A Methodology to Generate e-Commerce Systems:

A Process Pattern Perspective (P^3)

Prasad M. Jayaweera

Department of Computer and Systems Sciences

Stockholm University and Royal Institute of Technology

Forum 100

SE-164 40 Kista

Stockholm

Sweden

Email: prasad@dsv.su.se

http://www.dsv.su.se/~prasad

March 27th, 2002

Abstract

In electronic commerce, systems development is based on two fundamental types of models, business models and process models. A business model is concerned with value exchanges among business partners, while a process model focuses on operational and procedural aspects of business communication. Thus, a business model defines the *what* in an e-commerce system, while a process model defines the *how*. Business process design can be facilitated and improved by a method for systematically moving from a business model to a process model. Such a method would provide support for traceability, evaluation of design alternatives, and seamless transition from analysis to realisation. This work proposes a method for the systematic transformation of a business model to a process model, which can be executed on a Process Manager. The theoretical foundations of the approach, called Process Pattern Perspective (P^3) are value theory, the language/action perspective to information systems, and enterprise ontologies.

පිදුම,

ආදරණීය මවට සහ පියාට!

Dedication,

To

Loving Mum and Dad!

Acknowledgments

I would first of all like to offer very special thanks to my academic supervisor Prof. Paul Johannesson for his excellent assistance received during my research work, and for the knowledge I gained. I acknowledge Sida/SAREC-IT Sri Lanka project[1], for financing my studies through Split Ph.D. programme.

I would also like to thank Birger Andersson, M.Sc. for reading early versions of this work and for his valuable suggestions. My gratitude goes to Dr. Petia Wohed for working closely in P^3 and for the entire Process Broker[2] research group for all support and knowledge I gained through discussions and working together. Also, I would like to thank all personnel at the Department of Computer and Systems Sciences of Stockholm University and Royal Institute of Technology for providing me all necessary facilities and a pleasant environment to do my research work.

At last but not least, I express my gratefulness to my wife Jayani and to my loving daughter Nileesha for their unwearied support and forbearance.

[1] Sida/SAREC-IT Sri Lanka Project, http://www.ruh.ac.lk/Projects/Sida_IT/sida.html
[2] Process Broker Project, http://www.dsv.su.se/~pajo/processbroker/index.html

Table of Content

1 Introduction

This chapter is mainly dedicated to introduce the background of the work presented in this thesis and a general overview of the Process Patterns Perspective (P^3) approach. Then the chapter moves to the research method followed during this work and introduces two cases discussed in this thesis.

1.1 Background

Electronic Commerce (e-commerce) is the buying and selling of goods and services electronically by consumers or by companies via computerized transactions. Replacing manual and paper based business processes with electronic alternatives and by using information flow effectively in new and dynamic ways, e-commerce has speeded up ordering, production, delivering, payment for goods and services. At the same time, e-commerce has reduced marketing, operational, production, and inventory costs in such a way that customer will benefit indirectly.

No single force embodies digital economy like the Internet. The Internet will influence and change the way we work, the way we learn, the way we do business and will change our entire lifestyle. We are experiencing these changes at a growing rate as Internet grows exponentially. The Internet Economy Indicators reports that Internet economy grew at a 173.6% from 1999 to 2000 **[61]**.

Therefore, Internet will be the technology for e-commerce as it offers easier ways to access companies and individuals at a very low cost. Around the clock presence of companies on the Web gives competitive advantage to companies' businesses. This enabling technology requires organization to build new business models directly linking customers, suppliers and other parts of their organizations, hence to build new e-commerce systems.

When building e-commerce systems, two types of models are fundamental: business models and process models **[20]**. The purpose of a business model is to describe the fundamental business aspects of the e-commerce system to be built. A business model describes which actors are involved, what the actors offer each other, and what activities they perform when producing and consuming offerings. The central concept in a business model is that of *value*, and the model describes how value is exchanged between actors **[48]**, **[49]**. Detailed discussion on the business model associated in this work is given in Chapter 4.

The business model can be contrasted to a process model, which aims at describing the operational and procedural aspects of a process and specifies the control flow of the activities carried out in a process. A process model specifies the actors involved in the operations, which activities they perform as well as the sequencing of these activities. Thus, a business model defines the *what* in an e-commerce system, while a process model defines the *how*. In Chapter 5, an introduction to our process model and in Section 5 of Chapter 7, an example process model can be found.

A business model can be seen as more basic than a process model as it specifies the declarative aspects of an e-commerce system. A natural question is, therefore, whether it is possible to move from a business model to a process model in a systematic way. Methodological support for this task would provide several benefits:

support for identifying design alternatives in process modeling, support for motivating design choice, and a clarification of the relationships between the declarative business model and the procedural process model.

In this work, we argue that it is possible to move from a business model to a process model and we suggest methodological guidelines and modeling techniques that assist a designer in the task.

A starting point of the method proposed is that much of the procedural aspects of a process model concern communication among actors. This communication is carried out in order to establish commitments among the actors to perform exchanges of values. The commitments are created by speech acts, and the control flow in a process is determined by the interleaving of these speech acts with each other and with the value exchanging activities.

The proposed guidelines and techniques are based on three building blocks:

1. A business model describing the values exchanged in an e-commerce process.
2. A formal and executable language based on communicating state machines used for modeling processes.
3. An automated designer assistant that guides a user from a business model to an executable process model.

We are proposing a methodology called Process Pattern Perspective (P^3) to design e-commerce systems. The methodology is based on primitive process patterns described in Chapter 5.

1.2 Research Methodology

In this section, we discuss the methodology that has been associated with the research work presented in this thesis.

Research Problem

The motivation behind the problem to carry out this work can be reduced to a single sentence as shown below,

> *"The business process modeling is a very complicated and time consuming task requiring lot of expertise effort"*

Our investigations into the SAP Collaborative Business Maps **[52]**, RosettaNet **[51]** and other laboratory cases are the main reasons for our motivation for addressing the above stated problem.

Research Goal

The methodology that we are proposing, which we call P^3 methodology, is for modeling business processes in the electronic commerce domain. The research goal that we are aiming at is to investigate the P^3 methodology that supports process designers with complicated business process-modeling tasks. We hypothesize that the P^3 methodology facilitates process design by overcoming much of the business process designers' burden while automating much in the development workflow.

The proposed methodology with a solid theoretical foundation is intuitively depicted in **[Fig. 1]**.

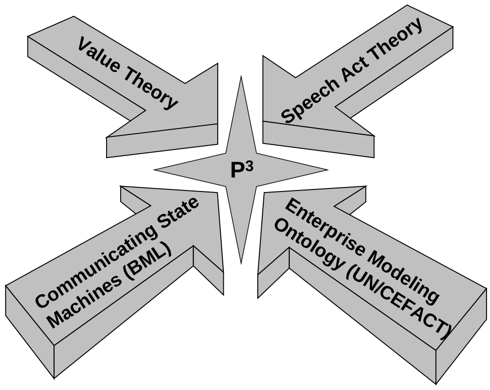


Fig. 1. Foundation of P^3

The meaning for any business is to create value for participating business partners. The means for creating value is through communication where partners request, response, acknowledge, and commit. Therefore, the development of our methodology was based on two well-established theories, the Language Action Perspective and the Value Theory.

The next challenge, after choosing foundation theories, was to identify, understand and develop the concepts that were to be used within the methodology. For this reason, we looked into different contributions in ontology development for adaptation to our work. The UN/CEFACT enterprise-modeling ontology has become a widely accepted standard in the business-modeling domain and we adopted the UN/CEFACT enterprise modeling ontology into the P^3 methodology. Finally, BML, a visual process-specification language was selected for process specification. The underlying model for BML is that of communicating state machines.

Validation of Research Goal

As the final stage, we needed to check and evaluate the methodology by conducting laboratory and real-world tests. With such evaluations, we could deduce the applicability and usability of the P^3 methodology. Due to time constraints and other difficulties, we have not carried out thorough real-world validation in this thesis. For illustrative purposes we have presented two cases: one is a very simple artificial case and the other a reduced real-world case.

As we will continue to develop this methodology our intention is to carry out a thorough evaluation of it in a learning environment and possibly also in industrial environments. In a learning environment, we will assess how students working with business process design benefit by using our methodological guidelines. The experience gathered will be taken to the industrial setting. The validation will not only include novices but also experienced and expert users of modeling techniques.

1.3 Thesis Overview

The thesis consists of nine chapters starting with an introductory chapter where the background to this work and the methodology is presented. The second chapter surveys approaches based on Language Action Perspective, one of the theoretical foundations of our work. The third chapter briefly discusses enterprise modeling ontology in general and the UN/CEFACT ontology in particular as enterprise modeling ontologies from the basis for the concepts din our framework. The fourth chapter highlights the Value Theory, which is the theoretical foundation for business idea development. In Chapter five, we explain how these theories have been adapted to the P^3 methodology.

In Chapter Six, process modeling in general and the processes modeling language, BML is explained. Also primitive process patterns used in the P^3 methodology are presented. Chapter seven goes through each stage of P^3 development process in detail using the simple case 1 examples presented in Section 1.4.1. In Chapter eight, a module of real-world case is discussed to illustrate the applicability of the approach while discussing its advantages and limitations. Concluding remarks and directions of further works listed in Chapter nine.

1.4 Case Studies

In this thesis, two case studies will be presented and discussed. The first, concerning e-catering, is an artificial case running though out the thesis mainly for pedagogical reasons when explaining different stages of the P^3 framework. This case is deliberately simplified to keep it compact so that intermediate stages of the development workflow can be easily understood. This case is presented in detail in Section 1.4.1.

The second case concerns a real European e-business selected to evaluate the applicability of the proposed methodology. The company involved in this case is presented briefly in Section 1.4.2 and the case details can be found in Chapter 7.

1.4.1 e-Caterer

The e-catering system is designed for an Internet meal catering enterprise. In the system a customer can search the menus for food and beverage from electronically published e-Catalogs. Then she can compose a meal as a package from different possible alternatives listed in those e-Catalogs and place an order on-line from the e-Caterer directly. The customer has to make a down payment together with the order placement to ensure meal delivery. Finally, she has to settle the final payment upon successful delivery of the ordered meal.

When e-Caterer receives a customer order for a meal package he first reserves and purchase beverage from beverage supplier and food from a food supplier. Then the beverage and food will be delivered to the customer in a single package.

The system for receiving customer orders has been integrated with the beverage and food purchasing systems so that those systems can place purchase orders immediately at the beverage supplier's and food supplier's systems.

The beverage supplier requests a down payment before attending to a specific beverage order. However, for the food supplier, the entire payment can be made after the e-caterer's customer paid for meal delivery.

1.4.2 transtec[1]

transtec is a leading European systems manufacturer and direct supplier of system upgrades for UNIX (DEC, SUN, IBM RS/6000, SGI, HP 9000), Windows NT systems, and network components. It was founded 1984 in Tübingen, Germany. transtec's products ranges from storage solutions, accessories and peripherals to numerous service options such as warranty extensions, support packages or on-site warranty contracts. transtec opened the 'black-box' process of order placement to delivery of product to its customer by providing access to internal merchandise management system via the web **[53]**, **[40]**.

One of the interesting features of transtec is the Web based system configurator through which customers can select different system components for the computer system that they are intended to purchase. For illustrative purposes of the applicability of P^3 methodology, we have chosen transtec "1300 Low Noise System Configurator" at Swedish branch **[64]**. The interpretation of the case that can be found in Chapter 8 totally based on our browsing through the "1300 Low Noise System Configurator", references mentioned above and our imaginations. The real situation of the module discussed here may vary a lot from our interpretations.

[1] We follow transtec's practice to write the enterprise name "transtec" in small letters.

2 Language Action Perspective

The main theoretical foundation of our work, the Language Action Perspective (LAP), is introduced in this chapter. The chapter starts with a brief survey of some common LAP approaches and highlights their distinguishing features. The chapter ends with an explanation of how LAP can be useful in e-commerce systems development in general and in the P^3 methodology particular.

2.1 Speech Act Theory

J. L. Austin **[1]** proposed the Speech Act Theory in the beginning of the 1960's. He explained that language not only refers to states of affairs in the world but also has the capability to change the world. Utterances of certain language statements constitute acts and he named those statements "performatives" or "speech acts". For example, when someone says "I promise …", "I apologize …", "I name …", the utterance immediately conveys a new psychological or social reality. Furthermore, Austin argued that the generally accepted view of truth and falsity of propositions was not applicable for many of these classes of speech acts.

2.1.1 Illocutionary Points and Illocutionary Forces

J. R. Searle **[55]** further investigated and formalized the classification of speech acts in his work during mid 1970's. He argued that it is senseless to ask whether a statement like "I promise that I meet you tomorrow" is true or false. It is only more or less appropriate in the context in which it is uttered.

Searle classified all speech acts according to one of five fundamental illocutionary points carried by all utterances, not just sentences with explicit performative verbs such as "I apologize" and "I declare". For instance, we may treat a statement like "I will do it" as a speech act promising someone to do a task in a particular context.

The five categories of speech acts with different illocutionary points are according to Searle:

Assertives: the purpose of which are to convey information about some state of affairs of the world from one agent, the speaker to another, the hearer. Examples of assertives are "It is raining" and "A lecture is in progress".

Directives: where the speaker requests the hearer to carry out some action or to bring about some state of affairs. "Please bring me coffee" and "I order you to leave the class" are examples of directives

Commissives: the purpose of which are to commit the speaker to carry out some action or to bring about some state of affairs. Examples of commissives are "I promise to meet you tonight" and "I'll make it for you"

Expressives: the purpose of which are to express the speaker's attitude about some state of affairs. Examples of expressive are "I like tea" and "I am satisfied with your service".

Declaratives: where the speaker brings about some change of state of affairs by the mere performance of the speech act. "I hereby pronounce you husband and wife" and "I hereby baptize you to Samuel" are examples for declaratives.

Searle differentiated between *illocutionary point* of an utterance, its *illocutionary force* and its *propositional content*. A statement "I promise that I meet you tomorrow" can be analyzed to "I promise" as indicator of its illocutionary force and "I meet you tomorrow" as its propositional content. There may be situations where speech acts with the same illocutionary point may differ in their illocutionary force (manner and degree). For instance, a polite question and a demand for information with same directive illocutionary point and same propositional content may differ in their illocutionary forces.

To get much use out of Speech Act Theory in modeling real communication situations, it has to be adapted and put in a modeling framework. A way of adapting the theory is to group elementary speech acts into different complex action patterns. These patterns can then be used to model, for instance, the coordination of actions in organizational settings.

The following sections describe a few different modeling frameworks that use adaptations of Speech Act Theory. Presented in Section 2.3 is the Conversation for Action, in Section 2.4 the Action Workflow Loop, in Section 2.5 the Dynamic Essential Modeling of Organization (DEMO), in Section 2.6 the Business Action Theory (BAT), and finally in Section 2.7 the Layered Transactional Patterns.

2.2 Conversation for Action

Conversation for Action is a well known example of an adapted application of Speech Act Theory. It was proposed by T. Winograd and F. Flores **[74]** in 1986.

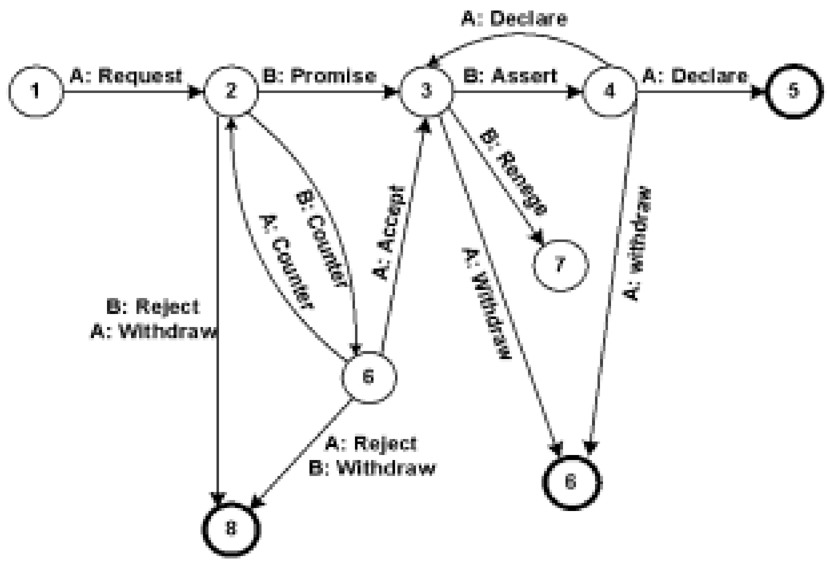


Fig. 2. A State transition Diagram of a Conversation for Action

The Conversation for Action is a generic schema where successive speech acts are related to each other forming a network of speech acts like the one in **[Fig. 2]**. Each circle represents a possible state of the conversation and arrows represent transitions accomplished by speech acts. With the request from initial speaker (A) to hearer (B), a transition is made from state 1 to state 2. In the above state transition diagram, there is a finite number of transitions that the conversation can take from a given state.

In the path showing successful completion of a conversation, B assert to A that the conditions of satisfactions have been met (state 4) and if A declares she is satisfied the conversation terminates successfully at the termination state 5. Note that there are also possible conversation failure termination states, for instance when a withdrawal of request from A leads to termination state 8 in the diagram.

2.3 Action Workflow Loop

Action Technologies **[45]** developed their speech act based modeling approach within Business Design Language. They extended the Conversation for Action pattern from Section 2.3 to a four-step Action Workflow Loop, which is used as the basic modeling unit.

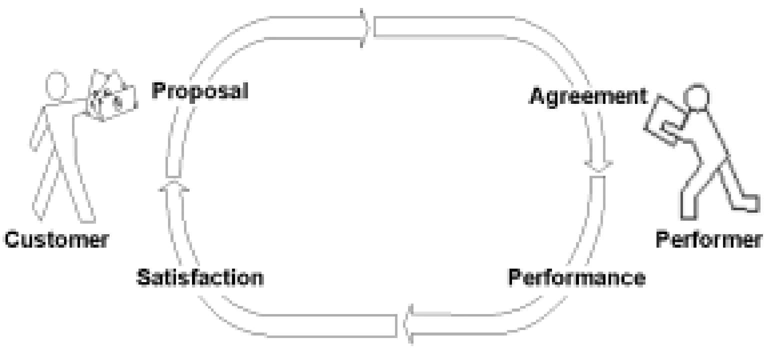


Fig. 3. A basic Action Workflow Loop

The above diagram shows the basic sequence of phases in the Action Workflow Loop. There is always an identified customer and a performer for the completion of a task as in **[Fig. 3]**.

The four phases are:

1. **Proposal**
   The customer requests (or the performer offers) completion of a particular action according to some stated conditions of satisfaction.
2. **Agreement**
   The two parties come to mutual agreement on the conditions of satisfaction, including the times by which further steps will be taken.
3. **Performance**
   The performer declares that the action is completed.
4. **Satisfaction**
   The customer declares that the completion is satisfactory.

There are possibilities to model additional actions at any phase of the Action Workflow Loop e.g. to include further negotiations for clarifying satisfaction conditions or changes of participants commitments. A detailed analysis of these further negotiations can be found in **[74]**.

The key difference between traditional workflow approaches and the Action Workflow Loop is the shift from task or information flow oriented action coordination to request and commitment oriented action coordination. That is, business processes are modeled as networks where different Action Workflow Loops are connected by links at different phases of the loops. See **[45]** for more details of business process modeling with networks of Action Workflow Loops.

The Action Workflow Loop is the main foundation and inspiration of the business process patterns that are proposed in this thesis.

2.4 Dynamic Essential Modeling of Organization (DEMO)

Dynamic Essential Modeling of Organization (DEMO) **[6]** is a reengineering and development methodology that offers concepts and modeling techniques for business processes. In DEMO, the construction of business is viewed as business transactions on three levels: the documental, the informational and the essential. A business transaction at higher level allows multiple realizations at the lower levels as shown in **[Fig. 4]**.

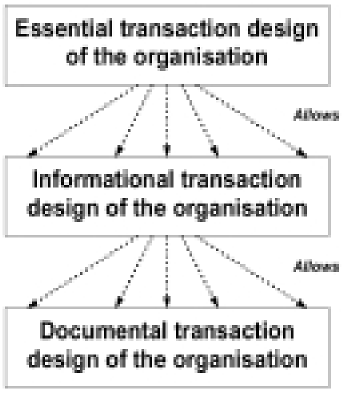


Fig. 4. DEMO Transaction design and levels of abstraction

At the documental level, an organization is viewed as a system of actors that produce, store, transport, and destroy documents. In other words, at the documental level the substance and form by which coordination becomes visible is considered. At the informational level, one abstracts from this substance and form (i.e. documents) and focuses on the actual meaning of documents. The organization is considered as systems of actors that send and receive information, and perform calculations on this information in order to create derived information.

The essential business transaction is a core concept in DEMO and it is performed by two actors: the Initiator and the Executor. The DEMO transactions passes through three phases: the Order (O) phase, the Execution (E) phase and the Result (R) phase. In the O-phase two actors come to an agreement about the execution of some future action through an *actagenic conversation*. In the E-phase, the negotiated action is executed. In the R-phase, actors negotiate an agreement about the result of the execution through a *factagenic conversation*. These phases are visualized in **[Fig. 5]**.

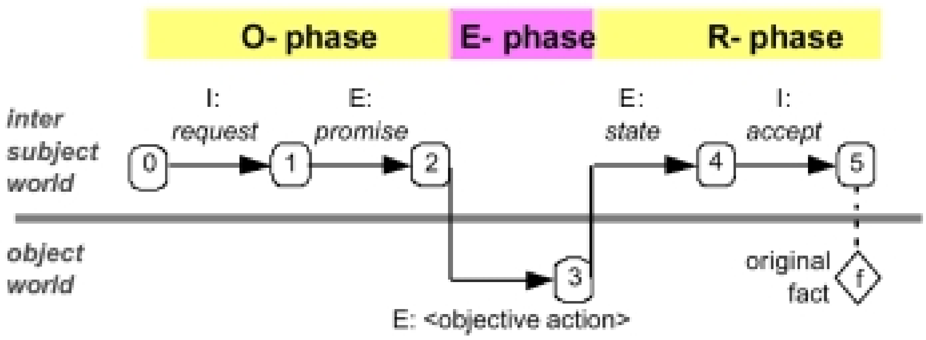


Fig. 5. The basic pattern of the DEMO transaction (adapted from **[9]**)

The successful execution of a transition in the subject world (the world of communication) results in a change in the object world (the world of facts) in which actors exist.

