Modelling Institutional, Communicative, and Physical Domains in Agent Oriented Information Systems

Maria Bergholtz, Prasad Jayaweera, Paul Johannesson, Petia Wohed

Department of Computer and Systems Sciences
Stockholm University and The Royal Institute of Technology
Forum 100, SE-164 40 Kista, Sweden

e-mail:{maria,prasad,pajo,petia}@dsv.su.se

Abstract. One role of a business system is to provide a representation of a Universe of Discourse, which reflects its structure and behaviour. An equally important function of the system is to support communication within an organisation, by structuring and co-ordinating the actions performed by the organisation's agents. These two roles of a business system may be represented in terms of business and process models, i.e. separating the declarative aspects from the procedural control flow aspects of the system. Although this separation of concerns has many advantages, the differences in representation techniques and focus of the two model types constitute a problem in itself. Abstracting business semantics out of, for instance, technical messaging protocols pose severe problems for business analysts. The main contribution of this paper is a unified framework based on agent oriented concepts for facilitating analysis and integration of business models and process models in a systematic way. The approach suggested bridges the gap between the declarative and social/economic aspects of a business model and the procedural and communicative aspects of a process model in a technology independent manner. We illustrate how our approach can simplify business and process models integration, process specification, process pattern interpretation and process choreography.

1 Introduction

Agent oriented concepts, like agents, events, actions, commitments, etc, have recently been introduced in the area of information systems analysis and design, thereby extending the conceptual toolkit for object-oriented analysis and design. The driving force for this has been the need of adequate capture and representation of the semantics of business processes. Two main points, when arguing for this [22], have been that i) the state of an agent also includes mental components, such as beliefs and commitments, which are not captured by the existing conceptual frameworks available within object-oriented analysis, and ii) the communication between agents is realised through speech acts, which are application independent in contrast to the application specific, ad-hoc manner within the object-oriented paradigm.

Furthermore, addressing the business process analysis for e-Commerce, a distinction between business models and process models has been made [11]. A business model is concerned with value exchanges among business partners [11], while a process model focuses on operational and procedural aspects of business communication. 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 realisation.

In the business requirement capture phase, coarse-grained views of business activities as well as their relationships and arrangements in business collaborations are represented by means of business model constructs at an abstract level. In contrast, the specification of a process model deals with more fine-grained views of business transactions, their relationships and choreography in business collaborations. 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 modelling views 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 modelling views and their inter-relationships. It can also provide a basis for design guidelines that can assist in developing process models.

The purpose of this paper is to propose a unified framework integrating the contents of business models and process models. The framework is based on agent-oriented concepts, like agent, commitment, event, action, etc., [20]. We use ebXML [9] and UMM [2] as the basis of our framework, more specifically the UMM Business Requirements View (BRV) for business models and the UMM Business Transaction View (BTV) for process models. UMM BRV already includes a number of agent-oriented concepts, which we extend by adding a number of constructs for bridging business and process models, in particular speech acts. The work presented in this paper builds on [6] and [5], where speech act theory [19] and the language/action approach [7], [21] are used for analysing processes, as well as for clarifying the relationships between agents in business and process models, or for coordinating web services.

The rest of the paper is organised as follows. Section 2 gives an overview of related research and introduces informally the basic concepts. Section 3 introduces the UMM BRV and BTV. Section 4 contains the main contribution of the paper and presents the integrated framework. Section 5 illustrates two applications of the introduced framework, and the analysis and design of business process patterns. Section 6 introduces rules for governing the choreography of transactions and collaborations. Section 7, finally, concludes the paper and discusses the results.

2 Basic Concepts and Related Research

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 a 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 behaviour 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 fulfil obligations. A similar approach is shown in [13] where a set of social patterns to bridge the gap between program driven and requirements driven paradigms to information systems modelling is introduced. Our work targets this work and aims at creating a set of concepts to unify already existing models in e-Commerce systems development.

The three-fold division above is based on an agent-oriented approach to information systems design, [22]. 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 fulfil commitments and contracts [8]. 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 [18] for a review. The work reported on in this paper differs from these approaches since it uses speech sct theory for analysing and integrating different modelling domains in e-Commerce, rather than facilitating electronic message handling per se.

One of the pioneers in the development of a theory of speech acts is John Searle, [19], who introduced a taxonomy of five different kinds of speech acts: assertive, directive, commissive, expressive, and declarative, also called illocutionary points.

An *assertive* is a speech act the purpose of which is to convey information about some state of affairs of the world from one agent, the speaker, to another, the hearer. A *commissive* is a speech act, the purpose of which is to commit the speaker to carry out some action or to bring about some state of affairs. A *directive* is a speech act, where the speaker requests the hearer to carry out some action or to bring about some state of affairs. A *declarative* is a speech act, where the speaker brings about some state of affairs by the mere performance of the speech act, e.g. “I declare you husband and wife”. Finally, an *expressive* is a speech act, the purpose of which is to express the speaker’s attitude to some state of affairs.

