and punctuality’. Functional goals must be defined for every business process, whereas non-functional goals can be defined for an organizational unit, a company, or even for a whole society.

In a similar manner, Craven and Mahling have categorized goals as explicit goals and implicit goals [1995]. An explicit goal of a workflow is its objective or end-product, such as the delivery of a functional software system. An implicit goal is the successful achievement of the explicit goal in terms of timeliness and quality, or in compliance with other measurements.

KAOS and GBRAM both involve focusing on the implementation or design of systems; they put too much emphasis into representing integrity constraints. In contrast, the OEM-based goal modeling in the present study operates at the conceptual level, and therefore classifications from the previous theories are not suitable for the present modeling method.

The distinctions between functionality and non-functionality and between explicit and implicit features are not directly used in this thesis to classify goals. Although the concepts are relevant to the present study, a categorization scheme based in linguistics

theory, i.e. negation interpretation [Pacholczyk, D et al, 1998], is used because it provides additional refinements to derive goals in OOEM.

2.3.3 Refinement and Abstraction of Goals

Goal refinement is the process through which goals are broken down into sub-goals. Goal abstraction proceeds in the opposite direction by providing super-goals. Goals can be refined into sub-goals by determining how these goals should be achieved, while super goals are found by ascertaining why particular goals are sought [Lamsweerde, 2000].

In other words, goal refinement is a top-down method, in which the highest level goal is identified first and decomposed into lower level goals. On the other hand, goal abstraction is a bottom-up method, in which lower level goals are first identified, and then abstract upper level goals are derived from them.

In the present thesis, the bottom-up method is used, i.e. we begin with the lowest level goals, and the concept of goal abstraction is extensively used to reach higher level goals. On the other hand, requests from customers or users are acknowledged from the beginning of the process of identifying goals. As these requests are the objectives the whole system is designed to achieve, goal refinement is also used in this thesis to a limited extent.

2.3.4 Related Concepts

In literature from previous research that has investigated goals, terms such as *super-goal* (parent goal), *sub-goal*, and *leaf goal* have been used frequently. Usually, super-goals (parent goals) are defined as the goals exactly one level higher than any specific level of goal in discussion. In contrast, sub-goals are exactly one level lower. When a sub-goal cannot be further decomposed, we call it a 'leaf goal'. In other words, a leaf goal is the lowest level goal in a goal structure [Guo, 2002].

These concepts can cause confusion, particularly when a specific goal has two or more super-goals, or two or more sub-goals. In this thesis, we give each layer of goals a distinct name to avoid possible confusion. .

2.4 Existing Goal Identification Methods

As we mentioned above, existing goal-driven modeling methods assume that the task of identifying specific goals is a straightforward process, which happens spontaneously. However, some researchers have suggested that substantial work must be extended in order to identify goals. In this section, we will review these goal identification methods and analyze their suitability for our OEM-based goal modeling. Most of the goal identification methods have been investigated in the field of requirement engineering. These methods are illuminative for the current study, inasmuch as requirement engineering is highly related to business processes, as follows:

(1) Both areas of study use goals for the same purpose, i.e., to handle exceptions and to

generate different alternatives.

(2) They both involve a variety of particular processes, because they both consist of various activities, which must be executed in a specified order to accomplish certain tasks [Jablonski and Bussler, 1996].

(3) In essence, automating business processes with information systems or information technology is similar to the processes involved in requirement engineering.

In this section, we introduce existing goal modeling methods. Among them, GBRAM and Rolland's Goal Modeling will be addressed in detail, because these two methods are the only available systematic approaches to identifying goals both in the area of requirement engineering and business processes.

2.4.1 The Goal-Based Requirements Analysis Method (GBRAM)

Existing goal-based methods have generally failed to address the initial identification of the origins of goals, taking previous documentation of the goals for granted [Anton, 98, 67]. The Goal-Based Requirements Analysis Method (GBRAM) [Anton, 98] is a method for the identification and refinement of goals into operational requirements for software-based information systems. GBRAM focuses on the initial identification and abstraction of goals from all available sources of information, regardless of the scope of the information. It also supports the elaboration of goals to the desired level of detail, as appropriate in particular applications.

Figure 2.8 depicts the entire process when an analyst applies GBRAM to identify goals. The ovals located within the dotted box on the upper right corner of the figure denote the goal analysis activities. In GBRAM, goal analysis involves the exploration of documentation for goal identification followed by the organization and classification of goals. The goal analysis activities may be summarized as follows:

- *Exploration* activities entail the examination of 'inputs'.
- *Identification* activities entail the extraction of goals and their responsible agents from the available documentation.
- *Organization* activities involve the classification of goals and organization of those goals according to goal dependency relations.

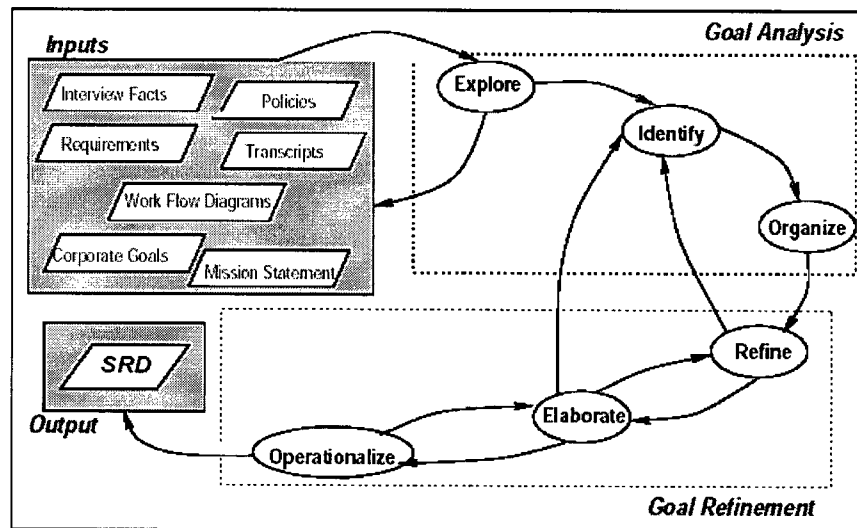


Figure 2. 8: Overview of GBRAM Activities [Anton, 1996]

The ovals within the dotted box on the lower half of the figure denote the activities that take place during goal refinement. Goal refinement involves the evolution of goals

from the moment they are first identified to the moment they are translated into operational requirements meeting system specifications. The goal refinement activities may be summarized as follows:

- *Refinement* activities entail the actual pruning of the goal set.
- *Elaboration* activities refer to the process of analyzing the goal set by considering possible goal obstacles and constructing scenarios to uncover hidden goals and requirements.
- *Operationalization* refers to the translation of goals into operational requirements according to the final required specifications.

The box in the top left corner of Figure 2.8 contains the possible inputs, which may vary in accordance with the documentation initially available to analysts. The output of GBRAM (as shown in Figure 2.8) is always a software requirement document (SRD). The SRD includes functional and nonfunctional requirements for a system, and should be very specific with regard to the external behavior of the system.

In the process of goal analysis activity, analysts begin by exploring existing documentation for the initial identification of goals, they then proceed to identify stakeholders and responsible agents, and they conclude by organizing goals according to dependency relationships and by classifying goals according to target conditions.

Goal identification techniques are applied to extract and identify goals from the

available documentation to initially specify them for further elaboration. Stakeholders are identified by considering who or what claims an interest in each goal. In addition to goals and stakeholders, the responsible agents must be identified, thereby allocating responsibility assignments to each goal.

To extract and identify goals, each statement (or piece of information) is analyzed by asking, “What goal(s) does this statement/fragment exemplify?” and “What goal(s) does this statement block or obstruct?” In this process, all action words are possible candidates for goals in the proposed system. The goals can be identified by searching for action words that point to a particular state that is achieved within the system once the action is completed. In order to operationalize goals for specification, analysts must be able to reason about any preconditions and postconditions inherent in the goals and the corresponding system operations. It is for this reason that the identified goals are worded to emphasize a state that is true or a condition that holds true, when the goal is realized.

Goal-identification can be illustrated by an example based on a career track training practice: “Congress has mandated that acquisition professionals in the Department of Defense (DoD) must improve their acquisition skills so that they may spend tax-payers’ money allocated for weapons systems more effectively and efficiently. A DoD-wide program that includes the introduction of new positions and training programs was established to develop career tracks for these acquisition professionals.” By examining each statement in the example and asking “What goal does this fragment exemplify?”,

several initial goals become evident from the description: *skills improved*, *position training provided*, *qualifying training provided*, *career tracks provided*, and *tax-payer money spent efficiently*. Then, all the initial goals are refined, a process that involves refining the goal set by eliminating redundant goals and reconciling synonymous goals. Goals are initially refined by eliminating redundancies and reconciling synonymous goals. Goals are considered synonymous if their intended states are equivalent or if they mean the same thing to different stakeholders who simply express the goal using different terminology. It is up to the analyst to identify these instances. As an example of reconciling synonymous goals, in a case like this one, if we identify another goal, called *skill training provided*, we will find this goal and *qualifying training provided* are synonymous and can be reconciled as one goal.

The objective of elaboration is to identify obstacles to goals, by considering possible reasons that goals might fail and by constructing scenarios to uncover hidden goals and requirements. The objective of operationalization is to represent the goals more formally (e.g. more formal than they appear in ordinary English usage) so that they may be mapped onto actions in a set of goal schemas. As these two activities are not directly related to goal identification, we will not explain them in detail. An example of the goal schema of GBRAM is provided in the Table 2.4.

Goal:	<i>Name</i>
Type:	<i>Name</i>
Description:	<i>Text</i>
Action:	<i>Name</i>
Agent:	<i>Name(s)</i>
Stakeholders:	<i>Name(s)</i>
Constraints:	<i>Items</i>
Obstacles:	<i>Items</i>
Preconditions:	<i>Condition</i>
Postconditions:	<i>Condition</i>
Subgoals:	<i>Name(s)</i>

Table 2. 4: Schema Syntax for Goal Models [Anton, 1996]

The principle contribution of GBRAM to the current study is that it provides proscriptive advice to analysts for the initial discovery and identification of goals; however GBRAM also has its own limitations. The main limitations of GBRAM are as follows:

1. The method provides informal semantics for goals, as opposed to formal semantics, and thus it does not support formal reasoning.
2. While well suited for identifying functional requirements which represent specific behavior the proposed system should exhibit, GBRAM has not been adequately proven and tested for nonfunctional requirements, other than for general maintenance purposes.
3. GBRAM focuses on searching for action words that point to a projected state, like *complete*, *achieve*, or *find out*, in order to identify goals. However, this approach might not work in business processes, which usually involve the general processes of the business, and they do not involve as many action words that point to particular states as desirable for the requirement engineering.

4. The processes of eliminating redundant goals and reconciling synonymous goals are in some sense too subjective; they rely on analysts' personal experiences and skills as they perform the analysis.

2.4.2 Rolland's Goal Modeling Using Scenarios

According to previous studies, the main sources for identifying goals have been identified as *scenarios*, *usage cases*, *interview transcripts*, *corporate mission statements*, and similar information resources [Anton 1997, van Lamsweerde et al. 1995]. *Scenarios* have been employed extensively in these studies, both to discover goals and to express how the goals can be implemented.

Rolland et al. [1998] have applied scenarios for determining goals, by proposing bi-directional links between goals and scenarios. According to this coupling, just as goals can assist in discovering scenarios, so can scenarios help in discovering goals. Thus, the process of determining requirements can be decomposed into two phases: *scenario authoring* and *goal discovery*.

A scenario is "a possible behavior limited to a set of purposeful interactions taking place among several agents" [Plihon et al., 1998]. Figure 2.9 demonstrates that a scenario is composed of one or several actions; the combination of which follow a unique path leading from initial to final states of agents. Thus, a combination of scenarios is used to describe the behavior of a complex system of agents. We are also aware that not all the

possible behaviors can be expressed through combinations of scenarios.

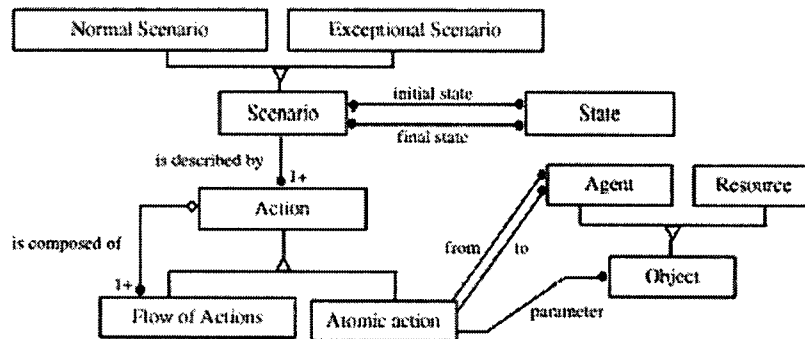


Figure 2. 9: The Scenario Structure

A scenario is characterized by its initial and final states. An initial state of a scenario is the precondition for the scenario to be triggered, while its final state is the state reached at the end of the scenario.

The concepts related to *scenarios* are similar to the concepts of OOEM. For example, OOEM has the same state (i.e., a *stable state*). The actions in scenarios are like the activities in the OOEM model, and a scenario is similar to a service in OOEM. An agent in a scenario is like an object in OOEM (refer to Table 2.5). However, OOEM does not consider resources, and it does not differentiate *normal services* from *exceptional services*. Therefore, this differentiation is introduced in our OOEM-based goal modeling. In the process of identifying goals of services, we informally categorize the services into two categories: services that are accomplished successfully; and services that fail. This will be addressed in detail in Chapter 3.

Roland's Concepts	OOEM Concepts
actions	activities
Scenarios	services
agents	objects

Table 2. 5: Mapping Roland's concepts into OOEM concepts

Actions in Figure 2.9 are of two types: *atomic* and *flows of actions*. Atomic actions are interactions from one agent to another which affect a parameter within the scenario. Flows of actions are composed of several atomic actions.

Another important concept in Rolland et al.'s method is *requirement chunks* (RC). A requirement chunk is the basic building block of the requirement chunk model. They are defined as pairs: $\langle G, Sc \rangle$, where G is a goal and Sc is a scenario. Because goals are intentional and scenarios are operational by nature, requirement chunks are possible ways of achieving goals. The requirement chunk is modeled as a class of objects that are an aggregate of the goal and scenario classes.

