A Unified Framework for e-Commerce Systems Development:

Business Process Pattern Perspective

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

September 24th, 2004

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 realization. This work proposes a unified framework that can be used as a basis to analyze, to interpret and to understand different concepts associated at different stages in e-Commerce system development.

In this thesis we illustrate how UN/CEFACT's recommended metamodels for business and process design can be analyzed, extended and then integrated for the final solutions based on the proposed unified framework.

Also as an application of the framework, we demonstrate how process-modeling tasks can be facilitated in reaching e-Commerce system design. The proposed methodology, called BP^3 stands for Business Process Patterns Perspective. The BP^3 methodology uses natural language interface to capture different business requirements from the designers. It is based on defined process patterns, and the final solution will be generated by applying the captured business requirements by means of a set of production rules to complete the inter-process communication among these patterns.

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 and the University of Ruhuna, Sri Lanka for granting me study leave.

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 and Maria Bergholtz for working closely in BP[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 the necessary facilities and a pleasant environment to do my research work. Also I would like to thank Dr. Gihan Wikramanayake of University of Colombo School of Computing, Sri Lanka for his effort to introduce the areas of our work in Sri Lanka.

At last but not least, I express my gratefulness to my wife Jayani, to my loving daughter, Nileesha and son, Savidu 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

List of Figures

1 Introduction

This chapter is dedicated to introduce the background, motivation of the work presented in this thesis and a general overview of the research methodology followed during the work. Also, the chapter contains a briefing of the theoretical basis for the proposed approach and lists work in relation to the approach that has been published in different forums.

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 also benefit indirectly.

No single force embodies digital economy like the Internet. The Internet has influenced and changed the way we work, the way we learn, the way we do business and has changed our entire lifestyle. We are experiencing these changes at a growing rate as the Internet grows exponentially. The Internet Economy Indicators reports that Internet economy grew at a 173.6% from 1999 to 2000 **[81]**.

Therefore, Internet is the technology for e-Commerce as it offers easier ways to access companies and individuals at a very low cost in order to carry out day-to-day business transactions. 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.

With the growing interest and activities in e-Commerce, there is an increasing need for methods and techniques that can help in the design and management of e-Commerce systems. In e-Commerce, systems design 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*. This means that the process of designing e-Commerce systems consists of two main phases. First, a business requirement capture phase focusing on value exchanges, and secondly, a phase focused on operational and procedural realization.

In the business requirement capture phase, coarse-grained artifacts as well as their relationships and arrangements in business collaborations are represented by means of business model constructs at a very abstract level. The objective of business requirement capture phase is to construct business models that represent descriptive aspects of the e-Commerce systems being developed such that they can be easily communicated with domain experts and other system users.

In contrast, the specification of a process model deals with more fine-grained views of business communications, their relationships and choreography in business collaborations. The objective of the procedural and operational realization phase is to construct process models that can communicate business requirements to the designers. Hence, designers can build systems from these process models that meet captured business requirements.

Although the two phases in e-Commerce design, and their related models, have different focuses, there is clearly a need for integrating them. A unified framework covering coarse-grained business modeling artifacts to fine-grained process specification views provides several benefits. It can be used for supporting different user views of the system being designed, and it can form the basis of a precise understanding of modeling artifacts and their inter-relationships. Another advantage of a unified framework is that it can be used for process integration, i.e. to provide measures for the establishment of correspondences between different structures in process models. Ways of measuring correspondences is a prerequisite for the transformation mappings and conflict analysis that must be undertaken before the models are integrated. Also the framework offers a general and uniform analysis mechanism with a number of semantic primitives with meanings that can be agreed upon across e-Commerce frameworks. Finally, a promising framework in this nature can be used as the basis for building tools that can support and automate much of the e-Commerce system development process from very early stages capturing business requirements until final system delivery.

One central contribution of this thesis is to introduce a unified framework that can integrate business models and process models in e-Commerce system development with respect to globally accepted modeling standards such as UN/CEFACT modeling methodology (UMM) [85], ebXML [22], and Business Process Modeling Notion (BPMN) [9]. These technologies are open allowing anybody to develop solutions based on them. The technology neutral nature of these standards allows solutions to be mapped into different underlying implementation platforms. The selection of UMM and BPMN as our conceptual and notational framework is simply motivated with afore mentioned factors.

Another main contribution of this work is a methodology called BP^3 (Business Process Pattern Perspective) explained in later chapters. The BP^3 methodology can be readily automated by means of Designers Assistant tools to support e-Commerce systems generation. Such business process model generations discussed in this thesis are targeted at specifically BPMN specifications, but the approach can be tailored into any other available process specification languages.

1.2 Purpose

With the growing interest and activities in e-Commerce, there is an increasing need for methods and techniques that can help developers with design and management of the e-Commerce systems. Today, there are many different approaches, and standardization efforts, to support e-Commerce systems development processes. Among them, there are some work with global level acceptance for business process specification and e-Commerce system development such as UMM [85], Business Process Management Initiative [8] and RosettaNet [70].

An e-Commerce system development process is not trivial, and in most cases it involves very complex and time consuming modeling tasks. Also it requires different levels of participation at successive stages from different categories of stakeholders. However, when working with currently available methods and techniques, our experience is that in many cases they are developed targeting a specific type of stakeholders. Moreover, these tools are not defined with a precise transition from one stage to another in e-Commerce systems development and at the same time some are not providing system designers with adequate assistance at some stages to make the design task easy and comprehensive.

The main purpose of the work presented in this thesis is to develop a methodology (BP^3) based on a unified framework that can facilitate the e-Commerce system development process. We are proposing a unified framework that can support and integrate different user views of the systems being developed. Our understating is that such a framework is helpful in the precise understanding of concepts in different modeling views and their inter-relationships. The proposed framework is the basis for design guidelines for e-Commerce systems that is introduced at latter stages of this thesis.

1.3 Research Goal

As mentioned in the background section, there is clear need to integrate different models associated in e-Commerce system development process. Besides that, methodologies, techniques, and tool support that have capacity in assisting different stakeholders involved in e-Commerce system development process are by all means welcome!

The main objective of the research work presented in this thesis is two folded.

1. **A Unified Framework** – An interdisciplinary framework that can be used for analysis, interpret and integrate Business Models and Process Models in e-Commerce systems design
2. **A Designers Assistant** – A methodology based on the framework proposed above to support and generate business process models for e-Commerce Systems

1.3.1 A Unified Framework

A Unified Framework is the central contribution of this work. The Framework we are proposing here is based on Speech Acts Theory [1] and Language Action Perspective [15]. The framework integrates the contents of business models and process models. We use UMM [84], ebXML [22] and BPMN [9] as a conceptual and notational basis for the illustration of the Unified Framework.

Specifically the UMM Business Requirements View (BRV) metamodel as the basis for business models and the UMM Business Transaction View (BTV) metamodel and ebXML Business Process Specification Schema (BPSS) as the basis for process models are taken into consideration in this work with some extensions. BPMN Business Process Diagrams (BPD) has been used to document process models in this thesis.

1.3.2 A Designers Assistant

As an application of the proposed framework, we have developed and illustrated in detail here a Designers Assistant that can facilitate e-Commerce systems development process. Our investigations into SAP Collaborative Business Maps [71], RosettaNet [70] and other laboratory cases explore the inherent complexity and time-consuming nature of real world business process modeling tasks in e-Commerce domain. The Designers Assistant will relieve much of the burden business process designers are facing in this context.

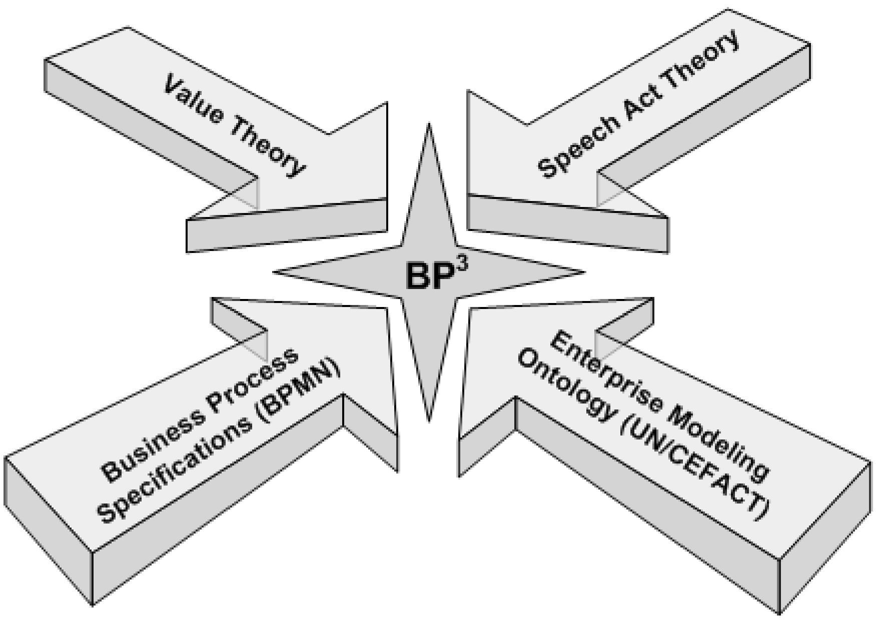


Fig. 1 Foundation of BP^3

The Designers Assistant that has been developed in this research work is named as BP^3. BP^3 stands for Business Process Pattern Perspective. As the name itself implies, its process specification is based on set of pre-defined primitive process patterns. However, it has solid theoretical foundation as shown in [Fig. 1].

The meaning of any business is to create value for participating business partners. Therefore, we have based our approach on some aspects of Value Theory as the first corner stone of our approach [41], [68] and [63]. The means for creating value is through communication where trading partners request, response, commit, fulfill and acknowledge. Secondly, as the central foundation of the proposed approach, it is based on Speech Act theory and Language Action Perspective, which deals with language pragmatics and is realized through physical actions, [1], [74] and [97].

Having founded our approach on well-established theories we had to face the challenge 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 developments for adaptation in our work. The UN/CEFACT [84] 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 BP^3 methodology. Global level participation in the UN/CEFACT's standardization effort, openness and technology neutral nature were the fundamental features that motivated our selection of UN/CEFACT's concepts and notations.

The approach we are proposing here is generic and can be adapted to generate its solution in any business process specification language (PSL) available in the field. However for illustrative purposes of BP^3 methodology in this thesis, we selected Business Process Modeling Notation (BPMN), a visual process-specification language [9]. BPMN has been proposed by BPMI to bridge the gap between visual business processes and formal process specification languages that utilize mathematical models such as pi-calculus [66] (for instance BPEL4WS [6]).

1.4 Research Methodology

As mentioned in the research goals sections, the first half of the thesis is dedicated to introduce the Unified Framework that we are proposing for e-Commerce system development. For that purpose the thesis starts with introducing underlying theories that the Unified Framework is founded on. Then, analysis of the UN/CEFACT and ebXML metamodels with respect the approach proposed is discussed.

Finally a detailed description and the process followed in the development BP^3 methodology is documented. The entire development process of BP^3 methodology can be visualized as in the diagram below.

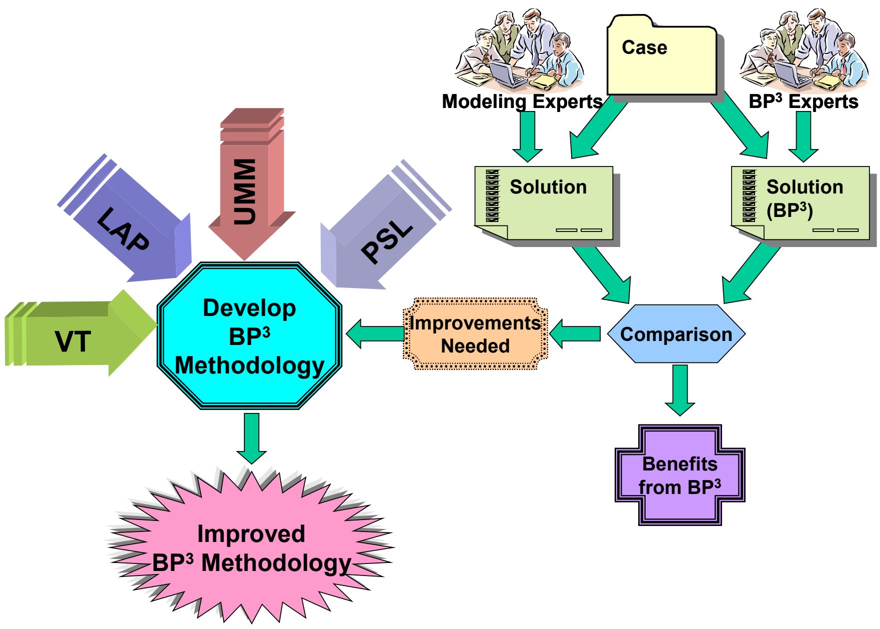


Fig. 2 Development Layout of the BP^3 Methodology

As mentioned, our understanding is that there is clear need in facilitating and in assisting e-Commerce systems designers to meet ever-increasing requirements. In order to meet this, designers should be able to develop precise and comprehensive models at different stages in development workflow, and they should be able to integrate these models so that these business requirements can be propagated seamlessly into the final implementations. In addition generation of e-Commerce systems with minimal human intervention is welcome, so that designer can reach solutions rapidly and also try out possible different design alternatives. Aiming at these targets we propose a novel approach called BP^3 methodology based on Value Theory, Language Action Perspective, UN/CEFACT modeling ontology and Process Specification Languages.

We put forward two main hypotheses as listed below. Through the proceeding chapters in this thesis the validity of those claims have been investigated.

Hypothesis 1: "The proposed unified framework provides precise and clear understanding of different modeling concepts associated with different models at development stages of e-Commerce Systems."

Hypothesis 2: "The BP^3 methodology can assist e-Commerce systems designers in reaching more precise solutions with minimal effort".

The BP^3 methodology that has been presented here is a refined and an extended version to the initial proposal of our research work. The earlier versions of BP^3 [43] have been tested in various ways and extended with respect to different process specification languages and their underlying concepts. The BP^3 methodology development process can be captured as in **Fig. 2**. Earlier versions of BP^3 have been tested against different laboratory cases and also with involvement of modeling experts in order to evaluate the overall approach. While identified advantages and benefits from our approach preserved and further improved in later versions of the methodology, limitations and error are debugged by feeding them back into the BP^3 development process as in **Fig. 2**.

1.5 Related Publications

This thesis is based on different contributions from different research groups and different research projects at Information Systems Laboratory at the Department of Computer and Systems Sciences, Stockholm University and Royal Institute of Technology, Stockholm, Sweden. The research results from work related to the thesis have been accepted and published in several occasions. Listed below is a selected set of our publications that have been accepted and presented in different forums.

- Maria Bergholtz, Prasad Jayaweera, Paul Johannesson and Petia Wohed, "A pattern and dependency based approach to the design of process models", *at the 23nd International Conference on Conceptual Modeling (ER 2004), Shanghai- China, Springer-Verlag*, LNCS.
- Maria Bergholtz, Prasad Jayaweera, Paul Johannesson and Petia Wohed, "Bringing speech Acts Into UMM", *at the 1st International REA Technology Workshop, Copenhagen-Denmark, April 22-24 of 2004*.
- Prasad Jayaweera and Paul Johannesson, "A Patient Centred Process Ontology for Information Visualisation in Health Care", *Poster Paper for EMOI - INTEROP 2004 (Enterprise Modeling and Ontologies for Interoperability)* at 16th International Conference on Advanced Information Systems Engineering (CAiSE '04), Riga-Latvia.
- Maria Bergholtz, Prasad Jayaweera, Paul Johannesson and Petia Wohed, "Modelling Institutional, Communicative, and Physical Domains in Agent Oriented Information Systems", in Post Proceedings *from both AOIS03 events (Chicago and Melbourne) Springer-Verlag*, LNCS.

- Maria Bergholtz, Prasad Jayaweera, Paul Johannesson and Petia Wohed, “Reconciling Physical, Communicative and Social/Institutional Domains in Agent Oriented Information Systems - a Unified Framework”, in Proceedings of *AOIS 2003 at the 22nd International Conference on Conceptual Modeling (ER 2003)- Chicago-USA, Springer-Verlag*, LNCS 2814.

- Maria Bergholtz, Prasad Jayaweera, Paul Johannesson and Petia Wohed, “Business and Process Models – a Unified Framework”, in Proceedings of *eCOMO 2002 at the 21st International Conference on Conceptual Modeling (ER’2002),Tampere-Finland*, Springer-Verlag LNCS 2784.

- Prasad Jayaweera, “*A Methodology to Generate e-Commerce Systems: A Process Pattern Perspective (P^3)*”, Licentiate of Philosophy Thesis March 27th, 2002.

- Prasad Jayaweera, Paul Johannesson and Petia Wohed, “Collaborative Process Patterns for e-Business”, *ACM SIGGROUP Bulletin 2001/Vol 22, No. 2*

- Prasad Jayaweera, Paul Johannesson and Petia Wohed, “Process Patterns to Generate e-Commerce Systems”, in Proceedings of *eCOMO 2001 at the 20th International Conference on Conceptual Modeling (ER 2001), Yokohama-Japan*, Springer-Verlag LNCS 2465.

- Prasad Jayaweera, Paul Johannesson and Petia Wohed, “From Business Model to Process Pattern in e-Commerce”, in Proceedings of *International Workshop on Language-Action Perspective on Communication Modeling (LAP 2001), Montreal-Canada*.

- Paul Johannesson, Benkt Wangler, and Prasad Jayaweera, “Application and Process Integration - Concepts, Issues, and Research Directions”, at *Symposium on the state of the art of Information*

Systems Engineering in the 12th Conference on Advanced Information Systems Engineering (CAiSE 2000), Stockholm-Sweden, Springer-Verlag.

- Prasad Jayaweera, "Process Algebraic Business Process Models", *Immerging Issues in Computer and Systems Sciences (IICSS 2000), Stockholm-Sweden*.

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 BP^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 **[74]** 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 **[97]** in 1986.

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. 3**]. 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.

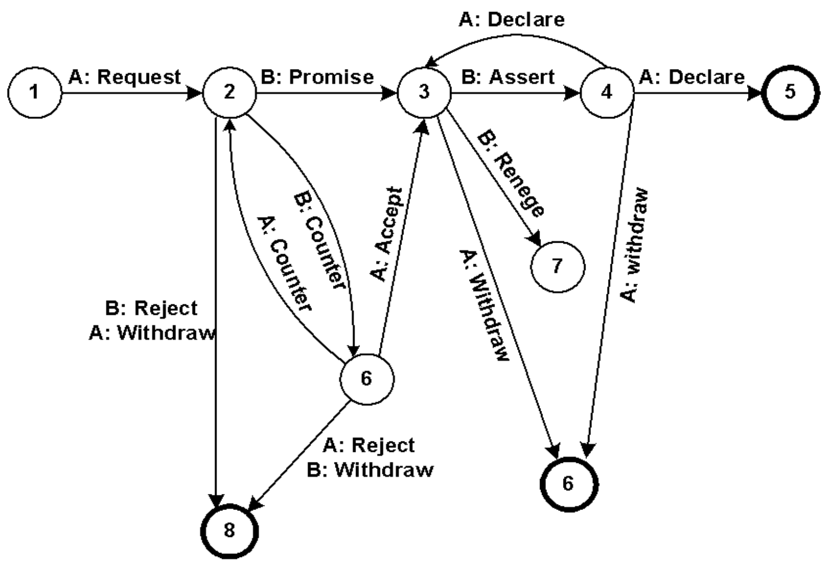


Fig. 3 A State transition Diagram of a Conversation for Action **[97]**

2.3 Action Workflow Loop

Action Technologies **[61]** 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.

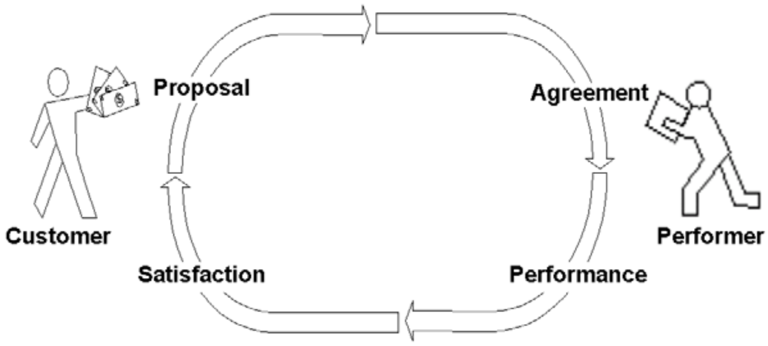


Fig. 4 A basic Action Workflow Loop **[61]**

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. 4**].

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 **[96]**.

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 **[61]** 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 proposed in this thesis.

2.4 Dynamic Essential Modeling of Organization (DEMO)

Dynamic Essential Modeling of Organization (DEMO) **[13]** 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. 5**].

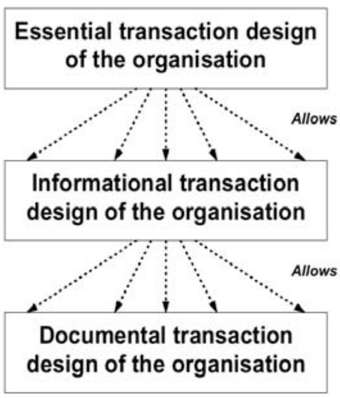


Fig. 5 DEMO Transaction design and levels of abstraction **[13]**

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 pass 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. 6].

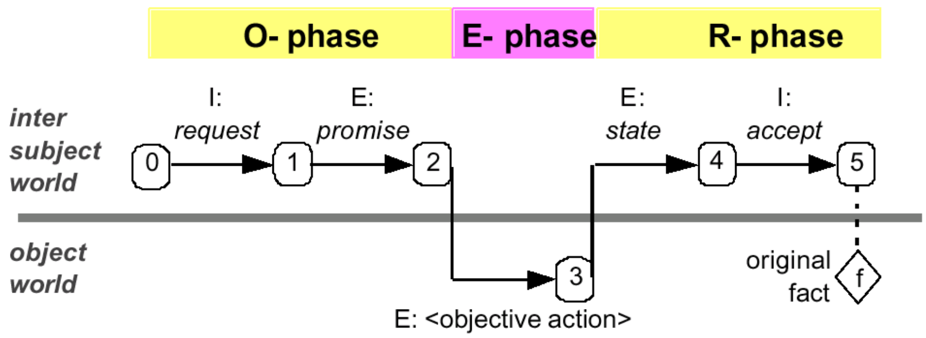


Fig. 6 The basic pattern of the DEMO transaction [18]

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), [29] 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 [31]:

5. ***Business prerequisites phase***, where prerequisites are established (both within the supplier's and customer's organization) for performing business (sales/ purchases)
6. ***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.
7. ***Contact establishment and proposal phase***, where the supplier presents available and possible offers to a specific customer showing some needs and purchase interest.

8. ***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.
9. ***Fulfillment Phase***, where the supplier and customer fulfill their commitments. The supplier fulfils the commitment by performing delivery and customer fulfils by paying for the received delivery.
10. ***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.

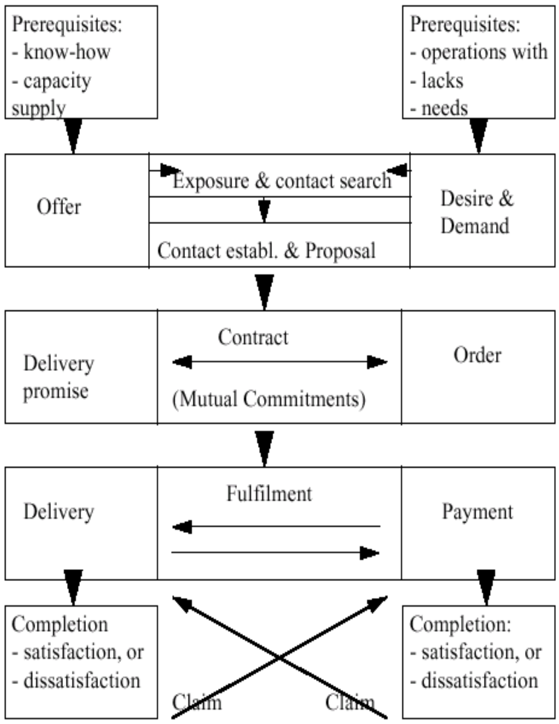


Fig. 7 The six generic phases of business processes [2]

The six generic phases of business processes in BAT and their relationships have been depicted in [Fig. 7]. 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 [93] distinguishes between five levels of (communicational) analysis meta-patterns from the lowest level of speech acts to the highest level of scenarios [Fig. 8].

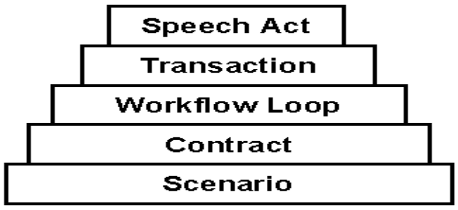


Fig. 8 Levels of Meta-Analysis Patterns [93]

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) [62]. 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 [95] 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 Weigand'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 defined 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 [94] 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) has been distinguished as meta-patterns since they do show structures. These meta-patterns have normally the following form: identification - essential transaction - ending of relationship. These metapatterns can be used and reused profitably in electronic commerce once they are made available in a well-documented form.