There are several method components of DEMO to represent structure of business transactions of an organization graphically. These structures of business transactions are modeled in five different partial models: the ***interaction model***, the ***process model***, the ***fact model***, the ***interstriction model***, and the ***action model***. Each of these models can be developed incrementally.

The interaction model captures the transaction types, and the actors involved in an organization as either initiator or executor of business transactions. The process model captures the causal and conditional relationships within transaction types, and the individual transaction scenarios. The fact model represents a complete and precise state space of the object world. The interstriction model specifies actors and the information needed for these actors to execute transaction types and finally the action model comprises the most detailed specification of the transaction structures of an organization.

2.5 Business Action Theory

The Business Action Theory (BAT), **[17]** is generic business action logic for business design, founded on communicative action theories. The fundamental idea in BAT is that a business always consists of customers and suppliers performing communicative and material actions. The framework captures business processes by means of six phases as listed in **[19]**:

1. ***Business prerequisites phase***, where prerequisites are established (both within the supplier's and customer's organization) for performing business (sales/ purchases)
2. ***Exposure and contact search phase***, where both parties, customer and supplier, seek contact. The suppliers' ability is offered and exposed to market. The customers' lacks and needs create demands.
3. ***Contact establishment and proposal phase***, where the supplier presents available and possible offers to a specific customer showing some needs and purchase interest.
4. ***Contractual Phase***, where the supplier and customer make commitments that are shown in an order from the customer and an acknowledgement of order from supplier.
5. ***Fulfillment Phase***, where the supplier and customer fulfil their commitments. The supplier fulfils the commitment by performing delivery and customer fulfils by paying for the received delivery.
6. ***Completion phase***, where the customer and supplier reach satisfaction or dissatisfaction. That is the customer uses delivered products with satisfaction and the supplier is happy with the payment for the delivery or certain claims are raised due to dissatisfaction of either party.

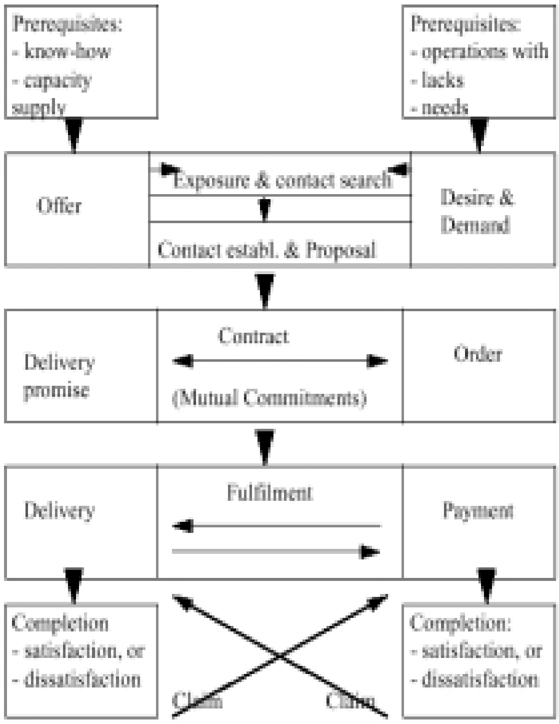


Fig. 6. Business Action Theory: The six generic phases of business processes (adapted from **[2]**)

The six generic phases of business processes in BAT and their relationships have been depicted in **[Fig. 6]**. As in DEMO, it comprises of different modeling components. They are ***Problem Analysis***, ***Goal Analysis***, and ***Strength Analysis*** as methodology for business process analysis and ***Action Diagrams***, ***Process Diagram*** for activity and process (re)-construction.

Only the Process and Action diagrams will briefly be discussed here. Action diagrams integrate flow orientation (describing information and material flow) and action orientation (describing the types of action performed) in a single description. Each action diagram therefore describes a business context within a business process and can be linked to other such action diagrams with descriptive connectors. Elementary descriptive objects that are being modeled are information, material, actions, activities, performers, and flows of information and material.

The process diagrams are the key maps of business processes and the methodology uses a bottom-up approach where elementary actions are grouped into process components. Examples of such processes are customer-to-customer, side processes and sub-processes. Each business process is assumed to have at least one customer-to-customer process and possible side processes. The customer-to-customer processes capture activities between a supplier and a specific customer e.g. customer inquiry or order placement. The side processes are supportive of the customer-to-customer processes as enablers or in other ways play a part in their performance. Both customer-to-customer and side processes may consist of sub processes that consist of, among other things, several contextually related activities.

2.6 Layered Transactional Patterns

In the layered transactional patterns for e-commerce, Weigand **[70]** distinguishes between five levels of (communicational) analysis meta-patterns from the lowest level of speech acts to the highest level of scenarios **[Fig. 7]**.

Fig. 7. Levels of Meta-Analysis Patterns (adapted from **[70]**)

2.6.1 Speech Act

The speech act is the lowest level elementary unit within the communication between subjects in methods based on language action perspective and representation languages such as Formal Language for Business Communication (FLBC) **[46]**. It consists of the propositional content, the illocutionary point and the illocutionary force. An example of a speech act in FLBC is given below.

```
Msg(pers(cus1), pers(sup3), request, delivery-product, mesg157)
```

"delivery-product" is the propositional content, the first "pers(cus1)" is the speaker from whom the message originates, the second "pers(sup3)" is the hearer to whom the message is directed, the "request" is the illocutionary force, and finally "mesg157" is the message identification number.

2.6.2 Transaction

In real world, speech acts typically occur in pairs, for example a commitment follows a request. Some says that it is not the speech act but a message pair, which is the basic unit in communication. Those message compositions are referred to as transactions in Weigand's work **[72]** where a transaction is defined as the smallest possible sequence of actions that has an effect in the social world of the participants, e.g. obligation, authorization, accomplishment.

A transaction is defined as a set of communicating subjects, communicative actions, constraints on the sequence of these actions and the goal and exit states. In FLBC an instance of a transaction can be defined as shown below.

```
Trans(

   [person(cus1), person(sup2)],
```

```
[msg(pers(cus1), pers(sup2), request,  delivery_product, msg3),

   msg(pers(sup2), pers(cus1), promise, delivery_product, msg4)],

[before(msg3, msg4)], trans5)
```

2.6.3 Workflow Loop

At the next level the Workflow Loop is defined. It is analogous to the DEMO transaction and the Action Workflow Loop of Action Technologies Inc. The Workflow Loop defined here corresponds to Winograd's Conversation for Action pattern and specifies the participants involved and set of transactions (in most cases two).

A Workflow Loop definition in FLBC can be given as follows.

```
Wfltype delivery_product(initiator($i), executor($e), product($p),
date($d)) ==

  ([person($i), person($e)], [request_product($I, $e, $p, $d),
delivery_product($e, $I, $p)] )
```

2.6.4 Contract

Contracts are widely discussed in the literature but in Wiegand's work they are used in the sense of two workflow loops with reciprocal transaction patterns. Different types of contracts have been distinguished for different types of conversations that may be intertwined with each other.

At instance level and directly in terms of FLBC messages, examples of such contracts can be found in **Table 1**.

Consumer-Supplier Transaction	Supplier-Consumer Transaction
Msg(pers(cust), pers(supp), request, delivery_product_X, msg1)	Msg(pers(supp), pers(cust), request, payment_of_money_for_X, msg2)
Msg(pers(supp), pers(cust), promise, delivery_product_X, msg3)	Msg(pers(cust), pers(supp), promise, payment_of_money_for_X, msg4)
Msg(pers(supp), pers(cust), assert, delivery_product_X, msg5)	Msg(pers(cust), pers(supp), assert, payment_of_money_for_X, msg6)
Msg(pers(cust),pers(supp), accept, delivery_product_X, msg7)	Msg(pers(supp), pers(cust), accept, payment_of_money_for_X, msg8)

Table 1. Instance level Contract deifned with FLBC messages.

The order in which the different communicative acts in a contract is uttered is dependent on the trading procedure that the involved parties has agreed upon. The semantics of these contracts have been addressed in **[71]** by means of Petri-nets.

2.6.5 Scenario

The scenario is the context in which a conversation can be understood. Defined at this level are the identities of speaker and hearer, physical and other incidental circumstances of time and place, the object of the conversational exchange, and the probable intentions of the speaker and hearer.

The structure of the scenario is the minimal story element or narrative function, composed of a begin, a development, and an end. This structure is valid for commercial transactions as well. The scenario (stories) have been distinguished as meta-patterns since they do show structures. These meta-patterns has normally the following form: identification - essential transaction - ending of relationship. These meta patterns can be used and reused profitably in electronic commerce once they are made available in a well documented form.

Incomplete scenario type definition can be given as below for an example.

```
 ScenarioType credit purchase;

   (domain IC(subject [person($customer), person($supplier),
person($bank)], identification["Chamber.of Sweden"], (community
club($consumer_society), ..], law["Swedish Law"]);

   contract([
```

```
        supplier/customer($supplier, $customer)]
        supplier/bank($supplier, $bank)
        customer/bank($customer, $bank)]);
     tranactions([
        deliver_goods($supplier, $customer, $good, $date)
        …]);
     termination
        termination_relation([$consumer_society, $customer)
  ])
  )
```

2.7 Alternative Generic Layered Patterns

In this section we briefly discuss Lind et al.'s **[39]** criticism of the layered transaction patterns explained above and their alternative proposals for each layer. Lind et al. argue against the speech act as being the unit of analysis. Some defects such as conceptual and terminological confusions and inability to derive higher levels in the Wiegand's hierarchical layered architecture are also pointed out. Lind et al.'s alternate hierarchical architecture is shown in **[Fig. 8]**.

Fig. 8. Layers of Generic Patterns for Business Modeling (adapted from **[39]**)

2.7.1 Speech Act vs. Business Act

While appreciating the idea of starting from a basic unit of analysis with the possibility to use it as a component when constructing higher layers in the hierarchy (as Weigand claims), the criticism is built on selecting speech acts only for this purpose. This is because of the reduction of business interactions to speech acts excludes possible material acts.

Instead, at lowest level ***Business Act*** has been proposed as the basic unit of analysis. A Business act can be a communicative and/or material act performed by someone (customer/supplier) aimed towards someone else (customer/supplier).

2.7.2 Transaction vs. Action Pair

Here the possible confusion of usage of the term "transaction" in Wiegand's work in to other language and communicative action perspectives has been highlighted. Also the elimitation of speech act combinations that lead to deontic state changes have been argued against as there are communicative acts that do not lead to deontic state changes. (find more about deontic in **[41]**, **[42]**, **[10]**) For example, a formal ordering of a product can be considered as a communicative act that leads to deontic state change while, showing interest of purchasing a product is, a communicative acts that doesn't lead to deontic state change.

Suggested as an alternative is, an ***Action Pair*** which is a group of two business acts where one business act functions as a trigger for another act which functions as a response.

2.7.3 Workflow Loop vs. Exchange

The construction of Workflow Loop layer upon asymmetry between the two parties involved in the conversation has been questioned. That is, in this closed pattern for a certain goal, ignorance of genuine business character of exchange actions has been questioned.

One or more action pairs can be grouped into an ***Exchange*** between actors. The intuition behind an exchange is one actor giving something in return for something given by another actor. The important feature of an exchange is that the business acts of constituting action pairs must be of the same type. A list of different exchanges can be found in Section 2.7.4 below.

2.7.4 Contract vs. Business Transaction

The usage of the term contract covering entire business transaction has been argued. Though a business transaction is an essential part within a contract but it is not the only part. Also the derivation of this layer from lower transaction layer where communicative acts that do not leads to deontic state changes has been ignored, is questioned.

Lind et al. propose a corresponding Business Transaction layer. This is a pattern of different types of Exchanges related to each other in a business transaction. Such a collection of exchanges in a business transaction can be listed as below;

1. Exchange of Interests
2. Exchange of Proposals
3. Exchange of Commitments
4. Exchanges of Value
5. Exchange of Assessments

2.7.5 Scenario vs. Transaction Group

Lind et al. are considering the scenario as a horizontal expansion rather than a vertical abstraction on the lower layers. At the higher levels, additional artifacts from the business context that were not considered at lower levels have to take into account. They questioned again on derivation of multi-party involvement at such higher level from lower layers where only two party involvement have been considered.

The suggestion they are proposing is ***Transaction Group***. At transaction group is grouped recurrent transitions that needs to be framed within wider agreements. The objectives of these long-term agreements are to establish, sustain, and develop business relations. Lind et al. have restricted their definition of transaction group to include two parties (Customer and Supplier).

2.8 Hindering Factors for Effective usage of LAP in e-Commerce

In e-commerce, business transactions are carried out using IT as a medium. The use of IT enables transactions to be carried out rapidly and at a low cost. As a consequence, new ways of working, new forms of organisation, and new business models are emerging, such as virtual enterprises, integrated supply chains, and value networks. A common theme is that of inter-organisational co-operation and communication. Business processes are not carried out within a single organisation but across organisational boundaries. As noted in **[73]**, inter-organisational processes have two distinguishing features. First, the resources needed for a process cannot be assigned centrally as they reside in different organisations. Second, the organisations involved in a process have a certain degree of autonomy meaning that no central authority has control over all the co-operating organisations. These features of processes in an e-commerce setting imply that in order to build effective IT-systems, it is required to explicitly model and manage communicative, institutional, and deontic notions **[10]** such as request, acknowledgement, commitment, obligation, responsibility, and trust. Thus, the Language Action approach to communication and information modelling seems to be a most promising framework for designing e-commerce systems. However, the penetration of the approach in industrial practice is still low although there exists a comprehensive body of theoretical as well as applied research in the area, **[74]**, **[73]**, **[18]**, **[7]**, and. **[35]**.

The limits of the applicability of the Language Action approach have been widely discussed in academia, e.g. the Suchman/Winograd debate, **[57]**. We acknowledge the importance of the arguments put forward in these discussions, but we believe they are less relevant in e-commerce settings as e-commerce processes are more formalised and structured than many intra-organisational work processes. We would like to add the following three factors to the list of hinders for effective use of the approach. The added factors are based on our experience in industrial case studies as well as in undergraduate teaching.

1. Using the Language Action approach for process modelling easily encourages a low-level perspective where the modelling quickly focuses on communicative acts like requests, replies, acknowledgements, cancellations, etc. Managers often experience this level as too detailed and not an adequate starting point for understanding the business objectives motivating the process design.

2. The underlying notions and terminology of the Language Action approach are unfamiliar to most users and designers. They find it difficult to reason and communicate using the specialised terminology.

3. There is a considerable distance between Language Action models and executable systems. After having designed a process model using the Language Action

approach, there is still much design and implementation work to be done before an executable system is completed.

Presented in this thesis is a methodology to overcome the aforementioned hindering factors.

3 Enterprise Modeling ontology[1]

In this chapter we introduce the Enterprise Modeling ontology underlying our work. It is an extension of the Resource-Event-Agent (REA) **[15]** ontology to accommodate the Language Action Perspective. The chapter begins with a general discussion of ontology, a brief description on some efforts in development of Enterprise Modeling ontology and then moves to the specifics relevant to the thesis.

3.1 A Definition of ontology

"An ontology is an explicit specification of a conceptualization" is a widely accepted definition by Gruber **[23]**. This definition is an elaboration on "An ontology is the object, concepts, other entities that are assumed to exist in some area of interest and the relationships that hold among them" found in Genesereth and Nilsson's work **[16]**.

Although Gruber's definition has much in common to the traditional description of a database conceptual schema, it differs in at least three important ways: objective, scope and content **[15]**. The objective of an ontology is to represent a conceptualization that can be shareable and reusable irrespective of any particular application. The scope of an ontology is to cover all applications in a domain, not just a specific one. Finally, the content of an ontology is an explicitly specified and constrained knowledge specification from which further knowledge can be inferred by application of rules.

There are two primary advantages of developing an ontology: increased knowledge about the domain being modeled and benefits from the resulting models. These benefits include a common terminology to be used in the domain, reference models for planning and controlling processes, etc.

3.2 A Classification

Guarino **[26]** has classified ontology according to two dimensions: their ***level of detail*** and their ***level of dependence*** on a particular task or point of view. In level of detail, he distinguishes between *reference ontology* (or *off-line ontology*) that holds sophisticated theories accounting for the meaning of terms used and *shareable ontology* (or *on-line ontology*) that holds very simple ontology agreed by all users.

In level of dependence, Guarino distinguishes between the following three levels as shown in **[Fig. 9]**.

[1] We have followed Guarino's distinction in using term 'ontology' as in [25].

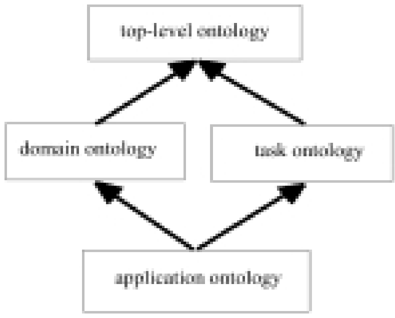


Fig. 9. Kinds of ontology according to their level of dependence (adapted from **[26]**)

1. *Top-level ontologies* describe very general concepts like space, time, matter, object, event, action, etc., which are independent of a particular problem or domain
2. *Domain ontologies* and *task ontologies* describe, respectively, the vocabulary related to a generic domain (like medicine or automobiles) or a generic task or activity (like diagnosing or selling), by specializing the terms introduced in the top-level ontology.
3. *Application ontologies* describe concepts depending both on a particular domain and task, which are often specializations of both the related ontologies. These concepts often correspond to *roles* played by domain entities while performing a certain activity, like *replaceable unit* or *spare componen*t.

3.3 Categories of ontology

With respect to IT, the research on ontology can be categorized into two classes as, Generic Enterprise ontology (GEM) and Deductive Enterprise ontology (DEM). The ESPRIT program's CIMOSA **[4]**, DOD-wide GEM of US Department of Defense **[11]** and other similar efforts have been categorized into Generic Enterprise Models (GEM). GEM is collection of concepts and concept relationships across a type of enterprise such as manufacturing or banking.

Contrasting these approaches are the Deductive Enterprise Models (DEM). The distinguishing feature of DEM is its ability to automatically deduce answers to many "common sense" questions. The artificial intelligence and knowledge management communities have contributed a lot in developing enterprise ontologies based on DEM as a result of efforts in achieving common sense reasoning on knowledge bases and agent communication.

The CYC project at MCC **[38]**, **[5]**, the Enterprise Project at University of Edinburgh **[12]** and the TOVE project at University of Toronto **[63]** are three noticeable projects in the area of DEM.

In early 1980's one of the recognizable efforts for creating industry wide standard for enterprise modeling can be found in US Air Force's ICAM (Integrated Computer-Aided Manufacturing) project. ICAM resulted in different Integration DEFinitions (IDEF), **[28]**: IDEF0 for functional and activity modeling, IDEF1 for information modeling, IDEF1x for data modeling, IDEF2 is dynamic modeling method for simulation, IDEF3 for process description capturing, IDEF4 for object oriented design, IDEF5 for ontology capturing.

The IDEF5 ontology development process consists of the following five activities.

- **Organizing and Scoping.** The organizing and scoping activity establishes the purpose, viewpoint, and context for the ontology development project, and assigns roles to the team members.
- **Data Collection.** During data collection, raw data needed for ontology development is acquired.
- **Data Analysis.** Data analysis involves analyzing the data to facilitate ontology extraction.
- **Initial ontology Development.** The initial ontology development activity develops a preliminary ontology from the data gathered.
- **ontology Refinement and Validation.** The ontology is refined and validated to complete the development process.

The Toronto Virtual Enterprise (TOVE) is discussed in brief here as it is relevant for the e-commerce domain. TOVE was targeted to achieve four main objectives as listed below:

1. To provide a shared terminology for the enterprise that each agent can agree upon and understand
2. To define the meaning of each term in as a precise and unambiguous manner as possible using First Order Logic.
3. To implement semantics in a set of axioms that will enable TOVE to automatically deduce the answer to "common sense" questions about the enterprise
4. To define a symbology for depicting a term or a concept graphically

Therefore, ontology can be seen in three different perspectives as mentioned in **[14]** and in **[Fig. 10]** below.

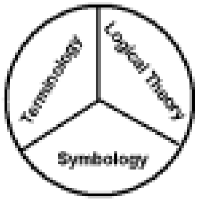


Fig. 10. Main Component of ontology (adapted from **[14]**)

In TOVE, the concepts of an ontology is classified into different subsets.

1. Process and Activities: includes representations of state, time, and causality.

2. Resource and Inventory: general representation of resources, inventory, location, etc
3. Organization Structure: representation of position, role, departments, processes, goal, constraints, etc
4. Product Structure and Requirements
5. Quality: basic representation in support of ISO9000, QFD **[50]**, etc.
6. Cost: representation of resource cost, activity cost, activity based costing, etc

3.4 REA ontology

Resource-Event-Agant (REA) **[44]**, **[56]** business model originates from the field of accounting. The fundamental idea behind REA is exchange of resources - give up some resources to obtain others.

The minimal ontology of the REA model can be visualized as in **[Fig. 11]**. It shows the minimal set of concepts in a business relationship. The same minimal REA model can be extended to accommodate other concepts, which may be needed for a particular industry or for a particular situation.

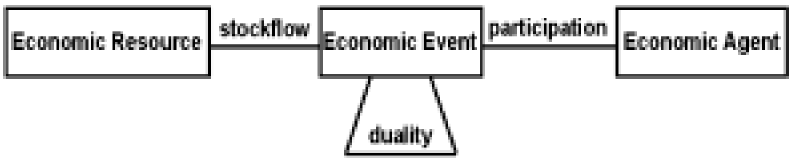


Fig. 11. The minimal REA model in UML

The REA model captures three intrinsic aspects of exchanges: the required *events*, the *resources* that are the subjects of the exchanges, and the participating *agents*. The *duality* represents the reciprocity relationship between inflow Economic Event and outflow Economic Event. There are two more associations in the minimal basic diagram. First, *stockflow* is the connection between Economic Resource and Economic Event that describes the movement of resources within an exchange. Finally the *participation* is the relationship between the agents involved in Economic Event. These involved agents can be either inside agents, which are accountable inside parties (say employees) or outside agents which are external parties (say customers).

The P^3 modeling methodology is centered on the base partner to whom a designed solution will be delivered. A base partner (always an economic agent) can be for instance a single person, an enterprise or an organization of employees within an enterprise.

With respect to base partner, two types of stockflow can be defined: the *inflow* economic event is an acquisition event where we take a resource (say cash) and the *outflow* economic event is consumption event where we give up a resource (say a finished good).

REA differentiates between two types of dualities: *transfer* duality and *transformation* duality. The value is created in transfer duality by market transactions usually with outside partners. Transformation duality creates value through changes in form or substance of a resource mainly within an organization.