In addition to its illocutionary point, a speech act also has a propositional content. The speech acts “I hereby pronounce you husband and wife” and “You are hereby divorced”, which are both declaratives, have different propositional contents. A speech act is often viewed as consisting of two parts, its propositional content and its illocutionary force. The illocutionary force is the illocutionary point together with the manner (for example ordering, asking, begging) in which the speech act is performed and the context in which it occurs.

3 UMM Business and Process Models – BRV and BTV

The Resource-Event-Agent (REA) [17] framework has recently been applied in the UN/CEFACT Modelling Methodology (UMM) for business process modelling [2]. The scope of UMM is to provide a procedure for specifying, in a technology-neutral and implementation-independent manner business processes involving information exchange. In UMM, a number of meta-models are defined to support an incremental model development and to provide different levels of specification granularity.

- A business meta-model, called the *Business Operations Map* (BOM) partitions business processes into business areas and business categories.
- A requirements meta-model, called the *Business Requirements View* (BRV) specifies business processes and business collaborations.
- An analysis meta-model, called the *Business Transaction View* (BTV) captures the semantics of business information entities and their flow of exchange between business partners as they perform business activities.
- A design meta-model, called the *Business Service View* (BSV) models the network components services and agents and their message exchange.

The two meta-models relevant for our work are BRV and BTV (see Fig. 1) and we describe them briefly in the following sub sections.

3.1 Business Requirements View

As it is based on REA, BRV models EconomicEvents, the Resources transferred through the EconomicEvents, and the Agents, here called Partners between whom the Economic Events are performed. An EconomicEvent is the transfer of control of a Resource from one Partner to another. Each EconomicEvent has a counterpart, i.e. another EconomicEvent that is performed in return and realising an exchange. For instance, the counter part of a goods transfer economic event could be a payment, i.e. a transfer of money economic event. This connection between two economic events is modelled through the relationship *duality*. Furthermore, an EconomicEvent fulfils an Economic Commitment. An EconomicCommitment 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 EconomicEvents is inherited into the Economic Commitments, where it is represented by the relationship *reciprocal*.

In order to represent collections of related commitments, the concept of Economic Contract is used. An EconomicContract is an aggregation of two or more reciprocal Economic Commitments. An example of an EconomicContract is a purchase order composed of one or more order lines, each one representing a corresponding EconomicCommitment in the contract. The product type specified in each line is the EconomicResourceType that is the subject for the EconomicCommitment. EconomicContracts are often made within the boundaries of different Agreements. An

Agreement is an arrangement between two Partners that specifies the conditions under which they will trade.

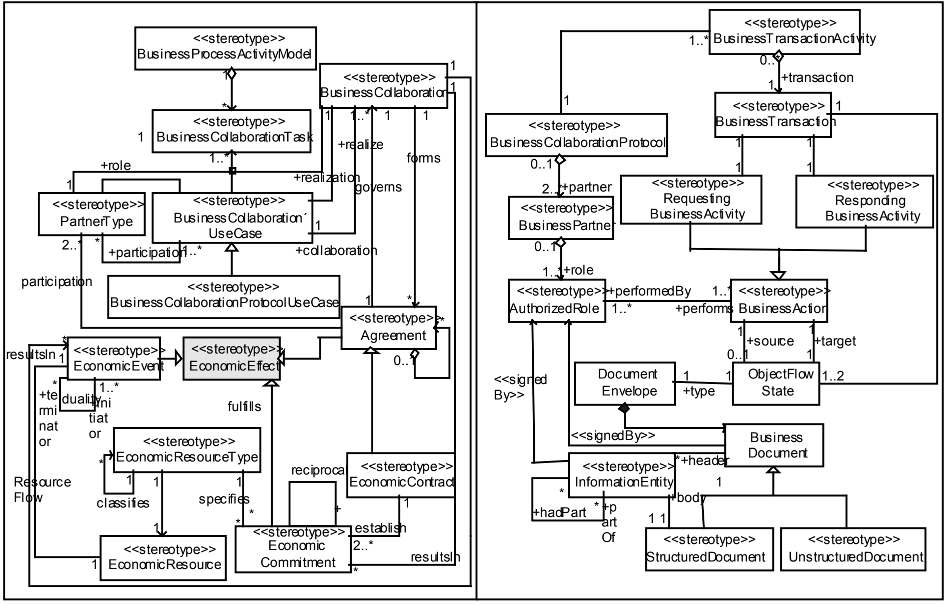
<<stereotype>> BusinessProcessActivityModel
<<stereotype>> BusinessCollaboration
<<stereotype>> BusinessCollaborationTask
+role
+realize
forms
+realization
governs
<<stereotype>> PartnerType
<<stereotype>> BusinessCollaboration´ UseCase
+participation
+collaboration
participation
<<stereotype>> BusinessCollaborationProtocolUseCase
<<stereotype>> Agreement
resultsIn
<<stereotype>> EconomicEvent
<<stereotype>> EconomicEffect
+terminator
duality
initiator
fulfills
<<stereotype>> EconomicResourceType
reciprocal
<<stereotype>> EconomicContract
Resource Flow
classifies
specifies
<<stereotype>> EconomicResource
<<stereotype>> Economic Commitment
establish
resultsIn
<<stereotype>> BusinessTransactionActivity
+transaction
<<stereotype>> BusinessTransaction
<<stereotype>> BusinessCollaborationProtocol
+partner
<<stereotype>> BusinessPartner
<<stereotype>> Requesting BusinessActivity
<<stereotype>> Responding BusinessActivity
+role
<<stereotype>> AuthorizedRole
+performedBy
+performs
<<stereotype>> BusinessAction
+source
+target
<<signed By>>
Document Envelope
+type
ObjectFlow State
<<signedBy>>
Business Document
<<stereotype>> InformationEntity
+header
+body
+hadPart
+partOf
<<stereotype>> StructuredDocument
<<stereotype>> UnstructuredDocument