An example of an incomplete scenario type definition.

```
 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 [55] 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 Weigand's hierarchical layered architecture are also pointed out. Lind et al.'s alternate hierarchical architecture is shown in [Fig. 9].

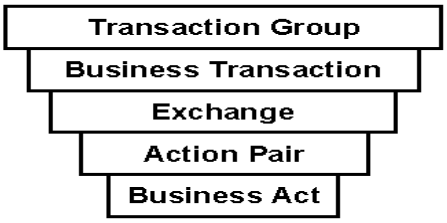


Fig. 9 Layers of Generic Patterns for Business Modeling [55]

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

The possible confusion of usage of the term "transaction" in Weigand's work compared to other language and communicative action perspectives has been highlighted. Also the elimination 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 [56], [57], [19]) 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 disputed. 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 against. Although 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 lead 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 be taken into account. They questioned again on derivation of multi-party involvement at such higher level from lower layers where only two party involvements 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 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 organization, and new business models are emerging, such as virtual enterprises, integrated supply chains, and value networks. A common theme is that of inter-organizational co-operation and communication. Business processes are not carried out within a single organization but across organizational boundaries. As noted in [96], inter-organizational processes have two distinguishing features. First, the resources needed for a process cannot be assigned centrally as they reside in different organizations. Second, the organizations involved in a process have a certain degree of autonomy meaning that no central authority has control over all the co-operating organizations. 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 [19] such as request, acknowledgement, commitment, obligation, responsibility, and trust. Thus, the Language Action approach to communication and information modeling 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, [97], [96], [30], [16], and. [50].

The limits of the applicability of the Language Action approach have been widely discussed in academia, e.g. the Suchman/Winograd debate, [77]. 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 formalized and structured than many intra-organizational work processes. We would like to add the following three factors to the list of hinder 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 modeling easily encourages a low-level perspective where the modeling 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 specialized 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 also to overcome the aforementioned hindering factors.

2.9 Discussion on LAP

In this Chapter, we have tried to explain briefly some of the popular LAP approaches. Our intension of introducing fundamentals of speech acts theory and its adaptation in e-Commerce as LAP is to indicate the readers how we have based our proposed framework on speech acts theory and how we have been influenced with these variants of LAP.

However, the significant difference in our approach compared to aforementioned is that the division of the domain in which e-Commerce systems being designed into three worlds as explained in detail in Section 4.3. This division facilitates to understand business models and process models are to develop in Social/Institutional and Physical worlds respectively, while communicative world based on speech acts integrates the two.

A future direction of research in this line could be a comparative empirical study to understand what advantages designers receives through the proposed approach in this thesis over other existing ones. Such thorough study could also suggest necessary amendments and possible enhancements to improve the usage in real environments.

3 Enterprise Modeling Ontology and Business Model

This chapter consists of two main sections: *Enterprise Modeling ontology* and *e-Business Model*. While Section 3.1 is dedicated to introduce Enterprise Modeling ontology, Section 3.2 introduces e-Business Model that has been associated in this work.

3.1 Enterprise Modeling Ontology

First, we introduce the Enterprise Modeling ontology[1] underlying our work. It is an extension of the Resource-Event-Agent (REA) [27] ontology to accommodate the Language Action Perspective. The chapter begins with a general discussion of ontology, a brief description of some efforts in development of Enterprise Modeling ontology and then moves to the specifics relevant to the thesis.

3.1.1 A Definition of Ontology

"An ontology is an explicit specification of a conceptualization" is a widely accepted definition by Gruber [36]. 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 [28].

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 [27]. 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.

[1] We have followed Guarino's distinction in using term 'ontology' as in [38].

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.

3.1.2 A Classification

Guarino [39] 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. 10].

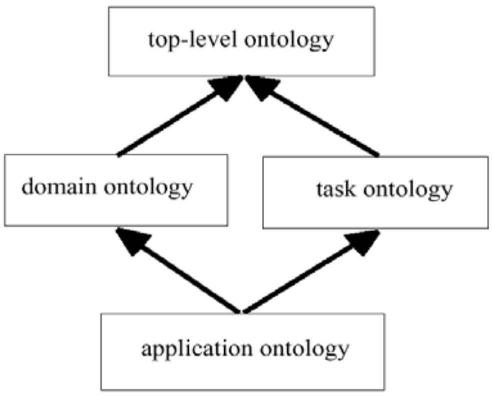


Fig. 10 Kinds of ontology according to their level of dependence [39]

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 component*.

3.1.3 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 [11], DoD-wide GEM of US Department of Defense [21] 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 [54], [12], the Enterprise Project at University of Edinburgh [23] and the TOVE project at University of Toronto [83] 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 **I**ntegration **DEF**initions (IDEF), [42]: 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

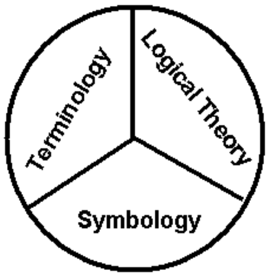


Fig. 11 Main Component of ontology [26]

Therefore, ontology can be seen in three different perspectives as mentioned in [26] and in [Fig. 11] above.

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 **[69]**, etc.
6. Cost: representation of resource cost, activity cost, activity based costing, etc

3.1.4 REA Ontology

Resource-Event-Agant (REA) **[60]**, **[75]** 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. 12]. 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.

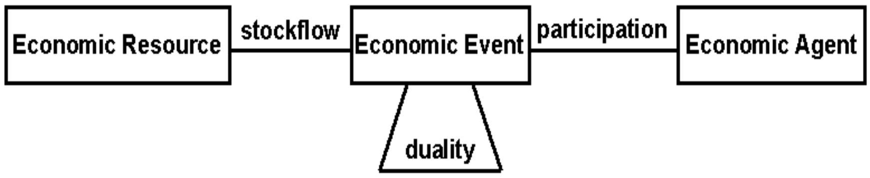


Fig. 12 The minimal REA model **[60]**

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).

With respect to 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 **[27]** 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.

UN/CEFACT modeling methodology [85] proposes set of metamodels that can be associated for modeling and designing of business systems in e-Commerce domain. Metamodels that deals with business requirement capture phases founded on the REA ontology.

3.2 e-Business Model

This section 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.

There are well-established theories from business, economics and related disciplines that such a business model can be founded on. However our main goal of this thesis is not to provide a complete economic analysis of such business models, instead to provide a methodology that can facilitate business process modeling tasks while leaving the model in its simplest possible form. Anyhow to provide the inspirations and the potential of business models in e-Commerce systems design, couple of work in this direction of briefed below.

3.2.1 Introduction to e-Business Model

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 completed.

The proposed BP^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 their process modeling tasks.

3.2.2 The Value Concept

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

A significant work can be found in Porter's competitive advantage series **[68]**. 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 primary 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 **[63]** depicted in the diagram below [Fig. 13]. It is possible to achieve competitive advantage in terms of successful marketing when one's offer is targeted the goal system of consumers (customer orientation) and is held by consumers to be better than competing offers.

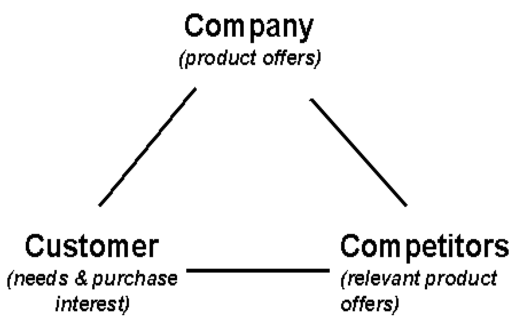


Fig. 13 Marketing Triangle **[63]**

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 **[41]**. 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 [41]. 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.

3.2.3 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. For the proposed BP[3] methodology in this thesis we keep associated business models simple, as taking all economic aspects into consideration at this stage is not the main concern. However the basis of our business model can be treated as concept of value discussed briefly above [**41**], [**68**] and REA ontology that has been adapted in UN/CEFACT modeling methodology [85].

A selected set of concepts has been used in defining business models that are to be developed during initial stages. It consists of Business Partners involved, Economic Resources they are to exchange, Economic Events they are going to perform in order to transfer Economic Resources, and Dualities that bundles reciprocal Economic Events. In the section below we will pictorially show what a typical business model looks like. However, we will first introduce a simple business scenario to which the business model is to build. Also we will use this business scenario as a running case to introduce different results from different stages of the proposed methodology.

3.2.4 Business Scenario for the Running Case

In order to explain and illustrate different stages and different outcomes from those stages in the proposed methodology, here we describe the running case. It is an extended version of Haugen's Drop-Dead Order business case described in [40].

3.2.4.1 Fried Chicken Business Scenario

In this business scenario, there are four trading partners: Customer, Distributor, Supplier and Carrier. First the Customer proposes the Distributor delivery of a specific number of fried chickens. The Customer orders products *only if* they can be delivered by a **hard deadline** – the Drop-Dead Date. The Distributor does not have products on hand; she needs to order products from a Supplier, plus arrange shipping service with a Carrier (ship from Supplier to end Customer).

When Customer decides on a specific variety of his chicken order, Distributor proceeds as follow by getting commitments from Supplier and Carrier.

```
If Distributor is able to get commitments from
both product Supplier and Carrier,

Then

Distributor accepts Customer order and also
confirms commitments from Supplier and Carrier.