In congruent exchanges, both inflow and outflow economic events take place simultaneously in space and time. e.g. cash-sales. Therefore, it has been argued in **[15]** for differentiation of congruent exchanges from duality in the ontology. But in our work we have left out the space and time dimensions at business modeling and treats congruent exchanges as a duality corresponding to two distinguishable economic events.

4 e-Business Model

This chapter introduces the most fundamental and initial models that one has to design in an e-commerce system development process. Here, we briefly discuss foundations for such e-Business models and how they differ from traditional business models.

4.1 Introduction

The distinguishing feature of an innovative e-Business project is that it needs to be completed fast and with intensive development effort in order to reach the market in time. For successful realization of an e-Business project, different categories of stakeholders (technical, business, and end user) have to agree on the feasibility of the innovative business idea being designed at a very early stage of the development process. During the short development time period not only the business design but also the implementation of the e-commerce system has to be done.

The proposed P^3 framework provides the designer with not only methodological guidelines but also with an automated design and implementation assistance for the development of e-commerce systems. Our intention of the framework is to assist e-commerce system designers with fast and reliable system delivery while facilitating her process modeling tasks.

4.2 The Value Concept

The main foundation of the business model is the concept of value. It has been analysed extensively in the economics and marketing literature for centuries.

A significant work can be found in Porter's competitive advantage series **[49]**. He builds the concept of value chain through which value is successively added to products to win a targeted customer. The value chain divides a company's activities into the technologically and economically distinct business activities which ultimately create value for the company. The physical creation of the product, its marketing and delivery to buyer, and its support and servicing after sale are some primitive value activities.

The challenge for any (electronic) commerce application is to do profitable business where the price for goods/services sold is higher than the production cost. This is done, according to Porter, by performing value adding activities at lower cost or performing them in a way that leads to differentiation from similar products so that customers will be ready to pay a premium price. Achieving this leads to competitive advantage.

The success of a product or service introduced to a competitive market is the basis of the survival of a company. This can be determined by relationships of the popular market triangle proposed by Ohmae **[47]** depicted in the diagram below **[Fig. 12]**. It is possible to achieve competitive advantage in terms of successful marketing when one’s offer is targeted to goal system of consumers (customer orientation) and is held by consumers to be better than competing offers.

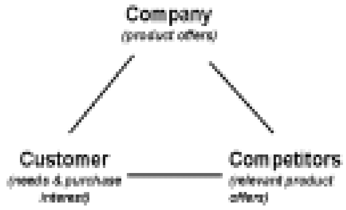


Fig. 12. Marketing Triangle

Consumer value is central for every successful marketing strategy in a market economy. An interesting and significant collection of contributions in the direction of consumer value can be found in **[27]**. There, Holbrook defines consumer value as “an interactive relativistic preference experience”. The evaluation of some *object* by some *subject* is called consumer value. In a typical case, a subject could be the consumer or customer while the object could be a product or a service offered by Manufacturing/service Company respectively.

The term “interactive”, in Holbrook’s definition of consumer value, means that consumer value entails an interaction between some subject and some object. This interaction has led to two schools: subjectivists and objectivists side of interaction.

The subjectivist argues that consumer value depends entirely on the nature of subjective experience, i.e. “man is the measure of all things”. This is the basis for customer orientation where a product is assumed to have value only if it pleases some customer or put simply, the customer is the ultimate arbiter of consumer value.

The objectivist argues that value reside in the object itself as one of its properties. These arguments have led to product orientation assuming that value is put into the offering by virtue of a certain resource, skill or manufacturing efficiencies. The classical economists including Karl Marx has contributed to the labor theory of value that specifies the value of an object as the amount of work invested in producing it.

The term “relativistic”, in Holbrooks definition of consumer value, means that consumer value is comparative, personal, and situational. Comparative is the value of one object compared to another when evaluated by the same individual. Here Holbrook has highlighted intra-personal comparisons rather than inter-personal comparisons. Personal means that the value of one object varies from individual to individual according to subjective preferences. Situational means that the value of one object depends on the context in which the evaluative judgment is reached. Finally, neither he states that the possession of the purchased product, nor the selection of the brand is the value but the consumption experience. This is the central point to treat all markets as service marketing when creating consumer value.

4.3 The Business Model

In an e-commerce systems development process, the initial phase includes the development of a business model. The fundamental objective of the business model development phase is two folded: The business idea being designed will be satisfactory to all involved parties and the technical feasibility of the realization of the business idea on the available IT platform will be determined.

The central concept of a business model in any trading set up is value. We assume that value can be created and it can be exchanged as economic resources among business partners. Among the objectives of a business model, answers to the following questions are essential.

1. What types of involved Business Partners are there?
2. What types of Economic Resources exchanges are there?
3. Which Business Partner offers what Economic Resource type to whom and in return for what Economic Resource type?

5 Theoretical Foundation of P³ Framework

This chapter presents the theoretical foundation of P^3 framework. In Section 1, adaptation of LAP within the framework and in Section 2, associated modeling ontology have been discussed. Finally, the P^3 business model based on value theory is presented.

5.1 Adoption of LAP in the P³ Framework

The ultimate goal of the P^3 framework is to generate an executable process model. The main enabler of the generation of this final process model is the process patterns that we are introducing in detail in Chapter 4. The process patterns can be grounded in LAP theories. At a low level these process patterns can be considered as different types of conversations designed to reach specific goals.

As a central concept in our approach, we have identified a specific role, called base partner to whom the final e-commerce system will be delivered. The term base partner in our work has been used in a broad sense meaning it can be single enterprise, a coalition of enterprises offering joint services and products or a company employee interacting with a customer. All process patterns are defined with respect to the base partner in the design situation.

This view has led to two fundamental process patterns where the base partner interacts with customer to offer her deliverables and where the base partner interacts with supplier to acquire necessary materials to make it possible for her to deliver to a customer. We have also defined two more additional intermediate process patterns, one that interacts with customers and suppliers when a counter offer situation arises and one that interact with suppliers for the reservation and booking of necessary materials.

In the P^3 framework, it is also possible to construct a hierarchical architecture that represents concepts governing underlying concepts. The diagram, **[Fig. 13]** below shows the P^3 layers. The objective of this four layered architecture is to understand context and serve the design of the business model.

Fig. 13. Hierarchical Architecture of P^3 Framework

We have followed the UN/CEFACT meta-business models **[62]** and proposals of the e^3 framework **[21]** (detailed discussion about these approaches can be found in Chapter 3) in designing our business model. Though the original ideas borrowed from those two approaches, they have not paid any consideration on LAP orientation in their business process analysis.

As the P^3 framework has one strong foundation in LAP concepts, we have analyzed and interpreted the business model with respect to LAP. In Section 5.2 a brief discussion on these layers and other concepts can be found.

5.2 Extended P^3 UN/CEFACT Modeling ontology

Techniques and Methodologies Working Group (TMWG) of United Nations Centre for Trade Facilitation and Electronic Business (UN/CEFACT) proposes UN/CEFACT Modeling Methodology (UMM) to model business processes and to support the development of existing and "The Next Generation" of Electronic Data Interchange (EDI) for e-Business.

The main objective of UMM is to capture common business practice into standardized business models. This will enable small and medium sized companies to engage in emerging e-Business practices in a protocol neutral and future proof manner independent of proprietary technologies.

The section below describes most of the ontology that UN/CEFACT proposed for their metamodels **[62]**. The original UN/CEFACT modeling ontology has been extended to suit the purpose of our work by including Language Action Perspective concepts that were discussed in detail in Chapter 2.

Partner
A partner is an independent economic and/or legal entity, e.g., John Doe, Stockholm University.

Partner Type
A partner type is the abstract classification of definition of a partner, e.g. Customer, University.

Economic Resource
An economic resource is a quantity of something of value (see Chapter 4 for a detailed discussion on value) that is under the control of an enterprise. The economic resource is transferred from one partner to another in an economic event, e.g. a car, cash, work performed by man or machine.

Economic Resource Type
An economic resource type is the abstract classification or definition of an economic resource, e.g. ItemMaster or ProductMaster of Enterprise Resource Planning (ERP) system.

Business Event
The business event is the basic unit in our work and can be treated as a performance of act as defined by Lind et al., i.e. it can be communicative and/or instrumental act that makes a state change of one or more entities within a business. There will be one

explicit originator who performs the business event and one recipient who is the beneficiary party of the performance. For an example in placing an order for a product, the ordering customer can be considered as originator and the sales organization of that product is the recipient.

Economic Event

An economic event is a business event, which specifically transfers control of an economic resource from one partner type to another type. A cash payment, shipment, and sale are examples of economic events.

We have noticed that some economic events can be further decomposed into ***atomic economic events***. A cash payment in installments or a delivery of a bulk order in batches are good examples. In the first case, an installment and in the latter case, a batch delivery are atomic economic events.

Duality

The duality is a relationship between two economic events, where one is the legal or economic consideration of the other. This corresponds to the value offering proposed by Gordijn **[20]** based on the "one good turn deserves another" principle. If T1 and T2 are two economic events such that T1 is the economic event transferring an economic resource from partner P1 to partner P2 and T2 is the corresponding economic event transferring an economic resource from P2 to P1, then duality represents the reciprocity between T1 and T2. That is, one partner is providing another with some thing of value and receiving some thing of value in return.

Economic Commitment

The economic commitment is an obligation to perform an economic event at a future point in time. An order line can be treated as a commitment where requestor commits to pay the mentioned price upon receipt of ordered item. There is a mandatory reciprocity relationship between two or more economic commitments. That is, in the above example, the requestor's commitment has reciprocity relationship with the supplier's commitment to deliver ordered item.

Agreement

At the highest level, we have agreements between two partner types that specify trading conditions in advance. But an agreement doesn't imply any specific economic commitment. It can be considered as correspond to establishment, sustaining, and developing business relationships as in the Transaction Group proposed by Lind et al. but not necessary limit to two partner types.

Economic Contract

A contract is a subtype of agreement between partner types that some actual economic exchange will occur in the future. Contracts are containers for collections of commitments and can have recursive relationships with other contracts, for example, yearly contracts with monthly and weekly and daily shipping schedules. Another example is a purchase order (a contract) wherein the order line items are commitments.

Speech Act

Speech act is the purely communicative action of one of the primitive illocutionary points from a speaker to a hearer (see Chapter 2 for details). This is the basic unit of analysis in our process modeling approach. For example request for an item and promise of delivery can be considered as two directive and commissive speech acts respectively.

Instrumental Act

Instrumental act is the material action that deals with material flow from specific originator to a specific recipient. It may constitute an economic event. For example, delivery of goods and cash payment are two instrumental acts.

Act

Act is the super type of an action, which either can be a speech act and instrumental act.

Transaction

Transaction is the smallest possible sequence of actions (speech acts) that has an effect in the social world of the participants. Typically speech acts occur in pairs, e.g. request/commit. Those pairs leading to obligations, authorizations, accomplishments are named transactions.

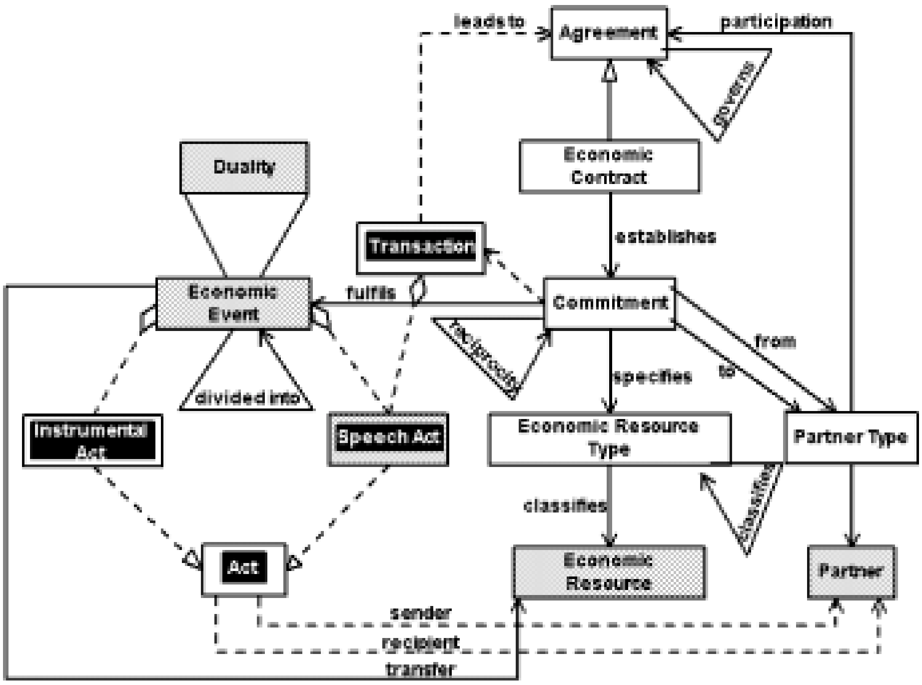


Fig. 14. An extended UMM economic model

One of the obvious requirements of a modeling methodology is that it is possible to understand its fundamental concepts. A well-known approach for gaining this understanding is to build a conceptual model (meta-model) representing concepts. There are several techniques to represent a conceptual model diagrammatically, we have chosen to use UML class diagrams notation since it is used in UMM. The conceptual model of our work can be represented in a diagram like **[Fig. 14]** above.

The rectangles in the diagram represent concepts and also note that we have used dotted shade for some rectangles that are considered as the most central concepts in the P^3 framework. The rectangles with white text labeling on black background are concepts that have been added to the original UMM meta-model.

Note that in the conceptual model, besides UML associations, we have used dotted lines between rectangles to indicate the extension to the original UN/CEFACT model. This is to represent the additional relationships resulting from the newly introduced concepts.

The conceptual framework we use is based on UMM's economic model describing resources, events and agents (REA model) **[66]**, **[15]**. In order to make the model suitable also for communication aspects, we have extended it by concepts from speech act theory. Furthermore, we include a number of notions proposed by Weigand et al. **[72]** used to distinguish between different levels of communication.

5.3 P^3 Business Model

In the current version of our P^3 framework, we are using a simplified form of a business model without analyzing in depth the value trading with respect to theories from various disciplines such as economics, marketing, etc. This is simply because our main intention of this thesis is to introduce the modeling methodology and tool support for transformation of a business model to an executable system.

We follow the business model proposed by Gordijn **[21]** and the UN/CEFACT concepts **[62]** for the development of our business model. Our business model can be considered as a triple that consists of set of partner types, set of economic resources types, and set of duality types.

The set of partner types defines all involved trading partner classes in an e-commerce setup. These partners create value offerings to others and consumes value offerings. Furthermore, we identify and distinguish the base partner to whom the e-commerce system is to be developed. Identifying the base partner is a main feature in the P^3 framework.

In the running example we introduced in Section 1.3.1 there are four partner types: eCaterer, Customer, Food Supplier, and Beverage Supplier. As we are to develop the ecommerce system for eCaterer, in this case the base partner is the eCaterer.

The set of economic resource types defines all value objects exchange between partners. For the above example, the four economic resource types are money, food, beverage, and meal.

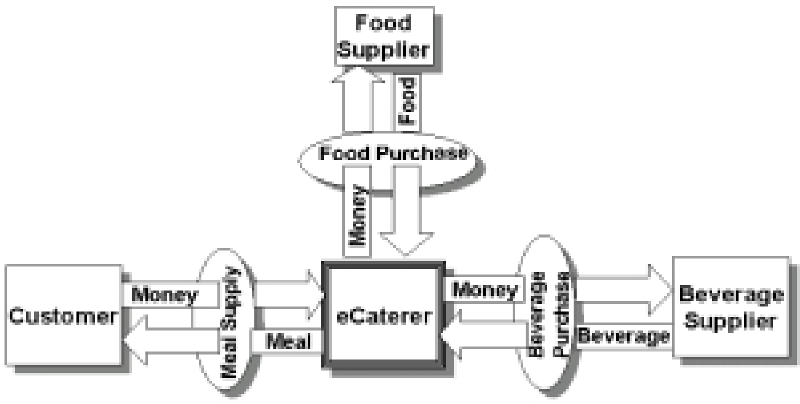


Fig. 15. The Business Model for e-Caterer System

The set of duality types defines value exchanges that bind two or more economic resources offered from one partner type to another. This is the central concept of business model as discussed in detail in Chapter 3. For the eCaterer case, there are six distinct economic resource exchanges as listed below,

1. Meal From eCaterer to Customer
2. Payment from Customer to eCaterer
3. Beverage from Beverage Supplier to eCaterer
4. Payment from eCaterer to Beverage Supplier
5. Food from Food Supplier to eCaterer
6. Payment from eCaterer to Food Supplier

The economic resource exchanges presented in **[Fig. 15]** are bounded into pairs making three dualities in this example. Tho three dualities named in **[Fig. 15]** are,

1. Meal Supply
2. Beverage Purchase
3. Food Purchase

6 Process Model

The main objective of this chapter is to describe the process models that realize the business requirement in a technical environment. First, process models in general will be addressed in brief and then the Business Modeling Language (BML) will be introduced. Finally the core of Process Patterns Perspective (P^3) framework, i.e. fundamental process patterns are defined, explained and presented in BML.

6.1 Introduction

Process models are to represent operational and procedural aspects of business ideas being designed. Development of business processes in practice is complicated, time consuming and often made with different model representations. Therefore, process modeling needs assistance from methodological guidelines and/or from tool support as highlighted in **[36]**.

Beside textual and graphical representations, there are two schools to process modeling: *activity oriented* and *communication oriented*. The activity oriented language diagrams usually represent a mix of automated and manual actions. A communication-oriented language, on the other hand, focuses on communicative processes describing the interaction between people and systems in terms of sending and receiving messages **[33]**. Our adaptation is to communication orientation in this work.

6.2 Business Modeling Language (BML)

This section briefly introduces a language based on communicating state machines, Business Modeling Language (BML) **[69] [68]**, **[33]**. The language has similarities to Specification and Description Language (SDL) **[54]**, **[3]**. BML is a communication oriented process language, which means that it focuses on describing interaction between actors through sending and receiving of messages. An important advantage of BML is that it can be used for the specification and design as well as operation of systems. This means that the same language can be used in different phases of a system's life cycle: in feasibility analysis, in requirement specification, in the design and implementation phases, and even in the operation phase. This enables different categories of stakeholders to use the same language for different purposes. The language can also be used directly as an implementation language and to some extent replace ordinary programming languages.

The main BML symbols and their descriptions are given in **[Table 2]**.

Symbol	Description
 Begin	This is the initial state named with label start, where each process instance starts with an initial event.
 Wait for Event	At the wait state a process instance is waiting for an event and there can be zero or more wait states in between Start and Stop states.
 End	This is the final state with label Stop, where a process instance terminates.
 Receive Message	Describes the consumption of a message from the input queue.
 Send Message	Describes the sending of a message
 Automated Business Decision	The control flow is dynamically changed depending on different business rules.
 Person A Actor	Human actor that interacts with a process instance
 Application A Application	An application that interacts with a process instance
 Start Timer	Start timer starts a timer and is represented by an hourglass "full of time".
 Expire Timer	Expire timer indicates timeout of a timer and is represented by an hourglass "out of time".
 Automated Business Activity	Describes operations that will be performed on the data instance of a process instance.

Table 2. Main BML symbols and their meaning

Further advantages of BML are its capability to describe and partition the interaction and interfaces between processes which work concurrently. The possibility of partitioning system into manageable and understandable parts with limited dependencies reduces the complexity of handling large systems. BML can describe the structure as well as the behaviour of a system by using two kinds of graphical diagrams. The structure of the system is visualised by a static diagram, see **[Fig. 16]**, which describes the processes in a static mode. The static diagram describes the messages sent between the processes and between the processes and the environment, i.e. external applications and people.

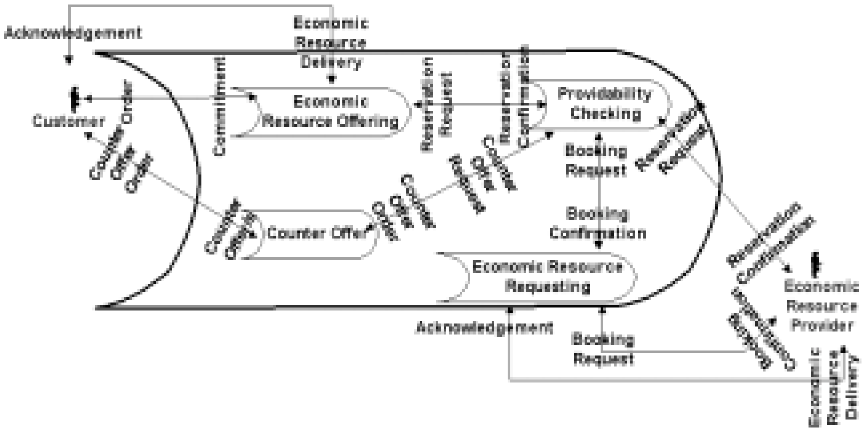


Fig. 16. The BML static diagram

The behavior of a system is described by using process models, which visualize the dynamics of a system, i.e. The order in which the messages shall be sent and received, see **[Fig. 17]**.