In the *requirements elicitation process*, the discovery of goals and the authoring of scenarios are complementary activities. Once a goal is discovered, a scenario can be authored, followed by the discovery of detailed-level goals. These goal-discovery/scenario-authoring sequences are repeated to incrementally populate the hierarchy of requirement chunks (Figure 2.10).

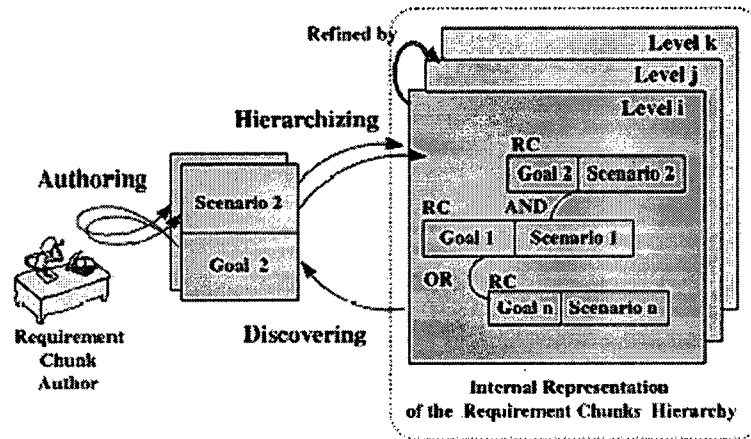


Figure 2. 10: Overview of the Discovery Process [Rolland, 1998]

Several advantages are associated with this model:

1. Scenario-based goal modeling sets up a tight coupling between goals and scenarios. It exploits goals in the reverse direction, from scenarios to goals. This contributes to removing the *fitness of use* problem identified by Potts [Potts, 1997], which leads to the generation of spurious, uninteresting, or non-critical goals.
2. Interactions expressed in scenarios are concrete and recognizable, and therefore the use of goal-scenario coupling for goal discovery helps to remove the 'fuzziness' that domain experts find in the notion of a goal. Instead, each interaction corresponds to goals. Again, goal discovery becomes a natural process through interactions of scenarios, and the goal-scenario coupling removes some of the mystery and ad hoc characteristics associated with it. In this sense, it helps in goal discovery.

Rolland's Goal Modeling clearly addresses how goal-discovery and scenario-authoring sequences are repeated to construct the requirement chunks hierarchy,

but the first pair of a goal and a scenario must be given, i.e., we still do not know how the first goal is generated. Our OOEM-based goal modeling identifies goals starting from documenting business processes, and thus addresses this problem. On the other hand, Rolland's Goal Modeling is a top-down approach, which is not suitable for OOEM-based goal modeling's bottom-up approach.

2.4.3 Other Goal Identification Methods

As noted above, most researchers have treated goal identification as a straightforward process. They either have not dealt with the origin of goals explicitly, or they have used simplistic approaches to the topic. For example, according to Haberfellner's goal-modeling method [1992], a process analyst begins with questions like "what are we trying to achieve?" and "what are we trying to avoid?" The answers to these questions are then treated as goals for business processes.

In this section, we have reviewed several existing goal-modeling methods. As this discussion has demonstrated, goal-modeling methods in the area of requirement engineering are more systematic than those used in business processes, and therefore some concepts have been borrowed from them for the current study. However, compared to methods used in business processes, they may be too simple and not systematic enough. Therefore, our OOEM-based goal modeling is customized to identify goals in business processes, and it is more systematic and based more on theory, i.e. OOEM and related

theories.

In the process of identifying goals, one thing that should be emphasized is that other expressions are used by different researchers to express the meaning of “identifying goals”, like *goal seeking* [Nishit, 2002], *goal finding* [Regev and Wegmann, 2002], *goal discovering* [Rolland et. al, 1998; Kavakli and Loucopoulos, 1998], and *goal achieving* [Craven and Mahling, 1995]. The term *identify* is used in this thesis because it was used in Anton’s goal modeling method (GBRAM), which was the first systematic method to talk about how to “identify” a goal.

2.5 Summary of Survey

In this chapter, we have introduced three research approaches, OEM, OOWM, and MOAP-Wf, which constitute the theoretical foundations of this thesis. We have also discussed various definitions of goals and related concepts, so as to build a common ground of terminology for analysis. Finally, we have reviewed two major research works on goal identification methods. As every previous research work has its merits and limitations, we will keep their merits and try to address the limitations in our OEM-based goal modeling.

Chapter 3: The Rules and Steps of OOEM-Based Goal Modeling

3.1 Modeling Rules

In this section, we describe a set of rules for developing OOEM-based goal modeling. The rules are primarily based on Object-Oriented Enterprise Modeling (OOEM) principles and address the following modeling issues:

- (1) Identifying the objects and services to be included in the model.
- (2) Identifying the goals of activities.
- (3) Identifying the goals of services.
- (4) Identifying the goals of objects.
- (5) Identifying the goals of a system.

We will use the ACME Warehouse case to demonstrate the application of our rules.

ACME Warehouse Management Inc. offers storage facilities and redistribution services (between their different warehouses) across the nation. A customer can request space in a particular warehouse, request items to be transferred to another warehouse, or request withdrawal of items from a particular warehouse (even for items not stored there).

A customer contacts ACME headquarters to request a withdrawal. An office clerk checks whether the customer has the authority to withdraw the items. The clerk then passes the withdrawal request to the warehouse where the customer wants to pick up the

items.

If the warehouse does not have the items or does not have a sufficient quantity of the items, the warehouse manager will contact other warehouses for the requested items. If the items are located, the warehouse manager will ask the planner to arrange transportation for the requested items.

The planner's responsibility is to schedule the company's truck fleet to accommodate requests for transportation, taking into account the existing schedule of each truck and its capacity. The warehouse manager will be notified whether the transportation request can or cannot be satisfied.

The warehouse manager will notify the office clerk if the request can be fulfilled or not, and the reasons. The office clerk will notify the customer as to the status of the request (approved, or declined due to lack of authority, no inventory, or no transportation).

The planner issues transport orders to truck drivers. After receiving a transport order, the truck driver informs the warehouse about the pickup of the items. The warehouse manager will make arrangements to have the items ready when the truck arrives. When the truck arrives at the warehouse, the items are loaded. The truck driver then informs the next warehouse about the delivery. When the truck has arrived at the next warehouse, the items are unloaded. A warehouse worker finds space for the items and arranges to have them moved to the allocated space. The worker updates the warehouse's inventory information. Truck drivers are required to report the status of the

truck and the delivery to the planner after each step.

The customer will come to the warehouse on the required date to pick up the items. A warehouse employee will check all the necessary documents and will deliver the items with accompanying documentation to the customer.

(Based on a case in: Jacobson, *Object-Oriented Software Engineering*, Addison-Wesley, 1992)

The purpose of this paper is to identify goals at different levels in business processes, including activities, services, objects and systems, so as to assist in handling exceptions in business processes. An accepted assumption of this study is that *all necessary information about goals will be derivable from activities and requests*. This is important because if a person does not model OEM correctly, or does not model activities correctly, then the whole approach would fail.

Services and objects are the two major constructs of Object-Oriented Enterprise Modeling (OEM), which is based on Bunge's ontology [Bunge, 1977]. According to OEM, an object is a model of a substantial thing in the problem domain that interacts with other objects [Zhao, 1995]. An object can represent an organizational unit, a division, a department or a role. A service models the transformation state of an object. It comprises a series of activities performed by an object with the purpose of satisfying a request [Zhao, 1995].

The definition of a goal in the assumption was already defined in the previous chapter. However, we have not yet defined the goal found at each level of business processes. In this paper, the goals at upper levels are defined as the stable states that are achieved through the accomplishments of related goals at immediately lower levels governed by the laws of the system. We will further operationalize how each layer's goal is identified using adverbial phrases and converting negative states into positive ones in the rules.

Exceptions in business processes are events that deviate from normal behavior and may prevent forward progress of a workflow [Chiu et al., 2001]. Exceptions can occur when business environments change, when organizational structures change, or when system errors occur [Guo, 2002].

3.1.1 Rule #1: Object and Service Identification Rule

The first task for modeling goals is the identification of objects and their services. In this paper, we simply use OEM's modeling rules to determine what will be included in the model, specifically the services that are included. In particular, we use the following rule to determine the scope of the model:

Rule 1: "The aspects of the system to be modeled are all and only those needed to represent the effects of the relevant external events".

The following rule is used to determine the objects:

Rule 2: “The model will contain all and only objects involved with generating requests for the system or responding directly or indirectly to the relevant external events”.

The following rule is used to determine services:

Rule 3: “A service will be included in an object if and only if it is involved by at least one request generated directly or indirectly by a client. Each service must use and change at least one attribute”. [Wand and Woo, 1999]

The Object-Oriented Enterprise Modeling (OOEM) methodology is based on Bunge’s ontological principles [Bunge, 1977]. These principles “provide concepts for how we can reason about the world”, and serve as “the basis to model and talk about organizational activities” [Jung, 1997]. The notion of these principles is that objects should reflect a “natural” view of the world [Wand, 1989]. With Bunge’s ontological principles applied to information system, the objects in business processes should also reflect the natural view of business processes.

As services comprise a series of actions performed by an object with the purpose of satisfying a request [Hui, 1997], services in business processes also must be identified in order to further understand the objects that perform the services.

Rule #1: The model will contain all and only those objects and services identified by the OOEM modeling rules.

In the example case detailed above, “office clerk”, “warehouse”, “planner” and “truck

driver” are all objects. “Process withdrawal request” is the service of the “office clerk”. The services of the “warehouse” include “check item availability”, “process withdrawal request”, “prepare loading”, “start loading”, “prepare unloading”, “start unloading”, and “process customer pickup requests”. The planner’s service is “plan transportation & truck assignment”, and “do transport orders” is the service of the “truck driver” (refer to Figure 3.1: OEM Diagram of ACME Warehouse example).

3.1.2 Rule #2: Activity Inclusion Rule

Although OEM provides a bird’s eye view of organizational activities within a problem domain by focusing on the interactions among objects, it in fact does not capture all behavioral aspects of an organizational process [Hui, 1997]. While a service is defined as a reaction to a request, it does not indicate the complexity that arises in handling the request. Therefore, in OEM we can frequently see that two rather different services may have similar names or even the same name. The following case provides an example:

Each supermarket submits a daily cash report to the area manager. The area manager's staff carries out random checks and then passes the reports to the cashier's department to perform further checks.

The cashier's department must contact the bank to check cash flow, and it also must contact the warehouse to check the balance. Then, the cashier's department carries out multiple checks on the arithmetic of the cash reports and passes the checked reports to the accounting department.

[Adapted from SuperMarket Chain case, assignment 3 of BAIT 506, Carson Woo, 2001]

In this case, the service of the cashier's department is far more complicated than that of the area manager, but the service names of these two objects could both be named "check cash report". The goals of these two departments are surely different but no difference will be apparent if the two services have the same name. Therefore, we cannot deduce goals directly from the services, but rather we need to attend to a more detailed layer: activities.

In OOWM, activities are the basic units of operations performed by an object, and they in turn are combined to form services. A service thus contains an ordered set of activities. The nature of any particular activity emerges in the interactions between objects. The activity begins with an incoming request or the response to a request from another object. It terminates when one of the following conditions is met:

1. The activity generates a request to another object.
2. The activity has completed all it needs to do. [Hui, 1997]

In its original presentation, OAT was composed of three principle categories: *interface attributes*, *internal attributes* and *services*. In the present thesis, we are not concerned with attributes, and therefore we simplify the OAT categorization by ignoring interface attributes and internal attributes. In the revised OAT, only *services* are retained,

composed of five sub-categories: *pre-condition*, *activity*, *termination conditions*, *request generated* and *receiver*. Table 3.1 provides an example of the revised OAT.

Pre-condition	Activity	Termination Condition	Request generated	Receiver
Service 1				
Pre-condition 1	Activity 1	Termination condition 1	Request generated by Activity 1	Object receiving a request generated by Activity 1
Pre-condition 2	Activity 2	Termination condition 2	Request generated by Activity 2	Object receiving a request generated by Activity 2
Service 2				
Pre-condition 1	Activity 1	Termination condition 1	Request generated by Activity 1	Object receiving a request generated by Activity 1
Pre-condition 2	Activity 2	Termination condition 2	Request generated by Activity 2	Object receiving a request generated by Activity 2

Table 3. 1: An Object Activity Template (OAT) (adapted from Hui, 1997)

Usually, OAT is a table, displaying information regarding *activities* and *business rules*, the latter of which are divided into *pre-conditions* and *termination conditions*.

The activities identified using the OOWM approach lead to one of the following types of activities:

1. At the completion of the activity, the service is not completed yet. We are not interested in this type of activity because it does not provide us any useful information about the accomplishment of the goal of the service.
2. At the completion of the activity, the service is potentially completed (i.e., whether it is completed or not depends on the result of the execution). This type of activity is important to our modeling because it signifies the possible accomplishment of the

goal of the service.

Definition: A *Type S activity* is a sequence of one or more OOWM activities where the last one is the only one within the sequence that can potentially put the object into a stable state.

To simplify our modeling, we introduce the following rule:

Rule #2: *Only Type S activities should be included in the model*

In the example case, the service “do transport order” of the object “truck driver” has four activities. They are Activity 1: “inform the warehouse to prepare to load”; Activity 2: “arrive at the warehouse”; Activity 3: “inform target warehouse” and Activity 4: “arrive at target warehouse”.

After Activity 1 is finished, the service “do transport order” of the object “truck driver” is not completed yet (the truck driver still must request that the other warehouses load the truck). Therefore, Activity 1 should be combined with Activity 2, and then there should be a check regarding whether or not the service is finished. After Activity 2 is finished, the service is still not finished, but rather the service won’t be finished until all four activities are finished. In this case, we combine the four activities and call this new activity Type S activity. We can see this process more clearly in Table App3 and Figure App4 in Appendix E.

Once a Type S activity is completed, the object which executes this service must be in a state in which the object does not need to do anything more. In ontological terms, this kind of state is a *stable state*.

A stable state is achieved when a thing can change state only due to external stimulation. An unstable state, on the other hand, occurs when a thing must change state [Wand and Woo, 1997]. This means that a stable state is the result of the service which executes a request. The object which contains the service can be in a stable state when and only when the request is processed (either fulfilled or failed).

In the case of the service “do transport order” by the object “truck driver”, for example, after Activity 4 is finished, the truck driver can reach one of two stable states, either “items are transported to the target warehouse” or “items are not transported to the target warehouse”.