Otherwise,

Distributor declines Customer order, and also
cancels commitments from Supplier and Carrier.
```

To put it another way, the Distributor wants both the chickens and the delivery service, but if not both, then neither. The Distributor does not want to get stuck paying for chickens that can’t be delivered, or an empty truck.

Distributor first asks for a down payment from Customer. Distributor will invoice for final payment as Carrier drops the order at Customers dock. Also Supplier first requires a down payment from Distributor in order to complete his chicken supply. Distributor pays Supplier the down payment from received Customer down payment.

Supplier has to complete his chicken supply before Carrier to start delivering them to Customer. As Supplier fulfills his commitments for Chicken supply, he invoices for the final payment Distributor due.

As Carrier drops the ordered fried chicken at Customer’s dock he invoices Distributor for his shipping service.

After receiving Customer’s final payment, Distributor settles both Supplier and Carrier payments due.

3.2.5 Business Model for Fried Check Scenario

The business model for the fired check scenario described above can be diagrammatically represented as in [Fig. 14].

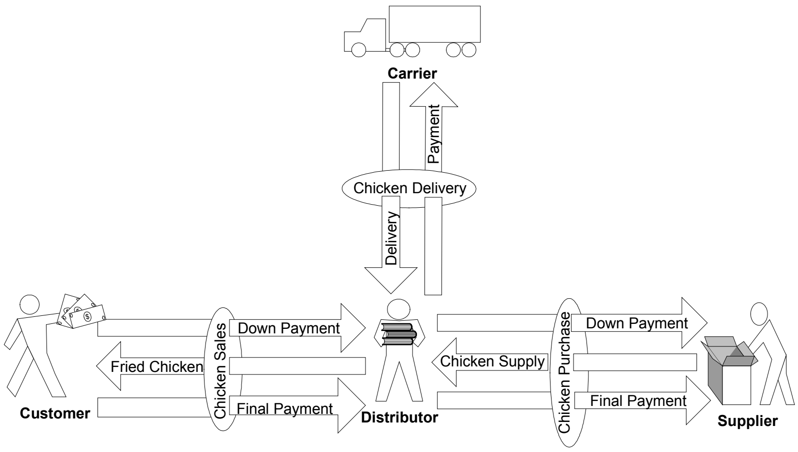


Fig. 14 Business Model for Fried Check Scenario

In [Fig. 14] partners involved in this trading situation are shown as stick human figures for Customer, Distributor and Supplier while figure of a vehicle has been used for Carrier. The thick arrows are named with relevant economic resources and their direction indicates the flow of economic resource from one partner to another. In other words, thick arrows depict economic events with directions. Ellipses that bundles two or more economic events represent different *dualities*. For the case descried about there are three such dualities names as Chicken Sales, Chicken Purchase, and Chicken Delivery. For an instance duality, Chicken Sales consists of three economic events: Customer Down Payment to Distributor, Distributor Fired Chicken Supply to Customer and Customer Final Payment to Distributor.

4 Foundational Analysis of the Framework

This chapter presents the theoretical foundation of the framework. First, adaptation of LAP within the framework and secondly, associated modeling ontology is explained. Finally, an analysis of UMM business and process models and ebXML Business Process Specification Schema by means of proposed approach has been discussed. By this analysis we illustrated how different modeling concepts associated at different stages in e-Commerce systems development can be understood and then to be integrated to the final solution according to the proposed framework.

4.1 Adaptation of LAP in the Framework

The main goal of the proposed framework is to provide basis for precise and clearer understanding of modeling concepts associated in e-Commerce systems design.

For the proposed framework, it is also possible to construct a hierarchical architecture that represents concepts governing underlying concepts. The diagram, [Fig. 15] below shows the BP^3 layers. The objective of this six-layered architecture is to understand context and serve the design of the business model first and then to move to business process model that deals with all associated communication in doing business.

Fig. 15 Hierarchical Architecture of BP^3 Framework

We have followed the UN/CEFACT meta-business models **[85]** and proposals of the e^3 framework **[34]** (detailed discussion about these approaches can be found in Chapter 3) in designing our business model. Though the some ideas borrowed from those two approaches, they are not based on LAP concepts in development of business and process models in e-Commerce domain.

As our framework has a 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. In proceeding sections, UN/CEFACT's metamodels [85] and ebXML Business Process Specification [22] that have been recommended for e-Business design has been critically analyzed together with some promising extension.

4.2 Extended BP^3-UN/CEFACT Modeling ontology

Techniques and Methodologies Working Group (TMWG) of United Nations Center 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 has proposed for their metamodels **[85]**. 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.

4.2.1 Partner

A partner is an independent economic and/or legal entity, e.g., John Doe, Stockholm University. The economic agent or simply agent in original REA ontology has been called as business partner or simply partner in UN/CEFACT's related work. UN/CEFACT uses the REA ontology as its basis.

4.2.2 Partner Type

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

4.2.3 Economic Resource

An economic resource is a quantity of something of value (see Chapter 3 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.

4.2.4 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.

4.2.5 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. [55], i.e. it can be communicative and/or instrumental act that makes a state change of one or more entities in a business world. 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.

4.2.6 Economic Event

An economic event is a business event that specifically transfers control of an economic resource from one partner type to another type. A completed economic event fulfills one or more commitments in the contract. A cash payment, shipment, and sale are examples of economic events.

4.2.7 Duality

The duality is a relationship between two or more economic events, where one is the legal or economic consideration of another [85]. This corresponds to the value offering proposed by Gordijn **[33]** based on the "one good turn deserves another" principle. If EE1 and EE2 are two economic events such that EE1 is the economic event transferring an economic resource from partner P1 to partner P2 and EE2 is the corresponding economic event transferring an economic resource from P2 to P1, then duality represents the reciprocity between EE1 and EE2. That is, one partner is providing another with some thing of value and receiving some thing of value in return.

4.2.8 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.

4.2.9 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.

4.2.10 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.

4.2.11 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.

4.2.12 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.

4.2.13 Act

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

4.2.14 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.

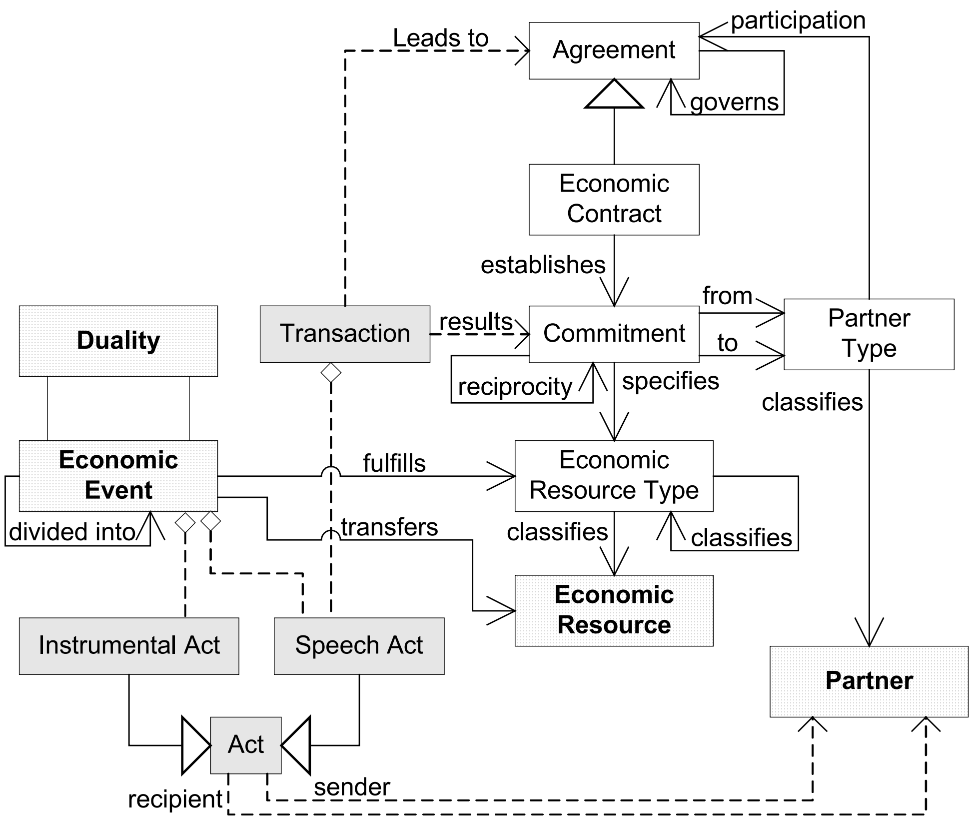


Fig. 16 An extended UMM economic model

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. 16] above.

The rectangles in the diagram represent concepts and also note that we have used dotted shaded rectangles that are found in REA and considered as the most central concepts in the BP^3 framework. The rectangles shaded with gray 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) **[84]**, **[27]**. 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. **[95]** used to distinguish between different levels of communication.

Fig. 16 is good as the starting point to see how LAP concepts can be roughly introduced into UN/CEFACT's business requirement view. However in order to motivate these extensions we have to provide a clear line of reasoning and introduction of coarse grained artifacts at the initial stages of an e-Commerce system design. At initial stages, business domain experts are not interested in low-level technical details that deal with necessary business communications to achieve business objectives. In proceeding sections we build our framework in this direction.

4.3 Three Worlds

As we mentioned in the Introduction (Chapter 1), a starting point for understanding the relationships between business models and process models is the observation that a person can carry out several different actions by performing one single physical act. An everyday example could be a person who turns on the water sprinkler and thereby both waters the lawn and fulfils the promise to take care of the garden – one physical act (turning on the sprinkler), which can be viewed as "carrying" two other actions (watering the lawn and fulfilling a promise). Relationships like these are particularly common for communicative actions, which are carried out by means of physical actions. One way to look at the role of communicative actions and their relationships to other actions is to view human actions as taking place in three different domains:

- ***The physical domain.*** In this domain, people carry out physical actions – they utter sounds, wave their hands, send electronic messages, etc.

- ***The communicative domain***. In this domain, people express their intentions and feelings. They tell other people what they know, and they try to influence the behavior of other actors by communicating with them. People perform such communicative actions by performing actions in the physical domain.

- ***The social/institutional domain.*** In this domain, people change the social and institutional relationships among them. For example, people become married or they acquire possession of property. People change social and institutional relationships by performing actions in the communicative domain.

Using this division, business models can be seen as describing the social/institutional domain, in particular economic relationships and actions like ownership and resource transfers. Process models, on the other hand, describe the communicative domain, in particular how people establish and fulfill obligations.

The three-fold division above is based on an agent-oriented approach to information systems design, [90], [98]. A key assumption of this approach is that an enterprise can be viewed as a set of co-operating agents that establish, modify, cancel and fulfill commitments and contracts [20]. In carrying out these activities, agents rely on so called speech acts, which are actions that change the universe of discourse when a speaker utters them and a recipient grasps them. A speech act may be oral as well as written, or even expressed via some other communication form such as sign language.

The feasibility of speech act theory for electronic communication systems is supported by several researchers, see [72] for a review. The work reported on in this thesis differs from these approaches since it uses SAT for analyzing and integrating different modeling domains in e-Commerce, rather than facilitating electronic message handling per se.

4.4 Pragmatic Actions

The basic notion introduced for relating Business Actions to economic concepts is that of a pragmatic action, see Fig. 17. A Pragmatic Action is a speech act, as defined in Chapter 2, and consists of two parts: a content and an illocutionary force. In e-Commerce applications, the content is always an economic concept. The illocutionary force of a pragmatic action indicates in what way the action is related to its content. An agent can perform a pragmatic action and thereby influence an economic concept in a specific way.

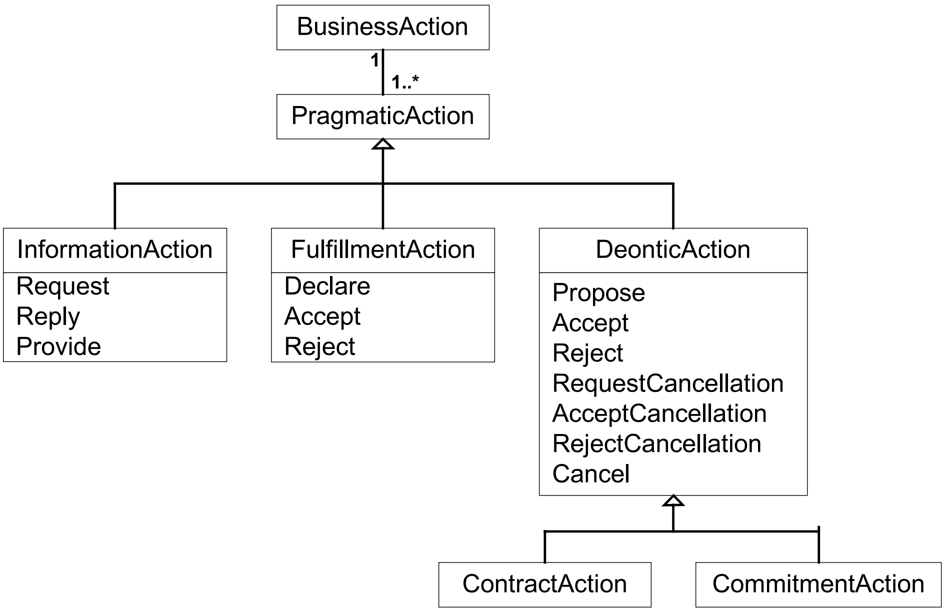


Fig. 17 Business Actions and Pragmatic Actions

Depending on which economic concept a pragmatic action addresses, different illocutionary forces are applicable. The pragmatic actions are, therefore, divided into several subclasses as indicated in Fig. 17. The three main subclasses are information actions, deontic actions, and fulfillment actions. The underling intuition for identifying these three sub classes of pragmatic actions is that in an e-Commerce scenario, trading partners exchange business information, then establish different obligations, and finally exchange Economic Resources, thereby fulfilling the obligations.

An Information Action can have any economic concept as its content and requests or provides information about the concept. There are three possible illocutionary forces for information actions: Request asks for information, Reply answers a preceding request, and Provide provides information without a preceding request. E.g. "query for price and availability".

A Fulfillment Action has an Economic Event as its content. The action may declare that an Economic Event has been performed, or it may express that such a declaration is accepted or rejected. There are three possible illocutionary forces for fulfillment actions: Declare states that an Economic Event has been performed, Accept states that a preceding declaration of performing an economic event is accepted, Reject states that such a declaration is rejected. E.g. “declare shipment completed”.

A Deontic Action can have a commitment or contract as its content. Thus, a deontic action concerns obligations to carry out events in the future. There are seven possible illocutionary forces for deontic actions: Propose means that an agent proposes the establishment of a commitment or contract. Accept is the acceptance of such a proposal while Reject is the rejection of a preceding proposal, Request-Cancellation is a request to cancel an established commitment or contract, AcceptCancellation is the acceptance of such a request, while RejectCancellation is the rejection of a preceding request to cancel, Cancel is a unilateral cancellation. E.g. “request purchase order”.

4.5 UMM Business Requirement View

The Business Requirement View of UN/CEFACT Modeling Methodology can be treated as the basis for Business Model that captures concepts in the Social/Institutional domain as described above in Section 4.3. The REA framework is used as the theoretical foundation of the Business Requirements View of UN/CEFACT Modeling Methodology (UMM) [85]. UMM is based on the Unified Modeling Language (UML) [64], and it provides a procedure for modeling business processes in a technology-neutral, implementation-independent manner. In UMM, a number of different view meta-models are defined to support an incremental model development and to provide different levels of specification granularity. Among these is the Business Requirement View (BRV), see [Fig. 18], capturing the business transactions with their interrelationships, which makes it the most relevant meta-model for our work. Below, we briefly re-introduce some of BRV concepts in order to get refreshed and some concepts of complete original BRV that we have suppressed in [Fig. 16] for simplicity.

Like REA, BRV models Economic Events, the Economic Resources transferred through the Economic Events, and the Economic Agents, here called Partner Types between whom the Economic Events are performed. Furthermore, an Economic Event fulfills an Economic Commitment. An Economic Commitment can be seen as the result of a commissive speech act and is intended to model an obligation for the performance of an Economic Event. The *duality* between Economic Events is inherited into the Economic Commitments, represented by the relationship *reciprocal*.

In order to represent collections of related commitments, the concept of Economic Contracts is used. An Economic Contract is an aggregation of two or more reciprocal commitments. An example of an Economic Contract is a purchase order with several order lines, which are the Economic Commitments involved in the purchase order contract. The products specified in each line are the Economic Resource Types that are the subject for the Economic Commitments.

Moving one level up, the Economic Contracts are often made within the boundaries of different Agreements. An Agreement is an arrangement between two Partner Types that specifies the conditions under which they will trade. Furthermore, a Business Collaboration choreographs the Business Collaboration Task performed in a contract formation and contract fulfillment with number of requesting and responding business interactions.

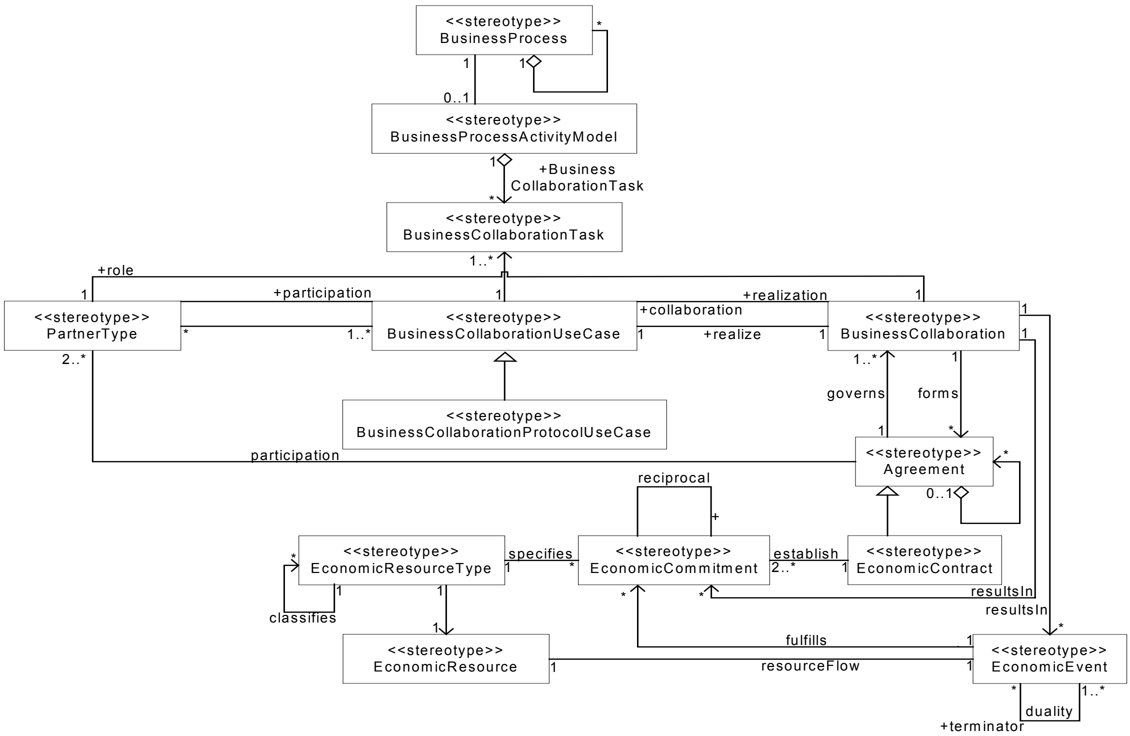


Fig. 18 UMM Business Requirements View (BRV) [85]

4.6 ebXML Business Process Specification Schema

There are well-established modeling techniques for e-Commerce development. However, there is still a considerable gap to be covered between e-Commerce process modeling and specification of software components. One major framework intended to bridge this gap is the ebXML Business Process Specification Schema (BPSS), [22]. In BPSS, a collaboration consists of a set of Business Transactions choreographed to obtain a control flow of the transactions. Each transaction consists of a pair of request and response activities carried out by two Business Partners in different roles as well as document flows between the activities, see Fig. 19. A collaboration may be between two parties, a Binary Collaboration, or between several parties, a Multi Party Collaboration. A UML class diagram of BPSS is shown in Fig. 20.

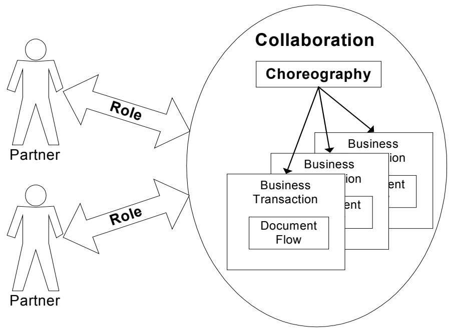


Fig. 19 Collaborations in ebXML BPSS [22]

A Business Transaction is an atomic unit of work; it consists of one Requesting Business Activity, one Responding Business Activity, and one or two document flows between them. There is always a Request document flow, while there does not have to be a corresponding Response document flow. A pair of Request and Response document flows is needed in cases where some kind of agreement is to be established. Some transactions, however, have the function of notifications, and in such cases only a Request document flow is needed. There is a common super class, Business Action, for Requesting Business Activity and Responding Business Activity, which holds common attributes specifying conditions on intelligibility checks, authorisation, time to acknowledge, and non-repudiation. An example of a Requesting Business Activity is “Request Purchase Order”, an example of a Responding Business Activity is “Accept Purchase Order” – together these two Business Actions constitute a Business Transaction.

Business Transactions are the basic building blocks of Binary Collaborations. A Binary Collaboration is always between two roles, and it consists of one or more Business Activities. These Business Activities are always conducted between the two roles that business partners can play in the Binary Collaboration. One of these roles is assigned to be the *initiatingRole* (from) and the other to be the *respondingRole* (to). An example of a Business Collaboration is "Manage Purchase" which could involve several Business Activities for querying about products or establishing the individual purchase order lines.

There are two kinds of Business Activities: Business Transaction Activities and Collaboration Activities. A Business Transaction Activity is the performance of a Business Transaction within the context of a Binary Collaboration. Thus, the same Business Transaction can be performed by multiple Business Transaction Activities within different Binary Collaborations or even within the same Binary Collaboration.

A Collaboration Activity is the performance of a Binary Collaboration. Analogous to Business Transaction Activities, a Binary Collaboration can be performed by multiple Collaboration Activities within different Binary Collaborations or even within the same Binary Collaboration.

A Binary Collaboration is not just an unordered set of Business Transaction Activities and Collaboration Activities. The Business Activities need to be ordered, which is done by means of a choreography. A choreography is specified in terms of Business States and Transitions between these states. The most important kind of a Business State is a Business Activity. Furthermore, there are a number of auxiliary Business States corresponding to diagramming artifacts on a UML activity chart: Start, Completion State, Fork, and Join.

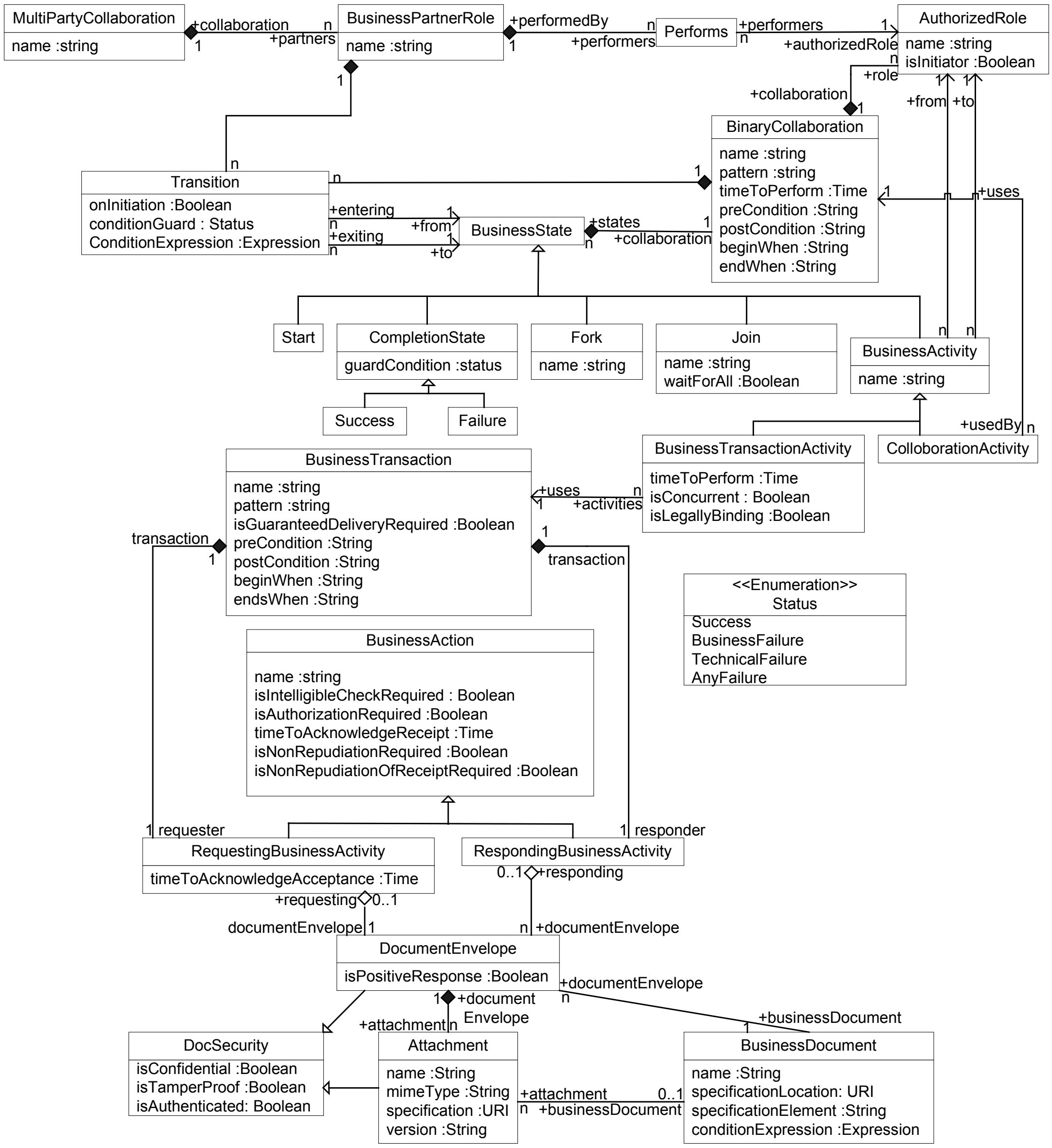


Fig. 20 ebXML Business Process Specification Schema, [22]

4.7 How Pragmatic Actions Bridge BRV and BPSS

In the UMM Business Requirements View, there is only a very general relationship between economic concepts (i.e. Economic Event, Economic Contract, Economic Commitment, Economic Resource, Economic Resource Type, Agreement, and Partner Type) and process concepts. Essentially, the relationship states that a commitment or agreement is created by means of a collaboration, but there is no indication of how the constituents of the collaboration are related to the economic concepts. In order to get a more fine-grained view of the relationships between collaborations and economic concepts, we need to specify how the individual Business Actions involved in a collaboration are related to the economic concepts.

In the analysis of the relationships between the UMM/BRV and the BPSS process model (Fig. 18 and Fig. 20) parts of the corresponding concepts in the respective models are modeled on different levels of abstraction. Two common abstraction levels defined in [24] and [27], are the operational level and the knowledge level. The *operational level* models concrete, tangible individuals in a domain. The *knowledge level* models information structures that characterize categories of individuals at the operational level. Martin and Odell, [59], employ the concept of *power types* to refer to the correspondence between the objects of the knowledge and operational levels. A power type is a class whose instances are subtypes of another class. The Economic Resource Type of Fig. 21 is a power type of Economic Resource. Instances of the Economic Resource Type are the different categories of Economic Resources, for instance "real estate". An instance of the Economic Resource class is a particular piece of land, e.g. "Hyde Park Mansions".

In BPSS, classes like Business Activity and Business Transaction are defined at the knowledge level only. Business Activities do not possess properties related to actually transferred recourses, nor are they associated with the agents or roles between whom the transfer occurs. This is, however, not the case with the economic concepts of UMM/BRV. An Economic Event of UMM/BRV is explicitly related to an actual Economic Resource on the operational level.

An Economic Resource refers to an actual and tangible resource, whereas an Economic Resource Type is the corresponding power type defined on the knowledge level, serving as a template for concrete Economic Resources. Pursuing this line of analysis it is possible to identify templates on several levels, each of which is the power type of the other.

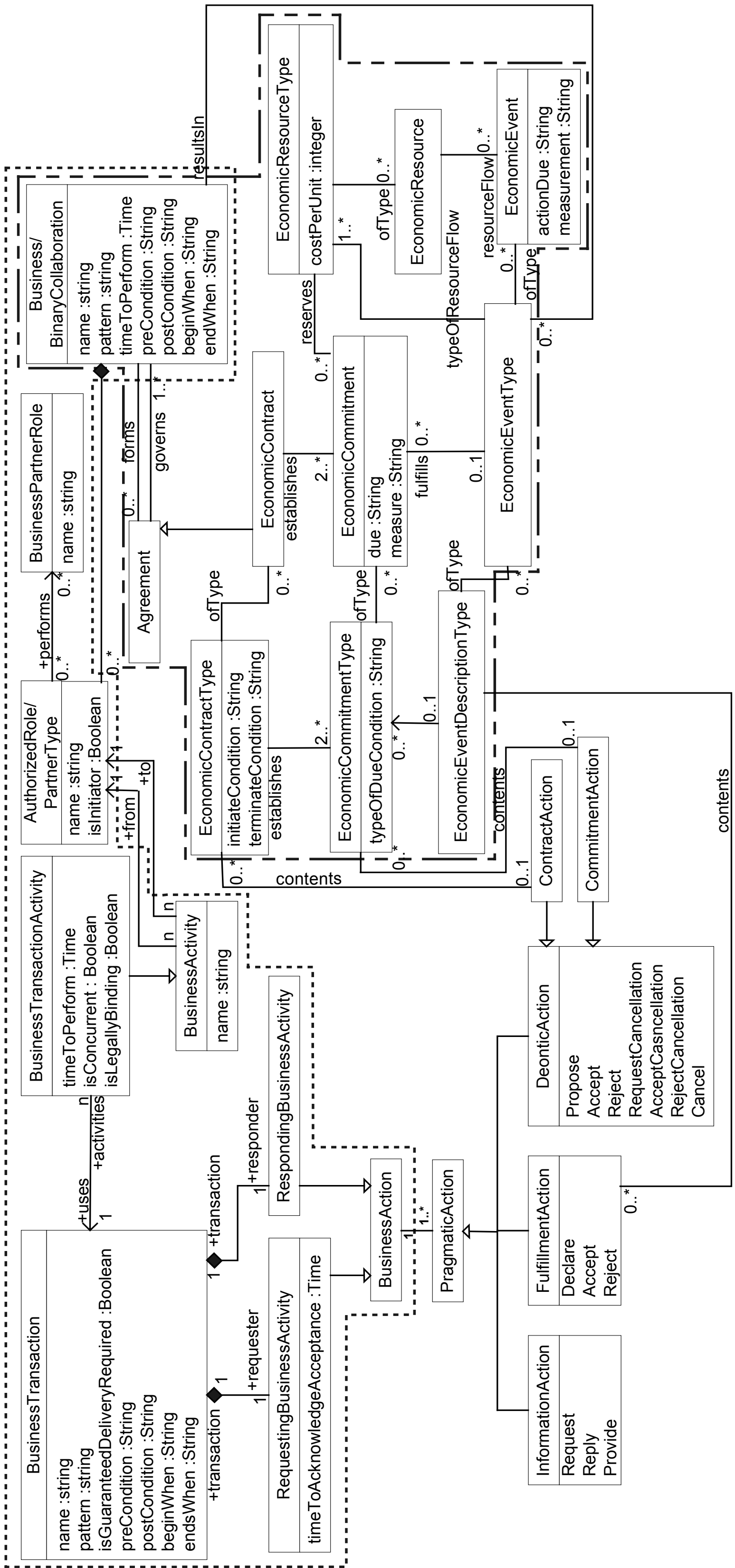


Fig. 21 Integrated view of business and process model