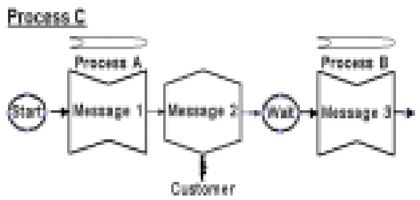


Fig. 17. The BML process diagram *Note that the figure only shows the beginning of a Process.*

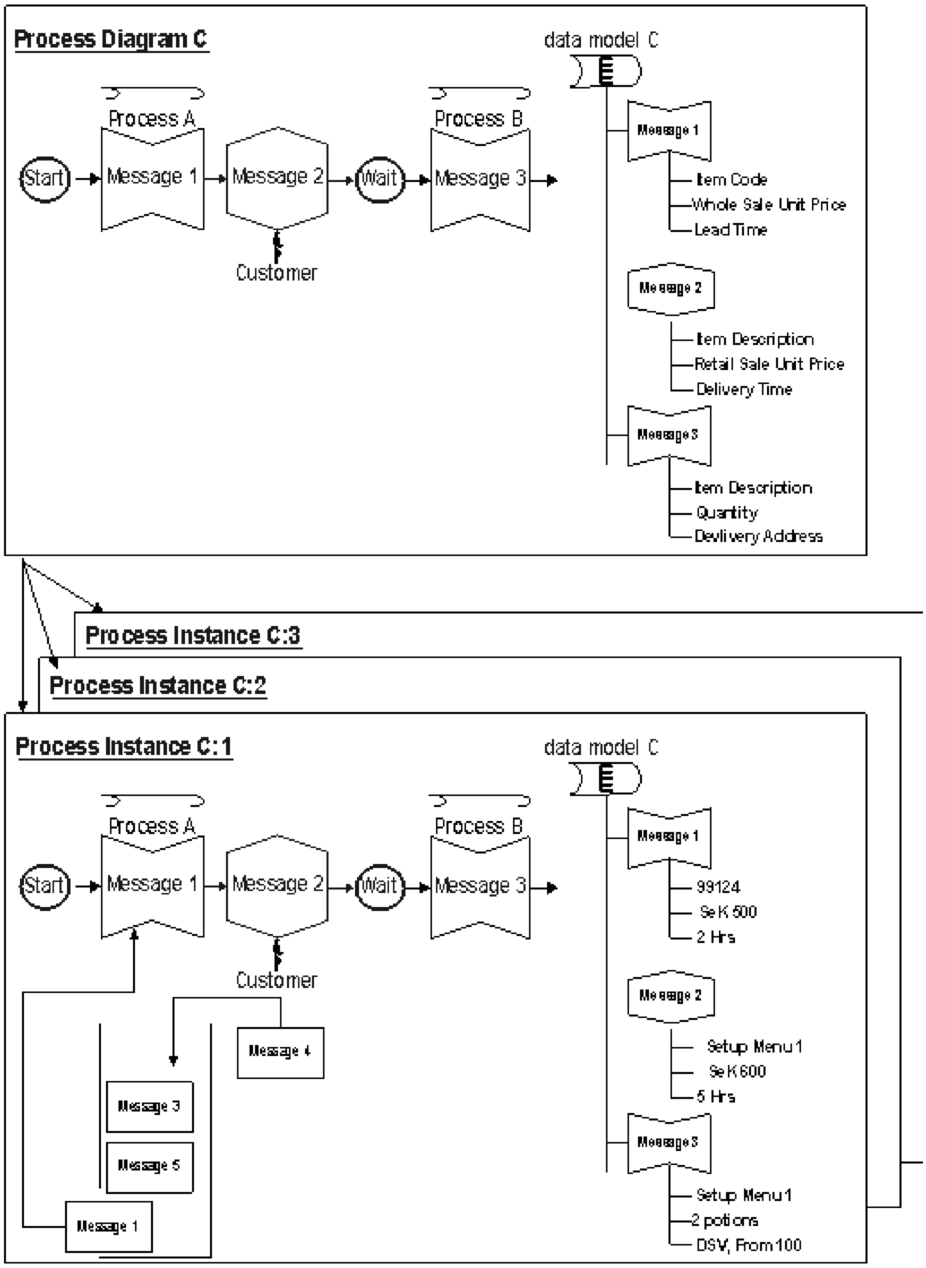


Fig. 18. The process diagram and its associated data model at the top. The process instances with the input queues and associated data instances at the bottom.

For each process diagram there is a number of process instances. A process diagram can be seen as a template for the process instances, which are created during runtime, see **[Fig. 18]**. The process instances execute independently of each other, but can communicate by sending and receiving messages asynchronously. Each instance has an input queue, see bottom left of **[Fig. 18]**, where received messages are stored. A process instance can either be waiting in a stable state (represented by an ellipse in BML) or be in transition from one state to another. A transition is initiated when a message in the input queue is chosen and consumed.

Following the example in **[Fig. 18]** (bottom), the process instance C:1 starts in a Start State (ellipse with label Start). Only the message Message 1 can initiate a transition. The Message 1 is first in the queue and is therefore consumed and the process instance performs a transition to the state Wait. During the transition a message Message 2 is sent out to Customer. Thereafter the message Message 5 is first in the queue. Since only message Message 3 can initiate a further transition from state Wait, the Message 5 is discarded or put back at the rear end of the queue (depending on the implementation). The next message in queue is then Message 3 which can initiate a transition from state Wait to some other wait for event state (not specified in the example). During the transition data can be manipulated and decisions can be made. New process instances can be created and messages can be sent to other process instances or to the process instance it self.

A basic characteristic of a BML diagram is that it is designed from one partner's perspective; and we will refer to him as a *base partner*. The base partner sends messages to, and receives messages from other actors. Typically, the base partner is the organization for which an e-commerce system is to be built.

In our work presented here, only a limited number of BML symbols have been presented for the sake of simplicity and understandability of BML process models. Only the first seven symbols listed in **Table 2**, have been used.

We have extended the original BML syntax slightly to achieve a higher level of compactness in the BML diagrams discussed in this thesis. The extension comprises of allowing Wait States symbols to receive messages. The extension makes it possible for us to leave out successive receive messages.

6.3 Generic Process Patterns

A business process modeling task is in general very complicated and time consuming, especially if one is to start from scratch in each new project. A good designer practice to overcome these difficulties is to use well-documented design patterns.

We hypothesize that most process models for e-commerce applications can be expressed as a combination of the following primitive process patterns. These basic patterns are defined and discussed below as BML process diagrams;

1. ***Economic Resource Requesting Diagram (ERRD)***
   This diagram is an action workflow loop from the perspective of the requesting actor, i.e. when the requesting actor is the base partner. For example, a base partner's purchase order enacts one ERRD instance to get supplier delivery.
2. ***Economic Resource Offering Diagram (EROD)***
   This diagram is an action workflow loop from the perspective of the supplying actor, i.e. when the supplying actor is the base actor. For example,

a customer's order enacts one EROD instance for the fulfillment of customer delivery.

3. ***Providability Checking Diagram (PCD)***

   This diagram models the reservation and booking of resources needed for carrying out an economic event.

4. ***Counter Offer Diagram (COD)***

   This diagram handles communication carried out in order to identify alternatives to a request that could not be fulfilled.

The final executable process model will be a collection of the above basic process patterns. From the business model introduced in the previous chapter it is possible to draft a process model, by instantiating one Economic Resource Requesting or Offering Process diagram for each economic event.

In addition, for identified economic resources which need counter offer handling, a pair of Providability Checking and Counter Offer process patterns have to be added. However, the communication among the process patterns is not uniquely determined by the business model, but may vary depending on the requirements for the process. How to determine this communication, i.e. how to move from a business model to a process model is addressed in Chapter 7.

6.3.1 Economic Resource Requesting Diagram (ERRD)

With respect to the base partner, the fundamental economic event is requesting some economic resource of value from another supplying party. Here the base partner is the receiving party. An economic resource requesting process pattern is the solution for a process-modeling task of economic resource requesting scenario and can be defined as follows with a BML diagram.

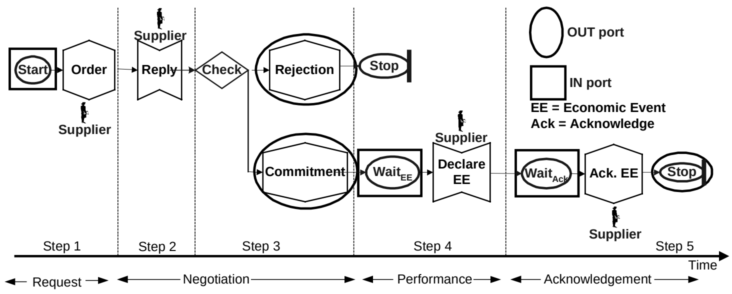


Fig. 19. Economic Resource Requesting Diagram (ERRD) (Identified IN ports, OUT ports and interactions with Supplier)

The above BML figure **[Fig. 19]** defines an Economic Resource Requesting Process (ERRD) Pattern, and actions at each step is explained below.

Step 1: The base partner sends an order to the supplier requesting an economic resource.

Step 2: The supplier sends a reply for the order made by the base partner.

Step 3: The supplier's reply is interpreted either as a rejection or as a commitment to fulfil the order.

Step 4: The supplier declares the delivery of the ordered economic resource.

Step 5: The base partner acknowledges the receipt of the requested economic resource.

6.3.2 Economic Resource Offering Diagram (EROD)

As mentioned above, with respect to the base partner, in a modeling situation, the other fundamental economic event is offering some economic resource with value to another receiving partner.

An economic resource offering process pattern is the solution for a process-modeling task of economic resource offering scenario. Here the base partner (the supplier) in a situation offers a receiving partner (the customer) some economic resource that is of value to her. In this case, we follow the suggestion by Taylor **[58]** and introduce additional qualification steps, where the base partner acquires ***direct*** and ***indirect*** means required for carrying out the requested action. Direct means are resources that go into final delivery and may be acquired from external suppliers in order to fulfill the request. Indirect means are resources necessary to carry out the request but may not go into final delivery. These indirect qualification checks often involve monetary qualification check such as getting promise for payment or asking for a down payment.

The acquisition of direct and indirect means leads to a design decision to be made out of four possible alternatives. First, not paying any attention of the order between qualification checking for direct and indirect means. Second, qualification checking for direct means followed by qualification checking for indirect means. Third, qualification checking for indirect means followed by qualification checking for direct means. Finally, qualification checks without such direct or indirect distinction especially for payment from base partner.

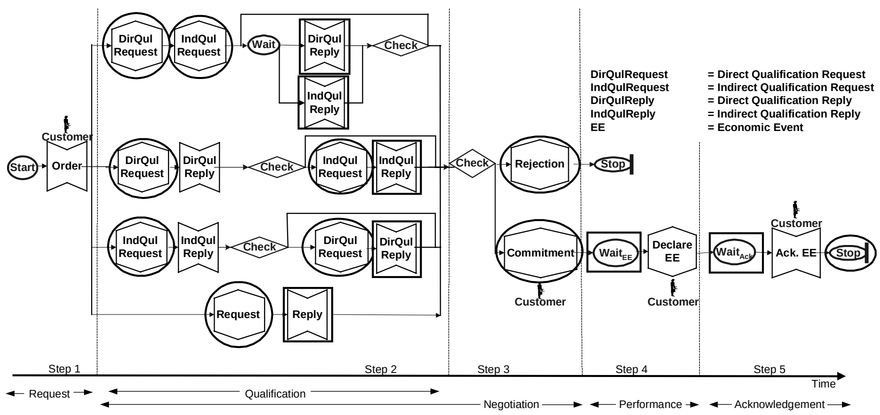


Fig. 20. Economic Resource Offering Diagram (EROD) (Identified IN ports, OUT ports and interactions with Customer)

The above BML diagram **[Fig. 20]** defines Economic Resource Offering Process (EROD) Pattern, and actions at each step is explained below.

Step 1: The customer sends an order for an economic resource to the base partner.

Step 2: The qualification steps are handled by choosing one of four alternatives. In the first one, there is no order between acquiring direct and indirect means. In the second one, direct means are acquired before indirect means. In the third one, indirect means are acquired before direct means. This includes one or more synchronous request/reply messages to acquire all necessary means prior to commit recipient with delivering an economic resource. Finally a single request-reply pair represents a qualification check for offers (say, monetary) where the direct/indirect distinction is irrelevant.

Step 3: The replies received when acquiring direct means are evaluated, and the base partner either rejects the customer's order or commits to deliver the ordered economic resource.

Step 4: The base partner declares the delivery of the ordered economic resource to the customer.

Step 5: The customer acknowledges the receipt of the ordered economic resource.

6.3.3 Providability Checking Diagram (PCD)

The providability checking process pattern for an economic resource models a situation where direct means for an economic resource offering is first reserved and then booked. In the preliminary reservation, PCD interacts directly with a supplier. When the booking request is received from EROD, it invokes an ERRD to make the formal ordering. The Providability Checking Diagram (PCD) may invoke counter offer handling when the base partner is not capable of providing for the original order.

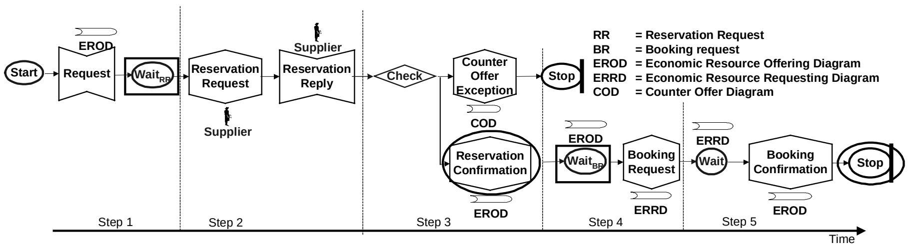


Fig. 21. Providability Checking Diagram (PCD) (Identified IN ports, OUT ports, interactions with EROD and COD)

The above BML **[Fig. 21]** figure defines economic resource Providability Checking Process (PCD) Pattern and related action at each step is explained below.

Step 1: A reservation request is received from the economic resource offering diagram (EROD).

Step 2: The reservation request is sent to a supplier and a reservation reply is received.

Step 3: The reservation reply is evaluated either to a counter offer exception, which is to be handled by a Counter Offer Diagram (COD) for the economic resource, or to a reservation confirmation that is to be sent to an EROD.

Step 4: A booking request from EROD is sent to an ERRD.

Step 5: When the booking confirmation from the ERRD is received, EROD is acknowledged with the received booking confirmation.

6.3.4 Counter Offer Diagram

A Counter Offer Diagram (COD) models the management of counter offers, which are received from a PCD. A COD communicate with a Counter Offer Provider and with a customer who made the original order in order to identify satisfactory alternatives for the original request made by the customer when the base partner is not in a position to provide for the original request.

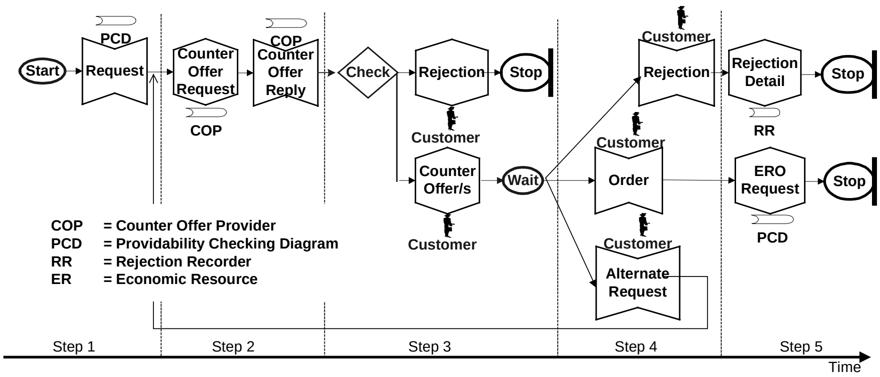


Fig. 22. Counter Offer Diagram (COD) (Identified interactions with PCD, COP, and Customer)

The above BML **[Fig. 22]** figure defines Counter Offer Process (COD) Pattern, and related actions at each step is explained below.

Step 1: A counter offer request is received from PCD.

Step 2: A counter offer request is sent to the Counter Offer Provider (COP) and the reply is received. (A COP is a process that creates counter offers. It may use algorithms and/or human involvement.)

Step 3: The response received from COP is evaluated either to a rejection that is to be notified to the customer or to a set of counter offers that the customer may choose among.

Step 4: The customer can reply with a rejection, request a selected offer, or request an alternative offer.

Step 5: If the customer rejects the counter offer, the rejection is directed to a relevant handler process here called Rejection Recorder. If the customer requests a selected offer, then the request is sent to a new PCD instance. If the customer requests an alternative, the request is sent to a new COD instance.

7 P^3 Business Process Modeling Methodology

In this chapter we will introduce the P^3 modeling framework and methodological guidelines for e-commerce system development. The chapter starts with a high level view of phases in the development process and then continues with a detailed view of the activities in each phase.

7.1 Introduction

The P^3 Business Process Modeling Methodology is the main contribution of this work. It enables process designers to systematically generate executable business process models in an e-commerce setting and to experiment with different design alternatives. The P^3 development framework has four distinct phases through which a designer can reach the final business process model.

1. Design of Business Model
2. Establishment of Economic Event Order
3. Establishment of Process Order
4. Generation of Process Model

Our four-phased approach can be visualized as in **[Fig. 23]**. Collectively these phases can be treated as a systematic transformation process of a business model into an executable process model. The first three phases are question and answering sessions where the designer assistant asks the user in a language comfortable to the user and where the answers lead to different design decisions. The final phase of the P^3 approach is an automatic generation of an executable process model in BML.

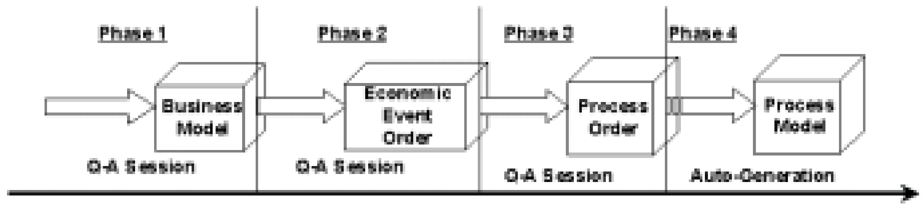


Fig. 23. Four phased approach to process models

7.2 Phase 1 - Designing of Business Model

In the first phase the main objective is to design a business model that can be agreed upon by all involved partners. In this phase the designer specifies for which organization the e-commerce system is to be developed, establishes who will be the customers of its processes and finally builds the model. During this phase, the answers to the following questions are obtained.

1. *Who are the partners?*
2. *Who is the base partner?*
3. *Who is the customer?*
4. *What are the economic resources exchanged?*
5. *What are the dualities, and who are the partners involved in each one?*
6. *What are the economic events for each duality?*

The questions guide the designer through the task and prompt her to provide names for partners, economic resources, economic events, and dualities. An example of set of answers to these questions is given in **[Fig. 24]** and the resulting business model is shown in **[Fig. 25]**.

1) Who are the partners?

Customer
Food Supplier
Wine Supplier
ECaterer

2) Who is the base partner?

Customer	
Food Supplier	
Wine Supplier	
ECaterer	X

3) Who is the customer?

Customer	X
Food Supplier	
Wine Supplier	

4) What are the economic resources?

Meal
Money
Food
Wine

5) What are dualities are there, and who are the partners involved in each exchange?

Duality	Partner	Partner
Meal Supply	Customer	eCaterer
Food Purchase	Food Supplier	eCaterer
Wine Purchase	Wine Supplier	eCaterer

6) What are the economic events of each duality?

Duality	Economic Event	Originator	Recipient
Meal Supply	MealToCustomer	eCaterer	Customer
	PaymentFromCustomer		
Food Purchase	FoodToCateringCompany	Food Supplier	eCaterer
	PaymentToFoodSupplier		
Wine Purchase	WineToCateringCompany	Wine Supplier	eCaterer
	PaymentToWineSupplier		

Fig. 24. Questions and Answers for case 1 in Phase 1

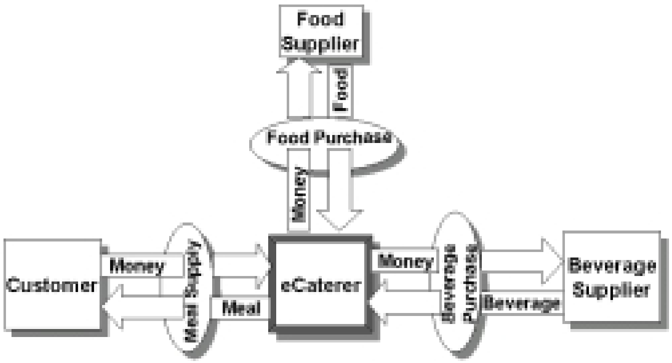


Fig. 25. Business Model for case 1

7.3 Phase 2 - Establishment of Economic Event Order

In this phase, the designer starts to order the activities of the process. First, the designer takes into account only economic events while disregarding the communicative acts that co-ordinate the process. By considering only the order of the economic events in this phase, the designer can concentrate on the main business logic and postpone until later more detailed design decisions about the coordination of communicative acts.

The designer first has to decide whether an economic event must or can be divided into parts; such a part is called an *atomic economic event*. The first question is therefore:

7. What are the atomic economic events of each economic event?

After having identified and named the atomic economic events, the designer is prompted to order them by determining the dependencies that exist between them. In an e-commerce context, we identify two main types of dependencies: trust dependencies and flow dependencies.

A *trust dependency* occurs between two atomic economic events within the same duality, e.g. that a product must be paid before it can be delivered. A trust dependency expresses the level of trust between the actors involved in a duality, e.g. requesting a down payment expresses low trust. A *flow dependency*, **[43]**, occurs between two atomic economic events in different dualities and expresses that the economic resource obtained by one of the economic events is needed for the other economic event. A simple example is that a retailer has to obtain a product from an importer before delivering it to a customer. In order to identify trust and flow dependencies, the following question is posed.

8. How do you order the atomic economic events?

An example of answers to questions 7 and 8 is given in **[Fig. 26]**.

7. What are the Atomic Economic Events (AEEs) of each Economic Event (EE)?

Economic Event	Atomic Economic Event	Abbreviation
<eCaterer, Customer, Meal>		MealToCustomer (MTC)
<Customer, eCaterer, Money>	<Customer, eCaterer, DownPay>	DownPayFromCustomer (DPFC)
	<Customer, eCaterer, FinalPay>	FinalPayFromCustomer (FPFC)
<BeverageSupplier, eCaterer, Beverage>		BeverageToCateringCOmpany (BTCC)
<eCaterer, BeverageSupplier, Money>	<eCaterer, BeverageSupplier, DownPay>	DownPayToBeverageSupplier (DPTBS)
	<eCaterer, BeverageSupplier, FinalPay>	FinalPayToBeverageSupplier (FPTBS)
<FoodSupplier, eCaterer, Food>		FoodToCateringCompany (FTCC)
<eCaterer, FoodSupplier, Money>		FinalPayToFoodSupplier (FPTFS)

8. How do you order the Economic Events and the Atomic Economic Events?
(Add only < symbol in the matrix)

	MTC	DPFC	FPFC	BTCC	DPTBS	FPTBS	TFCC	FPTFS
MealToCustomer (MTC)			<					
DownPayFromCustomer (DPFC)	<		<					
FinalPayFromCustomer (FPFC)								
BeverageToCateringCompany (BTCC)	<					<		
DownPayToBeverageSUpplier (DPTBS)				<		<		
FinalPayToBeverageSupplier (FPTBS)								
FoodToCateringCompany (FTCC)	<							<
FinalPayToFoosSupplier (FPTFS)								

The table shall be read in the following way: row header, cell, column header, e.g. the first cell on the seventh row gives FoodToCateringCompany precedes MealToCustumer.

Fig. 26. Questions and answers for the example in phase 2

7.3.1 Reservation and Booking

When the offered economic resource is assembled from more than one economic resource, the acquisition of direct means for each can be ordered by getting answers for the questions below. The economic event order can be completed with reservation and booking as follows.