3.1.3 Rule #3: The Identification Rule for Goals of Services

An object fulfills its goal by accomplishing its related services governed by the laws. Therefore, we must identify the goal of each service in order to identify the goal of the entire object. All potential stable states of an object as the result of executing the service(s) should be considered when identifying the goal of the service.

A request triggers a service to respond. The result of the response (i.e., whether or

not the request is acknowledged or acted upon) depends on whether the service is successfully executed. Therefore, an object should have two kinds of potential stable states after a service is executed. One is positive, when the request is accomplished after the service is successfully executed; the other is negative, when the request is not fulfilled because the service fails. Regardless of the quantity of stable states a service may be able to generate, all stable states can be separated into one of these two kinds of services, however, and only positive stable states are the objectives that services are pursuing.

Not all stable states generated by a service are considered. Stable states are the results after a service is concluded; however, whether this result is acceptable or not will be evaluated in *post conditions*. For example, a post condition in a service involving people might reject the combination of a weight of 50 pounds and a height of 7 feet, because this combination is impossible in the circumstances. If the service checks for valid combinations of weights and heights, then the stable states will reflect them. Otherwise, they can be checked by the post conditions. In this thesis, we assume that post conditions are handled within the service.

Given the above, we provide the following rule:

Rule #3: The goal of a service should capture all the potential positive stable states of the object as the result of executing the service , should not include integrity information or validations that belong to post-conditions, and should include any constraints specified in the request that invokes the service.

The term “constraints” in the rule involves any specifications that a request could address. For example, if a request specifies that weight and height have to be given in kg and cm, then this constraint should be included as part of the goal of the service.

In order to identify the goals of services, the first step is the categorization of all goals of Type S activities into two types: positive (request is accomplished) and negative (request is not accomplished) in the form of “request + service”, then we interpret the negative type into a set of positive goals of Type S activities. Finally, all positive goals of Type S activities are combined together using the attributes and values contained in the goals of Type S activities. We use “request + service” to describe the two types of goals of Type S activities in order to clearly identify that an object reaches a stable state when a request is processed by executing a service.

Step 1: Categorize all goals of Type S activities into two types: positive (request is accomplished) and negative (request is not accomplished) in the form of “request + service”.

In the ACME Warehouse case, the service “process withdrawal request” by the object “warehouse” has four goals of Type S activities. They are:

1. Items can be located at the target warehouse.
2. Items cannot be located within the warehouse.
3. Items can be located at other warehouses and can be transported to the target

warehouse.

4. Items can be located at other warehouses, but cannot be transported to the target warehouse.

We combine these four goals of Type S activities into two types according to whether the request is finished or not. We can see that the first and third items belong to the group in which the request is finished. The third state is more complicated than the first, but their results are the same, in that the request (items are located at the target warehouse) is accomplished after executing the service (searching ACME Warehouse). Thus, the first and third items can be combined in the form of "request + service", which is "items are located at the target warehouse by searching ACME warehouse". For similar reasons, the second and fourth states can be combined together into "items are not located at the target warehouse when searching ACME warehouse fails".

We use the form "subject + predicate (passive voice)" to describe the "request" part of a goals of Type S activities. A passive voice is used widely by many researchers describing goals, such as Anton [1998] and Lee [1993]. As goals are identified as desired states, the goals should be worded in the passive voice [Anton, 1998]. Besides, in the process of goal identification, we are more concerned regarding what function is executed or accomplished, rather than regarding the identity of the object that executes this function. In fact, the function could be executed by different objects in the case of business process redesign, but the function itself would remain through the changes.

Step 2: Interpret negative states into positive ones.

After the above step is completed, we interpret the negative goals of Type S activities into a set of positive ones. In the ACME example, the negative goals of Type S activities are described as “items are not located at the target warehouse when searching ACME warehouse fails”, which can be interpreted into a set of positive states, such as “items are located at the target warehouse by searching branch A’s warehouse”, “items are located at the target warehouse by searching vendor B’s warehouse” and so on.

The primary value in interpreting negative sentences into affirmative sentences arises because each sentence with a negative meaning fails to correspond to a single positive property, due to the imbalance between the one factor that is denied (by the negativity of the statement) and the multiple alternatives that may be implied by that denial [Pacholczyk, D, 1998]. For example, “John is not tall” does not necessarily refer to the sentence “John is small” but can correspond to several possible interpretations like “John is very small”, “John is small” and “John is medium”. We can clearly see that a model that implies denied properties cannot be viewed as involving only one-to-one correspondences, but also one-to-many relations, generating multi-set functions [Pacholczyk et al., 1998].

Step 3: When all of the potential goals of Type S activities have been expressed positively, combine them together using the attributes and values contained within them.

All the positive goals of Type S activities generated to this point in the process are related to the same service, and therefore it is possible to use the same set of attributes to model them. As demonstrated in the example above, an original positive goals of Type S activity can be characterized by a single attribute, which in this case is “items are located at the target warehouse”; while its value is “searching ACME warehouse”.

Similarly, the attributes and values of the negative type are expressed by the statement, “items are located at the target warehouse = < searching branch A’s warehouse, searching vendor B’s warehouse...>” based on the theory of interpreting linguistic negation [Pacholczyk et al., 1998].

Thus, the combination of all positive goals of Type S activities, i.e., the goal of this service, is “items are located at the target warehouse”, which encompasses a set of values including searching the ACME warehouse, searching branch A’s warehouse, searching vendor B’s warehouse, and so on.

In this case, if the original service, “process withdrawal request”, fails (i.e., items cannot be located within the ACME warehouse), other services like “searching Vendor B’s warehouse” may be employed. All these services involve different operations, but have the same goals at the level of the service.

3.1.4 Rule #4: The Identification Rule for Goals of Objects

An object possesses an interface comprised of a selection of interface attributes. Other than through the interface, attributes are not known by other objects and can be changed only by actions of the object. This has been referred to as *encapsulation* in the context of modeling business processes [Wand and Woo, 1999].

The concept of encapsulation is used widely in many fields. It is the process of separating the characteristics of an object into *external* and *internal* aspects. The external aspects of an object must be visible, or known, to other objects in the system. The internal aspects are details of an object that should not be affected by other parts of the system. Hiding the internal aspects of an object means that they can be changed without affecting other objects in the system. The primary benefit of encapsulation in Object Oriented modeling is that it permits the internal operation of a component to be changed without affecting other aspects. In OEM, only the interface attributes of an object can be changed by other objects, but the change of internal attributes always occurs within the object without affecting other objects in the system.

Thus, in the process of our goal modeling, we also desire the benefits of encapsulation. In effect, once the goal of any service fails to handle an exception in the workflow, the exception could still be handled by the goal of the object to which the service belongs. In this situation, more exceptions can be handled at the level of the object without affecting other objects.

Rule #3 allows us to introduce additional services to complete the goals of services; however, in practice the goals of a service could remain incomplete even when additional services are introduced. Therefore, similar to the duality of negative and positive goals of Type S activities, we can likewise have two kinds of goals of services: accomplished (positive) goals of services and unaccomplished (negative) goals of services.

Goals of services are used as a guide to satisfy the requests that trigger the services. As requests from external objects can change, therefore the goals of the services may also need to adapt in order to accomplish the altered requests. When an object has only one service, the object should be able to handle a variety of requests it receives. On the other hand, when an object has more than one service, finishing up to all goals of the services governed by laws means finishing all the things this object needs to do. Therefore, the goals of these services can be combined to get the emergent goal, which will be the final goal of the object.

The above considerations can be summarized in the following rule:

Rule #4: *If an object has only one service, the goal of the object should allow different goals of the service to be achieved when and only when the goals include adverbial clauses or phrases. The goal of an object is an emergent goal of a service with no adverbial information, when the object has two or more services;*

Continuing with the ACME warehouse example, the planner has only one service,

which is “plan transportation and truck assignment”. We can accomplish the goal of this service, i.e. “items are transported to target warehouse”, using Rule #3. As we mentioned above, a goal of a service can remain incomplete even if additional services are introduced, and therefore uncompleted goals might be involved in each particular service, i.e. “items are not transported to target warehouse” in this case.

Similar to Rule #3, in order to identify the goals of objects, we first interpret negative goals of services into sets of positive ones, and then combine all of the positive goals of services together.

Based on the theory of negation interpretation discussed above [Pacholczyk et al., 1998], the negative goal of service, “items are not transported to the target warehouse”, does not necessarily mean that the items cannot be transported at all; it can also be interpreted into a positive goal, like “items are transported to vendor A’s warehouse” or “items are transported to a customer’s location”. The reason we can interpret this negative goal of service into positive goals is because there is an adverbial phrase, “to the target warehouse” in this negative goal of service. Grammatically, adverbial phrases or clauses are descriptive elements that function like adverbs and adverb phrases. They are often optional, and peripheral to the meaning of the clause [Jarvie, 1993].

If a negative goal of a service contains no adverbial clause, this goal of the service cannot be interpreted into a set of positive goals, “maintaining the intended meaning of

negative information” [Pacholczyk, 1998]. This is the meaning of “when and only when the goal of service contains adverbial clauses or phrases”, in Rule #4.

Similar to processes involved in combining goals of Type S activities to identify the goals of services, the concept of *attributes* and *values* can be used to combine positive goals of services governed by laws to identify the goal of the object that performs this service. In the ACME Warehouse case, a set of goals of services can be addressed by the planner, like “items are transported to the target warehouse”, “items are transported to vendor A’s warehouse”, “items are transported to the customer’s location” and so on. When we attempt to combine these positive goals of services, it is evident that the three goals of service share one common attribute, “items are transported”, with the possible values of “to target warehouse”, “to Vendor A’s warehouse”, and “to the customer’s location”. Therefore, the goal of the object “planner” is “items are transported”.

When an object has more than one service, the process of identifying the goals of the object is more complicated. First, we use the same method to generate the combined goals of the services involved (original positive goals of services and additional positive goals of services interpreted from negative goals) and then identify the final emergent goal (i.e., the goal of the object). In this process, the adverbial clauses which could be contained in each goal of the services are not readily apparent in the final goals of the object. This is the meaning of “with no adverbial clauses or phrases”.

An emergent property is another concept from ontology relevant to the current study, and in particular regarding the functions and features of *composite objects*. Ontologically, a property of a composite object that is not possessed by any of the components of the object is called an *emergent property*. As all related services of an object governed by laws are executed one by one to finish a final stable state, once each goal of each service is finished, a new goal may be attained, although the property of the new goal might not be possessed by any of the single goals of the services. In the process of pursuing a goal of an object, a service may or may not be invoked. The goals of services that will be finally invoked are governed by the laws in the system.

Using the ACME Warehouse example, the object “warehouse” has seven services, they are “check item availability”, “process withdrawal request”, “prepare loading”, “start loading”, “prepare unloading”, “start unloading”, and “process customer pickup request”. Accordingly, the seven combined goals of these services are “items are located”, “items are located and can be transported”, “items are ready to load”, “items are loaded”, “items are ready to unload”, “items are unloaded” and “items are picked up”. In this case, all these seven goals of the service should be achieved in order to accomplish the goal of the object. After we combine these seven goals of the services together, a composite state appears, that is, “items are ready for pickup”. Thus, the goal of the warehouse is: ***Items are ready for pickup.*** (With respect to the more detailed documentation of this part, please refer to Appendix E.)

3.1.5 Rule #5: The Identification Rule for Goals of Systems

As noted above, in OEM, Rule #1 declares that only aspects relevant to the purposes of an (organizational) system should be included in the OEM model [Wand and Woo, 1997]. In the present thesis, we call the purpose of an organizational system the *goal of the system*.

Goals give an organization the direction it needs, and thus the goals help the organization to arrive where it wants to be. The goals play many important roles for organizations for three basic reasons: they give legitimacy to the organization; they provide direction, motivation, and commitment; and they establish performance standards [Simon, 1964].

Ontologically, the world is made of simple things and composite things. A composite thing is formed by simple things. Because the behavior of an organizational system is a result of the interactions among its component objects, it must possess emergent properties [Wand and Woo, 1997]. Similarly, the goal of a system should also possess the emergent properties of all related goals of the objects it encompasses. Same as Rule #4, in the process of pursuing the goal of the system, only the goals of objects governed by the laws will be included.

This observation can be summarized in the following rule:

Rule #5: The goal of a system is an emergent goal of objects in the system.

Using the ACME Warehouse example, the system has four objects: “office clerk”, “warehouse”, “truck driver”, and “planner”. The goals of the objects are “items can be picked up”, “items are ready for pickup”, “items are transported”, and “items are transported”, respectively. The emergent goal of the objects, which is the goal of the system, is “Items are picked up”.

The methodology we developed in this thesis starts with a correct OEM diagram in rule #1. The information on an OEM diagram (e.g., objects, internal requests, services, and interface attributes) is determined by its external request(s). Therefore, activities in OEM are derived using a top-down approach (external requests - objects - services - activities). On the contrary, our OEM-base goal modeling is a bottom-up methodology to identify goals (e.g., activities - services - objects - system). Assuming the top-down development of activities and the bottom-up derivation of goals are both correct and coherent with each other, the goal of the system identified through the bottom-up approach should still be able to reflect the external requests.

After applying all five rules, we can construct a goal system as a four-level structure, ordered from the lowest level to the highest; these levels are *goal of Type S activities*, *goals of services*, *goals of objects*, and *goals of systems*.