To facilitate the integration with BPSS, several economic concepts need to be added to UMM/BRV to include classes defined on the knowledge level. In the case of an Economic Contract, an Economic Contract Type is introduced to distinguish between the description of a contract and the actual contract between parties or abstract roles to be played by parties. The Economic Contract Type class models properties such as the types of conditions that may initiate or terminate a future contract, whereas an Economic Contract is associated to the authorized roles or partner types between whom a contract is established.

The introduction of new knowledge level classes into the economic concepts of UMM/BRV can be seen as schema conforming [76], i.e. transforming the schemas to be integrated in order to increase their similarity. The individual constituents of the BPSS model become possible to relate to the economic concepts of UMM/BRV as the two views now contain corresponding concepts defined on the same level of abstraction. The global, integrated view of UMM/BRV and BPSS is shown graphically in Fig. 21. The glue in this integrated view is the pragmatic actions defined in Section 4.4. Each individual Business Action of BPSS carries one or more Pragmatic Actions, which serve as categorizations of the Business Action. The categorizations are, in turn, defined in terms of the economic concepts of UMM/BRV. In Fig. 21, the original BPSS-parts are grouped with a dotted line boundary, UMM/BRV-parts are grouped with a dashed line boundary and the pragmatic actions that relate the two are depicted without any line boundary.

4.8 UMM Business Transaction View

The Business Transaction View (BTV) [Fig. 22] proposed by UN/CEFACT Modeling Methodology [85] specifies the flow of business information between business roles as they perform business activities. A

Business Transaction is a unit of work through which information and signals are exchanged (in agreed format, sequence and time interval) between two business partners. These information exchange chunks, called Business Actions, are either Requesting Business Activities or Responding Business Activities (depending on whether they are performed by a Partner Role who is requesting a business service or whether they are the response to such a request). A transaction completes when all the interactions within it succeed, otherwise it is rolled back. Furthermore, the flow between different Business Transactions can be choreographed through Business Collaboration Protocols. Business Collaboration Protocols should be used in cases where transaction rollback is inappropriate. For example, a buying partner requests a purchase order by a selling partner and the selling partner accepts the order but he does it only partially. Accepting the order completes the transaction (i.e., the transaction can not longer be rolled back). However, the behavior following after the partial acceptance, i.e. the delivery of the accepted parts, differs from the behavior of accepting an order in its whole which would imply the delivery of all products specified in the order.

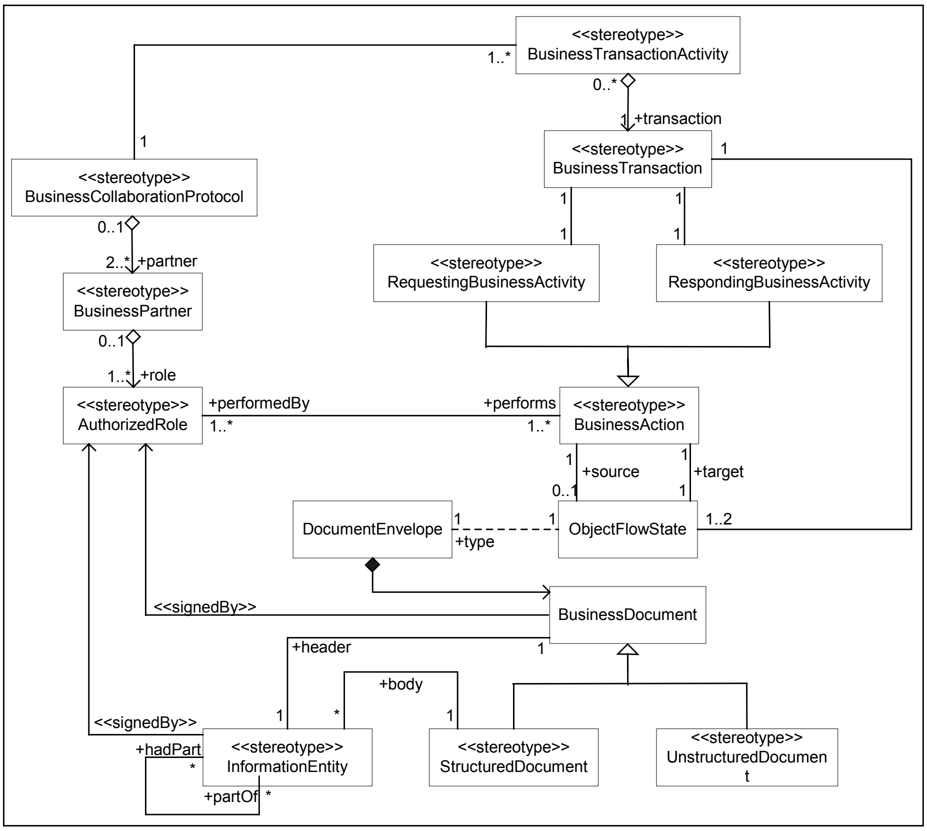


Fig. 22 UMM Business Transaction Views [85]

4.9 Economic Effect and Economic Effect Enactment

In terms of the three domains introduced in Section 4.3, UMM explicitly addresses only the physical and the social/institutional domains. The physical domain is modeled through classes like Business Transaction and Business Action, while the social/institutional domain is modeled through Economic Commitment, Economic Event, and other classes. The details of the communicative domain, however, are not explicitly modeled. This state of affairs causes two main problems. First, the relationship between the physical and the social/institutional domains is very coarsely modeled; essentially the UMM only states that a completed collaboration may influence objects in the social/institutional world, but it does not tell how the components of a collaboration affect the social/institutional objects. Secondly, there is no structured or systematic way of specifying how events in the physical domain influence the social/institutional domain. These problems can be overcome by introducing the communicative domain as an additional layer in the UMM, thereby creating a bridge between the physical and social/institutional domains.

As a preparation to modeling the communicative domain, a minor modification to UMM BRV is made, see [Fig. 23]. A class Economic Effect is introduced as a super class of Economic Commitment, Agreement, and Economic Event.

The power type [59] of Economic Effect, called Economic Effect Type, is also added for the purpose of differentiating between the modeling of concrete, tangible objects in a domain, and the abstract characteristic categories of these objects.

These modifications will allow for a more concise representation of the effects (as well as the characteristics of the effects) of communicative actions. In addition to these changes, the classes Business Action Enactment and Business Transaction Enactment are added. These represent the actual execution of a business action or business transaction, respectively.

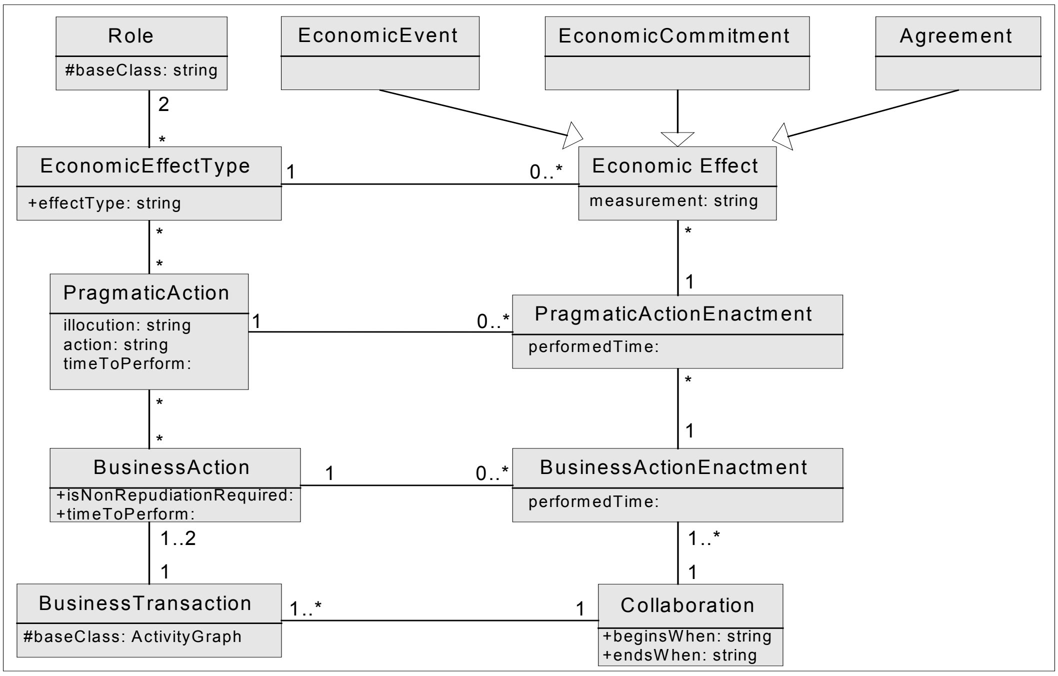


Fig. 23 Extended Business Requirement View

The basic notions introduced for modeling the communicative domain are those of a pragmatic action and its execution, i.e. Pragmatic Action and Pragmatic Action Enactment, see [Fig. 23]. A *pragmatic action* is a speech act as introduced in Section 4.4. It consists of three parts, denoted as a triple:

<Illocution, Action, EffectType>

Intuitively, these components of a pragmatic action mean the following:

- ***Effect Type*** specifies an Economic Effect Type, i.e. it tells what kind of object the pragmatic action may affect
- ***Action*** is the type of action to be applied – create, change, or cancel
- ***Illocution*** specifies the illocutionary force of the pragmatic action, i.e. it tells what intention the actor has to the Action on the Effect Type

Formally, Illocution and Action are defined through enumeration:

Action = {create, change, cancel, none}

Illocution = {propose, accept, reject, declare, query, reply, assert}

The meanings of the illocutions are as follows:

Propose	–	someone proposes to create, change, or cancel an object
Accept	–	someone accepts a previous proposal
Reject	–	someone rejects a previous proposal
Declare	–	someone unilaterally creates, changes, or cancels an object
Query	–	someone asks for information
Reply	–	someone replies to a previous query
Assert	–	someone makes a statement about one or several objects

For ‘query’, ‘reply’, and ‘assert’, there is no relevant Action involved, so only the “dummy” ‘none’ can be used.

The class Pragmatic Action Enactment is used to represent the actual executions of pragmatic actions. A Pragmatic Action Enactment specifies a Pragmatic Action as well as an Economic Effect, i.e. the agreement, commitment, or economic event to be affected. Some examples of Pragmatic Actions are:

“Query status of a sales order” would be modeled as <query, none, salesOrder>

“Request purchase order” would be modeled as <propose, create, purchaseOrder>,

where ‘salesOrder’ and ‘purchaseOrder’ are Economic Effect Types

4.10 Economic Effect that glues BRV and BTV

The glue between the physical domain and the communicative domain is made up by the associations between the classes Business Action and Pragmatic Action, and Business Action Enactment and Pragmatic Action Enactment. These associations express that a business action can carry one or more pragmatic actions, i.e. by performing a business action, an actor simultaneously performs one or several pragmatic actions. Often, only one pragmatic action is performed, but in some cases several can be performed, e.g. when creating a commitment and its contract at the same time.

The global integrated view of BRV and BTV is shown graphically in Fig. 24. The original BTV-parts are grouped within a checked area boundary, BRV-parts are grouped within a dashed area and the new parts introduced in this chapter are depicted in the white area.

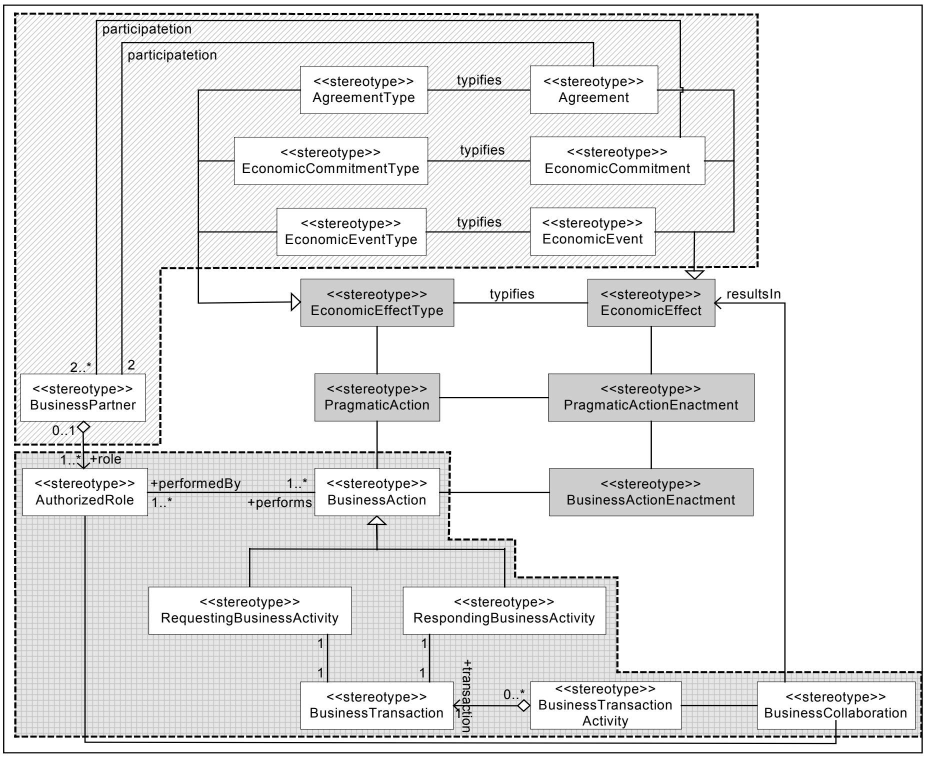


Fig. 24 Integrated Global view

4.11 Conclusion

In this chapter we have analyzed, interpret and integrated UMM Business Requirement Views (BRV), Business Transaction Views (BTV) and ebXML Business Process Specification Schema (BPSS). As the instrument for this analysis a LAP extension, Pragmatic Actions is defined.

Our understanding and experience with this work is this framework provide better understanding of modeling concepts associated in different stages of e-Commerce system development process and integration mechanism of business and process model in reaching final systems' designs.

The Pragmatics Actions have been defined in this chapter as triplet of *Illocution*, *Action* and economic *EffectType*. While listing tuples for illocution and action through enumeration, economic effect type has been introduced as an extension to UMM meta-models for business modeling.

However, in order to test the exhaustiveness of introduced pragmatic actions, some work is needed in future including several empirical case studies.

5 BPMN and Generic Process Patterns

This chapter starts by introducing Business Process Modeling Notation (BPMN) that has been adapted as conceptual and notational framework for targeted solutions. Then set of generic process patterns have defined by means of BPMN business process diagrams (BPDs). These pre-defined patterns are the basis for process generations discussed in the Chapter 7.

5.1 Business Process Modeling Notation (BPMN)

The notation we use for business process models in this thesis is BPMN [9]. BPMN is an open standard developed by Business Process Management Initiative (BPMI) [8]. The goal of BPMN is to provide a rich set of readily comprehensible notation for a wide spectrum of stakeholders from business domain experts to technical developers. The Ambition of the BPMN project is to bridge the gap between business process design and their technical realizations.

As XML languages designed for the execution of business processes has captured market for e-Business solutions, there was a need for higher-level visual symbolic languages to specify business processes such that they can be easily communicated even with non-technical domain experts. BPMN can be treated as bottom-up development to visualize those XML oriented low-level process specifications such as BPEL4WS (Business Process Execution Language for Web Services) [7]. BPEL4WS is based on both graph and block structures, and also has borrowed principles from formal mathematical models, such as π–calculus [66].

Therefore by staring business process modeling with BPMN there is a greater advantage of being readily able to map into a XML language designed to execute business processes like BPEL4WS. Besides openness of BPMN this is the other reason for selecting it as our notation for business process models to be discussed in this thesis. However, here we are not going to discuss the full potential of BPMN, instead a selected set of core elements from BPMN is introduced for the purpose of our work presented in this thesis. Complete specification of BPMN can be found at [9] under "BPMN Specification Releases:"

5.1.1 Three Basic Types of sub-models

Models result in business process modeling tasks in e-Commerce domain have to represent wide variety of information and they should possess the ability to communicate them to wide variety of audience comprehensively. In order to serve this purpose, BPMN proposes usage of three basic types of sub-models.

1. Private (Internal) Business Processes
2. Abstract (Public) Processes
3. Collaboration (Global) Processes

5.1.1.1 Private (Internal) Business Processes

Business processes that are internal to a specific organization are categorized into private business processes in BPMN.

5.1.1.2 Abstract (Public) Processes

Processes that model interaction between private processes and another processes or participant are called abstract processes in BPMN. These processes consist only of activities that are used to communicate outside the private business processes.

5.1.1.3 Collaboration (Global) Processes

The collaboration processes are to model interactions between two or more distinct business entities. These collaboration process models can be mapped in to various collaboration languages such as ebXML BPSS [22], RosettaNet[70], or proposed specifications from W3C Choreography Working Group[92]. However, basic type of sub-model within an end-to-end BPMN that we are interested in and relevant for our work is collaboration processes. Therefore all BPMN models that we are to introduce in proceeding sections belongs to this group.

5.1.2 Business Process Diagram Core Elements

The diagrams that visualize BPMN process specifications are called Business Process Diagrams (BPD). BPD can be constructed by combining different graphical objects according to rules that have been defined in [9] under "BPMN Specification Releases:". Again these graphical objects can be categorized into;

1. Primary modeling elements
2. Elements to connect primary modeling elements
3. Elements to group primary modeling elements

5.1.2.1 Primary Modeling Elements

There are three primary modeling elements in BPMN as,

1. Events
2. Activities
3. Gateways

5.1.2.2 Elements to Connect Primary Modeling Elements

There are three elements that can be used to connect primary modeling elements.

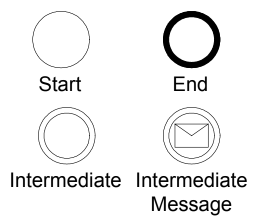
1. Sequence Flow
2. Message Flow
3. Association

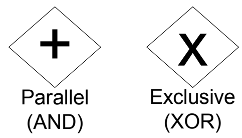
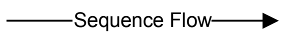
5.1.2.3 Elements to Group Primary Modeling Elements

There are two elements that can be used to group primary modeling elements.

1. Pools
2. Lanes

Below, the Table 2 borrowed from BPMN specification release [9] lists different notations that have been proposed to symbolize BPD core elements. This is not the complete set of core elements that can be found in original specification; instead it is a selected sub-set for our work.

Element	**Description**	**Notations**
Event	An event is something that "happens" during the course of a business process. These events affect the flow of the process and usually have a cause (trigger) or an impact (result). There are three types of Events, based on when they affect the flow: Start, Intermediate, and End.	

Activity	An activity is a generic term for work that company performs. An activity can be atomic or non-atomic (compound). The types of activities that are a part of a Process Model are: Process, Sub-Process, and Task. Tasks and Sub-Processes are rounded rectangles. Processes are either unbounded or a contained within a Pool.	
Gateway	A Gateway is used to control the divergence and convergence of Sequence Flow. Thus, it will determine branching, forking, merging, and joining of paths. Internal Markers will indicate the type of behavior control.	
Sequence Flow	A Sequence Flow is used to show the order that activities will be performed in a Process.	
Message Flow	A Message Flow is used to show the flow of messages between two participants that are prepared to send and receive them. In BPMN, two separate Pools in the Diagram will represent the two participants (e.g., business entities or roles).	

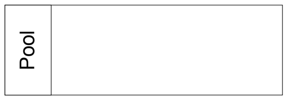
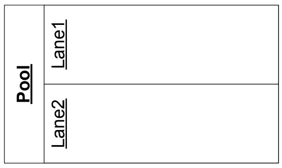
Pool	A Pool represents a Participant in a Process. It is also acts as a “swimlane” and a graphical container for partitioning a set of activities from other Pools, usually in the context of B2B situations.	
Lane	A Lane is a sub-partition within a Pool and will extend the entire length of the Pool, either vertically or horizontally. Lanes are used to organize and categorize activities.	
Collapsed Sub-Process	The details of the Sub-Process are not visible in the Diagram. A “plus” sign in the lower-center of the shape indicates that the activity is a Sub-Process and has a lower-level of detail.	
Transaction	A transaction is a Sub-Process that is supported by special protocol that insures that all parties involved have complete agreement that the activity should be completed or cancelled. The attributes of the activity will determine if the activity is a transaction. (Double-lined rectangle)	

Table 2 BPMN Business Process Diagrams Core Elements

5.1.3 Adaptation of BPMN in BP^3

An example of a BPMN diagram is shown in Fig. 25. The diagram shows a single Business Transaction in one pool with three lanes. We suggest to use the middle lane to place all events with global interest happening in the shared space and the two outermost lanes to place activities performed by two partners, respectively. A Business Transaction is a unit of work through which information and signals are exchanged (in agreed format, sequence and time interval) between two partners [85]. A Business Transaction consists of two Activities, one Requesting Activity where one partner initiates the Business Transaction and one Responding Activity where another partner responds to the Requesting Activity.

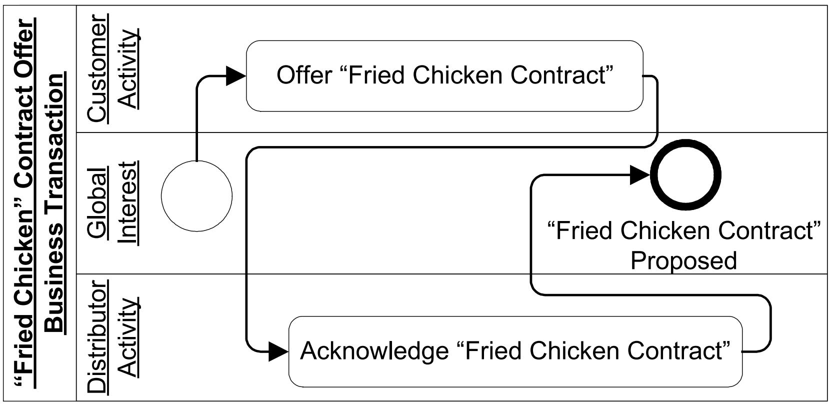


Fig. 25 Example of a BPMN diagram

Several Business Transactions between two partners can be combined into one binary Business Collaboration. It turns out that it is often fruitful to base binary Business Collaborations on Dualities, i.e. one Business Collaboration will contain all the Business Transactions related to one Duality. This gives a starting point for constructing a process model from a business model. Each Duality in the business model gives rise to one binary Business Collaboration, graphically depicted as a BPMN business process diagram contained in a Pool. In this way, a process model will be constructed as a set of interrelated Business Collaborations.

Furthermore, a binary Business Collaboration can naturally be divided into a number of phases. Dietz, [14], distinguishes between three phases. The Ordering phase, in which a partner requests some Resource from another partner who, in turn, promises to fulfill the request. The Execution phase, in which the partners perform Activities in order to fulfill their promises. The Result phase, in which a partner declares a transfer of Resource control to be finished, followed by the acceptance or rejection by the other partner. The ISO OPEN-EDI initiative [65] identifies five phases: Planning, Identification, Negotiation, Actualization and Post-Actualization. In this thesis, we use only two phases: a *Contract Negotiation phase* in which contracts are proposed and accepted, and an *Execution phase* in which transfers of Resources between partners occur and are acknowledged. In the next section, we will discuss how a binary Business Collaboration can be constructed utilizing patterns for these phases.

5.2 Generic Process Patterns

Designing and creating business and process models is a complicated and time consuming task, especially if one is to start from scratch for every new model. A good designer practice to overcome these difficulties is, therefore, to use already proven solutions. A *pattern* is a description of a problem, its solution, when to apply the solution, and when and how to apply the solution in new contexts [53]. The significance of a pattern in e-commerce is to serve as a predefined template that encodes business rules and business structure according to well established best practices. In this paper such patterns are expressed as BPMN diagrams. They differ from the workflow patterns of [87], [82], [88] by focusing primarily on communicative aspects, while control flow mechanisms are covered on a basic level only.

In the following subsections, a framework for analysing and creating transaction- and collaboration patterns are proposed. We hypothesize that most process models for e-commerce applications can be expressed as a combination of a small number of these patterns.

5.3 Modeling business transactions

When a transaction occurs, it typically gives rise to effects, i.e. Business Entities like Economic Events/Contracts/Commitments are effected (created, deleted, cancelled, fulfilled). Furthermore, the execution of a transaction may cause the desired effect to come into existence immediately, or only indirectly, depending on the intentions of the interacting Agents. For example, the intention of an Agent in a transaction may be to *propose* a Contract, to *request* a Contract or to *accept* a Contract. In all three cases the business entity is the same (a Contract) but the intention of the Agent differs.

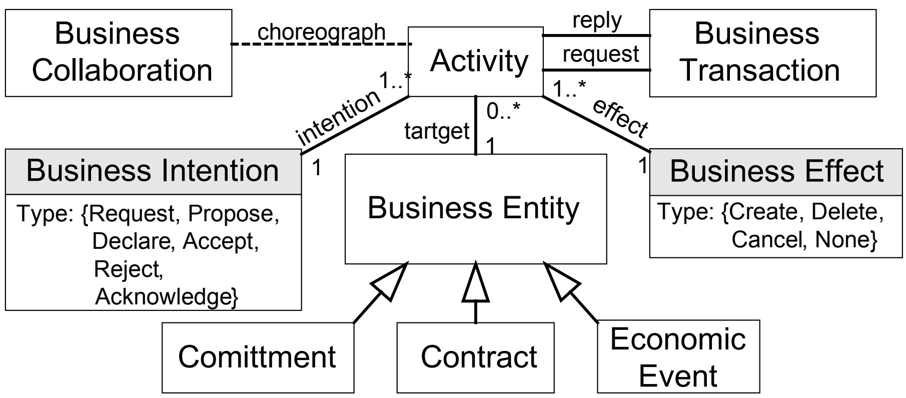


Fig. 26 Business Transaction analysis

Fig. 26 builds on REA and suggests a set of Business- Intentions, Effects and Entities. These notions are utilized in defining transaction patterns and transaction pattern instances as follows.

Definition: A *transaction pattern* (TP) is a BPMN diagram with two Activities, one Requesting Activity and one Responding Activity. Every Activity has a label of the form <Intention, Effect, Business Entity>, where Intention ∈ {Request, Propose, Declare, Accept, Reject, Acknowledge}, Effect ∈ {create, delete, cancel}, and Business Entity ∈ {aContract, anEconomicEvent, aCommitment}. All End Events are labeled according to the Intention and Business Entity of the Activity prior to the sequence flow leading to the End Event.

Intuitively, the components of an activity label mean the following:

- Business Entity tells what kind of object the Activity may effect.
- Effect tells what kind of action is to be applied to the Business Entity – create, delete or cancel.
- Intention specifies what intention the business partner has towards the Effect on the Business Entity.

The meanings of the intentions listed above are as follows:

- Propose – someone offers to create, delete or cancel a Business Entity.
- Request – someone requests other Agents to propose to create, delete or cancel a Business Entity.
- Declare – someone unilaterally declare a Business Entity created, deleted or cancelled.
- Accept/Reject – someone answers a previously given proposal.
- Acknowledge – an Agent acknowledges the reception of a message.

Definition: A *pattern instance* of a transaction pattern is a BPMN diagram derived from the pattern by renaming its Activities, replacing each occurrence of aContract in an activity label with the name of a specific Contract, replacing each occurrence of anEconomicEvent in an activity label with the name of a specific EconomicEvent, and replacing each occurrence of aCommitment in an activity label with the name of a specific Commitment.

5.4 BPMN Transaction Patterns (TPs)

In the following sections three basic Contract Negotiation and two Execution TPs are suggested based on the framework described above. Our observation is that these BPMN transaction patterns provide minimal set of recurring business communication patterns in contract negotiation phase and contract execution phase.