9a. Do you reserve economic resource, ER_1 before reserving economic resource, ER_2?
9b. Do you book economic resource, ER_1 before booking economic resource, ER_2?
9c. Do you book economic resource ER_1 before reserving economic resource ER_2?

An example answer to the questions is given in **[Fig. 27]**. The resulting partial order from phase 2 is the following (< means precedes):

EEO = {MealToCustomer < FinalPayFromCustomer,
DownPayFromCustomer < MealToCustomer,
DownPayFromCustomer < FinalPayFromCustomer,
BeverageToCateringCompany < MealToCustomer,
BeverageToCateringCompany < FinalPayToBeverageSupplier,
DownPayToBeverageSupplier < BeverageToCateringCompany,
DownPayToBeverageSupplier < FinalPayToBeverageSupplier,
FoodToCateringCompany < MealToCustomer,
FoodToCateringCompany < FinalPayToFoodSupplier,
BookBeverage < BookFood,
ReserveBeverage < ReserveFood}

Such a partial order between atomic economic events is called an *economic event order*. It expresses the order between the most important activities in the process and disregards coordinative activities.

9. For each Atomic Economic Event, identified the ones needed with counter offer handling.

- **Beverage Supply with counter offer handling**
- **Food Supply with counter offer handling**

Column / Row	Beverage	Food
Beverage		9a. BookBeverage < BookFood? Yes
		9b. ReserveBeverage < ReserveFood? Yes
		9c. BookBeverage < ReserveFood? No
Food	9a. BookFood < BookBeverage? -	
	9b. ReserveFood < ReserveBeverage? -	
	9c. BookFood < ReserveBeverage? -	

I.e. Get answers for three questions;
Book(row) < Book(column)?
Reserve(row) < Reserve(column)?
Book(row) < Reserve(column)?
considering a pair at a time.

Fig. 27. Questions and Answers for Reservation and Booking

7.4 Phase 3 - Establishment of Process Order

In phase 3, the designer will extend the economic event order from phase 2 by specifying dependencies between communicative acts. A starting point for this task is that for each atomic economic event, there will be one action-workflow loop (modeled by an ERRD or EROD diagram). The designer has to determine the interactions between the loops given by all the atomic economic events. Like earlier, the designer assistant supports this task through a number of questions. The intuition behind several of these questions is, roughly expressed, the following: before an actor does something of value to another actor, it will check whether that actor deserves it. By doing "something of value to another actor" is meant to carry out an economic event, to commit to carry out an economic event, or to initiate the acquisition of means needed to carry out an economic event.

The expression "check whether that actor deserves it" mirrors the fact that an economic event from an actor A to an actor B is always accompanied by another economic event from B to A; recall that these two economic events together constitute one duality. The expression states that before actor A is prepared to carry out its economic event (or some preparation for it) to B, it will check that B has done its corresponding economic event (or some preparation). Note that this check will be done only if the economic event order so prescribes. Furthermore, there are questions for ensuring that all required means for carrying out an economic event have been obtained.

In order to formulate the questions, we need to distinguish between an *incoming atomic economic event* (AEE), where the base partner receives an economic resource, and an *outgoing AEE*, where the base partner supplies an economic resource.

If In is an incoming AEE and Out is an outgoing AEE within the same duality, and In < Out in the economic event order, ask:

10a. Do you require that In be performed before you commit to perform Out?
10b. Do you require a commitment for In before you commit to perform Out?

Furthermore, if In is an incoming AEE and Out an outgoing AEE within the same duality, and Means is an incoming AEE in another duality, and In < Out, and Means < Out in the economic event order, ask:

11a. Do you require that Means be performed before you commit to perform Out?
11b. Do you require a commitment for Means before you commit to perform Out?
11c. Do you require that In be performed before you request Means?

An example of answers to these questions is given in **[Fig. 28]**. All answers will result in an extension to the economic event order from phase 2, which also includes ordering between communicative acts. Such an order is called a *process order*. In this case, we arrive at a process order PO:

PO = EEO U
{decl(DownPayFromCustomer) < com(MealToCustomer),
com(BeverageToCateringCompany) < com(FinalPayToBeverageSupplier),
com(FoodToCateringCompany) < com(FinalPayToFoodSupplier),
com(BeverageToCateringCompany) < com(MealToCustomer),
decl(DownPayFromCustomer) < dir(BeverageToCateringCompany),
com(FoodToCateringCompany) < com(MealToCustomer),
decl(DownPayFromCustomer) < dir(FoodToCateringCompany)}

The process order PO resulted here is a union of the economic event order EEO of the phase 2 and the ordering of communicative acts for work coordination done at phase 3. In specifying PO we have limited ourselves to order the directive, commissive and declarative speech acts introduced in Chapter 2. In the above PO schema *dir*, *com* and *decl* abbreviations have been used for these communicative acts.

10a. *Do you require DownPayFromCustomer be performed before you commit to perform MealToCustomer (Yes/No)? Yes*

It is unnecessary to ask 9b as the answer to 9a already implies it.

10a. *Do you require BeverageToCateringCompany be performed before you commit to perform FinalPayToBeverageSupplier (Yes/No)? No*

10b. *Do you require a commitment for BeverageToCateringCompany before you commit to perform FinalPayToBeverageSupplier (Yes/No)? Yes*

10a. *Do you require FoodToCateringCompany be performed before you commit to perform FinalPayToFoodSupplier (Yes/No)? No*

10b. *Do you require a commitment for FoodToCateringCompany before you commit to perform FinalPayToFoodSupplier (Yes/No)? Yes*

11a. *Do you require that BeverageToCateringCompany be performed before you commit to perform MealToCustomer (Yes/No)? No*

11b. *Do you require a commitment for BeverageToCateringCompany before you commit to perform MealToCustomer (Yes/No)? Yes*

11c. *Do you require DownPaymentFromCustomer be performed before you request BeverageToCateringCompany (Yes/No)? Yes*

11a. *Do you require that FoodToCateringCompany be performed before you commit to perform MealToCustomer (Yes/No)? No*

11b. *Do you require a commitment for FoodToCateringCompany before you commit to perform MealToCustomer (Yes/No)? Yes*

Fig. 28. Example of answers to question sets 10) and 11)

7.5 Phase 4 - Generation of Process Model

Phase 4 is the final phase in the P^3 framework. In this phase the executable BML process model will be generated from the process order obtained in earlier phases with a minimum of designer intervention.

The basis for generation of a process model is that for each atomic economic event, there is an action workflow loop represented in a process model by one of the direct process patterns, i.e. economic resource requesting process pattern (ERRD) or economic resource offering process pattern (EROD). In other words, a preliminary BML process model can be derived which inclues ERRDs for all economic events where the base partner offers economic resources and ERODs for all economic events where the base partner requests economic resources.

In addition to ERRDs and ERODs, optionally providability checking process patterns (PCDs) and counter offer process patterns (CODs) have to be included in the final solution for those economic resource-offers which requires for providability checking and counter offers. However, for the determination of inter-diagram communication the process order, identified through the second and third phases is necessary. Basically, each inequality identifies one inter-diagram connection that makes it possible for one BML process diagram to interact with another.

We have defined a formal set of production rules for this mapping of process order to process model. First, to give a basic intuition of the logic behind them, we are exemplifying the effect of a couple of the mapping rules and then list the entire set of these production rules.

For example: an inequality telling that the beverage has to be delivered to the catering company before the catering company can in turn deliver the meal to the customer (BeverageToCateringCompany < MealToCustomer), should result in a connection from the ***stop*** state of the ERRD modeling the caterer’s beverage request process, to the ***wait*** for confirmation state in the EROD modeling the caterer’s meal offering process **[Fig. 29]**. This inter-diagram connection indicates that the beverage delivery to the catering company process has to be completed before the meal delivery process completely fulfills a customer’s request.

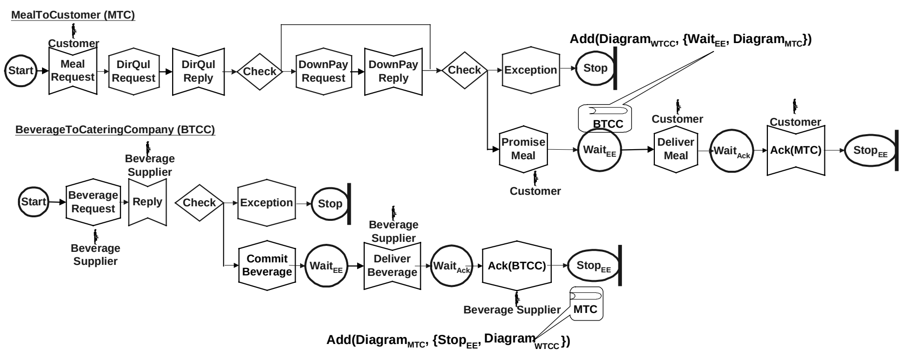


Fig. 29. BML process model segment (MealToCustomer and BeverageToCateringCompany processes with mapped inter-diagram communication from the inequality BeverageToCateringCompany < MealToCustomer)

The general rule that adds symbols to connect two diagrams for the above situation has the form,

$\texttt{If } EE_1 < EE_2$

$\texttt{add}(Diagram_2\ \{Stop_{EE},\ Diagram_1\})$

$\texttt{add}(Diagram_1\ \{Wait_{EE},\ Diagram_2\})$

The parameters used in the rule can be mapped to the above situation as EE1 = BeverageToCateringCompany (BTCC), EE2 = MealToCustomer (MTC) where EE stands for economic event and suffixes 1 = BTCC and 2 = MTC.

As an additional example, consider an inequality obtained from the process order where down payment from the customer has to be received in order to place the purchase order for beverage from a beverage supplier. In the process order, we have taken into account all communicative acts for work coordination, and the resulting inequality for this second situation is, decl(DownPayFromCustomer) < dir(BeverageToCateringCompany), (i.e. to order beverage from beverage supplier, base partner waits for down payment from customer). The necessity here is to communicate the successful completion of the process that collects a down payment from the customer to enact the process that purchases beverage from the beverage supplier. In BML technical terms, this inter-diagram communication can be interpreted as connecting *Stop* state of DownPayFromCustomer process to *Start* state of BeverageToCateringCompany process.

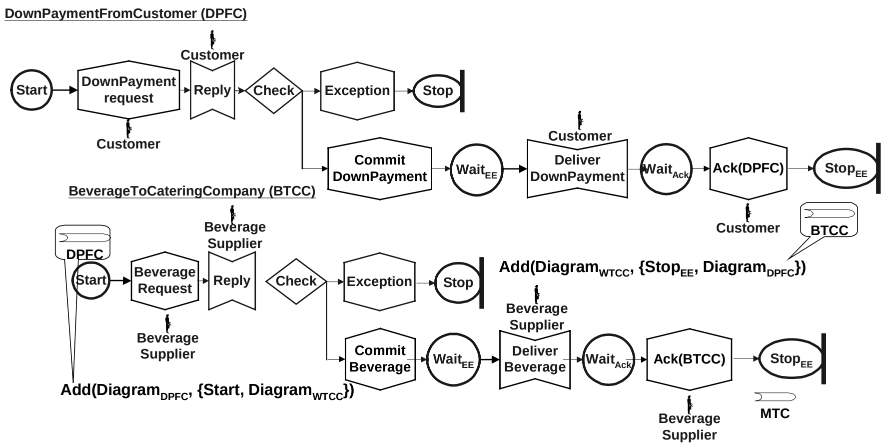


Fig. 30. BML process segment [DownPaymentFromCustomer and BeverageToCateringCompany with mapped inter-diagram communication from the inequality decl(DownPaymentFromCustomer < dir(WineToCateringCompany)]

As in the previous example the rule which adds symbols for completes inter diagram communication between the process patterns for the above situation can be listed as below. Note that in the BeverageToCateringCompany process diagram **[Fig. 30]** the MealToCustomer (MTC) process symbol at *Stop* state is included as a result of applying an earlier mapping rule.

If decl(EE_1) < dir(EE_2)

add($Diagram_2$ {$Stop_{EE}$, $Diagram_1$})

add($Diagram_1$ {Start, $Diagram_2$})

Process symbols will be added to the draft process model when production rules are applied to the process order.

7.6 Production Rules to Map Process Order to BML Process Model

When using the P^3 methodology, the successful completion of the first three phases results in a process order. This process order is specified as a set of inequalities showing the partial order among communicative acts for work coordination.

We have defined a formal system of production rules that can be applied on the process order to generate the final process model.

As mentioned earlier, the first cut BML process model can be generated by instantiating one primitive process pattern for each identified economic event at phase 2 together with optional inclusion of providability and counter offer processes.

A majority of the rules connects one IN port with an OUT port of another diagram. We have explicitly identified these IN and OUT port where inter-diagram communication can take place in the primitive pattern definitions (in Chapter 5).

For the formal definition of a production rule system we have associated two basic operations that either can add a process symbol to a diagram or remove an existing process symbol from a diagram. The former is to complete inter-diagram communication while the latter is to avoid unnecessary redundancies. The following syntax has been used for the definition of these two operations.

- `Add(<Source/Destination DIAGRAM>,{<Port>,<DIAGRAM that port belongs>})`

- `Remove(<Source/Destination DIAGRAM>,{<Port>,<DIAGRAM that port belongs>})`

"add (A, {B, C})" means there should be a state B in diagram C that has interaction (communication) with diagram A and "remove(A, {B, C})" means in diagram A the state B of the diagram C is removed. Basically the "remove" operation is used to reduce the redundancy of inter-diagram communications. Further we are using just (A, {B, C}) to represent the existence of communication from diagram A to the position B of diagram C.

7.6.1 P^3 Production Rule System

The entire P^3 production rule system can be grouped into three sub categories as listed below.

1. ***Main rules*** to handle economic events and communication for their coordination.
2. ***Supplementary rules*** to handle reservation and booking of economic resources.

Main rules

Main rules handle inter-diagram communication of BML process model by mapping the process order obtained from the first three phases, except optional reservation and booking of economic resources.

1) if AEE_i < AEE_j

Add($DIAGRAM_j$, {$Stop_{EE}$, $DIAGRAM_i$})

If AEEj ∊ {ERRD*}

Add($DIAGRAM_i$, {Start, $DIAGRAM_j$})

else

Add($DIAGRAM_i$, {$Wait_{EE}$ $DIAGRAM_j$})

This is the rule that delays the delivery of an atomic economic resource out or enacts request of supply for an economic resource, i.e. successful completion of AEE_i has to communicate either at $Wait_{EE}$ state or *Start* state prior to decl(AEE_j). ERRD* is the Economic Eesource Requesting Diagram set.

2) if com(AEE_i) < com(AEE_j)

if ($DIAGRAM_{PCi}$, {DirQulReply, $DIAGRAM_j$})

remove($DIAGRAM_{PCi}$, {DirQulReply, $DIAGRAM_j$})

remove($DIAGRAM_j$, {com(Rev), $DIAGRAM_{PCi}$})

remove($DIAGRAM_j$, {$Wait_{RRq}$, $DIAGRAM_{PCi}$})

if ($DIAGRAM_{PCi}$, {$Wait_{EE}$, $DIAGRAM_j$})

remove($DIAGRAM_{PCi}$, {$Wait_{EE}$, $DIAGRAM_j$})

Add($DIAGRAM_j$, {com(AEE_i), $DIAGRAM_i$})

Add($DIAGRAM_i$, {<Reply>, $DIAGRAM_j$})

The rule 2) is to get at promise for an atomic economic event before promising another, i.e. com(AEE_j) has to communicate with <Reply> which receives messages prior to promise AEE_j (com(AEE_j)) out.

Note that <Reply> has to be substituted by "DirQulReply" or "IndQulReply" if AEE_j is a transfer of direct material of customer delivery or if AEE_j is transfer of indirect material of customer delivery as named in process patterns respectively (see **[Fig. 19]** and **[Fig. 20]** of Chapter 5). "DirQulReply" is Direct Qualification Reply and "IndQulReply" is Indirect Qualification Reply as introduced in Section 6.3.2.

If AEE_j has been instantiated with a providability checking, by default there will be two interactions with process diagram of AEE_j ($DIAGRAM_j$), i.e. first to communicate preliminary reservation confirmation at DirQulReply of $DIAGRAM_j$ and next to communicate booking confirmation at $Wait_{EE}$ of $DIAGRAM_j$ (see **[Fig. 19]** and **[Fig. 20]** of Chapter 5. But the condition com(AEE_i) < com(AEE_j) requires booking confirmation (formal commitment) to be communicated at "DirQulReply". By removing existing communication of reservation, booking confirmations, and placing new commitment received directly from AEE_i at "DirQulReply" we avoid unnecessary redundancy.

```
3) if decl(AEE_i) < com(AEE_j)

     Add(DIAGRAM_j, {Stop, DIAGRAM_i})

     Add(DIAGRAM_i, {Reply, DIAGRAM_j})
```

Rule 3) is stronger than rule 2) and requires delivery of an atomic economic event prior to promise another, i.e. successful completion of AEE_i has to communicate at "Reply" which receives messages prior to promise AEE_j.

```
4) if decl(AEE_i) < dir(AEE_j)

     Add(DIAGRAM_j, {Stop, DIAGRAM_i})

     Add(DIAGRAM_i, {Start, DIAGRAM_j})
```

Rule 4) is the stronger than rules 2) and 3) in that it requires delivery of an atomic economic event prior to requesting another, i.e. successful completion of AEE_i has to communicate at "Start" which enacts the $DIAGRAM_j$.

Supplementary Rules

Supplementary rules are for handling optional reservations and bookings required by some economic resources. These rules complete inter-diagram communication and reflects the necessary order of temporarily acquiring and holding of direct means for customer delivery. For economic resource offering diagrams with Providability Checking and Counter Offer Handling the following rules are applied;

For Providability Checking, $Diagram_1$ and $Diagram_2$ for economic resources, ER_1 and ER_2 respectively,

```
5) if Book(ER_1) < Book(ER_2)

      add(Diagram_PCD2, {Stop, Diagram_PCD1})

      add(Diagram_PCD1, {Wait_BR, Diagram_PCD2})
```

Rule 1) of the supplementary rules is to get the booking confirmation of an economic resource prior to request for another booking confirmation for another economic resource. This can be achieved by communicating the booking confirmation of the former providability checking process from *Stop* state to $Wait_{BR}$ state of the later providability checking process which delays the booking request.

```
6) if Reserve(ER_1) < Reserve(ER_2)

      add(Diagram_PCD2, {ResConfirmation, Diagram_PCD1})

      add(Diagram_PCD1, {Wait_RR, Diagram_PCD2})
```

Rule 2) of the supplementary rules is to get the reservation confirmation of an economic resource prior to request for another reservation confirmation for another economic resource. This can be achieved by communicating the reservation confirmation of the former providability checking process from *ResConfirmation* state to $Wait_{RR}$ state of the later providability checking process which delays the reservation request.

```
7) if Book(ER₁) < Reserve(ER₂)

      add(Diagram_PCD2, {Stop, Diagram_PCD1})

      add(Diagram_PCD1, {Wait_RR, Diagram_PCD2})
```

Rule 3) of the supplementary rules is to get the booking confirmation of an economic resource prior to request for another reservation confirmation for another economic resource. This can be achieved by communicating the booking confirmation of the former providability checking process from *Stop* state to $Wait_{RR}$ state of the later providability checking process which delays the reservation request.

8 transtec Case

8.1 Functions of transtec AG

transtec is a computer system manufacturer and seller providing business customers with custom built systems and computers. It is a success story in Europe’s digital economy and covers a varity of aspects and dimensions of e-commerce.
transtec:

1. sells custom built systems and computers via the WWW [business-to-customer]
2. has an intranet server where special (authorized) customers can log in and get information and offers relevant to the particular customer [business-to-customer]
3. provides complete access to all company data to staff in different European branches e.g. for sales analyses or future demand estimates
4. acts as a value chain integrator by providing relevant access to company data for suppliers enabling them to promptly deliver parts to transtec [business-to-business]

For the evaluation of our proposed methodology in this work, we have selected the transtec 1300 Low Noise system configured and sold via the WWW. For this particular system configuration transtec communicates with many different external suppliers and of cause with the customer who places the order. For this case we have selected five different suppliers assuming all the rest of the system components received from transtect. Software to be installed on the system is supplied by Microsoft and Speakers from the Sony under established trading agreements. The Headphones and Zipdrive for this Low Noise system are purchased from teratek and lomega respectively. Finally, through the good relationship with UPS for delivering transtec’s customer orders, authorized customers can trace status of the product being delivered to him on-line via same site by accessing UPS's information.

8.2 Business Model for transtec

The layout of the web page for the 1300 Low Noise system configurator is shown in the **[Fig. 32]**. For the selected configuration of the transtec 1300 Low Noise system, the transtec configurator interacts with the customer, external manufacturers and local component delivery systems. **[Fig. 31]** lists the questions and answers from phase 1 of the P^3 development process.

Though the case discussed here is a real world one, but we have not looked into real situation how exact business functions. Therefore, business requirements captured P^3 designer assistant are mainly our common sense understanding and general assumptions. The real situation may differ than our interpretation here.

1) Who are the partners?

Customer
transtec
Microsoft
teratek
Iomega
UPS
Sony

3) Who is the customer?

Customer	X
Microsoft	
teratek	
Iomega	
UPS	
Sony	

2) Who is the base partner?

Customer	
transtec	X
Microsoft	
teratek	
Iomega	
UPS	
Sony	

4) What are the economic resources?

Money
1300 Low Noise System
Software
Micro+Headphones
Zip drive AT API 250MB
Logistics
SRS-A57 Speakers

5) What dualities are there, and who are the partners involved in each exchange?

Duality	Partner	Partner
1300 Low Noise System Sale	Customer	transtec
Software Purchase	Microsoft	transtec
Micro+Headphones Purchase	teratek	transtec
Zip drive AT API 250MB Purchase	Iomega	transtec
Logistics	UPS	transtec
SRS-A57 Speakers Purchase	Soney	transtec

6) What are the economic events of each duality?