3.2 A Systematic Approach to Using the Rules

The set of modeling rules presented in Section 3.1 can be used in a systematic way to produce an OEM-Based Goal Model. Thus, the rules can serve as the basis for defining a modeling process, which can be formalized in the following formula:

1. Using OEM to identify the objects and the services of each object and to draw an OEM diagram
2. Identifying the activities of each service using an Object Activity Template (OAT) and an Activity-Based Diagram
 - 2.1 Drawing an Object Activity Template (OAT) for each service
 - 2.2 Drawing an Activity-Based Diagram for each service based in an OAT
3. Identifying all the positive stable states of each service
4. Identifying the goals of each service
 - 4.1 categorizing all the stable states into two types (negative and positive)
 - 4.2 interpreting negative states into positive states
 - 4.3 combining all positive stable states together using the attributes and values contained within the positive stable states
5. Identifying the goal of each object
 - 5.1 interpreting negative goals of the services into sets of positive goals
 - 5.2 combining all positive goals of the services together
6. Identifying the goals of the system

Combining all of the goals of the objects to form an emergent goal of the object

7. Summarizing all the steps in a table

Step	Invoked rule	Use of Rule
1	1	Object and Service Identification
2	2	Activity Inclusion
2.1	2	Activity Inclusion
2.2	2	Activity Inclusion
3	2	Activity Inclusion
4	3	Goal of Service Identification
4.1	3	Goal of Service Identification
4.2	3	Goal of Service Identification
4.3	3	Goal of Service Identification
5	4	Goal of Object Identification
5.1	4	Goal of Object Identification
5.2	4	Goal of Object Identification
6	5	Goal of System
7	1,2,3,4, 5, 6	Object and Service Identification, Activity Inclusion, Goal of Service Identification, Goal of Object Identification

Table 3.2: Relating the steps to the rules

Note 1: If a service has only one activity, and if that activity has only one pre-condition and one termination condition, step 2.2 may be skipped

Note 2: If an object has only one service, step 5.2 may be skipped

We now use the ACME Warehouse example case to illustrate the systematic approach to developing an OEM-Based Goal Model, whereby we will only focus on the “planner” object. The application of the approach on other objects can be found in Appendix E

3.2.1 Step 1: Using OEM to Identify the Objects and the Services of Each Object and to Draw an OEM Diagram

The OEM diagram of the ACME Warehouse example case is as follows:

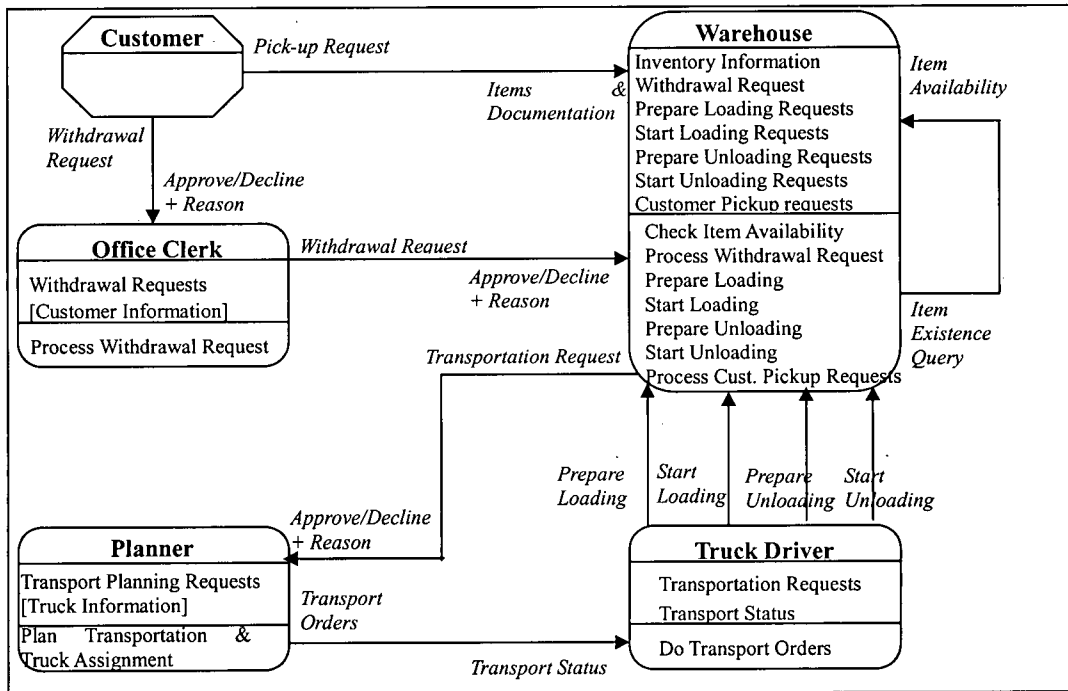


Figure 3. 1: OEM Diagram of ACME Warehouse example case

3.2.2 Step 2: Identify the Activities of Each Service Using an Object Activity Template (OAT) and an Activity-Based Diagram

An Object Activity Template (OAT) can be drawn for each service, and an Activity-Based Diagram based on OAT can then be drawn for each service. We draw the OAT and Activity-Based Diagram of “Planner” as follows:

Pre-condition	Activity	Termination Condition
Service 1: Plan transportation and truck assignment		

C1: request from warehouse to arrange transport	A1: check schedule of truck and capacity, and issue transport order to truck drivers	T1: warehouse is told the items cannot be transported OR transport order is sent to truck drivers to execute transport
---	--	--

Table 3.3: OAT of a Planner

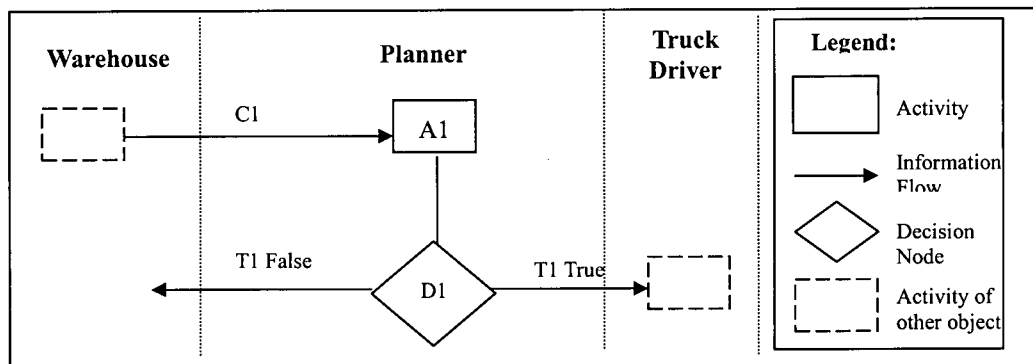


Figure 3. 2: Activity-Based Diagram of “Plan Transportation and Truck Assignment”

3.2.3 Step 3: Identify all the Type S activities of Each Service

There is only one activity in the service “plan transportation and truck assignment”, which is Activity 1, “check schedule of truck and capacity and issue transport order to truck driver”. Activity 1 can reach two stable states: “ACME truck is not arranged” (*Stable State 1*); and “Transportation is arranged using ACME truck and transport order is sent to truck driver” (*Stable State 2*). Thus, we have found two stable states of the object “planner” based on the service “plan transportation and truck assignment”. They are:

Stable State 1: ACME truck is not arranged.

Stable State 2: Transportation is arranged using ACME truck, and a transport order is sent to the truck driver.

3.2.4 Step 4: Identify the Goal of Each Service

There are two stable states in the service “plan transportation and truck assignment”, which are:

Stable State 1: An ACME truck is not arranged.

Stable State 2: Transportation is arranged using an ACME truck, and a transport order is sent to the truck driver.

Step 1: We categorize these stable states into two types: the positive state is “Items are transported to the target warehouse using ACME trucks”; and the negative state is “Items are not transported to the target warehouse when planning ACME trucks fails”.

Step 2: Interpret negative sentences into a set of positive ones: “Items are not transported to the target warehouse when planning ACME trucks fails” is interpreted into “Items are transported to the target warehouse by using a courier company”, “Items are transported to the target warehouse by using the post office” and so on.

Step 3: We combine all positive stable states together, so the goal of the service is:

Items are transported to the target warehouse.

3.2.5 Step 5: Identify the Goal of Each Object

We have two types of goals of services: positive and negative. The positive goal is “items are transported to target warehouses”, and the negative one is “items are not

transported to target warehouses”.

Step 1: Interpret negative sentences into a set of positive ones: “items are not transported to target warehouses” is translated to “items are transported to other warehouses”, “items are transported to customer’s location” and so on.

Step 2: We combine all positive stable states together, so the goal of the service is:

Items are transported.

3.2.6 Step 6: Identify the Goal of the System

The whole system has four objects: office clerk, warehouse, truck driver, and planner. The goals of the objects are “items can be picked up”, “items are ready for pickup”, “items are unloaded”, and “transportation is arranged and a transport order is sent”, respectively. The emergent goal of the object, which is also the goal of the entire system, is: *Items are picked up.*

3.2.7 Step 7: Summarize the Steps in a Table

	Service	Activity	Stable State	Goal of service	Goal of object
Goal of System: Items are picked up					
Office clerk	Process withdrawal request	1. Check customer’s authority; 2. Notify customer regarding the item’s availability;	1.Items cannot be picked up at the target warehouse because authority is not passed; OR 2.Items can be picked up	Items can be picked up at the target warehouse	Items can be picked up

			at the target warehouse because authority is passed; OR 3.Items cannot be picked up at the target warehouse because of unavailability of the items at the target warehouse.		
Ware house	Check item availability	1.Other warehouses check whether they have the items or not	1.Items are located at other warehouses OR 2.Items are not located at other warehouses	Items are located at the target warehouse	<i>Items are ready for pickup</i>
	Process with-drawal request	1. Check the target warehouse 2. Contact other warehouses 3. Arrange transportation	1. Items can be located at the target warehouse. 2. Items cannot be located within the warehouse. 3. Items can be located at other warehouses and can be transported to the target warehouse 4. Items can be located at other warehouses and cannot be transported to target warehouse.	Items are located at a the target warehouse	
	Prepare loading	1.Prepare to load the items	1.The items are ready to load at other warehouses 2. The items are not ready to load at other warehouse	Items are ready to load at other warehouses	
	Start loading	1.Load the items	1. The items are loaded at other warehouses 2. The items are not loaded at other warehouses	Items are loaded at other warehouses	
	Prepare unloading	1.Prepare to unload the items	1. The items are ready to unload at the target	The items are ready to	

			warehouse 2. The items are not ready to unload at target warehouse because the unloading is not prepared	unload at the target warehouse	
	Start unloading	1.Unload the items	1. The items are unloaded at target warehouse and unloading is started. 2. The items are not unloaded at target warehouse and unloading is not started	Items are unloaded at the target warehouse	
	Process customer pickup request	1.Check documents and deliver the items	1. The items are picked up at target warehouse by documents being checked. 2. The items are not picked up at target warehouse by documents being not checked.	Items are picked up at the target warehouse	
Planner	Plan transportation and truck assignment	1.Check schedule of truck and capacity and issue transport order to the truck driver	1.Items are transported to target warehouse using an ACME truck OR 2.Items are not transported to target warehouse when the ACME truck fails.	Items are transported to the target warehouse	Items are transported
Truck driver	Do transport order	1.Inform the warehouse to prepare to load 2.Arrive at the warehouse 3.Inform the target warehouse 4.Arrive at the target warehouse	1. Items are transported to the target warehouse by ACME transport. 2.Items are not transported to the target warehouse by ACME transport.	Items are unloaded at the target warehouse	Items are unloaded

Table 3.4: Summary of OEM-Based Goal Modeling Steps

3.3 Summary

In this chapter, we first described a set of rules for developing OEM-based goal modeling. Altogether five rules were proposed and the theories the rules relying on were also clearly discussed. Then, we addressed the detailed steps of applying this modeling method. Finally, we use the ACME Warehouse example case to demonstrate the application of our rules.

Chapter 4: Empirical Study

A user study is a necessary part of the present research. In this chapter, we will perform a small empirical study to discern the applicability of the goal modeling process proposed in the previous chapter, i.e. to assess the similarity between the goals identified using our OEM-based goal modeling on the one hand, and the goals professed by users of the same business processes.

The purpose of this empirical study is twofold. First, we diagnose any problems that might occur in any steps of the OEM-based goal modeling method, and to discern the causes of any problems that arise. Second, the empirical study also suggests solutions to these problems.

For the empirical study, three real world applications have been selected. All the applications have an application owner (e.g., an accounts payable manager) and a few organizational workers (e.g., an accounts payable clerk). Each of the corporations who own these three applications operates in three different areas.

Users and workers may have inconsistent goals, but it is beyond the scope of the present thesis to address these inconsistencies. Instead, the present study focuses on the owners of an application, a satisfactory parameter as long as the applications have only a few users.

Past research has indicated that fairly poor results are achieved by processes that rely on single evaluators, but it is reasonable to use only three to five evaluators to assure identification of most problems [Nielsen, 1999]. Nielsen has performed a series of experiments to determine the number of evaluators that are sufficient breadth to identify problems, finding that three evaluators may reveal almost two thirds of the problems (Figure 4.1) [Nielsen, 1999]. Other research by Egan et al. [1989] has used a pool of four to five subjects to identify major issues, with convincing results.

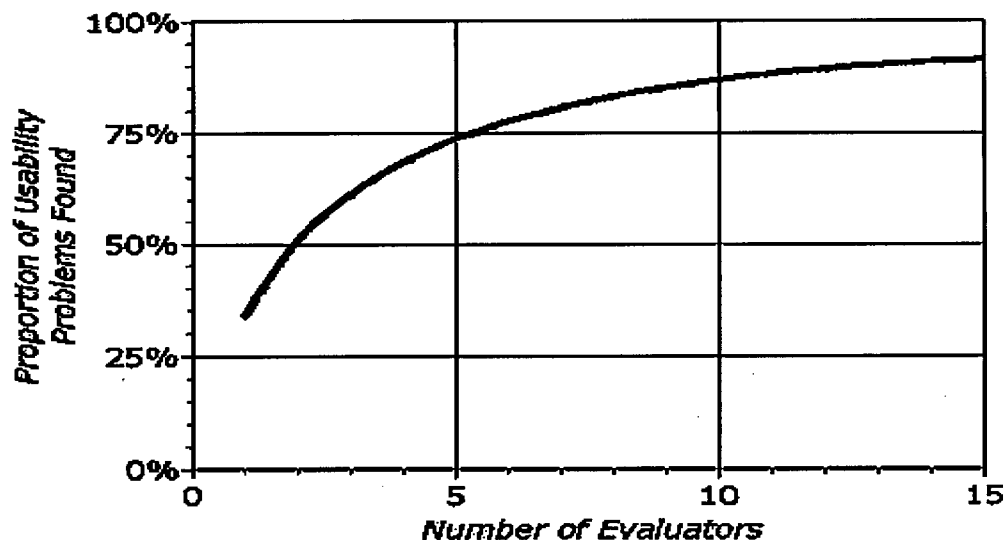


Figure 4.1: the Relationship between Proportion of Problems Found and the Quantity of Evaluators [Nielsen, 1999]

The present study has been conducted with three test subjects, due to resource limitations. No significant mathematical or statistical conclusion can be drawn from such a small sample size, and therefore the study is focused on qualitative observations. However, we can still expect to find most problems, according to Nielsen.