5.4.1 Contract Negotiation Transaction Patterns

In this category, we have defined primitive business communication (business transaction) patterns that can be composed together in order to form different complex interactions patterns to model binary collaborations in the contract negotiation phase. In order to cover minimal set of recurring business transactions patterns, we have defined three Contract Negotiation Transaction patterns.

5.4.1.1 Contract Offer Transaction Pattern

The Contract Offer transaction pattern models one partner proposing an offer (<propose, Create, aContract>) to another partner who acknowledges receiving the offer. See [Fig. 27]. When proposed contract has been accepted through Contract Accept/Reject transaction it will result in a residual obligations between involved two parties. This situation is explained in UMM as legally binding offers [85].

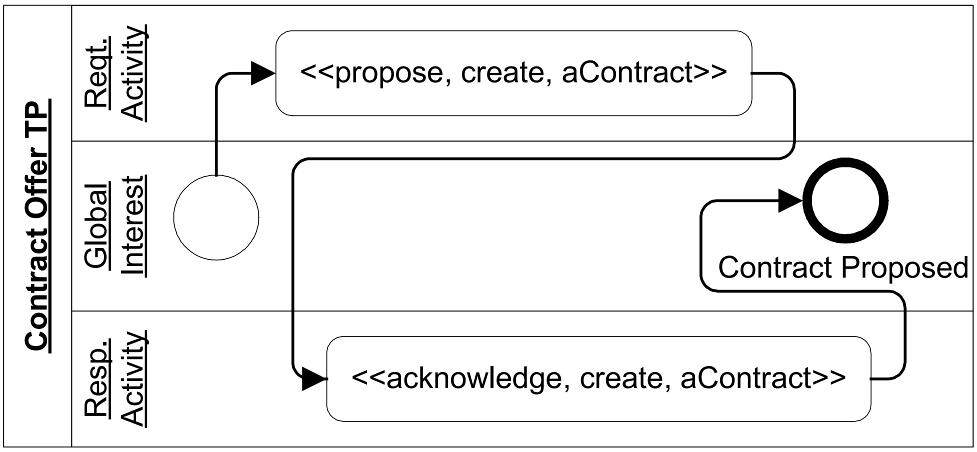


Fig. 27 Contract Offer Transaction Pattern

5.4.1.2 Contract-Accept/Reject Transaction Pattern

The acceptance or rejection of a Contract Offer is modeled in the Contract Accept/Reject transaction pattern. See [Fig. 28]. Here, a proposed contract from one partner is either accepted or rejected by another partner who in return receives acknowledgement from the proposed party. If accepted, as mentioned earlier, both parties end up with residual obligations to fulfill the terms of the contract.

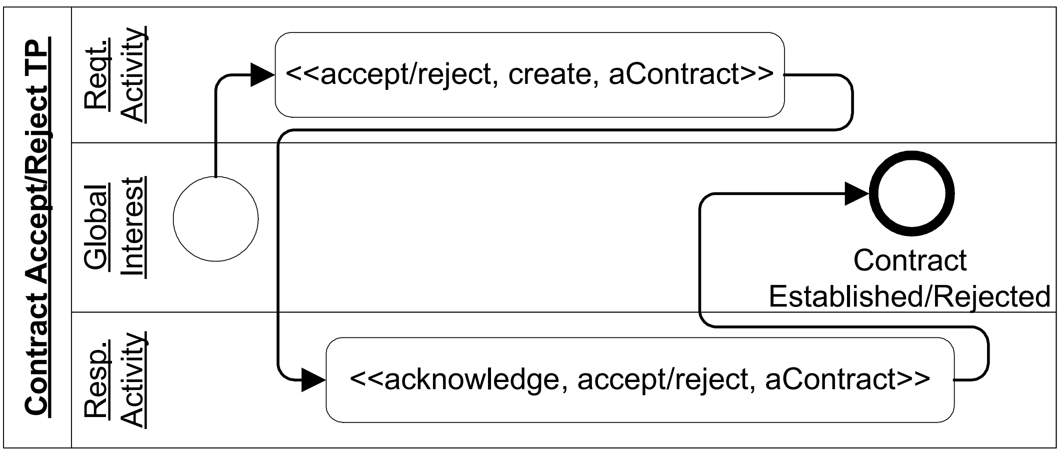


Fig. 28 Contract Accept/Reject Transaction Pattern

5.4.1.3 Contract-Request Transaction Pattern

Fig. 29 models the Contract Request case where one partner requests of other partner to make an offer for aContract on certain Economic Resources. This is the situation for requesting for a binding offer from a supplier in typical business.

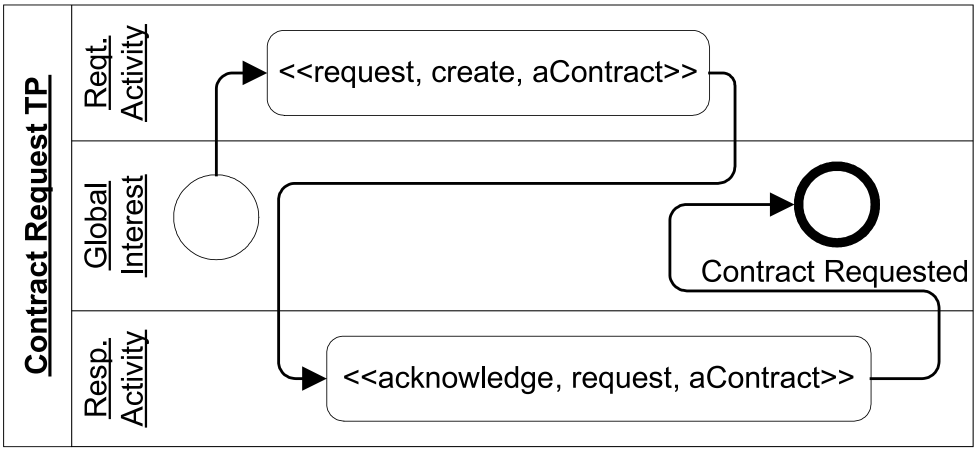


Fig. 29 TP for Contract Negotiation: Contract-Request

5.4.2 Contract Execution Transaction Patterns

In this category, we have defined primitive business communication (business transaction) patterns that can be composed together in order to form different complex interactions patterns to model binary collaborations in contract execution phase. This is the phase where different partners exchange contracted economic resources with each other through economic events in order to fulfill terms in established contracts. In order to cover minimal set of recurring business transaction patterns we have defined two Contract Execution Transaction patterns.

5.4.2.1 Economic Event Offer Transaction Pattern

Economic Event Offer Transaction pattern [Fig. 30] models an Economic Event that transfers Economic Resource from one partner to another. In typical business scenario this can be treated as delivery of ordered goods to a customer. However, in our analysis, acceptance of Economic Event offer materializes fulfillment of an Economic Commitment.

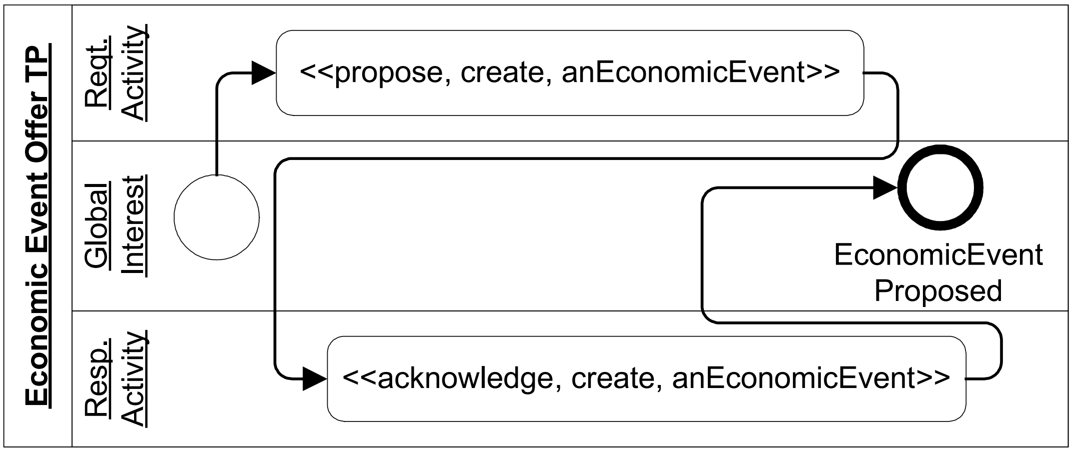


Fig. 30 Economic Event Offer Transaction Pattern

We introduce two Execution TPs (see Fig. 30) specifies the execution of an Economic Event, i.e. the transfer of Resource control from one Agent to another. An example is a Chicken Distributor selling Chickens (a Resource) for $3 (another Resource).

5.4.2.2. Economic Event-Accept/Reject Transaction Pattern

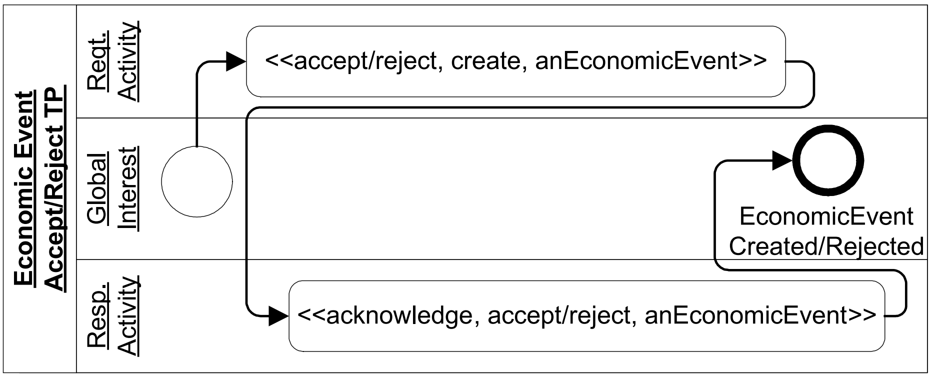


Fig. 31 Economic Event Accept/Reject Transaction Pattern

5.5 Assembling Transactions into Collaboration patterns

An issue is how to combine the transaction patterns described in the previous section, i.e. how to create larger sequences of communication patterns. For this purpose, collaboration patterns define the orchestration of Activities by assembling a set of transaction patterns and/or more basic collaboration patterns based on rules for transitioning from one transaction/collaboration to another [4].

To hide the complexity when transaction patterns are combined into arbitrarily large collaboration patterns, we use a layered approach where the transaction patterns constitute activities in the BPMN diagram of the collaboration patterns.

For this purpose we use two constructs that BPMN provides. BPMN transaction visual object has been used for abstraction of activities of defined transactions. BPMN collapsed sub-process visual object has been used to abstraction of transactions of defined collaborations.

Definition: A *collaboration pattern* (CP) is a BPMN diagram where the activities consist of transaction and collaboration pattern(s). A CP has exactly two end events representing success or failure of the collaboration, respectively. All end events are labeled according to the Intention and Business Entity of the Activity prior to the sequence flow that led to the end event.

5.5.1 Contract Negotiation Collaboration Patterns

5.5.1.1 Contract Establishment Collaboration Pattern

The Contract Establishment collaboration pattern, see Fig. 32 is assembled from the Contract-Offer and Contract-Accept/Reject transaction patterns. An example scenario is a Chicken Distributor proposing an offer to a customer on certain terms. The contract is formed (or rejected) by the customer's acceptance or rejection of the proposed offer.

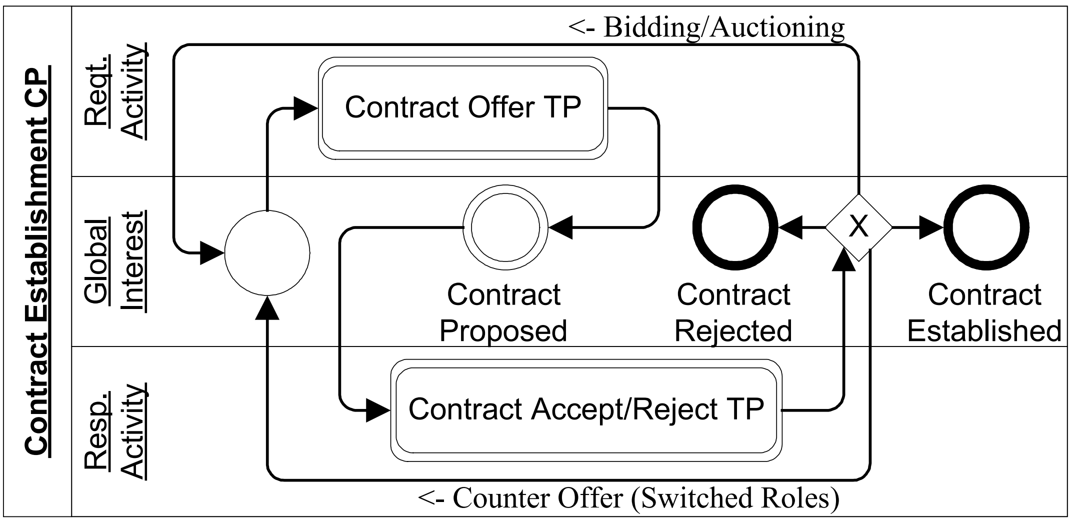


Fig. 32 Contract Establishment CP

The two-recursive paths when a contract offer has been rejected have a natural correspondence in the business negotiation concepts 'Counter Offer' and 'Bidding' (or 'Auctioning') respectively. 'Counter Offer' refers to the switch of roles between business partners, i.e. when the responding role has rejected the requesting role's offer, the former makes an offer of her own. 'Bidding' is modeled via the other sequence Flow from the gateway, i.e. when the responding role has turned down a offer (/contract offer), the requesting role immediately initiates a new Business Transaction with a new (changed) offer for Contract.

5.5.1.2 Contract Proposal Collaboration Pattern

The Contract-Proposal collaboration pattern, Fig. 33, is assembled from the Contract-Establishment collaboration pattern and the Contract Request transaction pattern defined above. This collaboration pattern represents the proceeding quotation request for contract establishment in trading environments. Therefore the contract establishment collaboration pattern discussed in section 5.5.1.1 represents the situation where contract is formed from already existing business information between trading partners.

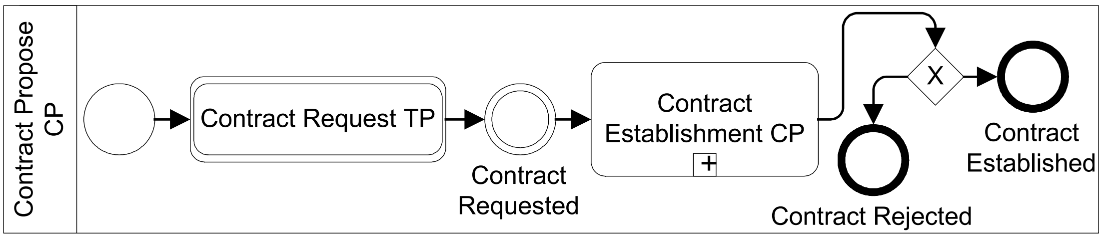


Fig. 33 Contract Propose Collaboration Pattern

However lanes that we introduced above to group requesting and responding activities when defining transaction patterns have been abstracted out in the above collaboration pattern. When collaboration pattern is composed of other collaboration patterns, lanes to place different business partner roles has been collapsed into collaboration patterns.

5.5.2 Execution Collaboration Pattern

The execution collaboration pattern specifies relevant transaction patterns and rules for sequencing among these within the completion of an Economic Event. The pattern is assembled from the economic event offer and economic event accept/reject transaction patterns [Fig. 34].

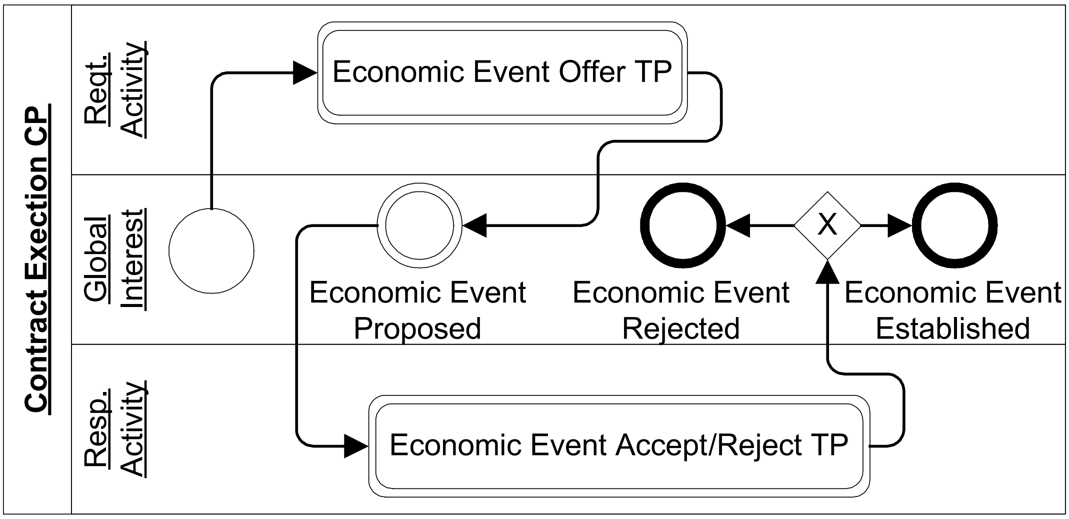


Fig. 34 Contract Execution Collaboration Pattern

5.6 Conclusion

Through this chapter, we have defined set of primitive business transaction and business collaborations patterns as BPMN business process diagrams. However, when defining these patterns, only Contract Negotiation and Contract Execution phases were considered.

The approach proposed here has the potential of being easily extended with many other patterns to modeling recurrent business communications in other phases that are possible in typical trading situations. Some of such phases have been discussed in [65], [14] and **[31]**.

6 BP^3 Designers Assistant

This chapter is dedicated to introduce BP^3 Designers Assistant that we have developed based on the proposed unified framework. However for the effective usage of BP^3 Designers Assistant basic understanding of different actions dependencies is needed. Therefore the chapter starts by defining different action dependencies that one can consider when choreographing activities and then introduces successive steps in BP^3 methodology to business process modeling.

6.1 Action Dependency

The process patterns introduced in Chapter 5 provide a basis for a partial ordering of the activities taking place in a business process, in particular the ordering based on contract negotiation and contract execution. We will refer to the activities involved in the different phases of a process as *contract negotiation* or *contract execution* activities respectively. However, the ordering derived from the process patterns only provide a starting point for designing complete process models, i.e. it needs to be complemented by additional interrelationships among the activities. These interrelationships should have a clear business motivation, i.e. every interrelationship between two activities should be explainable and motivated in business terms. We suggest to formalize this idea of business motivation by introducing the notion of action dependencies. An *action dependency* is a pair of actions (either economic events or activities), where the second action for some reason is dependent on the first one. We identify the following four kinds of action dependencies.

1. Flow Dependencies
2. Trust Dependencies
3. Control Dependencies
4. Negotiation Dependencies

6.1.1 Flow dependencies

A flow dependency, [58], is a relationship between two Economic Events, which expresses that the Resources obtained by the first Economic Event are required as input to the second Economic Event. An example is Distributor who has to obtain a chicken from Supplier before delivering it to Customer for the business scenario described for the running case. Formally, a flow dependency is a pair <A, B>, where A and B are Economic Events from different Dualities.

6.1.2 Trust dependencies

A trust dependency is a relationship between two Economic Events within the same Duality, which expresses that the first Economic Event has to be carried out before the other one as a consequence of low trust between the Business Partners. Informally, a trust dependency states that one partner wants to see the other partner do her work before doing his own work. An example is Distributor who requires a down payment from Customer before delivering Fried Chicken for the running case described. Formally, a trust dependency is a pair <A, B>, where A and B are Economic Events from the same Duality.

6.1.3 Control dependencies

A control dependency is a relationship between an execution Activity and a contract negotiation Activity. A control dependency occurs when one partner wants information about another partner before establishing a Contract with that partner. A typical example is a company making a credit check on a potential customer (i.e. an exchange of the Resources information and money in two directions). Formally, a control dependency is a pair <A, B>, where A is an execution Activity and B is a contract negotiation Activity and where A and B belong to different Dualities.

6.1.4 Negotiation dependencies

A negotiation dependency is a relationship between Activities in the contract negotiation phase from different Dualities. A negotiation dependency expresses that a partner is not prepared to establish a contract with another partner before she has established another contract with a third partner. One reason for this could be that a partner wants to ensure that certain Resources can be procured before entering into a Contract where these Resources are required. Another reason could be that a partner does not want to procure certain Resources before there is a Contract for an Economic Event where these Resources are required. Formally, a negotiation dependency is a pair <A, B>, where A and B are contract negotiation Activities in different Dualities.

6.2 BP^3 Designers Assistant

In this section, we will show how a process model can be designed based on the process patterns that we defined in Chapter 5 and action dependencies. Designing a process model is not a trivial task but requires a large number of design decisions. In order to support a designer in this task, we propose an automated Designers Assistant that guides the designer through the task by means of a series of question-answer sessions. These question-answer sessions are divided into four steps, followed by a fifth step where the process model is generated based on the responses received during step 1-4, see [Fig. 35].

Step 1. during which information is gathered about the Agents involved in the business process, the Resources exchanged between them, and the Economic Events through which these Resources are exchanged. The result from this step is a business model.

Step 2. during which information about the (partial) order between the Economic Events is gathered. The result from this step is an ordering of the Activities in the Execution phase of a process model.

Step 3. during which information about existing negotiation dependencies is gathered. The result from this step is an ordering of the Activities in the Negotiation phase.

Step 4. during which inter phase and inter pool dependencies are established. The result from this step is an ordering of Activities that crosses the Negotiation and Execution phases.

Step 5. during which a set of production rules are applied on the results of the previous steps in order to generate a process model.

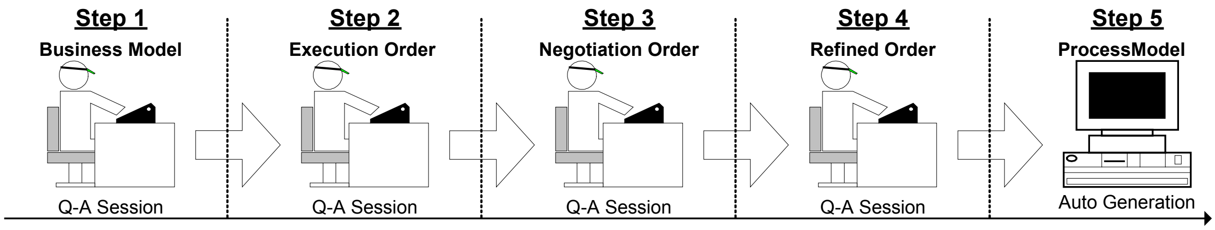


Fig. 35 Steps of the Designers Assistant

6.2.1 Step 1 - Business Model

Step 1 contains all relevant questions that have been formulated to capture necessary information to design the business model. The simpler version of business model that has been used in the BP^3 methodology requires four such questions, as described below.

1. Who are the Business Partners?
2. What are the Economic Resources?
3. What are the Economic Events?
4. What are the Dualities?

6.2.1.1 Who are the Business Partners?

In any trading situation at minimum there should be two business partners. It may also possible to have many business partners to be involved in today's electronic business environment. Therefore the objective of this question is to identify different involved partners those who exchange economic resources among them in a business scenario being investigation.

For the running case described in Chapter 3, the answer for this question will result following four business partners being listed.

1. *Who are the Business Partners?*

Answers:

Customer (Cust)
Distributor (Dist)
Chicken Supplier (Supp)
Carrier (Carr)

6.2.1.2 What are the Economic Resources?

Having identified different partners involved in a given business scenario, it is natural next to look into what economic resources that they are interested in exchanging. Here we ask a designer to list all economic resources that are being exchanged among all involved parties. It would be sufficient to list different types of economic resources exchanged at this point. Actual instance of economic resource that a partner transfers to another partner can be specifically listed when identifying economic events. For our running case the answers for this question is as below.

2. *What are the Resources?*

Answers:

Money
Chicken

6.2.1.3 What are the Economic Events?

Through this question, the designer identifies all economic events for a given business scenario. By identifying economic events, the designer can relate what economic resource has been transferred from which business partner to which business partner. For this question we propose the designer to fill a row for each economic event in a table like the one shown below [**Table 3**]. For the running case introduced in the Chapter 3, the completed table would look like [**Table 3**].

3. What are the Economic Events? Specify them by filling in the following table.

Name of EconomicEvent	EconomicResource	Business Partners	
		From	To
DownPayToDist	Money	Customer	Distributor
FinalPayToDist	Money	Customer	Distributor
ChickenToCust	Chicken	Distributor	Customer
DownPayToSupp	Money	Distributor	Chicken Supplier
FinalPayToSupp	Money	Distributor	Chicken Supplier
ChickenToDist	Chicken	Chicken Supplier	Customer
DeliveryToDist	Delivery	Carrier	Distributor
PayToCarr	Payment	Distributor	Carrier

Table 3 Economic Events for the running case

6.2.1.4 What are the Dualities?

As the last question in the Step 1 to design business model in BP^3 methodology, the designer requires to group different economic events into dualities that exist between business partner paires. These identified dualities bewteen business partner paires will be the basis for all binary business collaborations that are to be designed from deifined patterns.

Simply designer can groups relevant economic events with appropriate names for dualities for the considering case as in [**Table 4**], i.e as explained in the Section 4.2.7, economically to legally related economic events between two partners grouped by giving appropriate business names.

4. *Group the* Economic Events *into* Dualities *by filling in the following table.*

EconomicEvent	Duality
DownPayToDist	Chicken Sales
FinalPayToDist	
ChickenToCust	
DownPayToSupp	Chicken Purchase
FinalPayToSupp	
ChickenToDist	
DeliveryToDist	Chicken Delivery
PayToCarr	

Table 4 Economic Events grouped into Dualities for running case.

The answers to the four questions explained above provide sufficient information to produce the business model shown below.

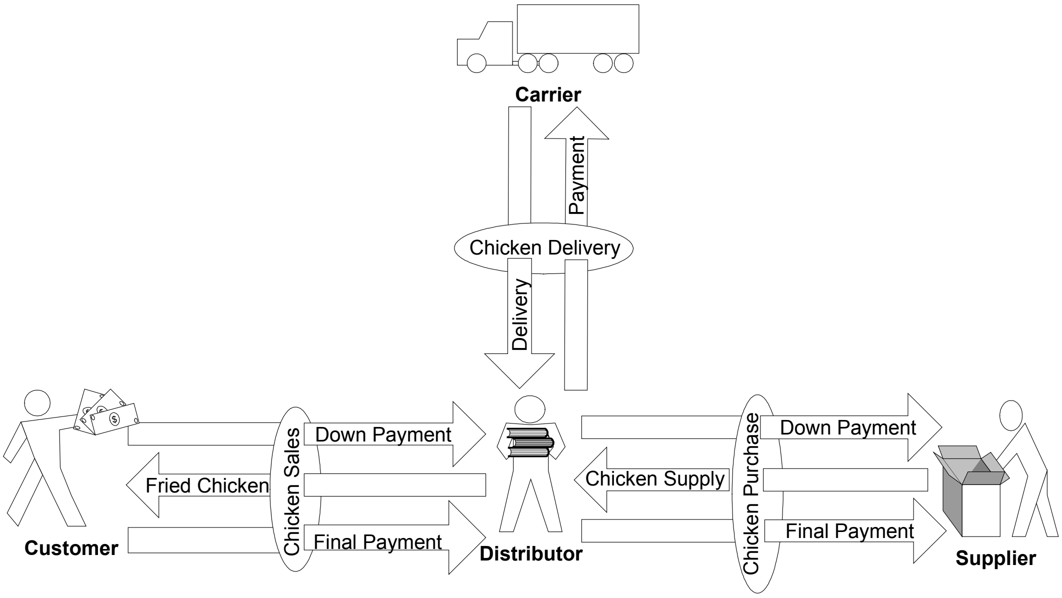


Fig. 36 Business Model for the running case

6.2.2 Step 2 – Execution Phase Order

In Step 2, the identified economic events are to be ordered. Since economic events come into existence as a result of execution of established contracts between business partners, we have named the economic event order resulting from this step as the execution phase order.

Also there are two different types of dependences: trust and flow to be considered while identifying the order between different economic events. The trust dependencies are to be considered when ordering economic events that belong to the same duality, while flow dependencies are to be considered when ordering economic events that belong to different dualities. These two types of dependencies that may exist between different economic events are captured through a matrix below [**Table 5**].

In this matrix, all economic events are checked against each other by listing them in rows as well as in columns. By placing “<” symbol in a cell [i,j], the designer can represent that economic event in i^{th} row has to be completed before starting economic event in j^{th} column. It is possible to reduce the search space to upper triangular form of the matrix by demanding the designer to consider both "<" and ">" for all cells. For simplicity of use, we kept the whole matrix while limiting the consideration only to "<" at each cell.

The diagonal of the matrix is left out, as it is meaningless to check dependency of an economic event against itself. Also notice that captured dependencies are subscripted either with "t" or "f" to indicate whether it is trust dependency or flow dependency. As all trust dependencies exist between economic events that belong to the same duality, notice that trust dependencies will occur in the diagonal formed by double line squares that represent different dualities. Therefore, all flow dependencies occur outside the diagonal formed by double line squares. Finally, for all dependency requirements through this matrix and in the following steps are numbered in order to make it easy to trace back to the final solution to be discussed in the appendix. The requirements in these steps motivate different sequencing orders of business process diagrams and different inter communication between business process diagrams in final solution.

5. *Specify* Flow *and* Trust Dependencies *between economic events by filling in the matrix below.*

	DPTD	**FPTD**	**CTC**	**DPTS**	**FPTS**	**CTD**	**DTD**	**PTC**
DownPayToDist (**DPTD**)			(1)$<_t$	(2) $<_f$				
FinalPayToDist (**FPTD**)					(3) $<_f$			(4) $<_f$
ChickenToCust (**CTC**)		(5)$<_t$						
DownPayToSupp (**DPTS**)						(10)$<_t$		
FinalPayToSupp (**FPTCS**)								
ChickenToDist (**CTD**)			(6)$<_f$		(7)$<_t$			
DeliveryToDist (**DTD**)			(8)$<_f$					(9)$<_t$
PayToCarr (**PTC**)								

Table 5 Flow and Trust Dependencies for running case.

The economic event order received by completing the above matrix is sufficient to complete the execution phase of the BPMN business process diagrams. With this information, in addition to determining the order of execution of different economic event within a binary collaboration process diagram (only trust – same duality) it is also possible to partially complete economic event order (flow – between duality) for inter binary collaboration process diagrams.

The execution phase is depicted with shaded areas in the final solution for the running case in the appendix. Numerical subscripts of the above matrix can be traced back to different sequencing and inter business process diagram communication in the final solution in appendix.

6.2.3 Step 3 – Contract Negotiation Phase Order

Having gathered sufficient information to produce the BPMN diagram for the Execution phase, development of BPMN business process diagrams in BP^3 continues for the Contract Negotiation phase. As there are two ways for initiating a binary Business Collaboration according to the suggested collaboration patterns in Section 5.5.1 where we introduced Contract Negotiation Collaboration Patterns in Chapter 5, it is first necessary to identify which of these patterns to use for binary collaborations. Question 6 captures relevant collaboration pattern to start binary collaboration for each duality. Also notice that two branches (a) and (b) of question 6 is explicitly expressed for clarity of exactly what patterns to be selected. For a duality if branch (a) of question 6 is answered with "Yes", then branch (b) can be skipped.

6. *For each binary Business Collaboration (for each Duality), ask whether*

(a) *a quotation already exists when the binary collaboration starts, or*

(b) *the binary collaboration is started by a partner requesting a quotation.*

If the answer is (a) then the contract establishment collaboration pattern of Fig. 32 will be chosen. If the answer is (b) then the contract proposal collaboration pattern of Fig. 33 will be chosen.