Duality	Economic Event	Originator	Recipient
1300 Low Noise System Sale	SystemToCustomer	transtec	Customer
	PaymentFromCustomer	Customer	transtec
Software Purchase	SoftwareTotranstec	Microsoft	transtec
	PaymentToMicrosoft	transtec	Microsoft
Micro+Headphones Purchase	HeadphonesTotranstec	teratek	transtec
	PaymentToteratek	transtec	teratek
Zip drive AT API 250MB Purchase	ZipdriveTotranstec	Iomega	transtec
	PaymentToIomega	transtec	Iomega
Logistic Services	Logistics	UPS	transtec
	PaymentToUPS	transtec	UPS
SRS-A57 Speakers Purchase	SpeckerTotranstec	Sony	transtec
	PaymentToSony	transtec	Sony

Fig. 31. Questions and Answers for the phase 1 of transtec case

http://www.transtec-ab.se/doit/loadva/sid=03BE82A53XX4BC1K/software/wwwdoit/applet/konfigurator/W13L_S.html

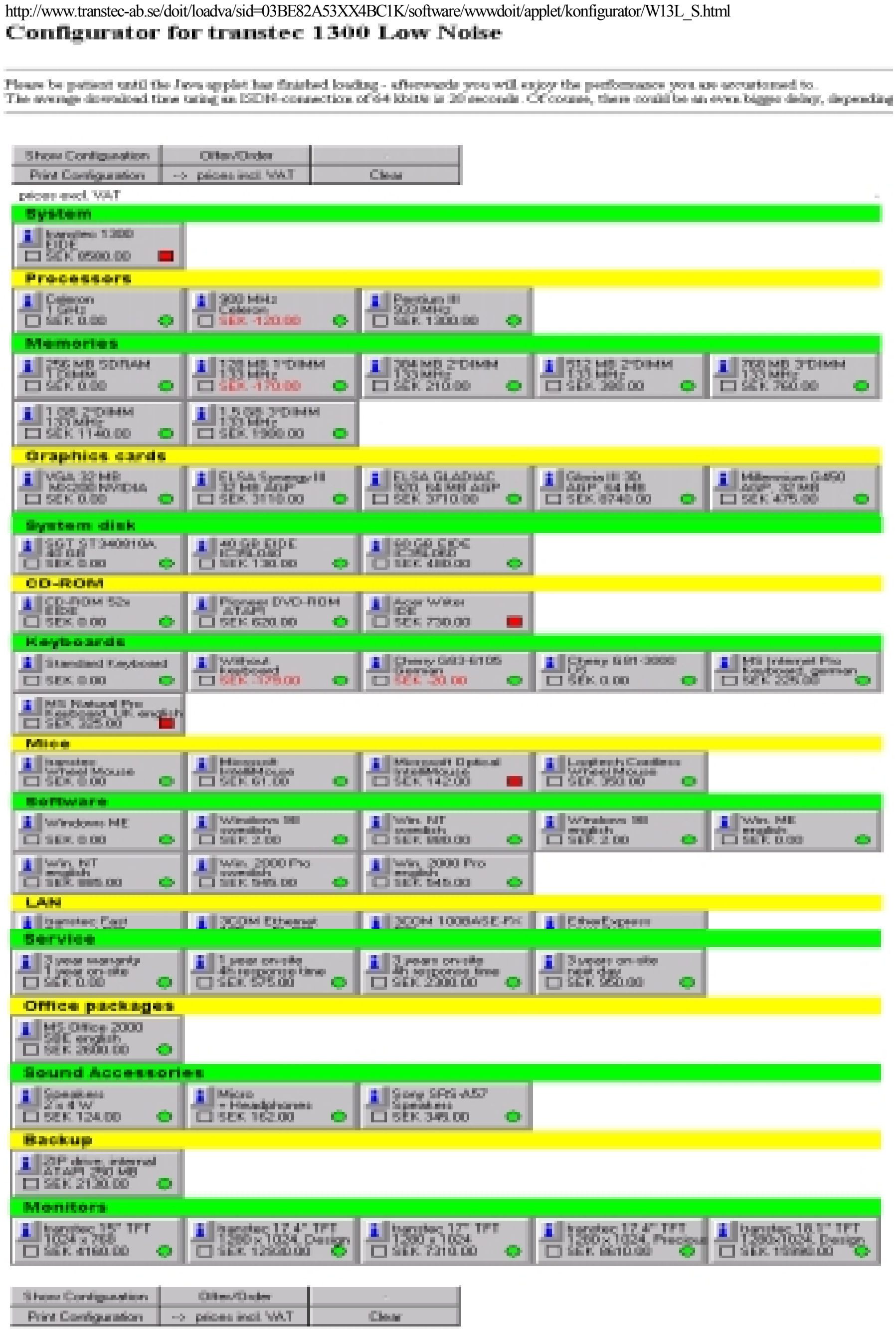


Fig. 32. The Configurator for transtec 1300 Low Noise system

With the help of the answers obtained in **[Fig. 31]**, the designer assistant will generate the business model for transtec as shown in **[Fig. 33]**.

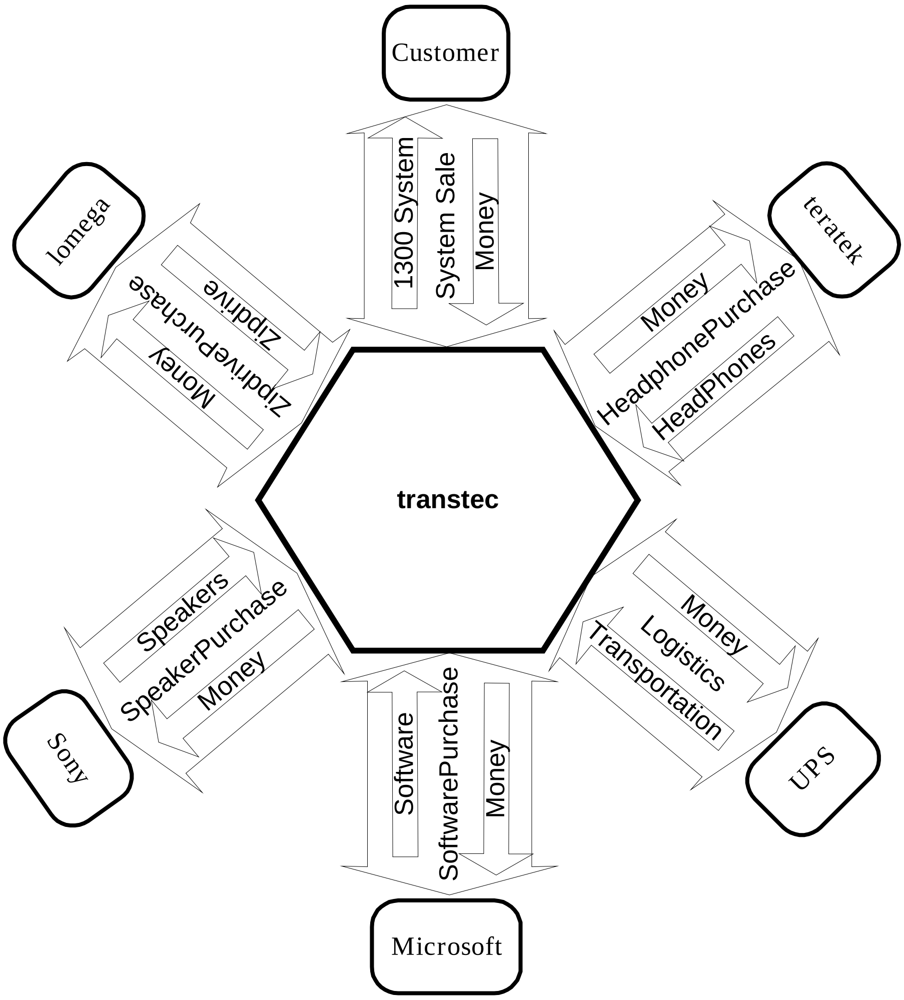


Fig. 33. Transtec Business Model

8.3 Economic Event Order for transtec

The economic event order for the selected business is obtained by getting answers for the question relevant for phase 2 of the P[3] development process. A possible set of answers obtained at phase 2 is given in tables **Table 3**, Table 4 and **Table 5**.

The abbreviations used in the following tables are:

EE = economic event, AEE = atomic economic event, t = transtec, C = Customer, S = 1300 System, M = Money, MS = Microsoft, SW = Software, So = Sony, Sp = Speakers, te = teratek, HP = Headphones, l = lomega, Z = Zipdrive, U = UPS, T = Transportation.

7. What are the Atomic Economic Events (AEEs) of each Economic Event (EE)?

EE	AEE	Abbreviation
<t, C, S>	<t, C, System>	SystemToCustomer (STC) N.B. 1) AEE ordering has been done only considering components from external suppliers and system as a whole.
	<t, C, Processors>	
	<t, C, Memories>	
	<t, C, Graphics cards>	
	<t, C, System disk>	
	<t, C, CD-ROM>	
	<t, C, Keyboards>	
	<t, C, Mice>	
	<t, C, Software>	
	<t, C, LAN>	
	<t, C, Services>	
	<t, C, Office packages>	
	<t, C, Sound Accessories>	
	<t, C, Backup>	
	<t, C, Monitors>	
<C, t, M>	<C, t, DownPay>	DownPayFromCustomer (DPFC)
	<C, t, FinalPay>	FinalPayFromCustomer (FPFC)
<MS, t, SW>		SoftwareTotranstec (SWTt)
<t, MS, M>	<t, MS, InitialPay>	<transtec, Microsoft, InitialPay>
	<t, MS, InstallationPay>	InstallationPayToMicrosoft (InPTM)
<So, t, Sp>		SpeakersTotranste (STt)
<t, So, M>	<t, So, DownPayment>	DownPayToSony (DPTS)
	<t, So, FinalPayment>	FinalPayToSony (FPTS)
<te, t, HP>		HeadphoneTotranstec (HTt)
<t, te, M>		PayToteratek(PTt)
<l, t, Z>		ZipdriveTotranstec (ZTt)
<t, l, M>		PayTolomega (PTl)
<U, t, T>		TransportationTot (TTt)
<t, U, M>		PayToUPS (PTUPS)

Table 3. Atomic Economic Events

8. How do you order the Economic Events and the Atomic Economic Events?
(Add only '<' or '>' or '-' symbol in upper triangular area of the matrix)

	STC	DPFC	FPFC	SWTt	IPTM	InPTM	STt	DPTS	FPTS	HTt	PTte	ZTt	PTl	TTt	PTUPS
SystemToCustomer (STC)		>	<	>	-	<	>	-	-	>	-	>	-	-	-
DownPayFromCustomer (DPFC)			<	-	-	<	-	-	-	-	-	-	-	-	-
FinalPayFromCustomer (FPFC)				-	-	-	-	-	<	-	<	-	<	-	<
SoftwareTotranstec (SWTt)					<	<	-	-	-	-	-	-	-	-	-
InitialPayToMicroSoft (IPTM)						<	-	-	-	-	-	-	-	-	-
InstallationPayToMicrosoft (InPTM)							-	-	-	-	-	-	-	-	-
SpeakersTotranste (STt)								>	<	-	-	-	-	-	-
DownPayToSony (DPTS)									<	-	-	-	-	-	-
FinalPayToSony (FPTS)										-	-	-	-	-	-
HeadphoneTotranstec (HTt)											>	-	-	-	-
PayToteratek (PTte)												-	-	-	-
ZipdriveTotranstec (ZTt)													<	-	-
PayTolomega (FPTl)														-	-
TransportationTotranstec (TTt)															<
PayToUPS (PTUPS)															

Table 4. Ordered Atomic Economic Events

9. For each Atomic Economic Event, identify the ones that need counter offer handling and fill in the answers to the following three questions in the matrix below.

a) **Do you want to book(row) < book(column)?**
b) **Do you want to reserve(row) < reserve(column)?**
c) **Do you want to book(row) < reserve(column)?**

Column → / **Row ↓**	**Processor**	**Memories**	**Graphics card**	**System disk**	**CD-ROM**	**Keyboard**	**Mice**	**Software**	**LAN**	**Service**	**Sound Accessories**	**Monitor**
Processor		c) Y	a) Y b) Y									
Memories												
Graphics card												a) Y b) Y
System disk												
CD-ROM												
Keyboard												
Mice												
Software												
LAN												
Service												
Sound Accessories												
Monitor												

Table 5. Reservation and Booking Order

The resulting economic event order can be written according to our formalism as follows.

EEO = DownPayFromCustomer < SystemToCustomer,
SystemToCustomer < FinalPayFromCustomer
SoftwareTotranstec < SystemToCustomer

SystemToCustomer < InstallationPayToMicrosoft
SpeakersTotranstec < SystemToCustomer
HeadphonesTotranstec < SystemToCustomer
ZipdriveTotranstec < SystemToCustomer
DownPayFromCustomer < FinalPayFromCustomer
DownPayFromCustomre < InstallationPayToMicrosoft
FinalPayFromCustomer < FinalPayToSony
FinalPayFromCustomer < FinalPayToteratek
FinalPayFromCustomer < FinalPayTolomega
FinalPayFromCustomer < FinalPayToUPS
SoftwareTotranstec < InitialPayToMicrosoft
SoftwareTotranstec < InstallationPayToMicrosoft
IntialPayToMicrosoft < InstallationPayToMicrosoft
DownPayToSony < SpeakerTotranstec
SpeakerTotranstec < FinalPayToSony
DownPayToSony < FinalPayToSony
HeadphonesTotranstec < PayToteratek
ZipdriveTotranstec < PayTolomega
TransportationTotranstec < PayToUPS
book(processor) < reserve(Memories)
book(processor) < book(GraphicCards)
reserve(processor) < reserve(GraphicCards)
book(GraphicCards) < book(Monitor)
reserve(GraphicCards) < reserve(Monitor)

8.4 Process Order for transtec

In this stage designer has to extend the economic event order from phase 2 by specifying dependencies between communicative acts. Again, the set of questions designed in Section 6.4 will be instantiated from each atomic economic event identified as shown below.

10a) Do you require that DownPayFromCustomer be performed before you commit to perform SystemToCustomer? Yes
10b) Do you require a commitment for DownPayFromCustomer before you commit to perform SystemToCustomer? -

10a) Do you require that SoftwareTotranstec be performed before you commit to perform InitialPayToMicroSoft? No
10b) Do you require a commitment for SoftwareTotranstec before you commit to perform InitialPayToMicroSoft? Yes

10a) Do you require that SpeakersTotranstec be performed before you commit to perform FinalPayToSony? No
10b) Do you require a commitment for SpeakersTotranstec before you commit to perform FinalPayToSony? Yes

10a) Do you require that TransportationTotranstec be performed before you commit to perform PayToUPS? No
10b) Do you require a commitment for TransportationTotranstec before you commit to perform PayToUPS? Yes

10a) Do you require that HeadPhonesTotranstec be performed before you commit to perform PayToteratek? No
10b) Do you require a commitment for HeadPhonesTotranstec before you commit to perform PayToteratek? Yes

10a) Do you require that ZipDriveTotranstec be performed before you commit to perform PayTolomega? No
10b) Do you require a commitment for ZipDriveTotranstec before you commit to perform PayTolomega? Yes

11a) Do you require that SoftwareTotranstec be performed before you commit to perform SystemToCustomer? Yes
11b) Do you require a commitment for SoftwareTotranstec before you commit to perform SystemToCustomer? -
11c) Do you require that DownPayFromCustomer be performed before you request SoftwareTotranstec? -

11a) Do you require that SpeakersTotranstec be performed before you commit to perform SystemToCustomer? No
11b) Do you require a commitment for SpeakersTotranstec before you commit to perform SystemToCustomer? Yes
11c) Do you require that DownPayFromCustomer be performed before you request SpeakersTotranstec?

11a) Do you require that TransportationTotranstec be performed before you commit to perform SystemToCustomer? No
11b) Do you require a commitment for TransportationTotranstec before you commit to perform SystemToCustomer? No
11c) Do you require that DownPayFromCustomer be performed before you request TransportationTotranstec? No

11a) Do you require that HeadPhoneTotranstec be performed before you commit to perform SystemToCustomer? No
11b) Do you require a commitment for HeadPhoneTotranstec before you commit to perform SystemToCustomer? Yes
11c) Do you require that DownPayFromCustomer be performed before you request HeadPhoneTotranstec? -

11a) Do you require that ZipDriveTotranstec be performed before you commit to perform SystemToCustomer? No
11b) Do you require a commitment for ZipDriveTotranstec before you commit to perform SystemToCustomer? Yes

11c) Do you require that DownPayFromCustomer be performed before you request ZipDriveTotranstec? -

Fig. 34. Answers for the questions in phase 3

The resulting process order (PO) of the phase 3 can be shown according to the formalism we have been using as follows. Here EEO is the economic event order obtained at the phase 2.

PO = EEO ∪{decl(DownPayFromCustomer)<com(SystemToCustomer),
com(SoftwareTotranstec)<com(InitialPayToMicroSoft),
com(SpeakersTotranstec)<com(FinalPayToSony),
com(TransportationTotranstec)<com(PayToUPS),
com(HeadPhonesTotranstec)<com(PayToteratek),
com(ZipDriveTotranstec)<com(PayTolomega),
decl(SoftwareTotranstec)<com(SystemToCustomer),
com(SpeakersTotranstec)<com(SystemToCustomer),
com(HeadPhoneTotranstec)<com(SystemToCustomer),
com(ZipDriveTotranstec)<com(SystemToCustomer),}

The complete process order can be listed as below with the economic event order (EEO) resulted from the previous phase.

PO = {DownPayFromCustomer < SystemToCustomer,
SystemToCustomer < FinalPayFromCustomer
SoftwareTotranstec < SystemToCustomer
SystemToCustomer < InstallationPayToMicrosoft
SpeakersTotranstec < SystemToCustomer
HeadphonesTotranstec < SystemToCustomer
ZipdriveTotranstec < SystemToCustomer
DownPayFromCustomer < FinalPayFromCustomer
DownPayFromCustomre < InstallationPayToMicrosoft
FinalPayFromCustomer < FinalPayToSony
FinalPayFromCustomer < FinalPayToteratek
FinalPayFromCustomer < FinalPayTolomega
FinalPayFromCustomer < FinalPayToUPS
SoftwareTotranstec < InitialPayToMicrosoft
SoftwareTotranstec < InstallationPayToMicrosoft
IntialPayToMicrosoft < InstallationPayToMicrosoft
DownPayToSony < SpeakerTotranstec
SpeakerTotranstec < FinalPayToSony
DownPayToSony < FinalPayToSony
HeadphonesTotranstec < PayToteratek
ZipdriveTotranstec < PayTolomega
TransportationTotranstec < PayToUPS

book(processor) < reserve(Memories)
book(processor) < book(GraphicCards)

```
reserve(processor) < reserve(GraphicCards)
book(GraphicCards) < book(Monitor)
reserve(GraphicCards) < reserve(Monitor),

decl(DownPayFromCustomer)<com(SystemToCustomer),
com(SoftwareTotranstec)<com(InitialPayToMicroSoft),
com(SpeakersTotranstec)<com(FinalPayToSony),
com(TransportationTotranstec)<com(PayToUPS),
com(HeadPhonesTotranstec)<com(PayToteratek),
com(ZipDriveTotranstec)<com(PayToIomega),
decl(SoftwareTotranstec)<com(SystemToCustomer),
com(SpeakersTotranstec)<com(SystemToCustomer),
com(HeadPhoneTotranstec)<com(SystemToCustomer),
com(ZipDriveTotranstec)<com(SystemToCustomer),}
```

8.5 Process Model for transtec

At this stage, it is possible to generate the final process model in the executable process modeling language, BML. This generation of process model is achieved by applying production rules in Section 6.6 on the process order (PO) resulted in Section 7.4. The final process model is shown in Appendix A.

8.6 Lessons learnt

There are several important things to be mentioned, as we have gathered experience from the transtec case discussed above. The development of process models without any assistance from methodological guidelines such as P^3, is a very complicated manual task. Among the difficulties we have been faced with we discuss some in this section.

Even for the selected transtec business module, even after reducing the number of business partners (five external suppliers and the customer) filling the two matrixes (**[Table 4]**, **[Table 5]**) to get the business order is very complicated. This is because as the number of atomic economic-events increase, the number of cells in those matrixes goes up resulting difficulties in filling answers to each cell to reflect exact business requirement.

We have extended our earlier work (**[31]**, **[32]**) in order to overcome this difficulty. That is, the search space within those two matrixes have been reduced to upper triangular form by allowing to fill ‘<’, ‘>’ or ‘-’ to indicate ‘occur after’, ‘occur before’ or ‘ordering is not desired’ respectively.

Another limitation is the large number of small process segments resulted from the process generation in P^3 methodology. Even, in this reduced transtec case with six business partners, there are 52 small process chunks causing much pain for technical designer who are to generate code when implementing.

Also, it is noticeable that according to the current version of P^3 the main economic event that delivers value to the ultimate customer gets overcrowded with the number of direct materials that goes into final delivery increases (see process segment in page 92 of Appendix A).

The monetary offering for the supplier deliveries have to be instantiated with synchronous Request/Reply pairs to check whether suppliers deserve for receive payment before send base partner's commitment. These monetary offering diagrams need further communication through these Request/Reply pairs to get the delivery status, which is not modeled here.

Another questionable point in final process model is the meaning of providability checking diagrams after applying rule 3 for a business requirement, e.g., com(SpeakersTotranstec) < com(SystemToCustomer). Rule 3 has removed most of the intermediate communication between the SystemToCustomer process and the CheckSpeakerProvidability process, as this business requirement demands a stronger direct communication between SystemToCuatomer and SpeakersTotranstec (see page 85 of Appendix A). But still it can be argued that the CheckSpeakerProvidability process is needed for many other different purposes such as auditing.

Beside all the limitation and problem listed above our experience is that process modeling without any assistant like the one in P^3 is too complicated. Therefore, the advantages by adapting such a modeling methodology can be generalized as follows.

1. ***Complexity of low level technical design***
   Even for the selected very small business module above, when it comes to low level technical design, the complexity of process models increases exponentially. However, the P^3 methodology takes a designer comfortably through various development stages mechanically to reach the final design.
2. ***Communication of a design choice to different stakeholders***
   As the final process model composes many process segments reflecting different business objectives, it improves the communication of design choices to different interested parties.
3. ***Uniformity of resulting models***
   A real world process model task is often teamwork, and ensuring uniformity of resulting solutions is very much necessary. P^3 ensures that irrespective of the individual developer the results are uniform.
4. ***Trying out different design alternatives***
   One of the time-consuming tasks is trying out possible design alternatives. With a methodology like P^3 this task can be in part automated thereby saving a lot of time and effort.
5. ***Transparent business designing with minimum technical expert involvement***
   The guidelines of P^3 methodology come in the form of a wizard where the user has to answer relevant questions. As this interface can be formulated in the domain terminology, many of the technical design burdens can be taken away from the users.
6. ***Rapid Development***
   One of the big challenges for e-commerce systems is to reach the market in time. This demands not only the business idea but also the executable system to be delivered in time. A readily automatable methodology like P^3 makes it possible.

9 Conclusion and Further Work

9.1 Concluding Remarks

The main contribution of this work is a set of methodological guidelines that support a designer in moving from a business model to a process model in a systematic way. The approach has a number of advantages:

Identifying alternatives. The designer assistant helps the designer to identify and evaluate possible design alternatives when building the process model and thereby ensures that no useful alternatives are overlooked.