4.1 Study Design

A pilot session was conducted first, with one of the subjects, to test the experimental design. The findings from the pilot study will be included in the analysis of overall results.

1. Step One (Background information forms)

Each test participant was required to sign a consent form (see Appendix A) and to fill in a questionnaire (see Appendix B). This questionnaire was used to collect background information from the participants, and information regarding the participants' familiarity with OEM and studies related to goals. A brief introduction of the corporation was also presented in this process (see Appendix C).

2. Step Two (Business process description)

For each of the applications, participants were asked to describe their business processes in detail. At this stage, we did not ask participants about their goals.

3. Step Three (OEM diagram and modeling goal)

We then generated an OEM diagram of the business processes that had been identified, and we applied the methodology detailed above, to distinguish the goals of the business process.

4. Step Four (Interview)

Short interviews were conducted, asking the business owners to identify the goals of their businesses. These interviews were used to extend the heuristic evaluation method and to provide design advice. Because these interviews were used to identify modeling problems and to seek potential solutions, they were open-ended; only two of the questions were set prior to the interviews (see Appendix D). All the answer sheets were collected after the test subjects completed their work.

Note: Participants could ask questions at any stage to ensure they fully understood all the work they were being asked to complete.

4.2 Subject Backgrounds

Three sample business processes were conducted in the study. They were chosen from three different corporations respectively. In the process of conducting the tests, eleven participants were invited.

4.2.1 Sample Corporations

The first test corporation is in the food industry. It has 300 staff, and its gross revenue in 2002 was around CDN\$10 million. We chose the *item purchasing* process from this corporation for the test. The process begins when a user applies to purchase an item, and ends when this purchase request is fulfilled.

The second corporation is an export-import bank located in China. It is a

policy-oriented bank mainly engaged in international business. The major business of this bank consists of export credit, foreign currency loans and bills discounting, foreign government loans, and similar transactions. The gross revenue in 2002 was CDN\$420 million. At present, the bank has a total staff of 244 people. We chose the corporation's *export credit application* process because this business is the most important one of this bank. This export credit application process begins when a user (usually an export-oriented corporation) applies for export credit, and ends when the application is approved or declined.

The third test corporation is a wireless internet service provider. Its major business is short message service (SMS), including short message games (SMS games). In 2002, its staff numbered 120 employees, and its gross revenue was CDN\$50 million. The process of developing SMS games begins when users request a new kind of SMS game, and the process ends when a new SMS game is successfully developed and launched on the market.

4.2.2 Sample Business Processes

4.2.2.1 Office Tools Purchasing Process

This process belongs to the test corporation which is in the food industry. The purchase process is initiated by users who submit purchase requisitions (PR) to their departmental managers. The departmental managers check to see if the user has the need

for a new computer, if it is within the department's capital expenditure budget, and if it is affected by any other constraints, e.g. company policies. After approving a PR, a department manager passes it to the Purchasing Department.

When the PR is approved, the Purchasing Department first decides if more detailed specifications or requirements for the computer need to be determined. If so, the Purchasing Department will send a request to the IT Department to provide the details. The Purchasing Department then prepares and sends a Purchase Order (PO) to a qualified vendor. Depending on the availability of vendors and products, the Purchasing Department may need to initiate a 'vendor selection and approval' process.

When the computer arrives, the Purchasing Department requests that the IT Department verify the quality and quantity of the items delivered. After the verification process is completed, the computer is handed over to the user and the Purchasing Department sends a payment request to the Finance Department. After an Accountant has checked the required authorization, payment terms and conditions set in the PO, and after they have verified that the required documents have been completed, the cashier of the Finance Department makes the final payment to the vendor, and the purchase process is completed [adapted from Guo, 2002].

4.2.2.2 Export Credit Process

The sample corporation in the banking industry offers export credit services to

support domestic businesses that want to increase their exports. Exporters contact the Export Credit Department to apply for an export credit loan with respect to a specific project. The Export Credit Department checks whether the customer has the need for this loan, and any other relevant constraints. After approving this application, the Export Credit Department passes this application to the Evaluation Department to perform further checks.

The Evaluation Department checks whether the project is profitable, and it requests an evaluation report of the project. After the project has been verified by the Evaluation Department, the Export Credit department negotiates with the customer. During the process of negotiation, the Legal Department is invited to provide comments and suggestions. Once the contract is signed, the Export Credit Department notifies the Accounting Department to make the payment to the exporters.

4.2.2.3 Short Message Service (SMS) Product Process

Customers require a new short message product, like an SMS game. The Product Department consults the Marketing Division to solicit suggestions after receiving this request, and then proposes a plan for this product.

After this, the Product Division sends the plan to the Technology Division to perform a feasibility study. Once the Technology Department proves that the plan is workable, the Product Division sends the plan to the R&D Division. The R&D Division

then develops the product based on the plan.

The final product is sent to the Sales Division. The Sales Division first seeks the support of its distributors, and then sells the product to the customers, assisted by the carriers.

4.2.3 Owners of the Business Process

All eleven participants in the study are managers of departments in a variety of companies. Among them, one is a former MIS (Management Information System) Masters student, six participants have Masters degrees in Economics, one has a degree in Electronic Engineering, one has a degree in Computer Science, and the last participant has a degree in Law. With the exception of the former MIS student, none of the participants know anything significant about OEM, OWM, MOAP-Wf and Artificial Intelligence. When questioned about any methods the participants might already use to identify their business objectives, seven participants indicated they used no formal methods, two stated that they used meetings and brainstorming sessions, and one claimed to use SWOT analysis. The former MIS student was chosen to be the subject of the pilot test.

When interviewed about their business goals, all test participants took a long time to understand what a business goal really is, however, they more easily discussed the tasks which are assigned to their divisions by the supervisors. When the goal was explained as

the objective of the business process they own, triggered by the single external request, they could not tell what the goal is straightforwardly. The owners asked a lot of questions about what the starting point and end point of business processes are, what that external request which triggers their business is, and so on. They took two to seven days to finish filling in all questionnaire forms.

4.3 Analysis of Results

All the goals given by the owners and the goals identified using OEM-based goal modeling are listed together in the following table:

No.	Department	Goal of department	Goals derived using OEM-based goal modeling
Purchasing Process Case			
1	User's Department	Provide adequate office tools for employees so as to improve and maintain satisfactory levels of employee productivity.	Office tool is provided
2	Purchase Department	Ensure that the correct goods and services are purchased and delivered with the right qualities, at the right time, and at the lowest cost. Handle purchase requirements quickly to meet the user's needs.	Office tool is sent
3	Finance Department	Process payment request efficiently based on the cash flow status of the company. Ensure payment requests are properly authorized and verified based on the company's internal policies.	Bill is paid
4	IT Department	Determine and maintain specifications for IT equipment to minimize total costs of	Specification is given and Office tool is verified

		ownership (TCO)	
Short	Message Game Case		
5	Product Department	Design and create short-message (SMS) products, which can obtain competitive advantage over other service provider (SP).	SMS product is launched
6	Marketing Department	Provide market intelligence and conduct market research in order to help company products fit the needs of the market	Suggestion is made
7	Technology Department	Provide a technology feasibility study for the product	Develop plan is approved
8	R&D Department	Technology realization according to the product requirements of the product department.	SMS product is developed
9	Sales Department	Sell the product to distributors and retailers.	SMS product is sold
Export	Credit Case		
10	Export Credit Department	Ensure the credit follows government policies in foreign trade and finance	Export credit loan is provided
11	Evaluation Department	Increase the quality of evaluations, avoid and defend against financial risks, and use quantitative and qualitative analysis to provide a fair decision reference.	Profitability is provided
12	Legal Department	Provide comments and suggestions on legal affairs during the credit negotiation; view and modify the contract to be signed	Contract is signed
13	Accounting Department	Pay the client safely and efficiently, and report the payment promptly and accurately.	Export credit loan is paid

Table 4.1: Goals comparison table

The thirteen department goals (i.e. goals of objects) identified from the three business processes can be categorized into four groups:

- 1) The goals set by users and the goals derived using OEM-based goal modeling are closely similar, although they are expressed in slightly different ways. Goals 1, 2, 3, 8, 9, and 13 can be categorized into this group. For these goals, the users use the active

voice to describe the goals instead of the passive voice, but the meaning is almost the same. Another difference is that the goals described by the users contain several modifiers, such as *adequately*, *efficiently*, *promptly*, *accurately*, *at the right time*, and so on. Because we are proposing a more systematic and scientific method, these kinds of subjective expressions are deleted by OOEM-based goal modeling on purpose, but the basic meaning still remains.

- 2) The goals given by users are a subset of goals derived using OOEM-based goal modeling. Goals 6, 7, and 12 can be categorized into this group. From the perspective of OOEM, the goals derived by OOEM-based goal modeling usually contain a set of value, while the goals given by users listed one or several values of them.
- 3) The goals derived using OOEM-based Goal Modeling are a subset of the goals stated by users. Goals 4 and 5 can be categorized into this group. When users identify goals, they consider both the objectives that relate to routine business operations, and the objectives that relate to the general strategy of the company. Table 4.2 clearly shows that multiple factors were considered to be related to the goals of particular departments.

Although, practically speaking, the strategy of a company is also very important to the goal of departments within the company, it is difficult for OOEM-base goal modeling to account for this relationship while identifying goals as responses to specific

external requests. OEM-based goal modeling is designed to identify goals within independent business processes. Strategies or policies of a company in which the processes are conducted are usually not limited to specific business processes. Therefore, improving our goal modeling method to apply it to strategic targets and tasks assigned could be a future line of research based on this thesis.

- 4) The goals identified by the test subjects differed from the goals derived using OEM-based goal modeling. Goals 10 and 11 demonstrate this disparity. The reason of this disparity is that the subjects only considered goals related to the strategy of the company as a whole instead of business processes. This also proves that goals are not that straightforward as some researchers claim. Although we provided detailed explanations about goals, business processes and related concepts, many test participants still faced difficulties to figure out the goals. Table 4.2 clearly shows how the business process owners considered multiple factors to identify their goals.

Process name	Department	How do you know your objective
Purchasing office tool	User's Department	Based on a thorough understanding of the business
Purchasing office tool	Purchase Department	Based on a thorough understanding of the business
Purchasing office tool	Finance Department	Based on a thorough understanding of the business
Purchasing office tool	IT Department	Based on a thorough understanding of the business
Export credit	Legal department	It is the general objective of the legal department
Export credit	Export credit department	Obtain guidance from the Board of Directors

Export credit	Accounting depart	It is common sense to increase the competitiveness of a bank
Export credit	Evaluation department	It is adjusted according to many internal and external factors, but most are drawn from previous experience in risk identification
SMS game	Product	A: Strategy of the company B: Market demand and customer requirements C: Analysis of competitors' products
SMS game	Marketing	The company strategy and the requirements of the product department
SMS game	Tech	Communication with the product department.
SMS game	R&D	The company's strategy and annual plan.
SMS game	Sales	The requirements of the company and its monthly quotas

Table 4.2: Identifying the objectives of business processes

According to the above analysis, we can see that the goals derived by OEM-based goal modeling generally reflect the objectives of various business practices. Among the thirteen goals, nine (group 1 and group 2) of them, i.e., 70 percent are fairly close to the goals derived by OEM-based goal modeling. This indicates that OEM-based goal modeling can successfully identify most of the goals of business practices. Two other goals may be considered as partly correct (4 and 5). If we consider these two goals, the rate of successful identification of goals can increase to 85 percent.

4.4 Summary

In this chapter, we have described a confined empirical study. We have studied three subjects and eleven owners of business processes. The results of this empirical study demonstrate that OEM-based Goal Modeling can successfully identify goals in business

processes in most cases. In the study, one weakness that needed to be addressed was also found. The OEM-based goal modeling cannot apply to requests originated from tasks assigned by supervisors or strategies or policies of a company, which also contribute significantly to the “objective” of a division.

Chapter 5: Conclusions and Future Research

5.1 Conclusion

The primary purpose of this thesis is to propose a systematic modeling method to model goals in business processes, which other goal-driven business process systems may be able to use to handle exceptions.

In this thesis, we have introduced the principle theoretical foundations for our OEM-based goal modeling: Object-Oriented Enterprise Modeling (OEM), Object-Oriented Workflow Model (OOWM), and MOAP-Workflow. We have reviewed the concept of goals and goal-related terms, including the *owner* of a goal, the *classification* of goals and the *refinement* and *abstraction* of goals. We have also reviewed established goal modeling methods both in the area of requirement engineering and business processes, the most prominent of which are the Goal-Based Requirements Analysis Method (GBRAM), proposed by Anton [98], and Rolland's Goal Modeling Using Scenarios technique [98].

OEM provides a natural view of an organizational process, and OEM objects and system representations contain all the information necessary for an analyst to understand the organizational activities; therefore we have based our goal modeling method, and its name (*OEM-based goal modeling*), on OEM. In our modeling method, we first

identify all objects and services in a system using OOEM rules. Then we separate services into activities by introducing OOWM and propose a new kind of activity: *type S* activity, to identify the stable states of each service. Subsequently, we categorize all stable states into two categories, positive and negative, and use negation interpretation combined with ontological concepts to identify the goals of the services. After all of the goals of the services belonging to an object are identified, the goals of the object are identified by combining all possible positive goals of all of the services. Finally, the goals of the system are identified using the concept of emergent properties.

We have applied our method to three real world applications: a purchasing process of a corporation in the food industry; a credit application process in an export-import bank; and a short message (SMS) game development process of an Internet service provider. For each of the applications, we asked the owners to describe the business processes they manage, and we generated appropriate OOEM diagrams. Then, using our OOEM-based goal modeling method, we determined the goals of the objects, and we asked the owners to identify the goals of their departments. Finally, we compared the results of our modeling against the test subjects' responses.

The results of the study are quite promising. Among the thirteen goals that were identified by the test subjects and the OOEM-based goal modeling, about 70 percent of the goals were pretty close to each other. This indicates that OOEM-based Goal Modeling can successfully identify most goals in business practices. Through this empirical study,

we also found that improving our goal modeling method to apply to strategic targets and business targets could be a valuable direction for future research.

5.2 Contributions

This thesis has developed a systematic and complete modeling method to identify the goals in business processes, based on OEM. Before this method was proposed, almost all goal modeling methods in business processes were either comparatively simple or subjective.

Using our goal-modeling method, we can identify a complete hierarchy of goals, in four layers, from the level of activities to the level of the whole system. This kind of multi-tiered structure is helpful in the process of handling exceptions.