The answers to question 6 for the running case are given below and only positive answers with "Yes" to be considered for the solution. Relevant branch letters are in bold.

6.1	***(a)***	*Does a quotation already exist when the Cust+Dist collaboration starts?, or*	*Yes*
	(b)	*Is the Cust+Dist collaboration started by a partner requesting a quotation?*	-
6.2	*(a)*	*Does a quotation already exist when the Dist–Supp collaboration starts?, or*	*No*
	(b)	*Is the Dist–Supp collaboration started by a partner requesting a quotation?*	*Yes*
6.3	*(a)*	*Does a quotation already exist when the Dist+Carr collaboration starts?, or*	*No*
	(b)	*Is the Dist+Carr collaboration started by a partner requesting a quotation?*	*Yes*

Note that abbreviated partner names are used here in naming the binary collaborations for dualities above. Between four partners, there are only three dualities to be considered.

The answers from this question are used to derive the beginning of binary collaborations (see the white area in the final solution of the appendix).

We continue further by identifying the negotiation dependencies. Similarly as in Step 2 where the execution phase order is captured through a matrix, here another such matrix is proposed. According to the way we defined collaboration patterns in Chapter 5, there are three possible events with global interest to be considered for further inter process communications, i.e. "Requested", "Proposed" and "Established" in the contract negotiation phase.

7. Specify the Negotiation Dependencies by filling in the following table, (where the row and column headings are pattern instantiations identified in questions 6). If an Activity[1] (in row i) precedes an Activity (from column j) put a '<' symbol in the corresponding cell (i.e. cell <i,j> in the table).

Completed matrix for the running case is shown in **Table 6**.

Note, that the relationships within a binary collaboration are given by the process patterns for contract negotiation phase and therefore we have crossed out the corresponding cells in the diagonal. The results from this question will give input for ordering of the activities from the Contract Negotiation phase. The alphabetical notes in the table refer to the resulting flows in the process model in the final process diagram in the appendix.

[1] Formally the contents of the rows are *pattern instances* (see Chapter 6), but without loss of generality we can equate each pattern instance to its first Activity.

	COCD	CECD	CRDS	CODS	CEDS	CRDC	CODC	CEDC
Contract-Offer: Cust+Distr (**COCD**)			(a)<			(b)<		
Contract-Establish: Cust+Distr (**CECD**)								
Contract-Request: Distr+Supp (**CRDS**)								
Contract-Offer: Distr+Supp (**CODS**)		(c)<						
Contract-Establish: Distr+Supp (**CEDS**)								
Contract-Request: Distr+Carr (**CRDC**)								
Contract-Offer: Distr+Carr (**CODC**)		(d)<						
Contract-Establish: Distr+Carr (**CEDC**)								

Table 6 Negotiation Dependencies for the running case.

6.2.4 Step 4 - Refined Order

In Step 1, business partners, economic events and dualities were identified in order to design the business model. During Step 2, identified economic events were ordered and this resulting order is used to sequence the activities of the execution phase for binary collaborations and their inter collaboration communication. In Step 3, the negotiation order is determined for activities in the contract negotiation phase to complete communication among business process diagrams for identified dualities.

Step 4 is the last question-answer session where further refinements for identified trust and flow dependencies are applied. In other words, through questions 8 and 9 so far captured flow and trust dependencies between economic events and are further strengthened as required in the considered business case. As the final question in Step 4, a matrix is prompted to capture possible control dependencies between negotiation and execution phase activities from different dualities.

8. *For each pair of* Economic Events *<EE_i, EE_j> (see Table 3), such that $EE_i <_f EE_j$: Is it required to perform EE_i before making a contract acceptance for EE_j, (i.e. a Contract Establishment between the* Agents *in EE_j)?*

The intuition behind this question is that a partner may want to ensure that she has definite access to certain Resources before she is prepared to enter into a Contract for some product where these Resources are needed as input. It is possible to think about this question as a strengthening of a flow dependency – we say not only that EE_j cannot be performed before we have got the Resources from EE_i, but even that we are not prepared to enter into a Contract for EE_j before we have got the Resources from EE_i.

Below, the implementation of question 8 for the running case is given with answers. The five flow dependencies identified in the **Table 5** can be listed as follows.

1. DownPayToDistributor $<_f$ DownPayToSupplier
2. FinalPayToDistributor $<_f$ FinalPayToSupplier
3. FinalPayToDistribtutor $<_f$ PayToCarrier
4. ChickenToDistributor $<_f$ ChickenToSupplier
5. DeliveryToDistributor $<_f$ ChickenToSupplier

In order to further refine the dependencies listed above, the following questions can be posed so that cases where stronger dependencies can be identified and reflected in the final process model to be generated.

8.1	*Must DownPayToDist be done before establishment of Dist+Supp Contract?*	**Yes**
8.2	*Must FinalPayToDist be done before establishment of Dist+Supp Contract?*	**No**
8.3	*Must FinalPaytToDist be done before establishment of Dist+Carr Contract?*	**No**
8.4	*Must ChickenToDist be done before establishment of Dist+Cust Contract?*	**No**
8.5	*Must DeliveryToDist be done before established of Dist+Cust Contract?*	**No**

*9. For Economic Events triplets in [***Table 3***], EE_i, EE_j, EE_k, such that $EE_i <_t EE_j$ and $EE_k <_f EE_J$: Is it required to perform EE_i before making a contract acceptance for EE_k (i.e a Contract Establishment between the Agents in EE_k)?*

This question can be seen as a strengthening of a trust dependency. It says not only that we want to see another partner perform EE_i before we perform EE_j, but that we want to see our partner to perform EE_i before we even start acquiring resources needed to perform EE_j.

Below, the implementation of question 9 for the running case with answers. In **Table 5** there are four pairs of trust and flow dependencies that hold necessary requirements to pose question 9. These are listed below.

1. Delivery ToDistributor $<_t$ PayToCarrier and FinalPayToDistributor $<_f$ PayToCarrier

2. ChickenToDistributor $<_t$ FinalPayToSupplier and FinalPayToDistributor $<_f$ FinalPayToSupplier

3. DownPayToDistributor $<_t$ ChickenToCustomer and ChickenToDistributor $<_f$ ChickenToCustomer

4. DownPayToDistributor $<_t$ ChickenToCustomer and DeliveryToDistributor $<_f$ ChickenToCustomer

For these four dependency pairs the following questions can be posed to capture situations that hold stronger dependencies.

9.1	*Must DeliveryToDist be done before establishment of Cust+Dist Contract?*	**No**
9.2	*Must ChickenToDist be done before establishment of Cust+Dist Contract?*	**No**
9.3	*Must DownPayToDist be done before establishment of Dist+Supp Contract*[2]*?*	**No**
9.4	*Must DownPayToDist be done before establishment of Dist+Carr Contract?*	**Yes**

A matrix similar to the one used to capture negotiation dependencies has been used also to capture possible control dependencies. The control dependency is defined over activity pairs from negotiation and execution phases as defined in above Section 6.1. Therefore the matrix used to capture control dependencies contains negotiation phase activities as row

[2] Case 9.3 is already covered by case 8.1 and only shown here for reasons of completeness.

headings and execution phase activities as column headings as shown below in **Table 7**.

10. Specify the Control Dependencies by filling in the following table, (where the row and column headings are pattern instantiations identified in questions 4 and 6). If an Activity (in row i) precedes an Activity (from column j) put a '<' symbol in the corresponding cell (i.e. cell <i,j> in the table).

However, as there are no control dependencies in the business scenario described for the running case, the **Table 7** is left empty. Also remember some of the cells in the matrix to be used to capture control dependencies may already be filled with answers received for question 8 and 9 as refining flow and trust dependencies. This may further reduce search space other than the crossed out diagonal formed with double line squares in **Table 7**.

	DownPayToDist	FinalPayToDist	ChickenToCust	DownPayToSupp	FinalPayToSupp	ChickenToDist	DeliveryToDist	PayToCarr
Contract-Offer: Cust+Distr								
Contract-Establish:Cust+Distr								
Contract-Request: Distr+Supp								
Contract-Offer: Distr+Supp								
Contract-Establish:Distr+Supp								
Contract-Request:Distr+Carr								
Contract-Offer: Distr+Carr								
Contract-Establish: Distr+Carr								

Table 7 Control Dependency for the running case.

6.3 Business Process Generation

The final step of the proposed Designers Assistant is the generation of a BPMN diagram based on the answers from steps 1 – 4. This is achieved using the binary collaboration patterns introduced in Chapter 5 and a set of production rules to interconnect those instantiated binary collaborations into a multi-party collaboration. Detail discussion of productions rule that complete final process models can be found in Chapter 8.

6.4 Conclusion

There are two main sections in this chapter to introduce Action Dependency and BP^3 Designers Assistant.

In the Action Dependency section, we have discussed different dependencies that user may consider in ordering different actions that trading partners are to perform within their collaborative environments. We have listed four such possible dependency types: Flow, Trust, Control and Negotiation. There is some thorough investigation needed to test whether these are sufficient and applicable.

In the other section BP^3 Designers Assistant is introduced. The BP^3 methodology advises to use five-step approach in designing business process models for given business scenario. The first four steps in this approach are question-answering sessions where different business requirements are captured by means of natural language interface that uses terminology closer to the domain. The final step is the business process generation, which is a step that can readily be automated. In this case, it generates BPMN business process diagrams as final target models.

Through the laboratory experiments carried out during the development of our approach, we have experienced, much of the designers' burden can be over come and rapid system development can be achieved with similar Designers Assistants.

7 Production Rules to generate Process Diagrams

In this chapter, set of production rule has been introduced. These rules can be applied on the answers received for question in Step 1 to Step 4 of BP^3 Designers Assistant introduced in Chapter 6 to generate BPMN business process diagrams. Also application of these rule illustrated with running case that has been considered through out this thesis.

7.1 Introduction

This chapter contains set of production rules that can be applied on the user responses received during first four steps in Designers Assistant (described in Chapter 6) to generate BPMN business process diagram (BPD) based on defined patterns (see Chapter 6).

We have categorized production rules into four main groups:

1. Rules for Binary Collaborations
   (to generate control flow within a pool)
2. Reduction Rules
   (to fine-tune final solution into comprehensive simpler model)
3. Rules for Inter-Collaborations
   (to completes the inter-connection between pools)
4. Deadlock Prevention Rules
   (to avoid starvation for events and to conflict resolution)

7.2 Rules for Binary Collaborations

These rules are for the responses received from question 1 to 6 in step 1 and in step 2 of the Designers Assistant (see Chapter 6). By applying these rules for binary collaboration, BPMN diagrams for different pools can be completed while leaving inter communication among them to be completed through the rules for inter-collaborations.

7.2.1 Rule 1 – Binary Collaborations (Pool)

1. Assume D_1, D_2, …, D_n are identified dualities at question 4 in step 1.

For each duality D_i
Create BPMN pool that will contain BPD for D_i
Name an empty pool with "P_{i1} + P_{i2} Collaboration" where P_{i1} and P_{i2} are the partners involved in the duality, D_i

Illustration for Rule 1:

After applying rule 1 for the "Chicken Sales" duality between Customer and Distributor for the running case, it will result a named empty pool to include BPD for binary collaboration for that duality as follows.

Fig. 37 BPMN Pool for Customer-Distributor Collaboration

Similarly application of the rule 1 for the remaining two dualities in the running case will result in three BPMN pools in total to place business process diagrams for three dualities among four business partners. See the final solution in appendix.

7.2.2 Rule 2 – Binary Collaborations (Contract Negotiation Phase)

2. For each identified duality, D_i

If answer to question (6) in Step 3 is (a) then
Start BPD by instantiating a "Contract Establishment" pattern in "Pi1+Pi2 Collaboration" pool as in Fig. 32.
else
Start BPD by instantiating a "Contract Propose" pattern in "Pi1+Pi2 Collaboration" pool as in Fig. 33.
endif

Illustration for Rule 2

For the running case, while binary collaboration between Customer and Distributor starts with out request for a quotation, collaborations between Supplier and Distributor starts with a preceding quotation request. See answers to question 6 in the Section 6.3 in Chapter 6.

Therefore for the BPD for Customer and Distributor collaboration starts as follows after instantiating the contract establishment patter. See Fig. 38. Notice that we have shown here is the happy path, as collaboration to proceed only success path of the previous needed to be considered. In other words, with out successful establishment of a contract, there is no point in carrying out termed economic events in the contract.

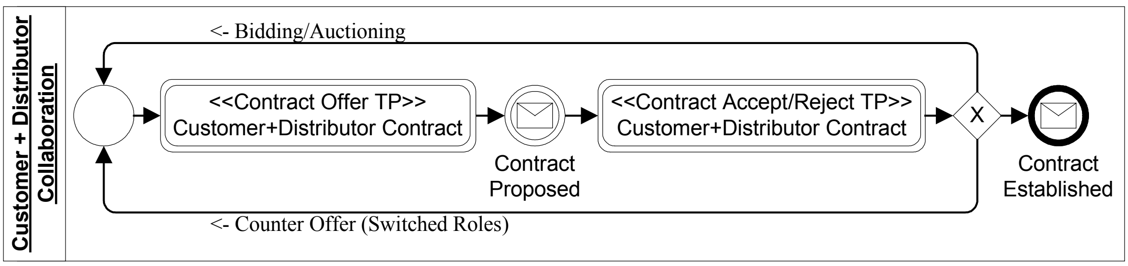


Fig. 38 BPD for Negotiation Phase of Customer & Distributor Collaboration

For the collaboration between, Distributor and Supplier, contract negotiation starts with a quotation request first. Therefore applying rule to for the answer received for question 6, resulting BPD for this collaborations will take the following form. See Fig. 39.

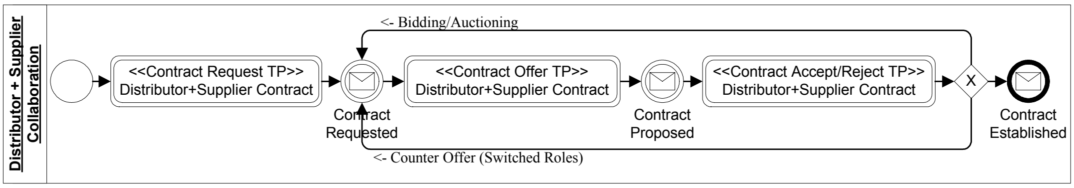


Fig. 39 BPD for Negotiation Phase of Distributor & Supplier Collaboration

7.2.3 Rule 3 – Binary Collaborations (Contract Execution Phase)

3. Assume A_{i1}, A_{i2}, …, A_{in} are economic events for a duality D_i

For each duality
Instantiate "Execution Collaboration" patterns in parallel for each $A_{i1}, A_{i2}, \ldots, A_{in}$ as in Fig. 40 *, in the pool "P_{i1} + P_{i2} Collaboration" for D_i connecting success path of instantiated collaboration pattern from rule 2.*

Illustration for Rule 3

There are three economic events for the "Chicken Sales" duality for the running case as identified at question 3 of Step 1 in Chapter 6. After applying rule 3, contract execution collaboration patterns for these three economic events instantiate in parallel as in Fig. 40.

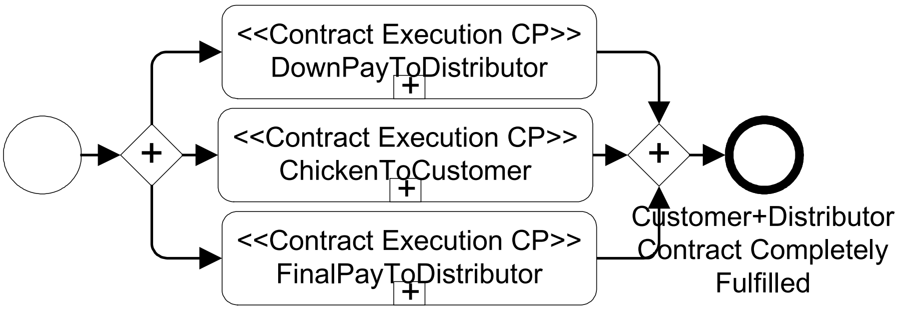


Fig. 40 Initial BDP for Execution Phase of Customer & Distributor Collaboration

By default execution collaborations are instantiated in parallel and then depending on the user-specified "Trust" dependency requirements at question 5) of Step 2 execution collaborations are ordered as illustrated below. Final ordered execution collaborations of a given duality connected to the resulted contract negations for the same duality from rule 2. The Contract Negotiation phase and Execution Collaborations phase has been discussed in detail in the Section 6.1.3 of Chapter 6.

A Modeling Assumption:

One of the assumptions we made for activities in BPMN business process diagram for collaborations is as follows. All activities (say activity Ai) are triggered with starting event; SE subscripted with activity name (as SE_{Ai}) and all activities are terminated with resulting event; EE subscripted with activity name (as EE_{Ai}). However if there is no business requirement that order these activities, we suppress these events without explicitly visualizing them in BPMN business process diagrams. For an example, BPD segment, Fig. 40 can be extended depending on the business requirements to Fig. 41 by explicitly visualizing these intermediate events in BPMN business process diagram.

Rule 4 – Binary Collaborations (Trust Dependencies)

4. For each trust dependency order specified in the **Table 5** of (see Chapter 6) between collaboration activities A1 and A2 for two economic events,

If $A_1 < A_2$ holds then

If no AND Gateway between SE_{A2} and A_2 then

Add a AND Gateway between SE_{A2} and A_2
endif
Link Intermediate Event, EE_{A1} to the AND Gateway between SE_{Ay} and A_y
endif

Illustration for Rule 4

Consider the trust dependency, (1) DownPayToDist < ChickenToCustomer specified in the **Table 5** for the running case. By applying rule 4 the initial BPD segment for execution phase of Customer and Distributor collaboration will change as follows. See Fig. 41.

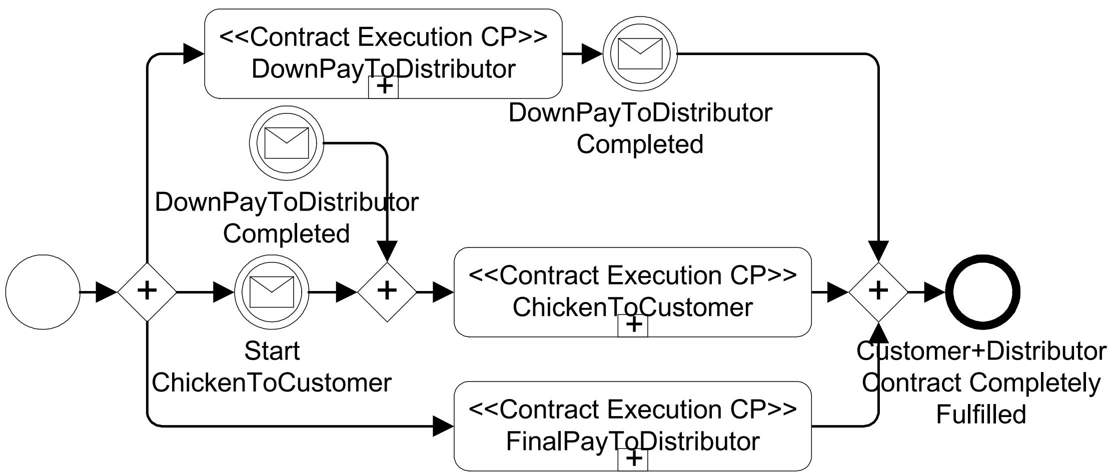


Fig. 41 BDP for Execution Phase of Customer & Distributor Collaboration after applying a trust dependency requirement

Similarly, once again rule 4 can be applied on the trust dependency requirement (5) ChickenToCust < FinalPayToDist as in the **Table 5** for the running case. It will further constraint the resulting BPD segment for execution phase of Customer and Distributor collaboration. The result is shown in See Fig. 42.

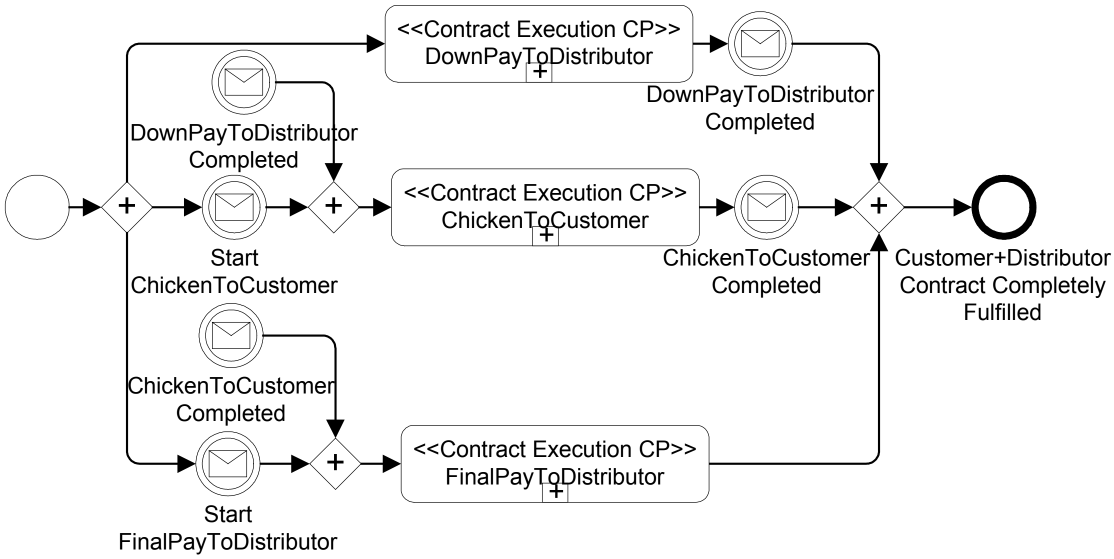


Fig. 42 BDP for Execution Phase of Customer & Distributor Collaboration after applying all trust dependency requirements (before apply reduction rules)

However, the above process diagram can be simplified into a comprehensive one by applying the reduction rule to be discussed next.

7.3 Reduction Rules

In this section, we have defined rules that can be applied on a completed solution to fine-tune it, so that final result will look much simpler and comprehensive. First, a rule that can be used to reduce the BPMN process segment for contract execution phase of a binary collaboration is defined. See Rule 5.

Then a rule that can be used to reduce the BPMN inter-pool communication is proposed. By applying this rule, redundant communication between BPMN pools can be simplified.

7.3.1 Rule 5 – Reduction (Contract Execution Phase, Binary Collaboration)

In the rule 5, reference numbers are give in each possible branches so that they can be referred back from the illustrative example give in the proceeding section.

Also there are two diagrams referred in this rule as original diagram and reduced diagram. The original diagram is the one before apply any reduction rule. The reduced diagram is the one successively reduced to its final form by applying the reduction rule until there are no more requirements left to be considered in the original diagram. Recollect that SE and EE stand for Starting Event and Ending Event of BPMN activities as introduced in the above assumption.

5. Assume A1 and A2 are two activities in execution phase of a binary collaboration. EE_{A1} and SE_{A2} are resulting event of A1 and triggering event of A2 respectively.

If EE_{A1} AND SE_{A2} < A2 holds (in Original diagram) then
If Ax < A2 already holds (in the Reduced diagram) then
If no "AND Gateway" between Ax and A2 [ref. 5.a]
Add "AND Gateway" between Ax and A2
Link A1 to "AND Gateway" between Ax and A2
Else [ref. 5.b]
Link A1 to "AND Gateway" between Ax and A2
Endif
Else
if no A1 < Ay (in the Reduced diagram) then [ref. 5.c]
Link A2 after A1
Else
if no "AND Gateway" between A1 and immediate event to proceed
A1 (in the Reduced diagram) then [ref. 5.d]
Add "AND Gateway" between A1 and immediate event to proceed A1 (in the Reduced diagram)
Link A2 to "AND Gateway" between A and immediate event to proceed A1 (in the Reduced diagram)
Else [ref. 5.e]
Link A2 to "AND Gateway" between A and immediate event to Proceed A1 (in the Reduced diagram)
Endif
Endif
Endif
Endif

Different Paths in the Rule 5

[ref. 5.a] When a predecessor activity (Ax) already placed (no "AND Gateway") in reduced diagram and an activity (A1) also to be placed as predecessor to A2.

[ref 5.b] When there are predecessors activities connected with "AND Gateway" and an activity (A1) also to place as predecessor to A2.

[ref 5.c] When there is no successor activity (Ay) already placed (no "AND Gateway") in the reduced diagram and activity (A2) to place as successor to A1.

[ref 5.d] When there is successor activity (Ay) already placed in the reduced diagram without "AND Gateway" and an activity (A2) also to place as successor to A1.

[ref 5.e] When there are successor activities (Ay) already placed in the reduced diagram connected with "AND Gateway" and an activity (A2) also to place as successor to A1.

Illustration of Rule 5

In this section we will pictorially demonstrate how resulted execution collaborations from rule 4 can be reduced into more simpler and comprehensive form. Once again the original BPMN diagram for execution phase process segment of Customer and Distributor collaboration is shown as Fig. 43.

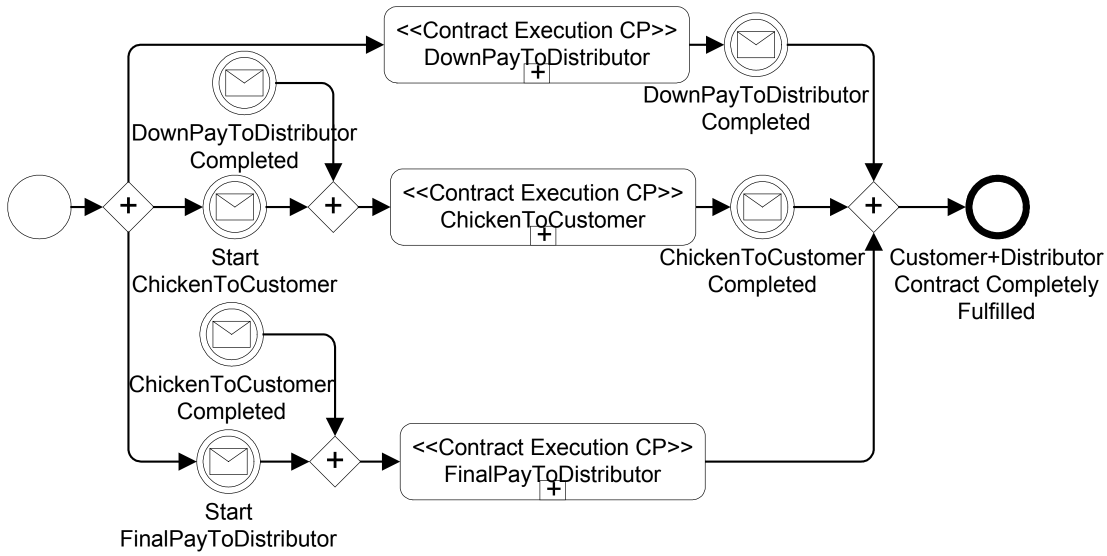


Fig. 43 BDP for Execution Phase of Customer & Distributor Collaboration (Original Diagram to rule 5)

In Fig. 43 there is a requirement (DownPayToDistributor AND Start ChickenToCustomer) < ChickenToCustomer that can be subject to rule 5. After following the branch [ref 5.c] of the rule 5, the resulting reduced process diagram will take the form Fig. 44.

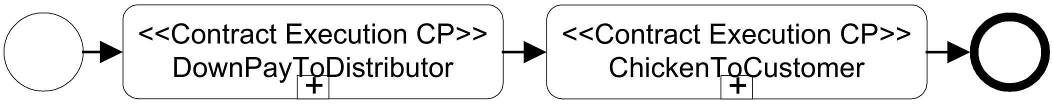


Fig. 44 Intermediate reduced process diagram for Customer and Distributor Collaboration (applying rule 5)

In Fig. 43, there is also a requirement (ChickenToCustomer AND Start FinalPayToDistributor) < FinalPayToDistributor that can be subject to rule 5. Similarly, after following the branch [ref 5.c] of the rule 5, the resulting reduced process diagram will take the form Fig. 45.

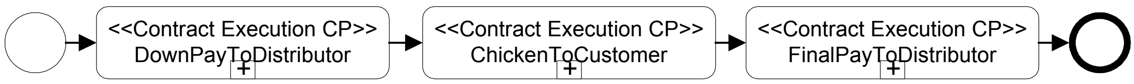


Fig. 45 Final reduced process diagram for Customer and Distributor Collaboration (application of rule 5) in execution phase