Traceability and motivation. When inspecting a process model, it is often difficult to understand why a particular solution has been chosen. By building a process model using the designer assistant, all design choices as well as their motivations are automatically and explicitly recorded.

Separation of concerns. The approach suggested makes an explicit distinction between the declarative aspects of a business model and the procedural aspects of a process model. This separation of concerns aids a designer in focussing on one problem at a time.

Seamless transition from analysis to realization. Using the designer assistant, the designer starts with a business model and builds successively a process model based on it. The end point of this activity is a set of diagrams that can be used for communication about the model as well as for actual execution.

In Chapter 2 we introduced the Language Action Perspective and we identified three problems that hinder an effective use of the Language Action approach. Following is a list of how the P^3 modeling techniques and guidelines introduced in the thesis have addressed those problems.

1. Using the Language Action approach encourages a low level perspective.

We suggest that the design of an e-commerce process be preceded by the design of a business model that focuses on actors and their value exchanges. A business model is a natural starting point for discussions with users and managers. When the business model has been designed, it is successively transformed and extended into a process model based on Language Action notions. In this way, the designer assistant helps the designer to investigate a large number of possible design alternatives before committing to one of them. Furthermore, it is also possible to move backwards and from a process model track the business objectives and decisions that motivated its design.

2. The notions and terminology of the Language Action approach are unfamiliar.

We propose an automated designer assistant that guides the designer through the task by means of a sequence of questions that use only terminology familiar to the ordinary user or manager. The user/manager can concentrate on the business idea being developed without paying any attention to underlying complex technologies.

3. There is a large distance between Language Action models and executable systems.

We suggest the use of communicating state machines, in the form of the executable language BML, for modelling processes. Thus, the specified process models are executable. Another advantage of using communicating state machines is that each state machines corresponds to an Action Workflow loop, which makes it easy to understand.

9.2 Further work

There is much future work to be done in order to get real application benefits from our proposal. In the section, we briefly list a few possible directions of future work.

1. ***Business Model***
   The business model discussed here can be extended by means of economical and marketing theories for a more in-depth analysis of real practical use. Work in the direction of identifying potential usage of business models has to be addressed.
2. ***Wizard of the designer assistant***
   The outlined appearance of the questions in the wizard has to be improved so that non-technical users will be able to understand them. In this direction, identification of high level concepts such as trust and flow dependencies, in which questions can be formulated, has to be studied further and has to re-shape the questions accordingly. Enhancing the graphical user interface of the wizard is also needed.
3. ***Production Rule System***
   The production rule system has to be tested against different cases to evaluate its correctness and completeness.
4. ***Extension for primitive process patterns***
   The four primitive process patterns discussed here are based on very basic conversation patterns. There is much work remaining to extend to cover cancellations, breakdowns, negotiations, etc. The scope of processes could also be extended to handle additional phases in e-commerce, like contact search as exemplified in BAT **[19]**.

In addition, we have identified some limitations of the P^3 framework. One of the noticeable limitations is that the generated process model from the methodology is composed of numerous small grained process segments resulting from each instantiated primitive process pattern. This may cause a big pain for the technical personnel who deal with script generation and code maintenance for a given design. One possible solution that could be tried out is parsing the generated process model through a "concatenator" capable to re-producing the process model with lager process segment granularity reflecting different business objectives.

Another limitation is that we have considered very primitive conversation styles in the definition of our process patterns. They may need to be extended to accommodate many other possible alternatives. However, those extended communication patterns have not been addressed in our framework. Consequently, further work is needed for the generation of process models by process designers to fine-tune the solutions to reflect exact business needs.

Although P^3 is not completed totally and needs many more extensions, still we feel that the approach has big potential in facilitating business process modeling.

10 References

1. Austin J. L., "*How to do things with Words*", Cambridge, MA: Harvard University Press.
2. Axelsson K., Goldkuhl G. and Melin U., "Using Business Action Theory for dyadic Analysis", accepted to *10th Nordic workshop on inter-organisational research*, Trondheim 2000.
3. Belina F., Hogrefe D. and Amarddeo S.: "*SDL with Applications from Protocol Specification*", Carl Hanser Verlag and Printice Hall International UK 1991.
4. CIMOSA: "*A Primer on key concepts, purpose and business value*, http://www.cimosa.de/cimosa/Primer/primer0.htm, (LVO[1] Feb 01st, 2002).
5. CYC project at MMC: http://www.cyc.com/publications.html, (LVO Feb 01st, 2002).
6. DEMO literature, http://www.demo.nl/DEMO/literatuur.htm, (LVO Feb 01st, 2002).
7. Dietz J.L.G. and Barjis J.: "Petri Net expressions of DEMO Process Models as a rigid foundation for Requirements Engineering", *The 2nd International Conference on Enterprise Information Systems (ICEIS'00)*, 2000.
8. Dietz J.L.G., "The What and the Why of Modelling Business Processes", R.M. van Es, A. Post (Eds.), "*Dynamic Enterprise Modeling*", Kluwer Bedrijfsinformatie, Deventer, 1996.
9. Dietz, J.L.G., and Mallens P.J.M., "Business Process Modeling as a Starting Point for Information Systems Design", *Data2Knowledge Newsletter, Part 1 (january 2001)*, *2 (march 2001) and 3 (may 2001).*
10. Dignum F. and Weigand H., "Communication and deontic logic", *Information Systems, Correctness and Reusability*, World Scientific, Singapore, 1995.
11. DoD Enterprise Model, Topic(s): "*Government BPR Project Reports, Process Modeling & Analysis*", The Department of Defense, http://www.c3i.osd.mil/bpr/bprcd/0035.htm, (LVO Feb 01st, 2002).
12. Enterprise Project at University of Edinburgh: http://www.aiai.ed.ac.uk/~entprise/, (LVO Feb 01st, 2002).
13. Fox M., Chionglo J.F. and Fadel F.G., "A Common Sense Model of the Enterprise", *Proceedings of the 2nd Industrial Engineering Research Conference,* pp. 425-429, Norcross GA: Institute for Industrial Engineers, 1993.
14. Fox M.S., Gruninger M., "On Ontologies and Enterprise Modelling", *International Conference on Enterprise Integration Modelling Technology*, Springer-Verlag 1997.
15. Geerts G. and McCarthy W. E., "The Ontological Foundaton of REA Enterprise Systems", *working paper, Michigan State University* (August 2000 and being revised for journal submission).
16. Genesereth M. R., & Nilsson N. J., "*Logical Foundations of Articial Intelligence*", Kaufmann, Los Altos, CA, 1987.
17. Goldkuhl G., "Generic business frameworks and action modelling", In proceedings of *Conference Language/Action Perspective*, Springer Verlag, 1996.
18. Goldkuhl G., "Generic Business Frameworks and Action Modelling", *First International Workshop on Communications Modelling - The Language/Action Perspective*, Springer Verlag 1996.
19. Goldkuhl G., "The six phases of business processes - business communication and the exchange of value", accepted to the *twelth biennial ITS conference*, Stockholm, 1998.
20. Gordijn J., Akkermans J. M. and Vliet J. C.: "Business Modeling is not Process Modeling", *eCOM2000 workshop, 19th International Conference on Conceptual Modeling 2000*.

[1] The abbreviation, LVO states last date visited on the URL.

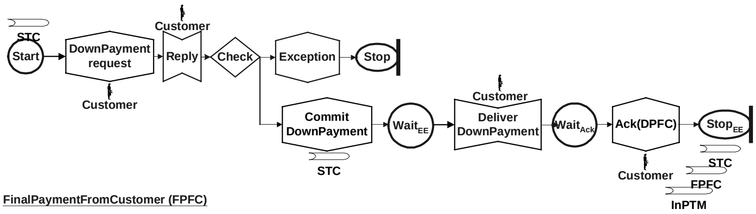
21. Gordijn J., Akkermans J. M. and Vliet J. C.: "What's in an Electronic Business Model? ", *Knowledge Engineering and Knowledge Management - Methods, Models, and Tools, 12th International Conference*, Springer-Verlag 2000.
22. Gordijn J., de Bruin H. and Akkermans J. M., "Scenario Methods for Viewpoint Integration in e-Business Requirements Engineering", *34th Hawaii International Conference On System Sciences*, IEEE, Hawaii, USA, 2001.
23. Gruber T., "A Translation Approach to Portable Ontology", *Knowledge Acquisition*, 1993.
24. Gruninger M., and Fox M.S., "The Role of Competency Questions in Enterprise Engineering", *Proceedings of the IFIP WG5.7 Workshop on Benchmarking - Theory and Practice*, Trondheim, Norway, 1994.
25. Guarino N., "Formal Ontology and Information Systems", *1st International Congerence on Formal Ontology in Information Systems (FOIS '98)*, 1998 Trento, Italy.
26. Guarino N., "Semantic Matching: Formal Ontological Distinctions for Information Organization, Extraction, and Integration", In M. T. Pazienza (ed.) *Information Extraction: A Multidisciplinary Approach to an Emerging Information Technology*, Springer Verlag, 1997.
27. Holbrook M. B., "*Consumer Value*", ISBN 0-415-19192, London Routledge 1999.
28. Integration Definitions: http://www.idef.com/Downloads/, (LVO Feb 01st, 2002).
29. Jayaweera P., "Mapping Function to Generate BML Process Model - A Technical Report", http://www.dsv.su.se/~prasad/html/MapFun.doc, (LVO Feb 01st, 2002).
30. Jayaweera P., Johannesson P. and Wohed P., "Collaborative Process Patterns for e-Business", *ACM SIGGROUP Bulletin*, 2001/Vol 22, No.2.
31. Jayaweera P., Johannesson P. and Wohed P., "From Business Model to Process Patterns in e-commerce", International Conference in Language Action Perspective (LAP '01) 2001, Montreal, Canada.
32. Jayaweera P., Johannesson P. and Wohed P., "Process Patterns to Generate eCommerce Systems", *2nd International Workshop on Conceptual Modeling Approaches for e-Business*, to be held in conjunction with the 20th International Conference on Conceptual Modeling (ER2001) Yokohama, Japan.
33. Johannesson P. and Perjons E., "Design Principle for Application Integration", *12th Conference on Advanced Information Systems Engineering*, eds. B. Wangler and L. Bergman, Springer LNCS, 2000.
34. Johannesson P. and Perjons E., "Design Principles for Application Integration", *12th Conference on Advanced Information Systems Engineering*, eds. B. Wangler and L. Bergman, Springer LNCS, 2000.
35. Johannesson P., "A Language/action based Approach to Information Modelling", in *Information Modeling in the New Millennium*, eds. M. Rossi and K. Siau, IDEA Publishing, 2001.
36. Johannesson P., Wangler B., and Jayaweera P, "Application and Process Integration - Concepts, Issues, and Research Directions", *Information Systems Engineering Symposium-CAiSE 2000*, eds. S. Brinkkemper, E. Lindencrona, and A. Sölvberg, Springer Verlag, 2000.
37. Kimbrough S. O. and Moore S. A, "On automated message processing in electronic commerce and work support systems: Speech Act", *ACM Transactions on Information Systems* (TOIS), 1997.
38. Lenat D., and Guha R.V., "*Building Large Knowledge Based Systems: Representation and Inference in the CYC Project*", Addison Wesley Pub. Co 1990.
39. Lind M., and Goldkuhl G., "Generic Layered Patterns for Business Modelling", *Sixth International Workshop on the Language-Action Perspective on Communication Modelling* Montreal, Canada, 2001.
40. Loebbecke C. and Schafer S., "Web portfolio based electronic commerce: the case of transtec AG", *Logistics Information Management*, Volume 14 No. 1; 2001.

41. Lokhorst G. J. C., “Ernst Mally's *Deontik* (1926)”. *Notre Dame Journal of Formal Logic*, 40 (2): 273-282, 1999 (2001). ISSN 0029-4527.
42. Mally E., “*Deontic Logic*”, http://mally.stanford.edu/deontic.html (LVO Feb 01st, 2002).
43. Malone et al.: “Towards a handbook of organizational processes”, MIT eBusiness Process Handbook, http://ccs.mit.edu/21c/mgtsci/index.htm, (LVO Feb 01st, 2002).
44. McCarthy W. E., “The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment”, *The Accounting Review* 1982.
45. Medina-Mora R. et al.: “The Action Workflow Approach to Workflow Management Technology”, *Proceedings of 4th Conference on Computer Supported Cooperative Work*, ACM Press, 1992.
46. Moore S., “*Formal Language for Business Communication*”, http://www-personal.umich.edu/~samoore/research/flbc/, (LVO Feb 01st, 2002).
47. Ohmae K., “*The mind of the strategist: The Art of Japanese Business*”, New York, 1982.
48. Perry R., “*Realms of value*”, ISBN 9-910-032287, Cambridge, Mass. Harvard Univ. Press 1954.
49. Porter M. E., “Competitive Advantages. Creating and Sustaining Superior Performance” The Free Press 1998.
50. QFD Institute - The official source for QFD, http://www.qfdi.org/transact.htm, (LVO Feb 01st, 2002).
51. RosettaNet Standards, http://www.rosettanet.org/, (LVO Feb 01st, 2002).
52. SAP Business Maps, http://www.sap.com/businessmaps/, (LVO Feb 01st, 2002).
53. Schäfer S. and Seibt D., “Using the Web to create the transparent enterprise - the success story of transtec AG”, Proceedings of the *12th International Bled Electronic Commerce Conference 'Global Networked Organizations'*, Bled/Slovenia, June 1999.
54. SDL Standards, http://www.sdl-forum.org/Publications/Standards.htm, (LVO Feb 01st, 2002).
55. Searle J. R., “A taxonomy of illocutionary acts”, K. Gunderson (Ed.), *Language, Mind and Knowledge*, Minneapolis: University of Minnesota, 1975.
56. Significant *REA Model* papers: http://www.msu.edu/user/mccarth4/paplist1.html, (LVO Feb 01st, 2002).
57. Taylor J. R., Groleau C., Heaton L. and VAN EVERY E. J., “*The Computerization of Work : A Communication Perspective*”, Thousand Oaks CA: Sage.
58. Taylor J., “The Limits of Rationality in Communication Modeling – a Semiotic Reinterpretation of the Concept of 'Speech Act', *Third International Workshop, The Language Action Perspective on Communication Modeling*, eds. G. Goldkuhl et.al. 1998.
59. Taylor, J. R., Groleau C., Heaton L. and VAN EVERY E. J.: “*The Computerization of Work : A Communication Perspective*”, Thousand Oaks CA: Sage.
60. Techniques and Methodology Working Group (TMWG): http://www.gefeg.com/tmwg/, (LVO Feb 01st, 2002).
61. The Internet Economy Indicators, Indicators Report, http://www.internetindicators.com/ keyfindings.html, (LVO Feb 01st, 2002).
62. TMWG N090: UN/CEFACT Modeling Methodology (UMM). N090, http://www.ebxml.org/project_teams/jdt/resources/, (LVO Feb 01st, 2002).
63. TOVE project at University of Toronto: http://www.eil.utoronto.ca/tove/ontoTOC.html, (LVO Feb 01st, 2002).
64. transtec AB, http://www.transtec-ab.se/, (LVO Feb 01st, 2002).
65. transtec AB, http://www.transtec-ab.se/, c.
66. UN/CEFACT - Document DownLoad, http://www.unece.org/cefact/docum/download/, (LVO Feb 01st, 2002).
67. Uschold M. and King M., “*Towards a Methodology for Building Ontologies*”, (also available from AIAI as AIAI-TR-183), 1995.
68. Visuera, http://www.visuera.com/, (LVO Feb 01st, 2002).

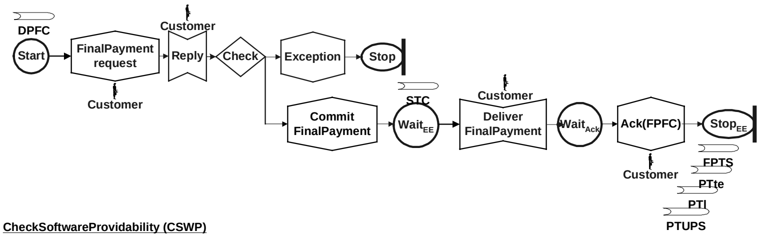
69. Wåhlander C., Nilsson M., and Törnebohm J, "Visuera Process Manager - Introduction", Copyright Visuera, 2001.
70. Weigand H. and Heuvel W. J., "Meta-Patterns for Electronic Commerce Transactions based on FLBC", *Hawaii Int Conf on System Sciences (HICSS'98)*, IEEE Press, 1998.
71. Weigand H. Verharen E. and Dignum F. "Dynamic business models as a basis for interoperable transaction design", *Information Systems*, 1997.
72. Weigand H., Moor A. de & Heuvel W-J. van den, "Supporting the evolution of workflow patterns for virtual communities", *33rd Hawaii Int. Conference of System Sciences*, 2000.
73. Weigand H., van den Heuvel W. and Dignum F., "Modelling Electronic Commerce Transactions – A Layered Approach", *Third International Workshop, The Language Action Perspective on Communication Modelling*, eds. G. Goldkuhl et.al. 1998.
74. Winograd T. and Flores F., "Understanding Computers and Cognition: A New Foundation for Design", Ablex, Norwood, N.J 1986.

Appendix A

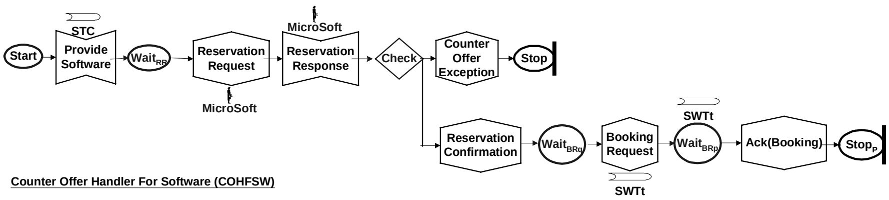
DownPaymentFromCustomer (DPFC)



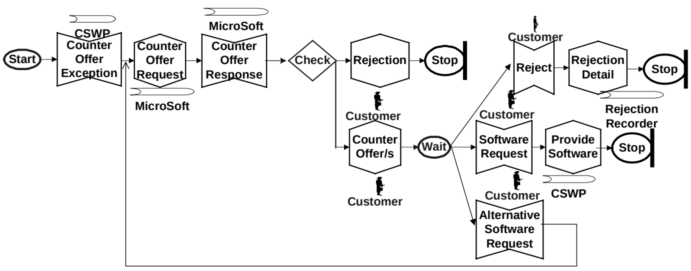
FinalPaymentFromCustomer (FPFC)



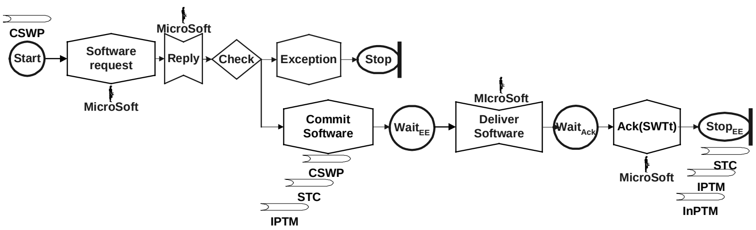
CheckSoftwareProvidability (CSWP)



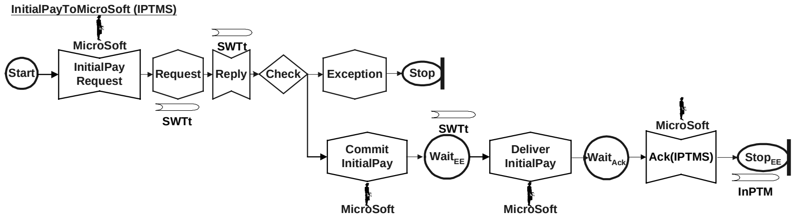
Counter Offer Handler For Software (COHFSW)



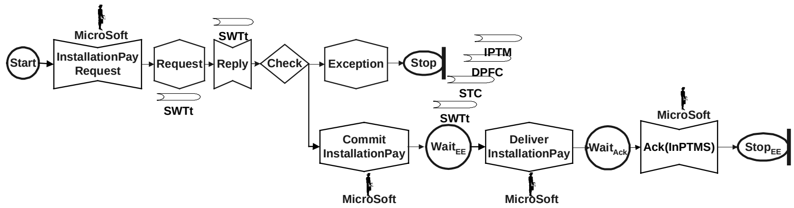
SoftwareTotranstec (SWTt)



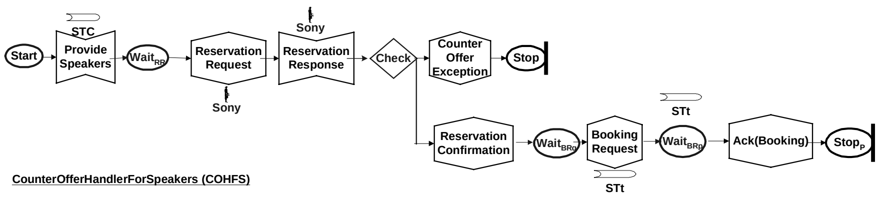
InitialPayToMicroSoft (IPTMS)



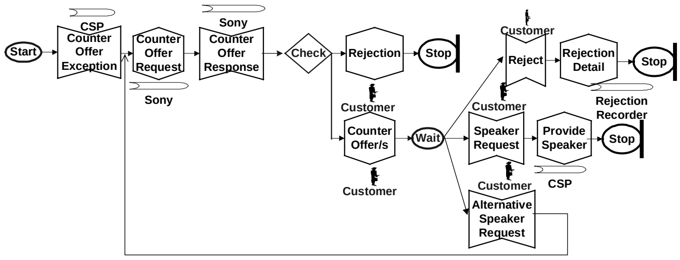
Installation PayToMicroSoft (InPTMS)



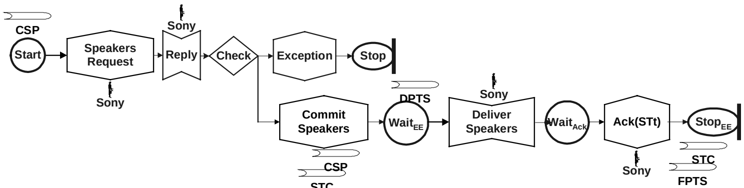
CheckSpeakersProvidability (CSP)