Generally speaking, OEM-based goal modeling is a bottom-up approach, i.e. we unite lower-level goals to form immediate higher-level goals. This approach is fairly straightforward, but it has the problem of lacking high level guidance. After we introduce the concept of *requests*, we address this problem to enable OEM-based goal modeling also boast some merits of top-down approach.

Furthermore, this method also makes some minor corollary contributions. First, we have introduced a new special type of activity, *Type S* activities, to OOWM. Using this Type S activity, we can find all stable states in business processes. This permits OOWM

to model business processes more precisely than before. Second, we have added a new construct, *other objects*, to the activity-based diagrams in OOWM. This expands activity-based diagrams to illustrate where objects send requests, and the sources from which objects receive requests more clearly.

5.3 Limitation and Future Research

Several research issues mentioned in this thesis require further research and attention.

First, the number of cases to which the goal modeling method has been applied is very limited; we only studied three cases in Chapter 4. A large number of empirical studies should be conducted to further examine the practicality of the goal modeling method.

Second, stable states are the starting point of our method and they are the results that a service will end up with. Whether this result is acceptable or not will be evaluated in *post conditions*. In this thesis, we have assumed that post conditions are handled inside the service. Further research is required to precisely define post conditions and how they determine whether stable states of a service are acceptable or not.

Third, in the empirical study, we have found that the strategy and policy of a company is also very important to the goal of a department, but OOEM-base goal

modeling cannot identify the goals related to the strategies and policies of a company, but rather it can only identify goals that correspond to single external requests. Future research may improve our goal modeling method to enable it to identify the goals responding to two or more different external requests.

Finally, all the goals identified by OEM-based goal modeling should be used to guide business process systems when they need to handle exceptions; however there are some other places where goals can be used. How these goals can handle exceptions is not addressed substantially in this thesis, and calls for more research. Once the exception handling functions are improved, we can obtain a more complete picture of goals, from identifying goals to handling exceptions using the goals identified.

References

1. Anton, A., "Goal Based Requirements Analysis," Proc. Second Int. Conference on Requirements Engineering, ICRE '96, pp. 136-144, 1996.
2. Anton, A., "Goal Identification and Refinement in the Specification of Software-Based Information Systems," Ph.D. Dissertation, Georgia Institute of Technology, Atlanta GA, 1997.
3. Anton, A., Potts, C., "The Use of Goals to Surface Requirements for Evolving Systems," Proc. of the 1998 Int. Conference on Software Engineering, pp. 157-166, 1998.
4. Anton, A., Earp, J., Potts, C., Aspaugh, T., "The Role of Policy and Stakeholder Privacy Values in Requirements Engineering," Proc. Fifth IEEE Int. Symposium on Requirements Engineering, pp. 138-145, 2001.
5. Bider, I., "Business Process Modeling as a Method of Requirements Engineering," ICEIS, pp. 13-15, 2002.
6. Bunge, M., "Treatise on Basic Philosophy Vol 3, Ontology I: The Furniture of the World," Reidel, Dordrecht, Boston, 1977.
7. Casati, F., Ceri, S., Paraboschi, S., and Pozzi, G. "Specification and Implementation of Exceptions in Workflow Management Systems," ACM Transactions on Database Systems (24:3), pp 405-451, 1999.
8. Chiu, K.W., "Exception handling in an Object-Oriented Workflow Management System," PhD Dissertation, Department of Computer Science, the Hong Kong University of Science and Technology, Hong Kong, 2000.
9. Chiu, D.K.W., Li, Q., and Karlapalem, K., "Web Interface-Driven Cooperative Exception Handling in ADOME Workflow Management System," Information Systems, Pergamon Press, Elsevier Science 26(2), pp93-120, 2001.
10. Cockburn, A., "Writing Effective Use Cases. Reading," MA: Addison-Wesley, 2000.
11. Constantine, L., "Essential modeling: Use cases for user interfaces," ACM Interactions, vol. II.2, pp. 34-46, Apr. 1995.
12. Constantine, L., and Lockwood, L., "Software for Use," New York: ACM Press 1999.

13. Cooper, A., "Goal-Directed software design," Dr. Dobbs Journal, Sept. 1996.
14. Craven, Noel and Mahling, Dirk, "Goals and Processes: A Task Basis for Projects and Workflows," Conference on Organizational Computing Systems, ACM Press, California, USA, 1995, pp. 237-248.
15. Dardenne, A., Lamsweerde, A. and Fickas, S., "Goal Directed Requirements Acquisition," Science of Computer Programming, vol. 20, pp. 3-50, Apr. 1993.
16. Darimont, R. and Lamsweerde, A., "Normal Refinement Patterns for Goal-Driven Requirements Elaboration," Proc. 4th ACM Symposium on the Foundations of Software Engineering (FSE4), San Francisco, pp. 179-190, Oct. 1996.
17. Eder, J., and Liebhart, W. "The Workflow Activity Model WAMO," Conference on Cooperative Information Systems, Vienna, Austria, pp. 87-98, 1995.
18. ELEKTRA consortium, "Electrical Enterprise Knowledge for Transforming Applications—Athena Deliverable: Initial Requirements for PPC," ELEKTRA Project Internal Report, <http://www.singular.gr/elektra>, 1997.
19. FDIS 15414, Information Technology - Open Distributed Processing - Reference Model - Enterprise language, 2002, p.6
20. Guo, K., "A Goal Oriented and Decentrally Controlled Workflow Model for Facilitating Exception Handling," MSc thesis, University of British Columbia, 2002.
21. Hagen, C., and Alonso, G. "Flexible Exception Handling in the OPERA Process Support System," International Conference on Distributed Computing Systems, Amsterdam, the Netherlands, 1998, pp. 526-533.
22. Haberfellner, Reinhard et al., "Systems Engineering – Methodik und Praxis," Verlag Industrielle Organisation, 7.Aufl. Zurich, 1992.
23. Jablonski, S. and Bussler, C., "Workflow management: Modeling Concepts, Architecture and Implementation," International Thomson Publishing, 1996.
24. Jarvie, Gordon, "Bloomsbury Grammar Guide," Bloomsbury Publishing Ltd, 1993.
25. Jung, D., "Object-Oriented Modeling: From Analysis to Logical Design," M.Sc. thesis, Faculty of Commerce and Business Administration, University of British Columbia, 1997.

26. Kaindl, H., "Design Process Based on a Model Combining Scenarios with Goals and Functions," IEEE Trans. Syst., Man, Cybern. A, vol. 30, no. 5, September 2000.
27. Katzenstein, Gary and Lerch, F., "Beneath the Surface of Organizational Processes: A Social Representation Framework for business Process Redesign," ACM Transaction on Information Systems, Vol 18, No. 4, October 2000, Pages 383-422.
28. Kavakli, V., and Loucopoulos, P., "Goal-driven Business Process Analysis Application in Electricity Deregulation", pp305-324. 1998.
29. Klein, M., and Dellarocas, C. "A Knowledge-Based Approach to Handling Exceptions in Workflow Systems," Journal of Computer Supported Collaborative Work, Special Issue on Adaptive Workflow Systems (9:3/4), 2000, pp 399-412.
30. Kueng, P., and Kawalek, P., "Goal-based Process models: Creation and Evaluation," Business Process Management Journal, Vol. 3, No. 1 (April 1997), pp. 17-38; ISSN 1355-2503.
31. Kueng, P., and Kawalek, P., "Goal-Based Business Process Models: Creation and Evaluation," Business process Management Journal, Vol. 3, No. 1, PP. 17-38, 1997.
32. Kueng, P., Bichler, P., Kawalek, P., and Schrefl, M., "How to Compose an Object-Oriented Business Process Model", in Brinkkemper, Sjaak et al. (Eds), Method Engineering – Principles of method construction and tool support; Proceedings of the IFIP TC8, WG8.1/WG8.2 Working Conference, Atlanta, 26-28 August 1996, Chapman & Hall, Longdon 1996, pp. 94-110.
33. Lamsweerde, A., Darimont, R. and Massonet, P., "Goal-Directed Elaboration of Requirements for a Meeting Scheduler: Problems and Lessons Learnt," 2nd Int. Symposium on Requirements Engineering (RE '95), York, UK, pp. 194-203, 1995.
34. Lamsweerde, A., "Requirements Engineering in the Year 00: A Research Perspective," Proc. 22nd Int. Conf. Software Engineering, Limerick, ACM Press, June 2000.
35. Lamsweerde, A. and Letier, E., "Handling Obstacles in Goal-Oriented Requirements Engineering," IEEE Trans. Software Eng. Special Issue on Exception Handling, 2000.
36. Leveson, N., "Intent specifications: An approach to building human-centered specifications," IEEE Trans. Software Eng., vol. 26, pp. 15-25, Jan. 2000.
37. Lee, Junta, "Goal-Based Process Analysis: a Method for Systematic Process

- Redesign*," In Proceedings of the Conference on Organizational Computing Systems (COOCS'94), 1993
38. Longman Dictionary of Contemporary English, 3rd Edition, Harlow: Longman, 1995.
 39. Loucopolos, Pericles and Karakostas, Vassilos, "*System Requirements Engineering*," McGraw-Hill, London, 1995.
 40. Mylopoulos, J., Chung, L., and Yu, E., "*From Object-Oriented to Goal-Oriented*," Communications of the ACM, vol. 42. No. 1, Jan. 1999.
 41. Narendra, N.C. "*Goal-Based and Risk-Based Creation of Adaptive Workflow, Bringing Knowledge to Business Processes*," Papers from the AAAI Spring Symposium, California, USA, 2000
 42. Nielsen, Jakob, "*How to Conduct a Heuristic Evaluation*," URL: http://www.useit.com/papers/heuristic/heuristic_evaluation.html. Online-Verbindung: 23.6.1999.
 43. Nishit, Chapagain. "*A Study on Goal-Oriented Business Process Modeling*," master thesis, Tokyo Institute of Technology, Japan, 2002
 44. Pacholczyk, Daniel, "*A New Approach to Linguistic Negation of Nuanced Information in Knowledge-Based Systems*," AIMS, pp 363-376, 1998.
 45. Plihon, V., Ralyte, J., Benjamin, A., Maiden, N., Sutcliffe, A., Duios, E., Heymans, P., "*A Reuse-Oriented Approach for the Construction of Scenario Based Methods*," proc. Int'l Software Process Assoc. Fifth Int'l Conf. Software Process, Chicago, pp14-17, June 1998.
 46. Pohl, K. and Haumer, P., "*Modeling Contextual Information about Scenarios*," Third Int. Workshop on Requirements Engineering: Foundation for Software Quality RESFQ, Barcelona, Spain, June 16-17, 1997.
 47. Potts, C., "*Fitness for use: The System Quality that Matters Most*," Proc. Third Int'l Workshop requirements Eng.: Foundation of Software Quality REFSQ'97, pp.15-28, Barcelona, June 1997.
 48. Randolph, W. Alan, and Blackburn, Richard S., "*Managing Organizational Behavior*," 1989


49. Regev, G., and Wegmann, A., "Regulation Based Linking of Strategic Goals and Business Processes," GBPM, 2002.
50. Rolland, C., Souveyet, C., Achour, C., "Guiding goal modeling using scenarios," IEEE Trans. Software Eng., vol. 24, pp. 1055-1071, Dec. 1998.
51. Scherer, Eric and Zölch, Martina, "Design of activities in shop floor management: A holistic approach to organization at operational business levels in BPR projects," in Browne, Jim and O'Sullivan, David (Eds.), *Re-engineering the Enterprise, Proceedings of the IFIP TC5/WG5.7 Working Conference*, Galway, Ireland, 20-21 April 1995, Chapman & Hall, London, 1995, pp. 261-272.
52. Simon, H., "On the Concept of Organizational Goal," *Administrative Science Quarterly*, March 1964, pp1-22
53. Sutcliffe, A. and Maiden, N., "Bridging the Requirements Gap: Policies, Goals and Domains," Proc. 7th Int. Workshop on Software Specification and Design, Redondo Beach, CA, December 1993.
54. Sutcliffe, A. and Maiden, N., "Domain Modeling for Reuse. In *Proceedings of the Third International Conference on Software Reuse Advances in Software Reusability*," pages 160-77, Rio de Janeiro, Brazil, November 1994.
55. Wand, Y., "A Proposal for a Formal Model of Objects," in: *Object-Oriented Concepts, Languages, Applications, and Databases*, Kim, W. and F.H. Lochovsky, eds, pp. 537-559, ACM Press, Addison Wesley Publishing Co., 1989, pp. 537-559.
56. Wand, Y., and Woo, C. "Ontology-Based Rules for Object-Oriented Enterprise Modeling," Faculty of Commerce and Business Administration, University of British Columbia, Vancouver, 1999.
57. Warboys, Brian (Ed.), "Business Systems Engineering – A Process Approach," draft book, Informatics process Group, Department of Computer Science, University of Manchester, 1996.
58. Weir, M., "Goal-Directed Behavior, Gordon and Breach Science Publishers," New York, 1984.
59. Yu, E., Mylopoulos, J., "Using Goals, Rules and Methods to Support Reasoning in Business Process Reengineering," Proc. 27th Hawaii Int. Conf. System Sciences, Maui, Hawaii, vol. 4, pp. 234-243, Jan. 1994.

60. Yu, E., Mylopoulos, J., "*Toward Modeling Strategic Actor Relationships for Information Systems Development - with Examples from Business Process Reengineering*," Proc 4th Workshop on Information Technologies and Systems, 1994.
61. Zhao, Howard. "*Object-Oriented Enterprise Modeling*," MSc thesis, University of British Columbia, 1995.