Now it can be understood that the resulting diagram [Fig. 40] from rule 4 is semantically equivalent to the reduced diagram [Fig. 45] resulted from rule 5. How ever before connection the final reduced process segment to the BPD completed so far we visualize terminating vents of these collaboration activities as they may needed to be communicated to other BPD for different dualities later on. Then the completed BDP for Customer and Distributor collaboration can be shown now as Fig. 46.

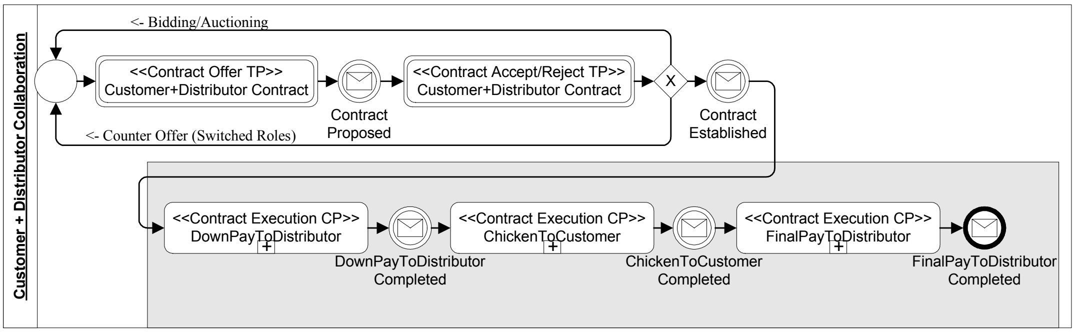


Fig. 46 Complete BPD for Customer and Distributor Collaboration

Following the same rules the three pools containing BPD for three dualities identified for the running case can be completed. See the final solution in the appendix. Notice that shaded area with in the pool contains all activities that are belonging to contract execution phase while activities in the white area belong to contract negotiation activities. See Fig. 46.

7.3.2 Rule 6 – Reduction (Inter Collaboration)

Though the rules for completion of inter-pool communication of BPMN solution are discussed under Section 7.4, we discuss the rule 6 to reduce inter-collaboration communication here under reductions rules category here.

There may be situations where captured action dependency requirements are refined further through structured questioning for refined order as defined in Chapter 6. In such situations, it is possible to remove weaker inter-pool communications while leaving only the communication that holds stronger dependency, which in-return implies that redundant weaker communication.

6. Assume A1 and A2 are two activities of a binary collaboration and A3 is another activity from a different binary collaboration. Assume that

within the binary collaboration mentioned first, A1 < A2 holds, there is an action dependency such that A3 < A2 and a captured refine order A3 < A1.

If EE_{A3} < A2 AND EE_{A3} < A1 holds (in Original diagram) then
If EE_x AND SE_{A2} < A2 then
(EE_x is any other event A2 depends on)
Remove the link of EE_{A3} to AND Gateway between SE_{A2} and A2
Else(no more events that A2 depends on)
Remove the link of EE_{A3} to AND Gateway between SE_{A2} and A2
Remove AND Gateway between SE_{A2} and A2
Link SE_{A2} to A2 directly
Endif
Endif

Illustration of Rule 6

For the running case, there is flow dependency requirement captured at the **Table 5** and a refined order as captured through the question 7.1 in Chapter 6 as listed below respectively.

1. DownPayToDistributor $<_f$ DownPayToSupplier
2. DownPayToDistributor < Contract Establish (Distributor + Supplier)

Before applying the reduction rule 6, the generated BPMN inter-collaboration by applying inter collaboration rules can be depicted as in Fig. 47. The inter collaborations rules will be discussed next.

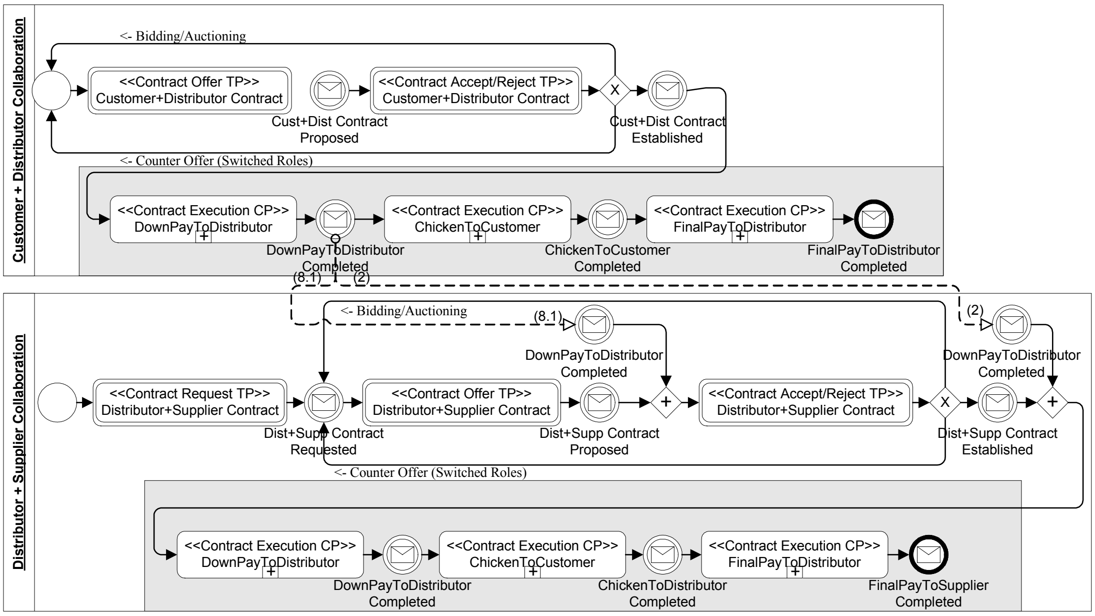


Fig. 47 Inter Connected BPDs for Customer+Distributor and Distributor+Supplier Collaborations [Flow (2) and Refined (7.1) dependencies]

After applying the rule 6, the reduced inter-collaboration communication will take the following form as shown in Fig. 48. Notice that since there is only one external event that DownPayToSupplier depends is Distributor+Supplier collaboration, both DownPayToDistributor Completed event and the AND Gateway have been removed.

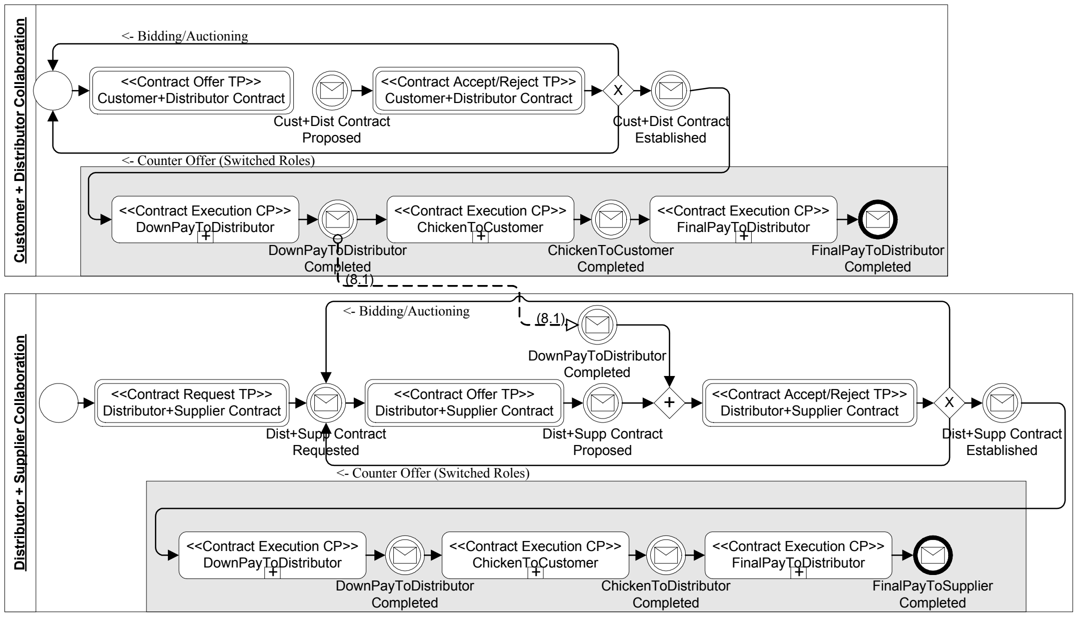


Fig. 48 Inter Connected BPDs for Customer+Distributor and Distributor+Supplier Collaborations [Reduced only to Refined (7.1) dependency]

7.4 Rules for Inter-Collaborations

These rules are for the responses received for remaining questions in step 2 to step 4 of the Designers Assistant (see Chapter 6). By applying these rules BPMN diagrams completed so far in pools can be inter-related depending on captured requirements. There is no distinction between different activities at BPMN diagram levels. But for the clarity we have divided them into following three groups with respect to business meaning of relevant activities to be ordered.

7.4.1 Rule 7 - For Flow Dependency

7. For each flow dependency order specified in the Table 3 of (see Chapter 6) between collaboration activity A1 and A2 for two economic events,

if $A_1 < A_2$ *holds then*
    If no AND Gateway between SE_{A2} *and* A_2 *the*
        Add an AND Gateway between SE_{A2} *and* A_2
    endif
        Link Intermediate Event, EE_{A1} *to the AND Gateway between* SE_{A2} *and* A_2
endif

Illustration of Rule 6

For the running case that has been introduced, there will be three pools to be place with BPDs for respective binary collaborations for identified three dualities as in the Chaper 6. In Fig. 49 the BPDs for binary collaborations between Customer + Distributor and Distributor + Supplier is shown.

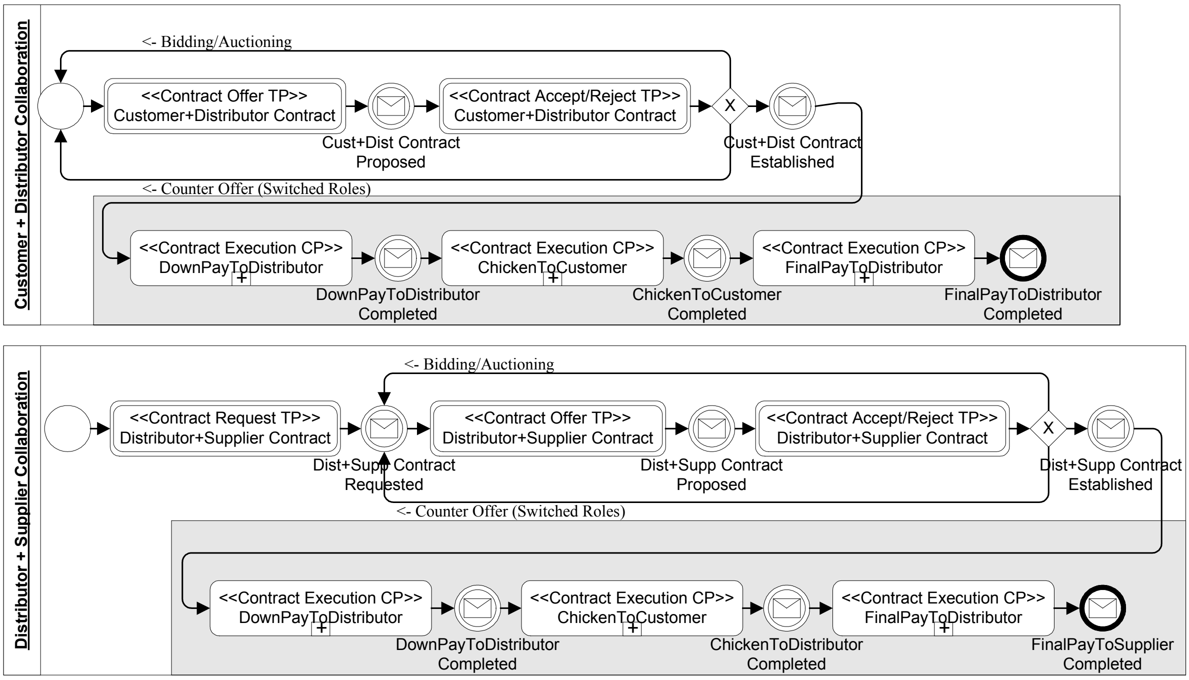


Fig. 49 BPDs for Customer+Distributor and Distributor+Supplier Collaborations

Now consider the trust dependency, (6) ChickenToDistributor < ChickenToCustomer specified in the **Table 5** for the running case. By applying rule 6 the BPDs completed so far in Fig. 49 for binary collaborations between Supplier + Distributor and Customer + Distributor will inter-connected as in Fig. 50 below.

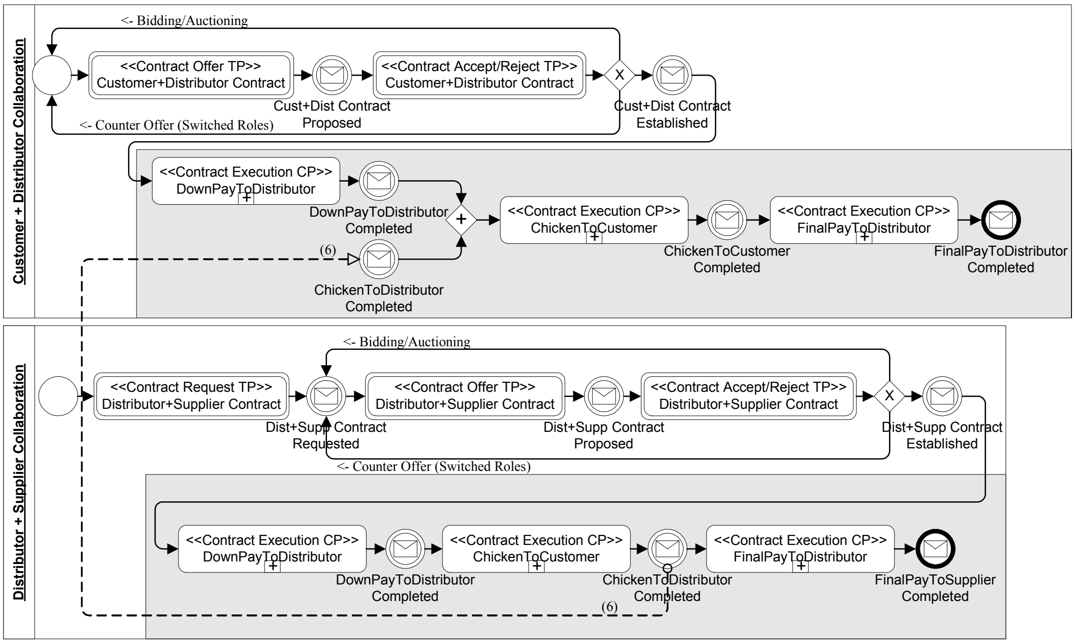


Fig. 50 Inter Connected BPDs for Customer+Distributor and Distributor+Supplier Collaborations [Flow dependency (6)]

7.4.2 Rule 8 - For Control and Negotiation Dependencies

8. For each control or negotiation dependency order specified between two activities A1 and A2 in contract negotiation phase or execution phase,

if $A_1 < A_2$ holds then
If no AND Gateway between SE_{A2} and A_2 the
Add an AND Gateway between SE_{A2} and A_2
endif
Link Intermediate Event, EE_{A1} to the AND Gateway between SE_{A2} and A_2
endif

***Illustration of Rule** 7*

For the running case, a requirement, (a) Customer+Distributor Contract Offer < Distributor + Supplier Contract Request is received through the **Table 6** that has been proposed to capture different negotiation dependencies. By applying rule 7 for the above requirement will further inter-connect the two BPDs in Fig. 50 as shown below in Fig. 51.

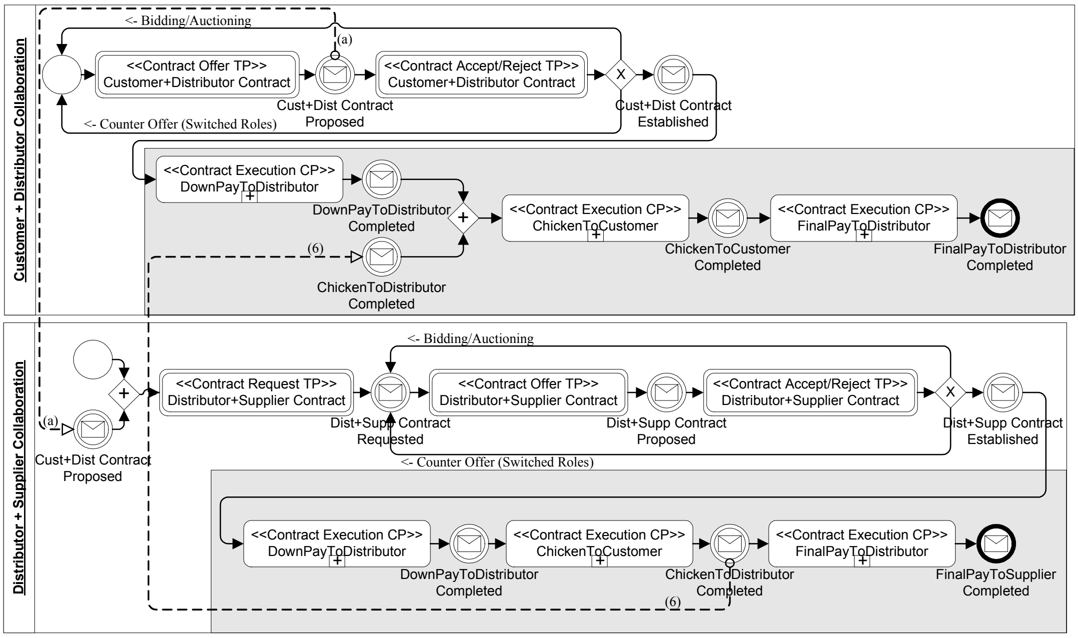


Fig. 51 Inter Connected BPDs for Customer+Distributor and Distributor+Supplier Collaborations [Negotiation (a) and Flow (6) dependencies]

7.4.3 Rule 9 - For Refined Order

9. For each refined order specified through answering question 8 or 9 in Step 4 between two activities A1 and A2 in contract negotiation or execution phases,

if $A_1 < A_2$ holds
then
If no AND Gateway between SE_{A2} and A_2 the
Add an AND Gateway between SE_{A2} and A_2
Endif
Link Intermediate Event, EE_{A1} to the AND Gateway between SE_{A2} and A_2
Endif

Illustration of Rule 8

There is a refined order for the business scenario that we have being considering for the running case as defined under question 8.1 DownPayToDist < Dist+Supp Contract Establishment. BY applying the rule 7 for as well for this requirement will further completes the BPDs that we have so far to the one shown in Fig. 52.

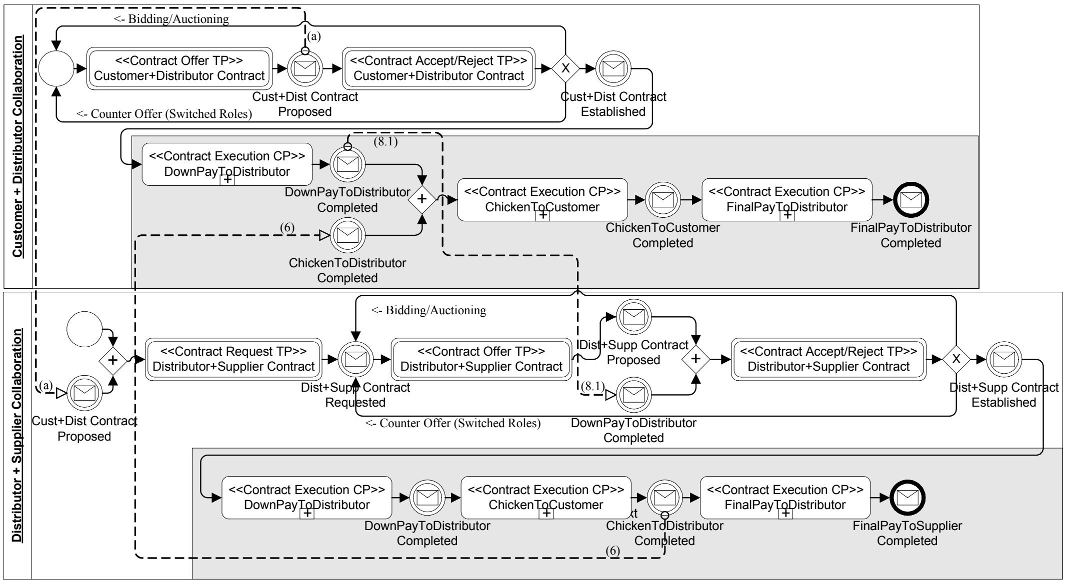


Fig. 52 Inter Connected BPDs for Customer+Distributor and Distributor+Supplier Collaborations [Refined (7.1), Negotiation (a) and Flow (6) dependencies]

7.5 Deadlock Prevention Rules

In this category we are proposing set of rules that can be applied to avoid any possible starvation like situations for BPMN events in the final result and to get rid of conflicting business requirements that may capture during Designers Assistant steps in (see Chapter 6).

Though we have defined these dead lock prevention rules as the last category of production rule system they have to be observed by users during BP^3 Designers Assistant question answering sessions.

7.5.1 For Economic Events of a Duality (Within a Pool)

Rule 10 is to avoid cyclic starvation among three collaboration activities in execution phase.

10. Assume A1, A2 and A3 are three collaboration activities in execution phase and user has specified A1 < A2 and A2 < A3.

If A1 < A2 and A2 < A3 holds
Then avoid getting A3 < A1 in **Table 5** *of* (see Chapter 6) *for trust dependency*
endif

Rule 10 is to avoid cyclic starvation in execution phase among more than three collaboration activities. Here not only what user has entered so far, but possible orders that can be deduced also have to take into account. For instance, if user has entered A1 < A2 and A2 < A3 for three activities in execution phase, by transitivity it is possible deduce that A1 < A3.

11. Assume A1, A2 and A3 are three collaboration activities in execution phase. The A1 < A2 and A2 < A3 are either user entered or deduced from specified order so far. Then,
If A1 < A2 and A2 < A3 holds then
avoid getting A3 < A1 in Table 3 of (see Chapter X) *for trust dependency*
endif

7.5.2 For Inter-Collaborations (Between Pools)

Rule 12 is to avoid cyclic starvation among three collaboration activities in contract negotiation and execution phases of different pools.

12. Assume A1, A2 and A3 are three activities and user has specified A1 < A2 and A2 < A3 or it can be deduced from governing rules for collaboration and transaction patterns introduced in (see Chapter 6).

If A1 < A2 and A2 < A3 holds then
avoid getting A3 < A1 in any step of Designers Assistant
endif

Rule 12 is to avoid cyclic starvation in contract negotiation and execution phases but among more than three activities. Here not only dependencies mentioned in rule 10 but possible orders that can be deduced have to take into account. For instance, if user has entered A1 < A2 and A2 < A3 for three activities in execution phase, by transitivity it is possible deduce that A1 < A3.

13. Assume A1, A2 and A3 are three collaboration activities in execution phase. The A1 < A2 and A2 < A3 are either user entered or deduced from specified order so far. Then,

If A1 < A2 and A2 < A3 holds then
avoid getting A3 < A1 in Table 3 of (see Chapter X) *for trust dependency*
endif

8.6 Conclusion

Here, we have defined set of production rules that can be applied on dependency requirements captured through BP^3 Designers Assistant. There are four categories of these rules: for Binary Collaborations, for Inter-Collaborations, for Reduction and for Deadlock Prevention.

The Deadlock Prevention rules have to observe during the dependency requirement capturing steps. The other three categories are to be applied after completion of first fours steps of BP^3 methodology.

However, thorough investigations are needed to see how defined rules systems behave with complex and exceptional cases.

8 Concluding Remarks and Further Research Directions

In this chapter we discuss benefits and limitations of the proposed approach. This work is wrapped up by giving some indications of possible improvements and directions for future research.

8.1 Concluding Remarks

8.1.1 A Unified Framework

Integrating process and business models poses a number of problems along several dimensions. Differences in focus, abstraction level, and domain give rise to different types of discrepancies that must be resolved. Process models may be seen as describing the communicative world, in particular how agents establish and fulfils obligations, while business models depict the social/institutional world where economic relationships such as 'ownership' holds and actions such as transfer of economic resources occurs.

The main contribution of our work is a unified framework to facilitate the integration of business models and process models in e-Commerce. The approach suggested bridges the gap between the communicative aspects of a process model and the social/institutional aspects of a business model. A key assumption of this approach is that an enterprise can be viewed as a set of co-operating trading partners that establish, modify, cancel and fulfil commitments and contracts. In carrying out these activities, partners rely on so-called pragmatic acts (speech acts), which are actions that change the universe of discourse when a speaker utters them and a recipient grasps them.

Besides facilitating process and business model integration, the proposed framework offers several benefits:

Simplified Analysis and Design. It will be easier for business users to participate in analysis and design if they are able to express themselves using concepts that have a business meaning (like propose, declare, commit, cancel) instead of using technical concepts like message structures and state machines. Furthermore, the specification of a pragmatic action is simple, as it can be viewed as filling in a template.

Technology Independence. An approach based on pragmatic actions makes it possible to abstract business semantic conversations out of technical messaging protocols, so that pragmatic actions can be used with any technical collaboration protocol (UMM BCP [85], ebXML BPSS[22], BPEL4WS [6], etc). Thus, pragmatic actions provide a clean interface to collaboration protocols.

8.1.2 BP^3 Designers Assistant

Another 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 Designers 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 focusing on one problem at a time.

Seamless transition from analysis to realization. Using the Designers 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.

Users of BP^3 Designers Assistant. In this thesis, we have mainly considered e-Commerce systems designers as users of the proposed BP^3 approach. However, the proposed framework has the potential to be extended for requirement capturing from domain experts, for technical realization with business process execution languages by systems developers as well.

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 BP^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 BPMN activities and events, for modelling processes. Thus, the specified process models can be readily mapped into executable XML process specifications. Another advantage of using communicating state machines is that each state machines corresponds to a Pragmatic Action, based on language action perspectives, which makes it easy to understand.

8.2 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 economic and marketing theories for a more in-depth analysis of real use in deciding economical and business feasibility of the solution being developed. Work in the direction of identifying potential usage of business models has to be addressed.

2. *Wizard of the Designers 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 including reasonably complex cases to evaluate its completeness and possible extensions for enhancing the quality of the final BPMN solutions. In the process of automation of the BP^3 Designers Assistant to generate BPMN Business Process Diagrams, proposed productions rules are the foundation.

4. Extension for primitive process patterns

In the thesis, a few transaction and collaboration patterns were introduced in Chapter 5 to model recurrent conversation patterns during Contract Negotiation and Contact Execution Phases. There is much work remaining to extend these primitive patterns to cover cancellations, breakdowns, negotiations, etc. The scope of processes could also be extended to handle additional phases in e-commerce, such as the ones discussed in [65], [14] and **[31]**.

5. Action Dependencies

We also introduce the notion of action dependencies in Chapter 6 for capturing relationships between the activities through which trading partners collaborate. Four kinds of dependencies are identified: flow, trust, control and negotiation dependencies. They can be stated declaratively, have a clear business motivation, and are used for the final derivation of a process model in the proposed BP^3 approach. A topic for further work is to investigate whether additional kinds of action dependencies are required.

6. Soundness and Completeness

A further line of future work is to examine the quality of the produced models, during which their completeness as well as their logical soundness should be investigated. While the work on completeness can primarily be done through empirical studies, the work on logical soundness can be supported by theoretical work like the one given in [10].

9 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", *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. Bergholtz M., Jayaweera P., Johannesson P. and Wohed P., "Reconciling Physical, Communicative and Social/Institutional Domains in Agent Oriented Information Systems - a Unified Framework", in Proceedings of *AOIS 2003 at the 22nd International Conference on Conceptual Modeling (ER 2003)-Chicago, Springer-Verlag*, LNCS 2814.
5. Bergholtz M., Jayaweera P., Johannesson P., Wohed P., "A pattern and dependency based approach to the design of process models", submitted to the *23rd,Int. Conference on Conceptual Modeling (ER'2004)*
6. *Business Process Execution Language for Web Services*, OASIS WS-BPEL Technical Committee, Valid on 20040404, http://www.ebpml.org/bpel4ws.htm
7. *Business Process Execution Language for Web Services*, OASIS WS-BPEL Technical Committee, Valid on 20040419, http://www.ebpml.org/bpel4ws.htm
8. *Business Process Management Initiative (BPMI)*, Valid on 20040419, http://www.bpmi.org/
9. *Business Process Modeling Notion (BPMN) Information*, Valid on 20040419, http://www.bpmn.org/
10. Chrzastowski-Wachtel P., Benatallah B., Hamadi R., O'Dell M., and Susanto A., "A Top-Down Petri Net-Based Approach for Dynamic Workflow Modeling", in Proc. of *Business Process Management Int. Conf.*, W. van der Aalst, A. ter Hofstede and M. Weske (Eds.), LNCS 2678, pp.336-353, Springer, 2003
11. CIMOSA: "*A Primer on key concepts, purpose and business value*, Valid on 20040419, http://www.cimosa.de/cimosa/Primer/primer0.htm
12. CYC project at MMC, Valid on 20030404, http://www.cyc.com/publications.html
13. DEMO literature, Valid on 20030404, http://www.demo.nl/DEMO/literatuur.htm
14. Dietz J., "Deriving Use Cases from Business Process Models", in Proc. of *22nd Int. Conference on Conceptual Modeling (ER'2003)*,Chicago, Illinois, USA. LNCS 2813, pp. 131-143, 2003
15. Dietz J., "Modelling Communication in Organisations", in Linguistic Instruments in Knowledge Engineering, Ed. R.v.d.Riet, pp 131-142, Elsevier Science Publishers, 1992
16. 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
17. Dietz J.L.G., "The What and the Why of Modeling Business Processes", R.M. van Es, A. Post (Eds.), "*Dynamic Enterprise Modeling*", Kluwer Bedrijfsinformatie, Deventer, 1996
18. 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)*
19. Dignum F. and Weigand H., "Communication and deontic logic", *Information Systems, Correctness and Reusability*, World Scientific, Singapore, 1995
20. Dignum F. and Weigand H., "Modelling Communication between Cooperative Systems", in proceedings of *the 7th Conference of Computer Advanced Information System Engineering (CaiSE)*, Lecture Notes in Computer Science, Springer Verlag, 1995
21. DoD Enterprise Model, Topic(s): "*Government BPR Project Reports, Process Modeling & Analysis*", The Department of Defense, Valid on 20030404, http://www.c3i.osd.mil/bpr/bprcd/0035.htm
22. ebXML Specification and Technical Documents, Valid on 20030404, http://www.ebxml.org/specs/index.htm
23. Enterprise Project at University of Edinburgh, Valid on 20030404, http://www.aiai.ed.ac.uk/~entprise/
24. Fowler M., "*Analysis Patterns: Reusable Object Models*", Addison-Wesley, 1997
25. 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
26. Fox M.S., Gruninger M., "On Ontologies and Enterprise Modeling", *International Conference on Enterprise Integration Modelling Technology*, Springer-Verlag 1997
27. 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)
28. Genesereth M. R., & Nilsson N. J., "*Logical Foundations of Articial Intelligence*", Kaufmann, Los Altos, CA, 1987
29. Goldkuhl G., "Generic business frameworks and action modelling", In proceedings of *Conference Language/Action Perspective*, Springer Verlag, 1996

30. Goldkuhl G., "Generic Business Frameworks and Action Modelling", *First International Workshop on Communications Modelling - The Language/Action Perspective*, Springer Verlag 1996
31. Goldkuhl G., "The six phases of business processes - business communication and the exchange of value", accepted to the *twelth biennial ITS conference*, Stockholm, 1998
32. Gordijn J., Akkermans J. M. and Vliet J. C., "Business Modelling, is not Process Modelling", Proc. of the 1th International Workshop on Conceptual Modeling Approaches for e-Business (eCOMO'2000), held in conjunction with the 19th International Conference on Conceptual Modeling (ER'2000), Salt Lake City, Utah, USA
33. Gordijn J., Akkermans J. M. and Vliet J. C.: "Business Modeling is not Process Modeling", *eCOM2000 workshop, 19th International Conference on Conceptual Modeling 2000*
34. 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
35. 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
36. Gruber T., "A Translation Approach to Portable Ontology", *Knowledge Acquisition*, 1993.
37. 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
38. Guarino N., "Formal Ontology and Information Systems", *1st International Congerence on Formal Ontology in Information Systems (FOIS '98)*, 1998 Trento, Italy
39. 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
40. Haugen B., Fletcher T., "*Multi-Party Electronic Business Transactions*", Valid on 20040419, http://www.ebpml.org/archive1.htm
41. Holbrook M. B., "*Consumer Value*", ISBN 0-415-19192, London Routledge 1999
42. Integration Definitions, Valid on 20030404, http://www.idef.com/Downloads/
43. Jayaweera P., "A Methodology to Generate e-Commerce Systems: A Process Pattern Perspective (P 3)", Licentiate of Philosophy Thesis March 27th, 2002
44. Jayaweera P., "Mapping Function to Generate BML Process Model - A Technical Report", Valid on 20030404, http://www.dsv.su.se/~prasad/html/MapFun.doc
45. Jayaweera P., Johannesson P. and Wohed P., "Collaborative Process Patterns for e-Business", *ACM SIGGROUP Bulletin*, 2001/Vol 22, No.2
46. 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
47. 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
48. 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
49. 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
50. 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
51. 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
52. 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
53. Larman C., "*Applying UML and patterns: an introduction to object oriented analysis and design*", ISBN 0-13-74880-7
54. Lenat D., and Guha R.V., "*Building Large Knowledge Based Systems: Representation and Inference in the CYC Project*", Addison Wesley Pub. Co 1990
55. 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
56. Lokhorst G. J. C., "Ernst Mally's *Deontik* (1926)". *Notre Dame Journal of Formal Logic*, 40 (2): 273-282, 1999 (2001). ISSN 0029-4527