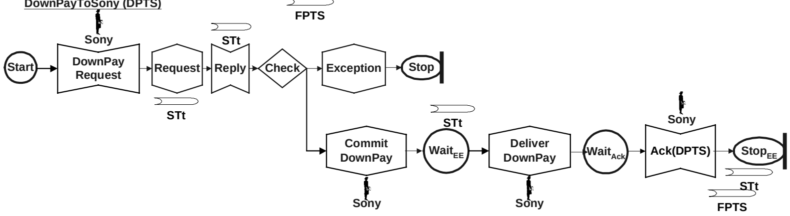
CounterOfferHandlerForSpeakers (COHFS)



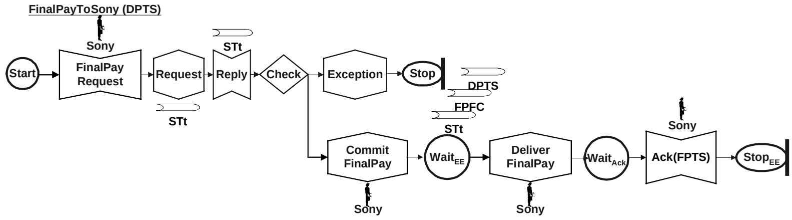
SpeakersTotranstec (STt)



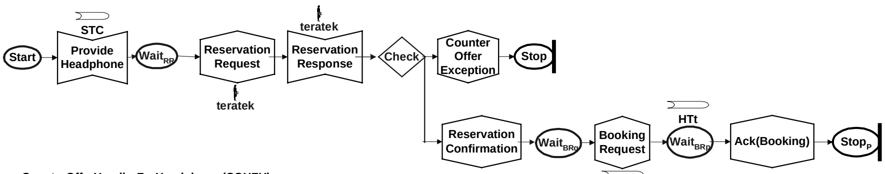
DownPayToSony (DPTS)



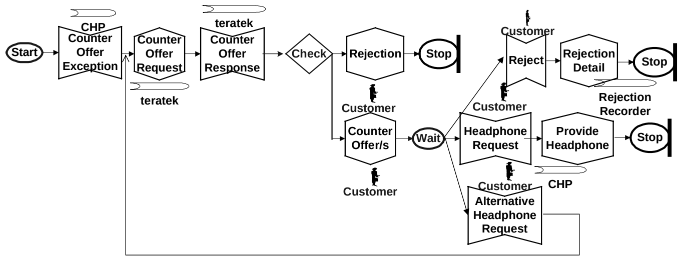
FinalPayToSony (DPTS)



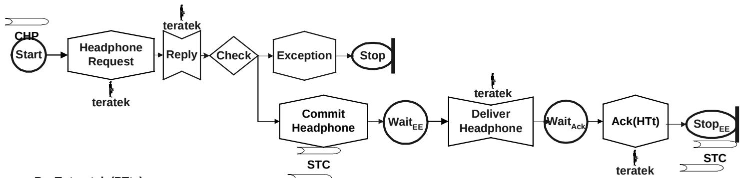
CheckHeadphoneProvidability (CHP)



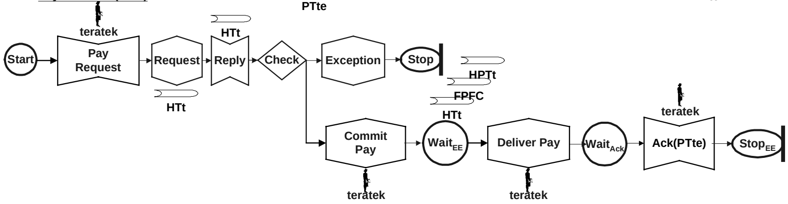
CounterOfferHandlerForHeadphone (COHFH)



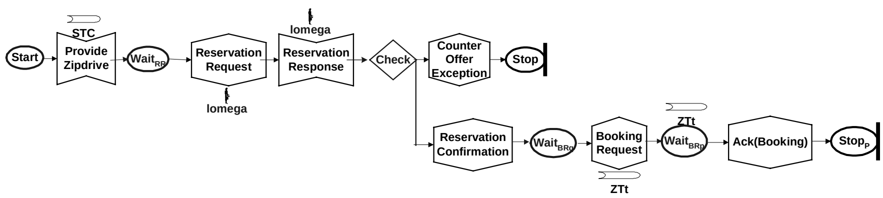
HeadphoneTotranstec (HTt)



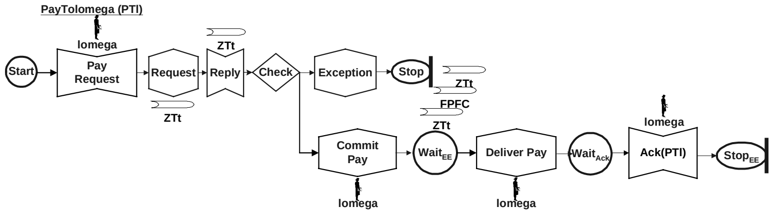
PayToteratek (PTte)



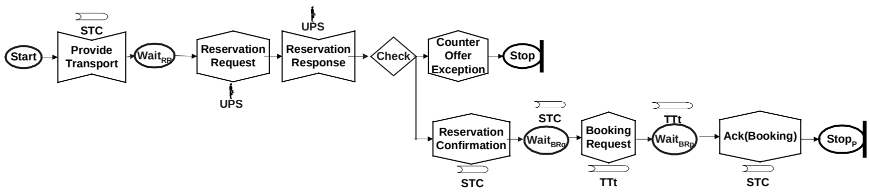
CheckZipdriveProvidability (CZP)



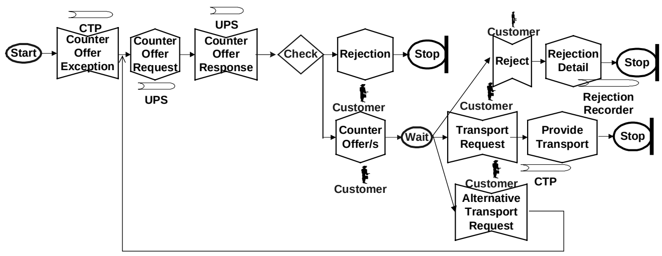
PayToIomega (PTI)



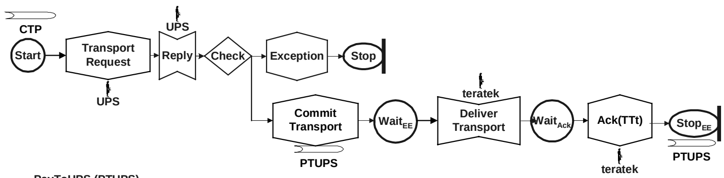
CheckTransportationProvidability (CTP)



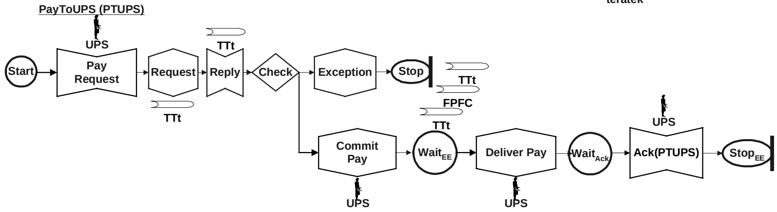
CounterOfferHandlerForTransportation (COHFT)



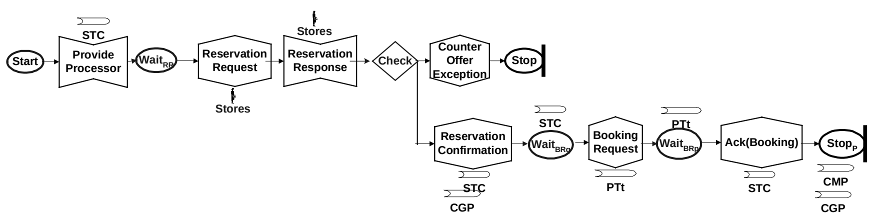
TransportationTotranstec (TTt)



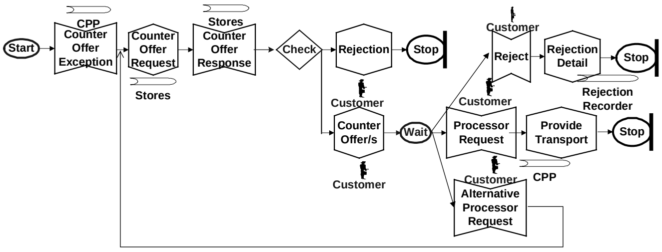
PayToUPS (PTUPS)



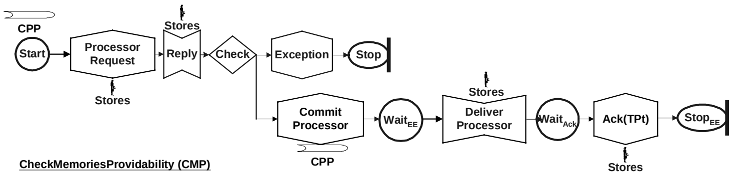
CheckProcessorProvidability (CPP)



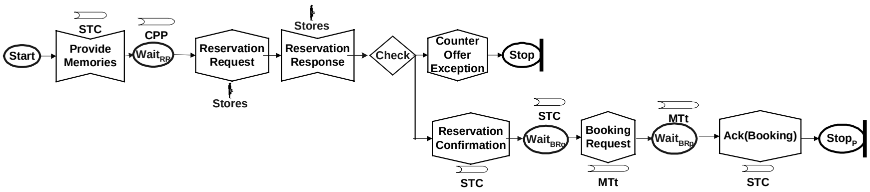
CounterOfferHandlerForProcessor (COHFP)



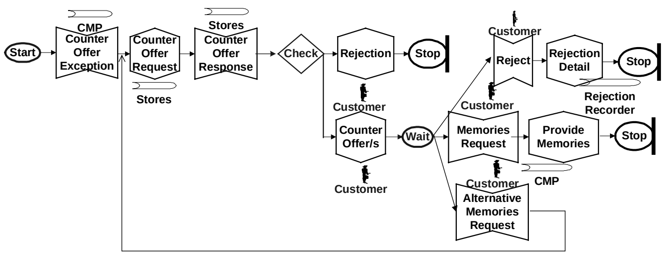
ProcessorTotranstec (PTt)



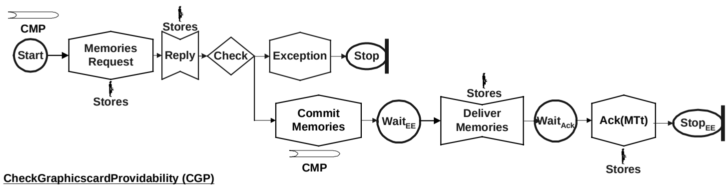
CheckMemoriesProvidability (CMP)



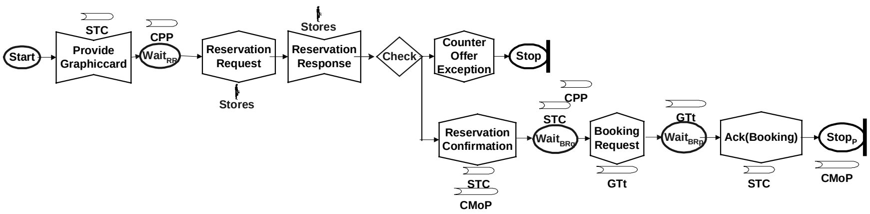
CounterOfferHandlerForMemories (COHFM)



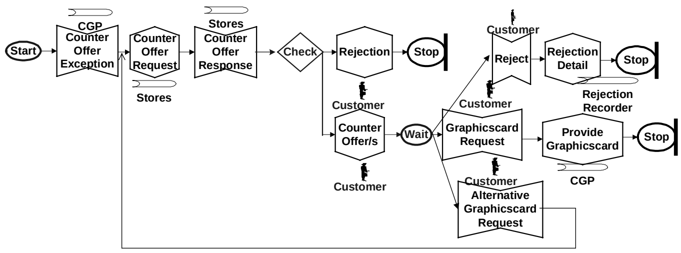
MemoriesTotranstec (MTt)



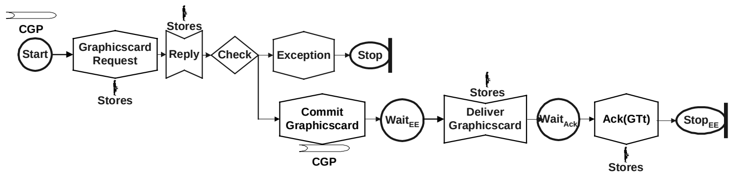
CheckGraphicscardProvidability (CGP)



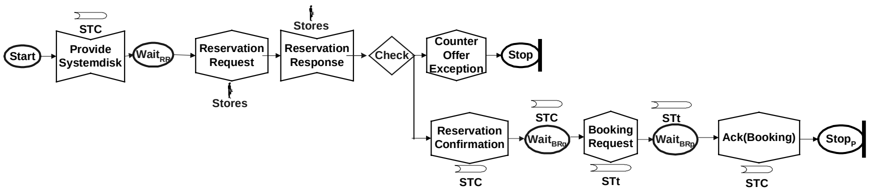
CounterOfferHandlerForGraphicscard (COHFG)



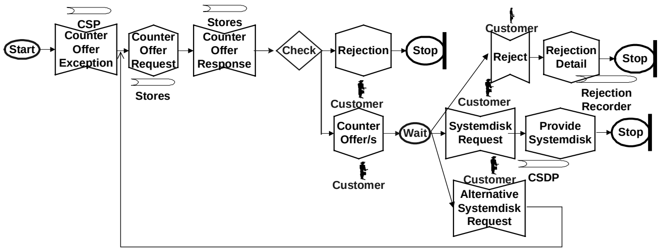
GraphicscardTotranstec (GTt)



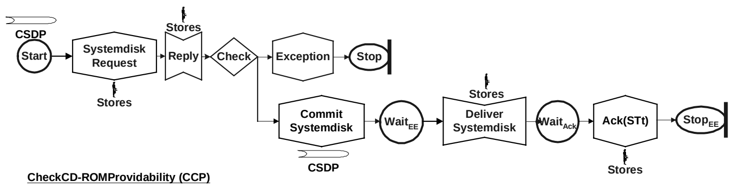
CheckSystemdiskProvidability (CSDP)



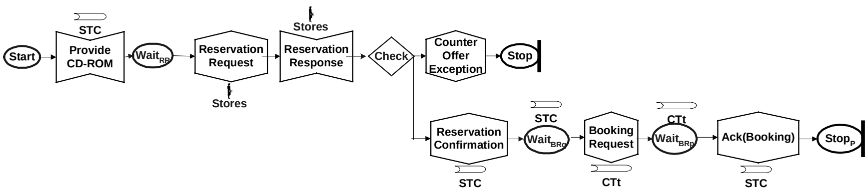
CounterOfferHandlerForSystemdisk (COHFSD)



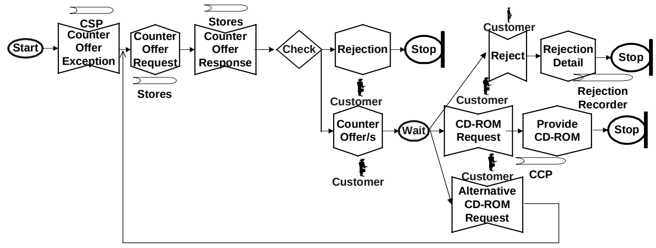
SystemdiskTotranstec (SDTt)



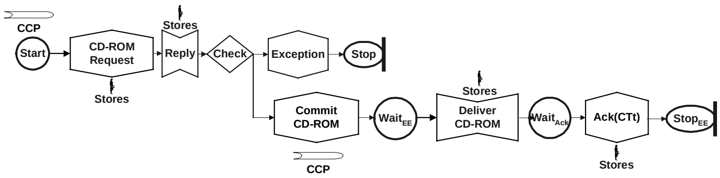
CheckCD-ROMProvidability (CCP)



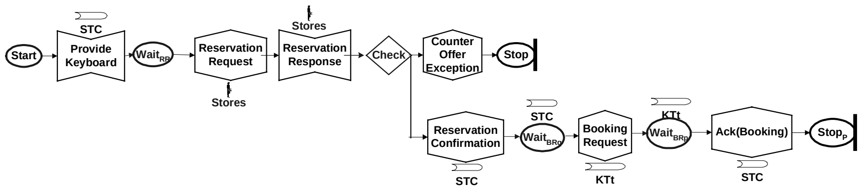
CounterOfferHandlerForCD-ROM (COHFC)



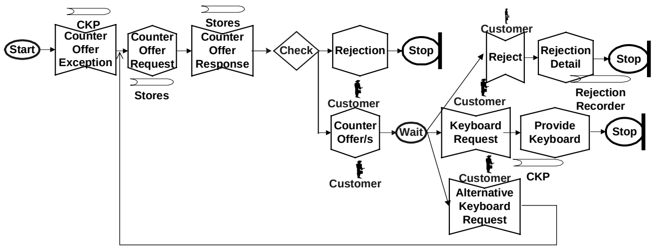
CD-ROMTotranstec (STt)



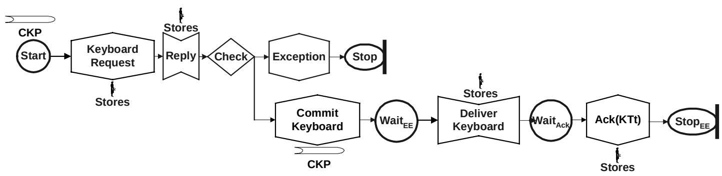
CheckKeyboardProvidability (CKP)



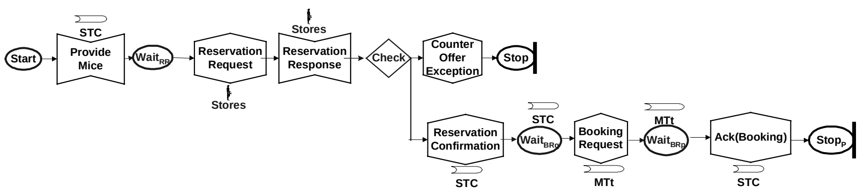
CounterOfferHandlerForKeyboard (COHFK)



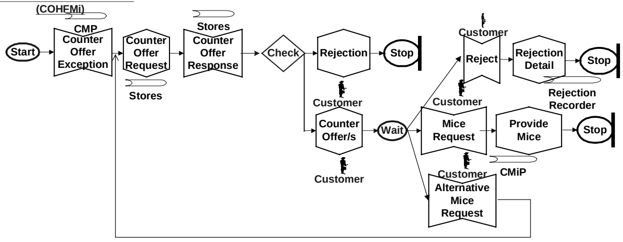
KeyboardTotranstec (STt)



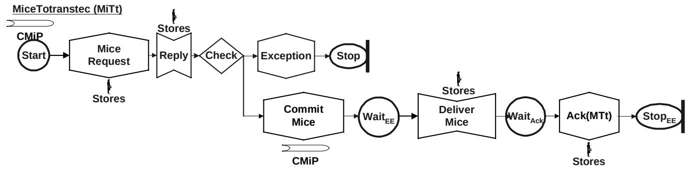
CheckMiceProvidability (CMiP)



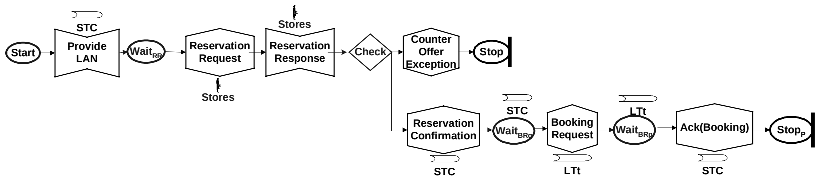
CounterOfferHandlerForMice (COHFMi)



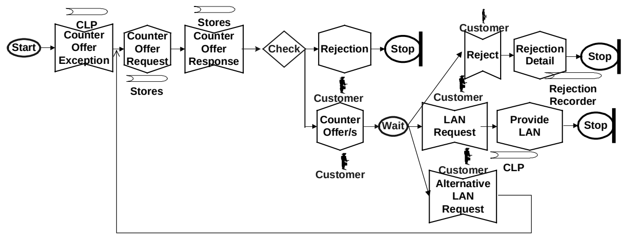
MiceTotranstec (MiTt)



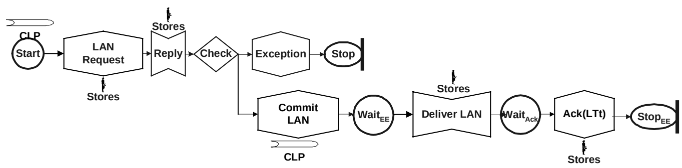
CheckLANProvidability (CLP)



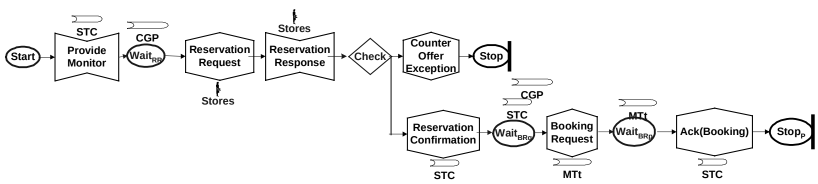
CounterOfferHandlerForLAN (COHFL)



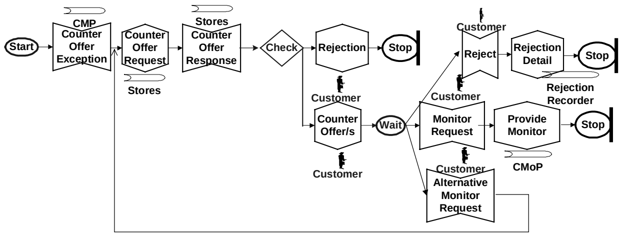
LANTotranstec (LTt)



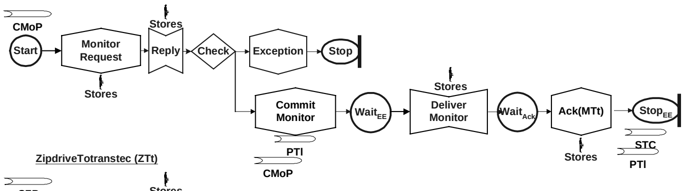
CheckMonitorProvidability (CMoP)



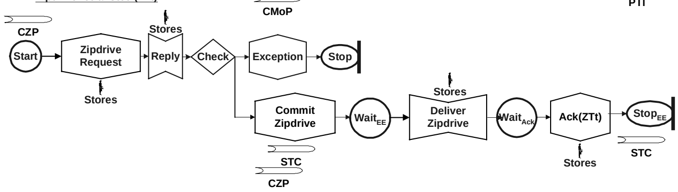
CounterOfferHandlerForMonitor (COHFMo)



MonitorTotranstec (MoTt)



ZipdriveTotranstec (ZTt)



SystemToCustomer (STC)