Appendix B: Subject Background Questionnaire

OOEM-Based Goal Modeling

July, 2003

 <p>Faculty of Commerce and Business Administration 2053 Main Mall Vancouver, B.C. Canada V6T 1Z2</p>	<p>Subject Background Questionnaire</p>	<p>Supervisor: Dr. Carson Woo Henry Angus Building e-mail: woo@interchange.ubc.ca Tel: 604-822-8390 Investigator: Weida Wang e-mail: weida@interchange.ubc.ca</p>
--	--	---

Name: _____ Date: _____

Gender: _____ Age: _____

Major: _____ Corporation: _____

Department: _____ Title: _____

The length of time you worked in this corporation: _____

1. How well do you know OOEM, OOWM and MOAP-Wf methods?

1 2 3 4 5 6 7

very unconfident

very confident

2. Have you ever taken courses on Artificial Intelligence?

Yes / No

If yes, please list the course names _____

3. How often do you have to identify objectives of the tasks you need to finish?

1 2 3 4 5 6 7

very rarely


very frequently

4. Have you used any formal methods to guide you in identifying goals or objectives?

Yes / No If yes, please list them _____

Appendix C: Corporation Brochure

OOEM-Based Goal Modeling Study
July, 2003

 Faculty of Commerce and Business Administration 2053 Main Mall Vancouver, B.C. Canada V6T 1Z2	OOEM-based Goal Modeling Steps	Supervisor: Dr. Carson Woo Henry Angus Building e-mail: woo@interchange.ubc.ca Tel: 604-822-8390 Investigator: Weida Wang e-mail: weida@interchange.ubc.ca
---	---	---

1. Export-Import Bank of China

Established in 1994, the Export-Import Bank of China (hereinafter referred to as “the Bank”) is a state policy bank directly under the leadership of the State Council and solely owned by the central government. Its international credit ratings are compatible with national sovereign ratings, ranking the highest among domestic financial institutions. At present, it has three business branches and nine representative offices in China, two overseas representative offices and 135 correspondent banks worldwide.

The main mandate of the Bank is to implement state policies in industry, finance, and foreign trade, to promote the export of Chinese mechanical and electronic products, complete sets of equipment, and high- and new- tech products, and to enhance Sino-foreign economic and technological cooperation and exchanges by means of providing financial support.

2. PepsiCo Food (China)

PepsiCo is a world leader in convenient foods and beverages, with revenues of about \$25 billion and over 142,000 employees. The company consists of the snack business of Frito-Lay North America and the beverage and food businesses of PepsiCo Beverages and Foods, which includes PepsiCo Beverages North America (Pepsi-Cola North America and Gatorade/Tropicana North America) and Quaker Foods North America. PepsiCo International includes the snack businesses of Frito-Lay International and

beverage businesses of PepsiCo Beverages International. PepsiCo brands are available in nearly 200 countries and territories.

Many of PepsiCo's brand names are over 100 years old, but the corporation is relatively young. PepsiCo was founded in 1965 through the merger of Pepsi-Cola and Frito-Lay. Tropicana was acquired in 1998 and PepsiCo merged with The Quaker Oats Company, including Gatorade, in 2001.


PepsiCo Food (China) Co., a unit of U.S. soft-drink giant PepsiCo, Inc., began producing Frito-Lay snack food products at its new plant outside Shanghai, China. The plant was built with an initial investment of \$30 million. PepsiCo Food, which entered the China market in 1994, has become the dominant snack food brand on grocery store shelves in China. PepsiCo Food (China) employs 300 staff, and its gross revenue in 2002 was around CDN\$10 million.

3. Newpalm (China) Information Technology Co., Ltd

Newpalm (China) Information Technology Co., Ltd. was founded in April of 2000, based in Beijing, China. Newpalm is a leading provider of both mobile Internet enabling technology and application services. The company's products and services enable carriers and enterprises to deploy a broad range of wireless applications in a speedy and cost-effective manner as well as empower mobile subscribers with instant access to time-sensitive, personalized services. The company is committed to expanding the scope and depth of its products and services and to bringing the Internet to the palm of mobile users. Currently, Newpalm employs approximately 80 staff members. In addition to Beijing headquarter, Newpalm has branch offices in Tianjin, Shanghai and Guangzhou. In October of 2000, Newpalm completed a second round of equity financing with a total of \$15 million investment from reputable overseas institutional investors. This new investment has further strengthened Newpalm's ability to offer its customers continued excellence within its products and services.

Appendix D: Answer Sheet

OOEM-Based Goal Modeling Study
July, 2003

	Faculty of Commerce and Business Administration 2053 Main Mall Vancouver, B.C. Canada V6T 1Z2	Answer Sheet	Supervisor: Dr. Carson Woo Henry Angus Building e-mail: woo@interchange.ubc.ca Tel: 604-822-8390 Investigator: Weida Wang e-mail: weida@interchange.ubc.ca
---	---	---------------------	---

Name (Please Print): _____

This answer sheet is designed to collect your answers to the following questions. Of course, if you like, any other questions and comments are welcome.

1. What is the objective of your department related to this particular business-process?
2. How do you know your objective?

Other Comments:

Appendix E: Identifying Goals of ACME House Case

Using OEM-Based Goal Modeling

Step 1: Using OEM to Identify the Objects and the Services of Each Object
and to Draw an OEM Diagram (refer to Figure 3.1)

Step 2: Identify the Activities of Each Service Using an Object Activity
Template (OAT) and an Activity-Based Diagram

1. Object Name: Office Clerk

1.1 OAT of Office Clerk

Pre-condition	Activity	Termination Condition
Service 1: Process withdrawal request		
C1: receive a withdrawal request from a customer	A1: check the customer's authority	T1: the customer is notified that they are declined because of lack of authority OR a request is sent to the target warehouse to locate the items
C2: the target warehouse responds to the request of the Office Clerk	A2: the customer is notified of the item's availability	T2: the customer is notified that the items are available at the target warehouse OR the customer is notified that their request is declined because of unavailability of the items at the target warehouse

Table App.1: OAT of an Office Clerk

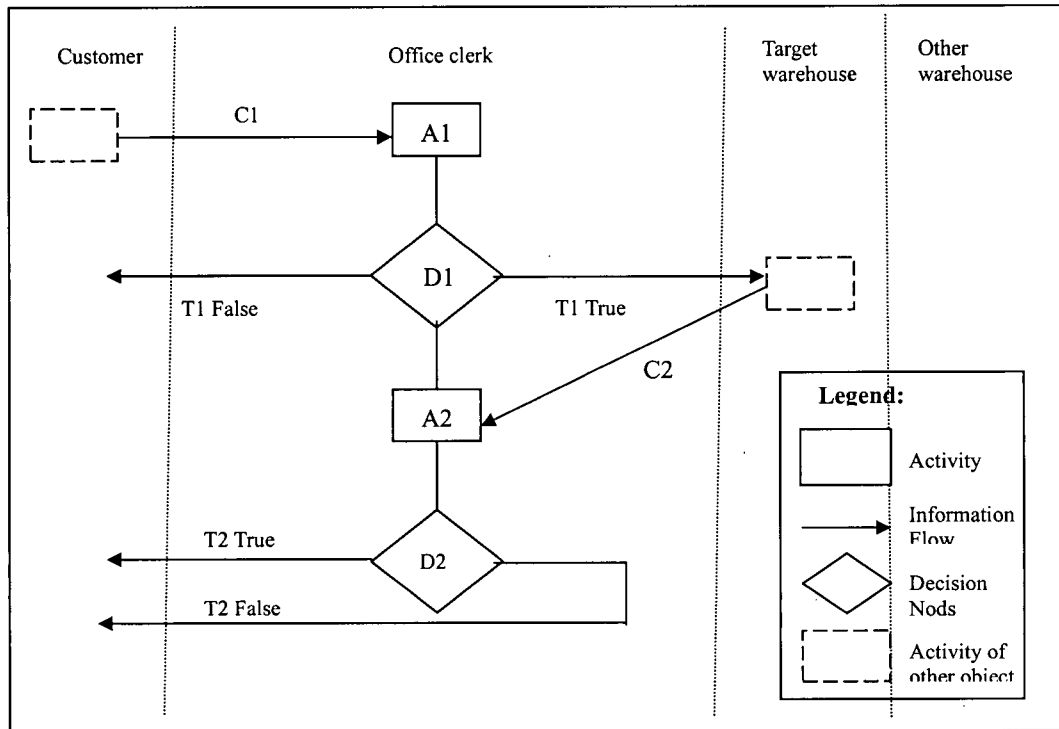


Figure App.1: Activity-Based Diagram of "Process Withdrawal Request"

2 Object Name: Warehouse

2.1 Table Appendix2: OAT of a Warehouse

Pre-condition	Activity	Termination Condition
Service 1: Check Item Availability		
C1: other warehouses receive a request from the target warehouse	A1: other warehouses check whether they have the items in sufficient quantity	T1: other warehouses provide the warehouse manager with the search result
Service 2: Process Withdrawal Request		
C1: withdrawal request from an Office Clerk	A1: check the target warehouse	T1: items are found at the target warehouse and reported to the Office Clerk OR other warehouses are asked to search for the item
C2: response to the request sent from other	A2: contact other warehouses	T2: items are not found at other warehouses and reported to the Office

warehouses		Clerk OR items are found at another warehouse and a request is sent to the Planner
C3: response to the request from the Planner	A3: arrange transportation	T3: the Office Clerk is informed that items are located at other warehouses and can be transported to the named warehouse. OR the Office Clerk is informed that the items cannot be picked up due to lack of transportation..
Service 3: Prepare Loading		
C1: the Truck Driver sends a request to the warehouse to prepare the items	A1: prepare the items	T1: the warehouse notifies the Truck Driver that the items are ready OR the warehouse notifies the Truck Driver that the items are not ready
Service 4: Start Loading		
C1: the Truck driver sends request to warehouse to start loading	A1: load the items	T1: items are loaded OR items are not loaded
Service 5: Prepare Unloading		
C1: the Truck Driver sends a request to the warehouse to prepare to unload the items	A1: the items are unloaded; space is found and the items are moved to the allocated space	T1: the warehouse notifies the Truck Driver regarding the readiness for unloading OR the warehouse notifies the Truck Driver that is not ready for unloading
Service 6: Start Unloading		
C1: the Truck driver sends a request to the warehouse to start unloading	A1: start to unload	T1: the items are unloaded OR the items are not unloaded
Service 7: Process Customer Pickup Requests		
C1: pickup request	A1: check all of the customer's documents and deliver the items	T1: items are picked up OR items are not picked up

Table App.2: OAT of a Warehouse

2.2 Activity-Based Diagram of the Service “Check Item Availability”

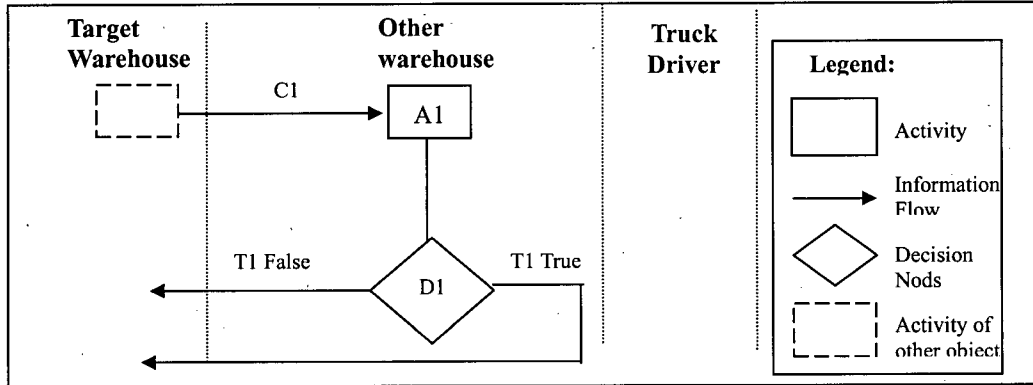


Figure App.2: Activity-Based Diagram of the Service “Check Item Availability”

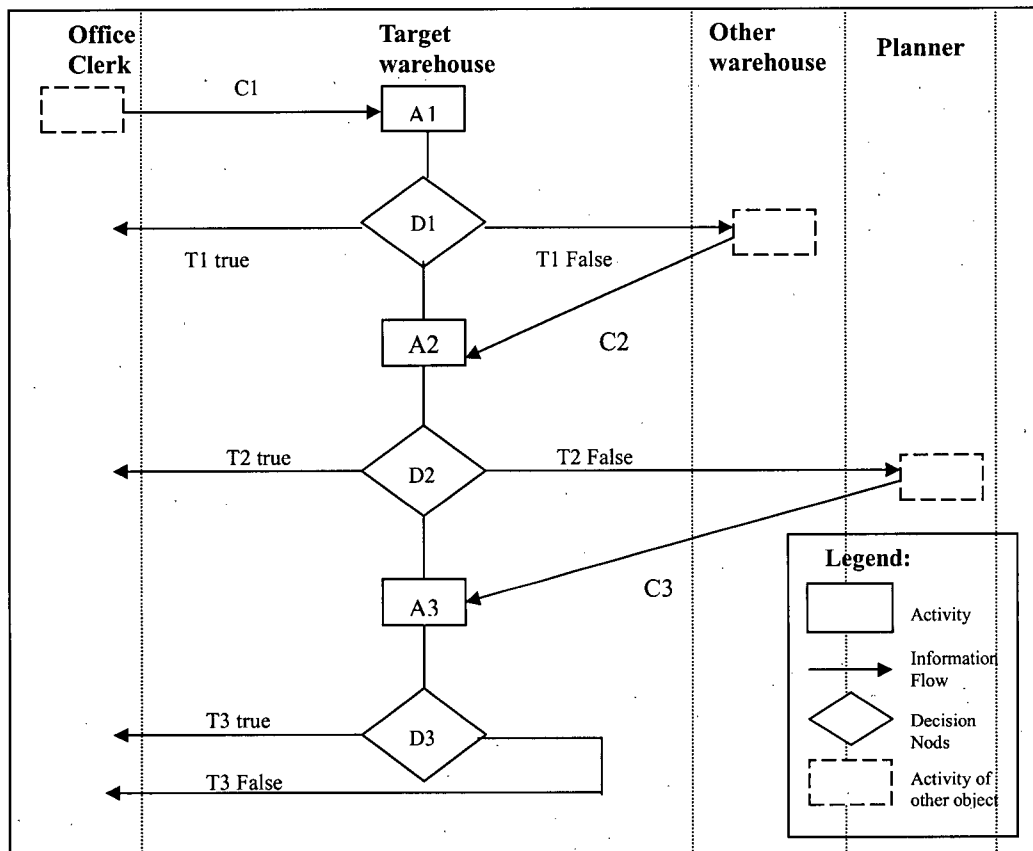


Figure App.3: Activity-Based Diagram of the Service “Process Withdrawal Request” of object Warehouse

Note: As Services 3, 4, 5, 6 and 7 are simple and straightforward; therefore we are

omitting the Activity-Based Diagrams of these activities.