57. Mally E., "*Deontic Logic*", Valid on 20030404, http://mally.stanford.edu/deontic.html
58. Malone et al.: "Towards a handbook of organizational processes", MIT eBusiness Process Handbook, Valid on 20030404, http://ccs.mit.edu/21c/mgtsci/index.htm
59. Martin, J., Odell, J.: *Object-Oriented Methods. A Foundation,* Prentice Hall 1994

60. McCarthy W. E., "The REA Accounting Model: A Generalized Framework for Accounting Systems in a Shared Data Environment", *The Accounting Review* 1982
61. 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
62. Moore S., "*Formal Language for Business Communication*", Valid on 20030404, http://www-personal.umich.edu/~samoore/research/flbc/.
63. Ohmae K., "*The mind of the strategist: The Art of Japanese Business*", New York, 1982
64. *OMG Unified Modeling Language Specification,* Version 1.4, September 2001, ftp://ftp.omg.org/pub/docs/formal/01-09-67.pdf
65. *Open-EDI phases with REA*, UN-Centre for Trade Facilitation and Electronic Business, Valid on 20040419, http://www.unece.org/cefact/docum/download/02bp_rea.doc
66. Parrow Joachim, "An Introduction to the π-calculus", Valid on 20030404, http://user.it.uu.se/~joachim/intro.ps
67. Perry R., "*Realms of value*", ISBN 9-910-032287, Cambridge, Mass. Harvard Univ. Press 1954
68. Porter M. E., "Competitive Advantages. Creating and Sustaining Superior Performance" The Free Press 1998
69. QFD Institute - The official source for QFD, Valid on 20030404, http://www.qfdi.org/transact.htm
70. RosettaNet Standards, Valid on 20030404, http://www.rosettanet.org/
71. SAP Business Maps, Valid on 20030404, http://www.sap.com/businessmaps/
72. Scott A. Moore, "A foundation for flexible automated electronic communication", *Information Systems Research, 12:1 (March 2001)*
73. SDL Standards, Valid on 20030404, http://www.sdl-forum.org/Publications/Standards.htm
74. Searle J. R., "A taxonomy of illocutionary acts", K. Gunderson (Ed.), *Language, Mind and Knowledge*, Minneapolis: University of Minnesota, 1975.
75. Significant *REA Model* papers, Valid on 20030404, http://www.msu.edu/user/mccarth4/paplist1.html
76. Spaccapietra S., Parent C. and Dupont Y., "Model Independent Assertions for Integration of Heterogeneous Schemas", *The VLDB Journal*, vol. 1, no.2, pp. 81-126, 1992
77. Taylor J. R., Groleau C., Heaton L. and VAN EVERY E. J., "*The Computerization of Work : A Communication Perspective*", Thousand Oaks CA: Sage
78. 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
79. Taylor, J. R., Groleau C., Heaton L. and VAN EVERY E. J.: "*The Computerization of Work : A Communication Perspective*", Thousand Oaks CA: Sage
80. Techniques and Methodology Working Group (TMWG), Valid on 20030404, http://www.gefeg.com/tmwg/
81. The Internet Economy Indicators, Indicators Report, Valid on 20030404, http://www.internetindicators.com/keyfindings.html
82. *The workflow patterns*, Valid on 20040419, http://tmitwww.tm.tue.nl/research/patterns/
83. TOVE project at University of Toronto, Valid on 20030404, http://www.eil.utoronto.ca/tove/ontoTOC.html
84. *UN/CEFACT - Document Download*, Valid on 20030404, http://www.unece.org/cefact/docum/download/
85. *UN/CEFACT Modeling Methodology (UMM) Revision 10*, Valid on 20030404, http://www.untmg.org/doc_bpwg.html
86. Uschold M. and King M., "*Towards a Methodology for Building Ontologies*", (also available from AIAI as AIAI-TR-183), 1995
87. van der Aalst W.M.P., ter Hofstede A.H.M., Kiepuszewski B., and Barros A.P., "Workflow Patterns", *Distributed and Parallel Databases*, 14(3), pages 5-51, July 2003
88. van Dijk A., "Contracting Workflow and Protocol Patterns", *in Proc. of Business Process Management Int. Conf.*, W. van der Aalst, A. ter Hofstede and M. Weske (Eds.), LNCS 2678, pp.152-167, Springer, 2003
89. Visuera, Valid on 20040404, http://www.visuera.com/
90. Wagner G., "The Agent-Object-Relationship Meta-Model: Towards a Unified View of State and Behaviour", *Information Systems 2003*, Valid on 20030404, http://tmitwww.tm.tue.nl/staff/gwagner/AORML/AOR.pdf
91. Wåhlander C., Nilsson M., and Törnebohm J, "Visuera Process Manager - Introduction", Copyright Visuera, 2001
92. Web Services Working Groups, *Web Services Choreography Working Group*, Valid on 20030404, http://www.w3c.org/2002/ws/chor/
93. 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
94. Weigand H. Verharen E. and Dignum F. "Dynamic business models as a basis for interoperable transaction design", *Information Systems*, 1997
95. 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
96. 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

97. Winograd T. and Flores F., "Understanding Computers and Cognition: A New Foundation for Design", Ablex, Norwood, N.J 1986
98. Yu L., and Schmid B. F., "A conceptual framework for agent-oriented and role based workflow modelling", In G. Wagner and E.Yu, editors, Proc. of the 1st Int. Workshop on Agent-Oriented Information Systems, 1999

Appendix A

In the Appendix A, all the steps of BP^3 Designers Assistant and responses that can be received with respect to the running case description are recollected. Finally, the solution generated by BP^3 approach has been documented as BPMN Business Process Diagrams.

1 BP^3 Designers Assistant Steps for Fried Chicken Case

In the following sections all responses received for successive steps of BP^3 Designers Assistant have been documents.

1.1 Step 1 – Business model for Fried Chicken Case

1. Who are the Business Partners?

Customer (Cust)

Distributor (Dist)

Chicken Supplier (Supp)

Carrier (Carr)

2. What are the Economic Resources?

Money

Chicken

Delivery

3.What are the Economic Events?

Here, we have used following quadruplets for rows of the [**Table 3**] to document relevant responses for the fired chicken case being modeled, i.e. {<EconomicEventName>, <EconomicResource>, <From>, <To>}.

{DownPayToDistributor, Money, Customer, Distributor}

{FinalPayToDistributor, Money, Customer, Distributor}

{ChickenToCustomer, Chicken, Distributor, Customer}

{DownPayToChickenSupplier, Money, Distributor, ChickenSupplier}

{ FinalPayToChickenSupplier, Money, Distributor, ChickenSupplier }

{ChickenToDistributor, Chicken, ChickenSupplier, Distributor}

{PayToCarrier, Money, Distributor, Carrier}

{DeliveryToDistributor, Delivery, Carrier, Distributor}

4. What are the Dualities?

We group identified economic events above into different dualities with appropriate names. A duality groups all economic events that belongs to a specific deal between two business partners in a given business scenario.

<Duality> = {<EconomicEvent$_1$>,
<EconomicEvent$_2$>,
…,<EconomicEvent$_n$>}

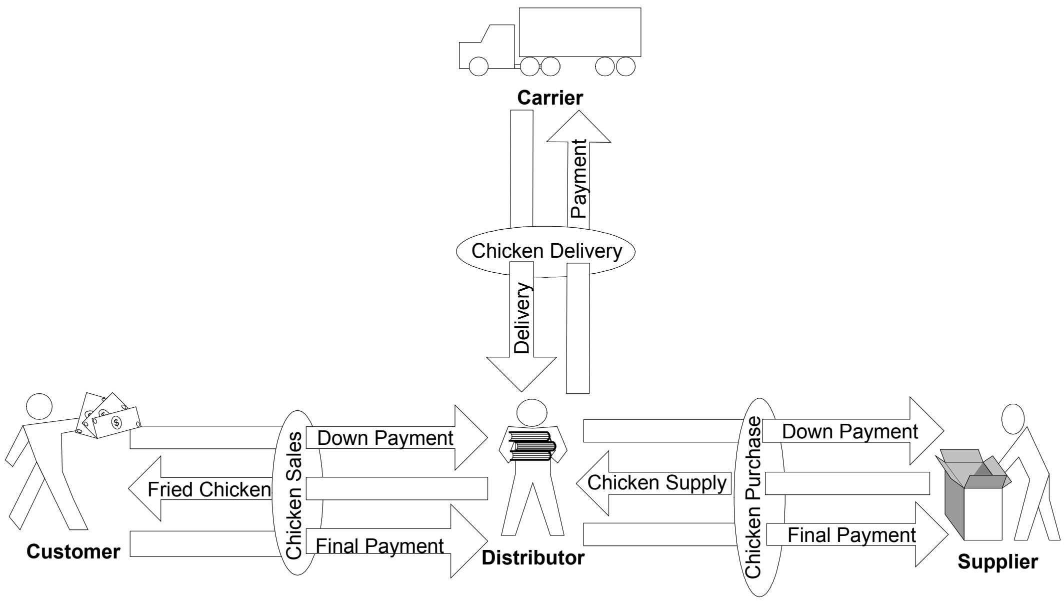
Above template has been used to document captured dualities in [**Table 4**] here.

1. Chicken Sales = {DownPayToDist,
FinalPayToDist,
ChickenToCust}

2. Chicken Purchase = {DownPayToSupp,
FinalPayToSupp,
ChickenToDist}

3. ChickenDelivery = {DeliveryToDist,
PayToCarr}

Having received responses for the four questions in the Step 1 of BP^3 Designers Assistant, the relevant business model can be generated as below for the Fried Chicken introduced.



1.2 Step 2 – Execution Phase Order for Fried Chicken Case

In the Step 2 of the Designers Assistant, trust and flow dependencies between economic events are to capture. For this purpose, dependencies are expressed as below where LHS < RHS means LHS has to be performed before RHS. Further < can be suffixed with '*t*' or '*f*' to indicate whether it is a trust or flow dependency respectively. For detail interface see [**Table 5**].

5. *Specify* Flow *and* Trust Dependencies *between economic events.*

1. DownPayToDistributor $<_t$ ChickenToCustomer
2. DownPayToDistributor $<_f$ DownPayToChickenSupplier
3. FinalPayToDistributor $<_f$ FinalPayToChickenSupplier
4. FinalPayToDistributor $<_f$ PayToCarrier
5. ChickenToCustomer $<_t$ FinalPayToDistributor
6. ChickenToDistributor $<_f$ ChickenToCustomer
7. ChickenToDistributor $<_t$ FinalPayChickenSupplier
8. DeliveryToDistributor $<_f$ ChickenToCustomer
9. DeliveryToDistributor $<_t$ PayToCarrier
10. DownPayToChickenSupplier $<_t$ ChickenToDistributor

1.3 Step 3 – Contract Negotiation Phase Order for Running Case

In this step, selection of contract negotiation patterns and inter-related communication between different binary collaborations at contract negotiation phase are to be decided on. For the fried chicken case, answers are shown below.

6. For each binary Business Collaboration (for each Duality), ask whether

a) *a quotation already exists when the binary collaboration starts, or*
b) *the binary collaboration is started by a partner requesting a quotation.*

6.1	***(a)***	*Does a quotation already exist when the Cust+Dist collaboration starts?, or*	*Yes*
	(b)	*Is the Cust+Dist collaboration started by a partner requesting a quotation?*	-

6.2 (a)	*Does a quotation already exist when the Dist–Supp collaboration starts?, or*	*No*
(b)	*Is the Dist–Supp collaboration started by a partner requesting a quotation?*	*Yes*
6.3 (a)	*Does a quotation already exist when the Dist+Carr collaboration starts?, or*	*No*
(b)	*Is the Dist+Carr collaboration started by a partner requesting a quotation?*	*Yes*

Having decided contract negotiation patterns from the responses received above, the [**Table 6**] is used to capture the required order between different communication activities at contract negotiation phase. Here for each cell[i,j] at i^{th} row and j^{th} column in the table, following natural language question can be posed.

7. Do you require $Contract_i$ *Request/Offer/Establish between* P_{i1} *and* P_{i2} *to be completed before* $Contract_j$ *Request/Offer/Establish between* P_{j1} *and* P_{j2}*?*

Where, P_{i1} and P_{i2} are business partner between whom $Contract_i$ is to be established and P_{j1} and P_{j2} are business partner between whom $Contract_j$ is to be established. Here all possible combinations are checked except for the contract negotiation activities that belongs to same dualities, i.e. except the shaded area in the [**Table 6**]. The answers received for the running case can be documented as follows similar to the economic events order above.

a) Cust+Dist Contract-Offer < Dist + Supp Contract-Request

b) Cust+Dist Contract-Offer < Dist + Carr Contract-Request

c) Dist + Supp Contract-Offer < Cust+Dist Contract-Establish

d) Dist + Carr Contract-Offer < Cust+Dist Contract-Establish

1.4 Step 4 – Refined Order for Running Case

For each pair of flow dependencies identified in Step 2, ask following question.

8. For each pair of Economic Events *$<EE_i, EE_j>$ (see Table 3), such that $EE_i <_f EE_j$: Is it required to perform EE_i before making a contract acceptance for EE_j, (i.e. a Contract Establishment between the* Agents *in EE_j)?*

There are following five flow dependencies for the fired chicken case.

1. DownPayToDistributor $<_f$ DownPayToSupplier
2. FinalPayToDistributor $<_f$ FinalPayToSupplier
3. FinalPayToDistribtutor $<_f$ PayToCarrier
4. ChickenToDistributor $<_f$ ChickenToSupplier
5. DeliveryToDistributor $<_f$ ChickenToSupplier

For each of them pose the following questions.

8.1	*Must DownPayToDist be done before establishment of Dist+Supp Contract?*	**Yes**
8.2	*Must FinalPayToDist be done before establishment of Dist+Supp Contract?*	**No**
8.3	*Must FinalPaytToDist be done before establishment of Dist+Carr Contract?*	**No**
8.4	*Must ChickenToDist be done before establishment of Dist+Cust Contract?*	**No**
8.5	*Must DeliveryToDist be done before established of Dist+Cust Contract?*	**No**

We have captured only one refined order for flow dependencies as,

8.1 DownPayToDist $<$ Dist+Supp Contract-Establish

For each economic event triplet ask the following question.

*9. For Economic Events triplets in [***Table 3***], EE_i, EE_j, EE_k, such that $EE_i <_t EE_j$ and $EE_k <_f EE_J$: Is it required to perform EE_i before making a contract acceptance for EE_k (i.e a Contract Establishment between the Agents in EE_k)?*

For the running case there are following pairs satisfying the above requirement.

1. DeliveryToDistributor $<_t$ PayToCarrier and FinalPayToDistributor $<_f$ PayToCarrier

2. ChickenToDistributor $<_t$ FinalPayToSupplier and FinalPayToDistributor $<_f$ FinalPayToSupplier

3. DownPayToDistributor $<_t$ ChickenToCustomer and ChickenToDistributor $<_f$ ChickenToCustomer

4. DownPayToDistributor $<_t$ ChickenToCustomer and DeliveryToDistributor $<_f$ ChickenToCustomer

For the above four pairs, following four questions has to pose.

9.1 *Must DeliveryToDist be done before establishment of Cust+Dist Contract?* **No**

9.2 *Must ChickenyToDist be done before establishment of Cust+Dist Contract?* **No**

9.3 *Must DownPayToDist be done before establishment of Dist+Supp Contract?* **No**

Dist+Supp Contract?

9.4 *Must DownPayToDist be done before establishment of Dist+Carr Contract?* **Yes**

Here we have captured another refined trust dependency as,

9.4 DownPayToDist < Dist+Carr Contract-Establish

Finally, we are to capture control dependencies between different contract negotiation and contract execution activities of intre-binary collaborations through the matrix [**Table 7**]. However there are no such dependencies of the running case and therefore we skip the question 10 discusses in Section 6.2.4, here.

These are the only responses we have received for the running case and in the next section we illustrate how these dependencies can be used to complete the final BPMN business process diagrams (BPD) for the fried chicken case.

2 BPMN BPD Generation for Fried Chicken Case

From the Step 1 above we have identified four dualities for the running case.

1. Chicken Sales (D_1)
2. Chicken Purchase (D_2)
3. Chicken Delivery (D_3)

As the first step in the BPMN BPD generation we have to apply the rule 1 stated below from the production rule system.

1. For each duality D_i
Create BPMN pool that will contain BPD for D_i
Name an empty pool with "P_{i1} + P_{i2} Collaboration" where P_{i1} and P_{i2} are the partners involved in the duality, D_i

This results the three named pools listed below as in [Fig. 53].

1. Customer+Distributor Collaboration
2. Distributor+Supplier Collaboration
3. Distributor+Carrier Collaboration

Then by applying rule 2 (shown below) of the production rule system, the process patterns for contract negotiation phases (white areas of pools in the Fig. 53) for above collaborations can be decided. For this responses received for question 6 of the Step 2 above have to be used.

2. For each identified duality, D_i
   If answer to question (6) in Step 3 is (a) then
   Start BPD by instantiating a "Contract Establishment" pattern in "Pi1+Pi2 Collaboration" pool as in Fig. 32.
   else
   Start BPD by instantiating a "Contract Propose" pattern in "Pi1+Pi2 Collaboration" pool as in Fig. 33.
   endif

By applying above rule on the three dualities identified, contract negotiation process patterns can be decided as in the white areas of the [Fig. 53].

Then by applying rule 3, rule 4 and rule 5 on identified economic events at Step 1 and execution order of economic event at Step 2 of BP^3 Designers Assistant, process pattern for execution phases can be completed as in shaded areas of [Fig. 53].

3. For each duality
Instantiate "Execution Collaboration" patterns in parallel for each $A_{i1}, A_{i2}, \ldots, A_{in}$ as in Fig. 40 , in the pool "$P_{i1} + P_{i2}$ Collaboration" for D_i connecting success path of instantiated Collaboration pattern from rule 2.

After applying above rule 3 for the identified three dualities, there will be three contract execution activities in parallel for Customer+Distributor (as in [Fig. 40]) and Distributor+Supplier Collaborations and only two contract execution activities for Distributor+Carrier Collaboration.

In the Step 2 where execution order is captured, we have identified following trust dependencies.

1. DownPayToDistributor $<_t$ ChickenToCustomer
2. ChickenToCustomer $<_t$ FinalPayToDistributor
3. ChickenToDistributor $<_t$ FinalPayChickenSupplier
4. DeliveryToDistributor $<_t$ PayToCarrier
5. DownPayToChickenSupplier $<_t$ ChickenToDistributor

Now, the following rule 4 has to apply on the execution activities instantiated in parallel initially.

4. For each trust dependency order specified above between say collaboration activities A_1 and A2 for two economic events,

If $A_1 < A_2$ holds then

If no AND Gateway between SE_{A2} and A_2 then

Add a AND Gateway between SE_{A2} and A_2
endif
Link Intermediate Event, EE_{A1} to the AND Gateway between SE_{Ay} and A_y
endif

However, by applying above rule 4 over trust dependencies that have been identified in Step 2 of Designers Assistant above execution activities initially arranged in parallel are over loaded with more and more triggering events as in [Fig. 42].

The over loaded parallel activities in contract execution phases is simplified into more comprehensive and compact forms by applying rule 5 over and over again.

5. Assume A1 and A2 are two activities in execution phase of a binary collaboration. EE_{A1} and SE_{A2} are resulting event of A1 and triggering event of A2 respectively.

If EE_{A1} AND SE_{A2} < A2 holds (in Original diagram) then
If Ax < A2 already holds (in the Reduced diagram) then
If no "AND Gateway" between Ax and A2 [ref. 5.a]
Add "AND Gateway" between Ax and A2
Link A1 to "AND Gateway" between Ax and A2
Else [ref. 5.b]
Link A1 to "AND Gateway" between Ax and A2
Endif
Else
if no A1 < Ay (in the Reduced diagram) then [ref. 5.c]
Link A2 after A1
Else
if no "AND Gateway" between A1 and immediate event to proceed
A1 (in the Reduced diagram) then [ref. 5.d]
Add "AND Gateway" between A1 and immediate event to proceed A1 (in the Reduced diagram)
Link A2 to "AND Gateway" between A and immediate event to proceed A1 (in the Reduced diagram)
Else [ref. 5.e]
Link A2 to "AND Gateway" between A and immediate event to Proceed A1 (in the Reduced diagram)
Endif
Endif
Endif
Endif

After reducing the execution activities of different binary collaborations take the form as in shaded areas of the [Fig. 53].

There are two refined order for flow and trust dependencies captured in Step 4 through question 8.1 and 9.4 expressed below.

8.1 DownPayToDist < Dist+Supp Contract-Establish

9.4 DownPayToDist < Dist+Carr Contract-Establish

In [Fig. 53] the two dotted arrows are labeled with 8.1 and 9.4 to represent the resulting inter-collaboration communication. The completion event of Customers down payment has been communicated to AND Gateway immediately before Distributor+ChickenSupplier Contract Accept/Reject activity. See Customer+Distributor and Distributor+ChickenSupplier Collaboration pools in [Fig. 53].

Similarly, completion event of Customer down payment has been communicated to AND Gateway immediately before Distributor+Carrier Contract Accept/Reject activity. Customer+Distributor and Distributor+Carrier Collaboration pools in [Fig. 53].

The rule that completes necessary inter-collaboration communication is resulted from the application of the following rule-9.

9. For each refined order specified through answering question 8 or 9 in Step 4 between two activities A1 and A2 in contract negotiation or execution phases,

if $A_1 < A_2$ holds
then
If no AND Gateway between SE_{A2} and A_2 the
Add an AND Gateway between SE_{A2} and A_2
Endif
Link Intermediate Event, EE_{A1} to the AND Gateway between SE_{A2} and A_2
Endif

Also there are following flow dependencies captured in Step 2.

1. DownPayToDistributor $<_f$ DownPayToSupplier (2)
2. FinalPayToDistributor $<_f$ FinalPayToSupplier (3)
3. FinalPayToDistribtutor $<_f$ PayToCarrier (4)
4. ChickenToDistributor $<_f$ ChickenToSupplier (6)
5. DeliveryToDistributor $<_f$ ChickenToSupplier (8)

For the above flow dependencies, application of the following rule 7 completes inter-collaboration communication at contract execution phases. See the relevant numbers at the end of above dependencies also in the [Fig. 53].

7. For each flow-dependency order specified in the [**Table 5**] of (see Chapter 6) between collaboration activity A1 and A2 for two economic events,

if $A_1 < A_2$ holds then
If no AND Gateway between SE_{A2} and A_2 the
Add an AND Gateway between SE_{A2} and A_2
endif
Link Intermediate Event, EE_{A1} to the AND Gateway between SE_{A2} and A_2
endif

However notice that the flow dependency, DownPayToDistributor $<_f$ DownPayToSupplier (2) has been overridden by the refined order, 8.1 DownPayToDist < Dist+Supp Contract-Establish that we captured above.

The following reduction rule-6 removes redundant such inter-collaboration communication leaving the stronger requirement to hold in the final solution.

6. Assume A1 and A2 are two activities of a binary collaboration and A3 is another activity from a different binary collaboration. Assume that within the binary collaboration mentioned first, A1 < A2 holds, there is an action dependency such that A3 < A2 and a captured refine order A3 < A1.

If EE_{A3} < A2 AND EE_{A3} < A1 holds (in Original diagram) then
If EE_x AND SE_{A2} < A2 then
(EE_x is any other event A2 depends on)
Remove the link of EE_{A3} to AND Gateway between SE_{A2} and A2
Else(no more events that A2 depends on)
Remove the link of EE_{A3} to AND Gateway between SE_{A2} and A2
Remove AND Gateway between SE_{A2} and A2
Link SE_{A2} to A2 directly

Endif

Endif

Next, we have to apply the rule-8 expressed below for control and negotiation dependencies. For the running case there are no any control dependency requirements.

8. For each control or negotiation dependency order specified between two activities A1 and A2 in contract negotiation phase or execution phase,

if $A_1 < A_2$ *holds then*
If no AND Gateway between SE_{A2} *and* A_2 *the*
Add an AND Gateway between SE_{A2} *and* A_2
endif
Link Intermediate Event, EE_{A1} *to the AND Gateway between* SE_{A2} *and* A_2
endif

The above rule-8 has to apply on following negotiation dependencies captured in Step-3 of the Designers Assistant.

a) Cust+Dist Contract-Offer < Dist + Supp Contract-Request

b) Cust+Dist Contract-Offer < Dist + Carr Contract-Request

c) Dist + Supp Contract-Offer < Cust+Dist Contract-Establish

d) Dist + Carr Contract-Offer < Cust+Dist Contract-Establish

Resulting inter-collaboration communication with the application of rule-9 can be seen in the [Fig. 53] with relevant alphabetic labels as in above negotiation dependencies.

Finally with the application of all relevant rules on the captured dependency requirements results the completed BPNM business process diagrams for all dualities identified with necessary inter-collaboration communications as in [Fig. 53].

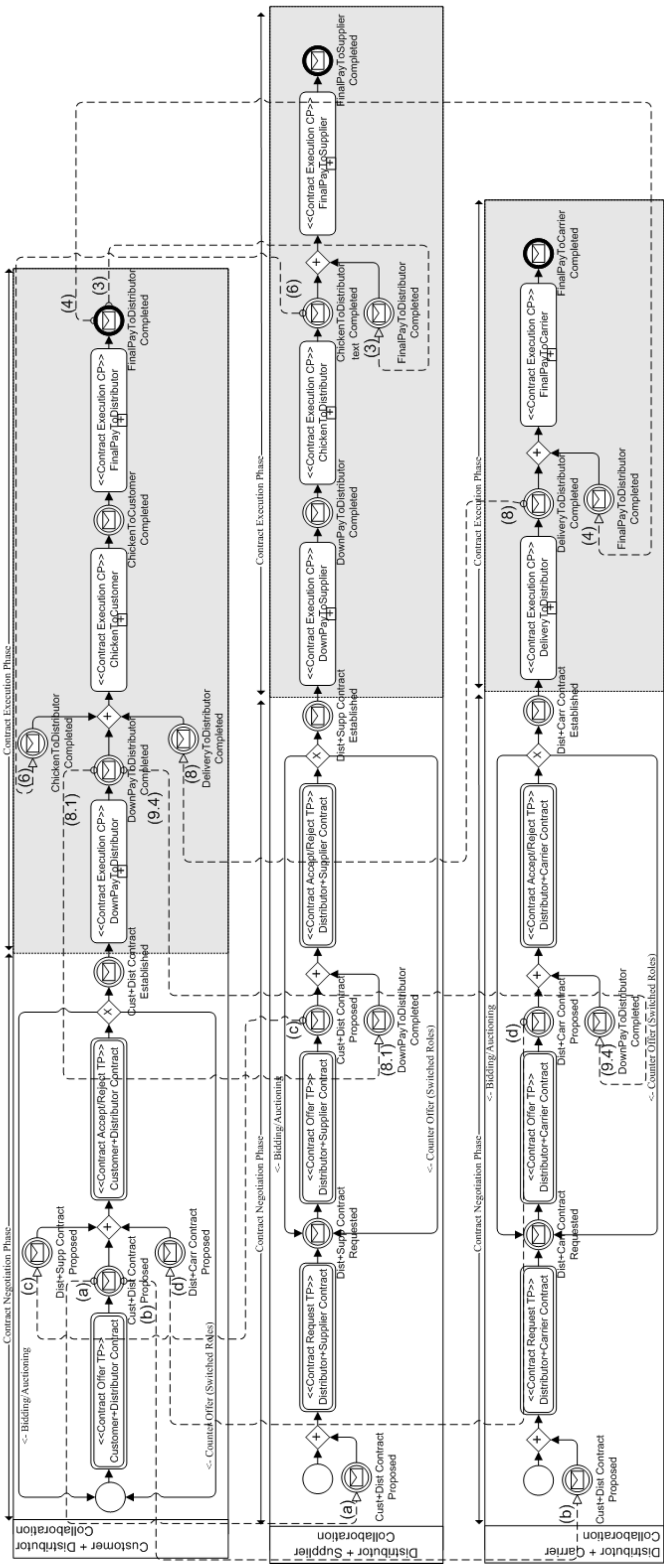


Fig. 53 BPMN Business Process Diagram for Running Case