3. Object Name: Truck Driver

3.1 OAT of Truck Driver

Pre-condition	Activity	Termination Condition
Service 1: Do Transport Order		
C1: request to transport the items to target warehouse from planner	A1: inform the warehouse to prepare to load	T1: request to warehouse to prepare to load
C2: response sent back to truck driver	A2: arrive at the warehouse to load	T2: request to warehouse to start to load
C3: response sent back to truck driver	A3: inform target warehouse to prepare to unload	T3: request to target warehouse to prepare to unload
C4: response sent back to truck driver	A4: arrive at target warehouse to unload	T4: request to target warehouse to start unloading

Table App.3: OAT of Truck Driver

Note: all four activities in the above table form a Type S activity.

3.2 Activity-Based Diagram of the Service “Do Transport

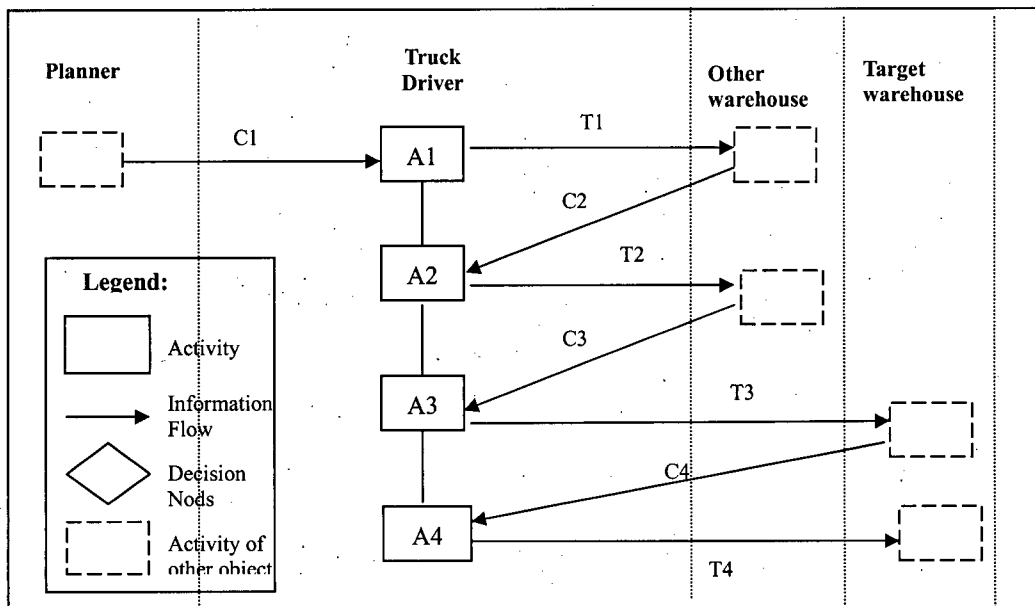


Figure App 4: Activity-Based Diagram of the Service “Do Transport”

Step 3: Identify all the Stable States of Each Service

1. Object Name: Office Clerk

1.1 Service Name: Process Withdrawal Request

There are two activities in this service, which are Activity 1: “check the customer’s authority”, and Activity 2: “notify the customer regarding the item’s availability”.

Activity 1 can reach one stable state:

Stable State 1: Items cannot be picked at target warehouse because of lack of authority.

Activity 2 can reach two stable states:

Stable State 2: Items can be picked up at the target warehouse; and

Stable State 3: Items cannot be picked up at the target warehouse because of unavailability of items at the target warehouse.

Thus, we have identified three stable states of the object “office clerk” based on the service “process withdrawal request”.

1.2. Object Name: Warehouse

1.2.1 Service Name: Check Item Availability

There is only one activity in this service, which is “other warehouses check whether they have the items or not”. This activity can reach two stable states: “Items are located at other warehouses”, called Stable State 1; and “Items are not located at other warehouses”, called Stable State 2.

Thus, we have found two stable states of the object “warehouse”, based on the service “check item availability”. They are:

Stable State 1: Items are located at other warehouses.

Stable State 2: Items are not located at other warehouses.

1.2.2 Service Name: Process Withdrawal Request

Activity 1 can reach a stable state, which is “items are located at the target warehouse”. We call this Stable State 1. Activity 2 can reach a stable state, which is “items cannot be located within the warehouse”. We call this Stable State 2.

Activity 3 can reach two stable states: “items can be located at the other warehouses and can be transported to target warehouse”, called Stable State 3; and “items can be located at other warehouses and cannot be transported to target warehouse”, referred to as Stable State 4. Thus, we have identified four stable states of the object “warehouse”, based on the service “process withdrawal request”. They are:

Stable State 1: Items can be located at the target warehouse.

Stable State 2: Items cannot be located within the warehouse.

Stable State 3: Items can be located at other warehouses and can be transported to the target warehouse.

Stable State 4: Items can be located at other warehouses and cannot be transported to the target warehouse.

1.2.3 Service Name: Prepare Loading

There is only one activity in this service, which is “prepare to load the items”. This activity can reach two stable states: “items are ready to load at other warehouses”, titled stable state 1; and “items are not ready to load at other warehouses”, called stable state 2. Thus, we have found two stable states of the object “warehouse”, based on the service “prepare loading”. They are:

Stable State 1: Items are ready to load at other warehouses.

Stable State 2: Items are not ready to load at other warehouses

1.2.4 Service Name: Start Loading

There is only one activity in this service, which is Activity 1, “load the items”. Activity 1 can reach two stable states: “items are loaded at other warehouses” (stable state 1); and “items are not loaded at other warehouses” (stable state 2). Thus, we identify two stable states of the object “warehouse”, based on the service “start loading”. They are:

Stable State 1: Items are loaded at other warehouses.

Stable State 2: Items are not loaded at other warehouses.

1.2.5 Service Name: Prepare Unloading

There is only one activity in this service, which is Activity 1, “prepare to unload the items”. Activity 1 can reach two stable states: “items are ready to unload at target warehouse” (stable state 1); and “items are not ready to unload at target warehouse” (stable state 2). Thus, we have found two stable states of the object “warehouse” based on the service “prepare unloading”. They are:

Stable State 1: Items are ready for unloading at the target warehouse.

Stable State 2: Items are not ready for unloading at the target warehouse.

1.2.6 Service Name: Start Unloading

There is only one activity in this service, which is Activity 1: “unload the items”.

Activity 1 can reach two stable states: the first is “items are unloaded at target warehouse” (stable state 1); and “items are not unloaded at target warehouse” (stable state 2). Thus, we have found two stable states of the object “warehouse” based on the service “start unloading”. They are:

Stable State 1: Items are unloaded at the target warehouse.

Stable State 2: Items are not unloaded at the target warehouse.

1.2.7 Service Name: Process Customer Pickup Request

There is only one activity in this service, which is Activity 1, “check documents and delivery of the items”. Activity 1 can reach two stable states: “items are picked up at the target warehouse” (stable state 1); and “items are not picked up at the target warehouse” (stable state 2). Thus, we have found two stable states of the object “warehouse” based on the service “process customer pickup request”. They are:

Stable State 1: Items are picked up at the target warehouse.

Stable State 2: Items are not picked up at the target warehouse.

1.3. Object Name: Truck Driver

1.3.1 Service Name: Do Transport Order

There are four activities in this service, which are Activity 1: “inform the warehouse to prepare to load”; Activity 2, “arrive at the warehouse”; Activity 3, “inform target warehouse”; and Activity 4, “arrive at target warehouse”. Only Activity 4 can reach two stable states: “items are unloaded at the target warehouse” (stable state 1); and “items are not unloaded at the target warehouse” (Stable State 2). Thus, we have identified two stable states of the object “Truck Driver” based on the service “do transport order”. They are:

Stable State 1: Items are unloaded at the target warehouse.

Stable State 2: Items are not unloaded at the target warehouse.

Step 4: Identify the Goal of Each Service

1. Object Name: Office Clerk

1.1 Service Name: Process Withdrawal Request

There are three stable states in this service, they are:

Stable State 1: Items cannot be picked up at the target warehouse because of lack of authority.

Stable State 2: Items can be picked up at the target warehouse.

Stable State 3: Items cannot be picked up at the target warehouse because of unavailability of the items at the target warehouse.

Step 1: we categorize these stable states into two types: positive and negative. The positive one is “Items can be picked up at the target warehouse by processing a

withdrawal request”; and the negative one is “Items cannot be picked up at the target warehouse when the withdrawal request fails”.

Step 2: interpret negative sentences into a set of positive ones: Items are picked up at the target warehouse is translated to “process withdrawal request” and “check the customer’s history record”.

Step 3: we combine all the positive stable states together, so the goal of the service is:

Items can be picked up at the target warehouse.

Note: we have skipped the detailed steps here. For detailed steps, refer to the service “process withdrawal request” of the object “warehouse”.

1.2 Object Name: Warehouse

1.2.1 Service Name: Check Item Availability

There are two stable states in this service, they are:

Stable State 1: Items are located at other warehouses.

Stable State 2: Items are not located at other warehouses.

Step 1: Categorize these stable states into two types: positive and negative. The positive one is “Items are located at the target warehouse by items found at other warehouses”, and the negative one is “Items are not located at the target warehouse when by items are found at other warehouses”.

Step 2: Interpret negative sentences into a set of positive ones: “Items are not located at the target warehouse” translates into “check item availability”, and “check Vendor

A's warehouse".

Step 3: Combine all positive stable states together, so the goal of the service is:

Items are located at the target warehouse.

1.2.2 Service Name: Process Withdrawal Request

In the ACME Warehouse case, the service "process withdrawal request" of the object "warehouse" has four stable states. They are:

- 1) Items are located at the target warehouse.
- 2) Items are not located within the warehouse.
- 3) Items are located at other warehouses, and transportation is arranged using an ACME truck.
- 4) Items are located at other warehouses, and transportation is not arranged using an ACME truck.

Step 1: Categorize these stable states into two types: positive and negative. The positive state is "Items are located at the target warehouse by searching the warehouse", and the negative state is "Items are not located at the target warehouse when searching the warehouse fails".

Step 2: Interpret negative sentences into a set of positive ones: "Items are located at the target warehouse" translates into "search the warehouse", "search Vendor A's warehouse", and "search branch B's warehouse".

Step 3: Combine all positive stable states together, so the goal of the service is:

Items are located at the target warehouse.

1.2.3 Service Name: Prepare Loading

There are two stable states in this service, which are:

Stable State 1: Items are ready to load at other warehouses.

Stable State 2: Items are not ready to load at other warehouses.

Step 1: Categorize these stable states into two types: positive and negative. The positive one is “items are ready to load at other warehouses”, and the negative one is “items are not ready to load at other warehouses when loading preparations fail”.

Step 2: Interpret negative sentences into a set of positive ones: “Items are ready to load at other warehouse” translates to “prepare loading”, and “dispatch stuff to complete the preparations.”

Step 3: We combine all positive stable states together, so the goal of the service is:

Items are ready to load at another warehouse.

1.2.4 Service Name: Start Loading

There are two stable states in this service, which are:

Stable State 1: Items are loaded at other warehouses.

Stable State 2: Items are not loaded at other warehouses.

The same method is used as in 4.2.3, and the goal of this service is:

Items are loaded at another warehouse.

1.2.5 Service Name: Prepare Unloading

There are two stable states in this service, which are:

Stable State 1: Items are ready to unload at the target warehouse.

Stable State 2: Items are not ready to unload at the target warehouse.

The same method is used as 4.2.3, and the goal of this service is:

Items are ready to unload at the target warehouse.

1.2.6 Service Name: Start Unloading

There are two stable states in this service, which are:

Stable State 1: Items are unloaded at the target warehouse.

Stable State 2: Items are not unloaded at the target warehouse.

The same method is used as 4.2.3, and the goal of this service is:

Items are unloaded at the target warehouse.

1.2.7 Service Name: Process Customer Pickup Requests

There are two stable states in this service, which are:

Stable State 1: Items are picked up at the target warehouse.

Stable State 2: Items are not picked up at the target warehouse.

The same method is used as 4.2.3, and the goal of this service is:

Items are picked up at the target warehouse.

1.4 Object Name: Truck Driver

1.4.1 Service Name: Complete Transport Order

There are two stable states in this service, which are:

Stable State 1: Items are unloaded at the target warehouse.

Stable State 2: Items are not unloaded at the target warehouse.

Step 1: Categorize these stable states into two types: the positive state is “Items are unloaded at the target warehouse by completing the transport order”; and the negative state is “Items are not transported to the target warehouse when the transport order fails”.

Step 2: Interpret negative sentences into a set of positive ones: “Items are unloaded to the target warehouse” is translated to “use an ACME truck”, “use a courier company”, and “use the post office”.

Step 3: Combine all positive stable states together, so the goal of the service is:

Items are unloaded to the target warehouse.

Step 5: Identify the Goal of Each Object

1. Object Name: Office Clerk

We have two types of goals of services: positive and negative. The positive type is “items can be picked up at target warehouses”; and the negative one is “items can be picked up at target warehouses”.

Step 1: Interpret negative sentences into a set of positive ones: “Items can be picked up” is changed to “other warehouse”, and “Vendor A’s location”.

Step 2: Combine all positive stable states together, so the goal of the service is:

Items can be picked up.

1.2 Object Name: Warehouse

As the object “warehouse” has more than one service, we skip step 5.1. The object “warehouse” has seven services, they are “check item availability”, “process withdrawal request”, “prepare loading”, “start loading”, “prepare unloading”, “start unloading”, and “process customer pickup request”. Accordingly, the seven combined goals of the services are “items are located”, “items are located and can be transported”, “items are ready to load”, “items are loaded”, “items are ready to unload”, “items are unloaded” and “items are picked up”. After we combine these seven goals of services together, we find that a composite item appears, that is, “items are ready for pickup”.

Therefore, the goal of the warehouse is: *Items are ready for pickup.*

1.3 Object Name: Truck Driver

We have two types of goals of service: positive and negative. The positive one is “items are unloaded to target warehouses”; and the negative one is “items are not unloaded to target warehouses”.

Step 1: Interpret negative sentences into a set of positive ones: “Items are unloaded” is translated to “other warehouses”, and “Vendor A’s location”.

Step 2: Combine all positive stable states together, so the goal of the service is:

Items are unloaded.

Step 6: Identify the Goal of the System

The whole system has four objects: Office Clerk, Warehouse, Truck Driver, and Planner. The goals of the objects are “items can be picked up”, “items are ready for pickup”, “items are unloaded”, and “transportation is arranged and a transport order is sent”, respectively. The emergent goal of the object, which is also the goal of the entire system, is: *Items are picked up.*