Fig. 1 UMM Business Requirement (BRV)[1] and Business Transaction Views (BTV)

3.2 Business Transaction View

The Business Transaction View (BTV) specifies the flow of business information between business roles as they perform business activities. A BusinessTransaction 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 BusinessActions, are either RequestingBusinessActivities or RespondingBusinessActivities (depending on whether they are performed by a AuthorizedRole who is requesting a business service or whether they are the response to such a request). Furthermore, the flow between different BusinessTransactions can be choreographed through BusinessCollaborationProtocols.

4 An Agent-oriented Integration Framework

In terms of the three domains introduced in Section 2, UMM explicitly addresses only the physical and the social/institutional domains. The physical domain is modelled through classes like BusinessTransaction and BusinessAction, while the social/-institutional domain is modelled through EconomicCommitment, EconomicEvent, and other classes. The details of the communicative domain, however, are not explicitly

[1] EconomicEffect is an extension to UMM BRV and is described in the next section

modelled. This state of affairs causes two main problems. First, the relationship between the physical and the social/institutional domains is very coarsely modelled; essentially the UMM only states that a completed collaboration may influence objects in the social/institutional domain, 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.

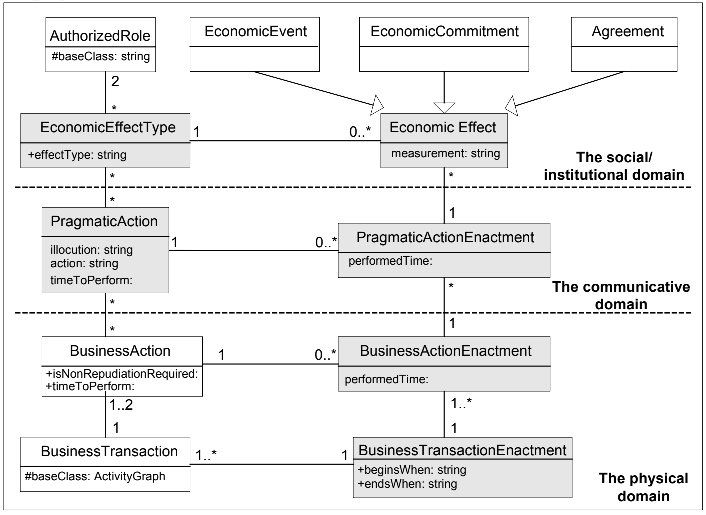

Fig. 2 Extended Business Requirement View

As a preparation to modelling the communicative domain, a minor modification to UMM BRV is made, see Fig. 2. A class EconomicEffect is introduced as a superclass of EconomicCommitment, Agreement, and EconomicEvent.

The power type [16] of EconomicEffect, called EconomicEffectType, is also added for the purpose of differentiating between the modelling 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 of communicative actions. In addition to these changes, the classes BusinessActionEnactment and BusinessTransactionEnactment are added. These represent the actual execution of a business action or business transaction, respectively.

The basic notions introduced for modelling the communicative domain are those of a pragmatic action and its execution, i.e. PragmaticAction and PragmaticActionEnactment, see Fig. 2. A *pragmatic action* is a speech act as introduced in Section 2. It consists of three parts, denoted as a triple:

<Illocution, Action, EffectType>

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

- EffectType specifies an EconomicEffectType, 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 EffectType

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 PragmaticActionEnactment is used to represent the actual executions of pragmatic actions. A PragmaticActionEnactment specifies a PragmaticAction as well as an EconomicEffect, i.e. the agreement, commitment, or economic event to be affected. Some examples of PragmaticActions are:

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

“Request purchase order” would be modelled as <propose, create, purchaseOrder>, where ‘salesOrder’ and ‘purchaseOrder’ are EconomicEffectTypes

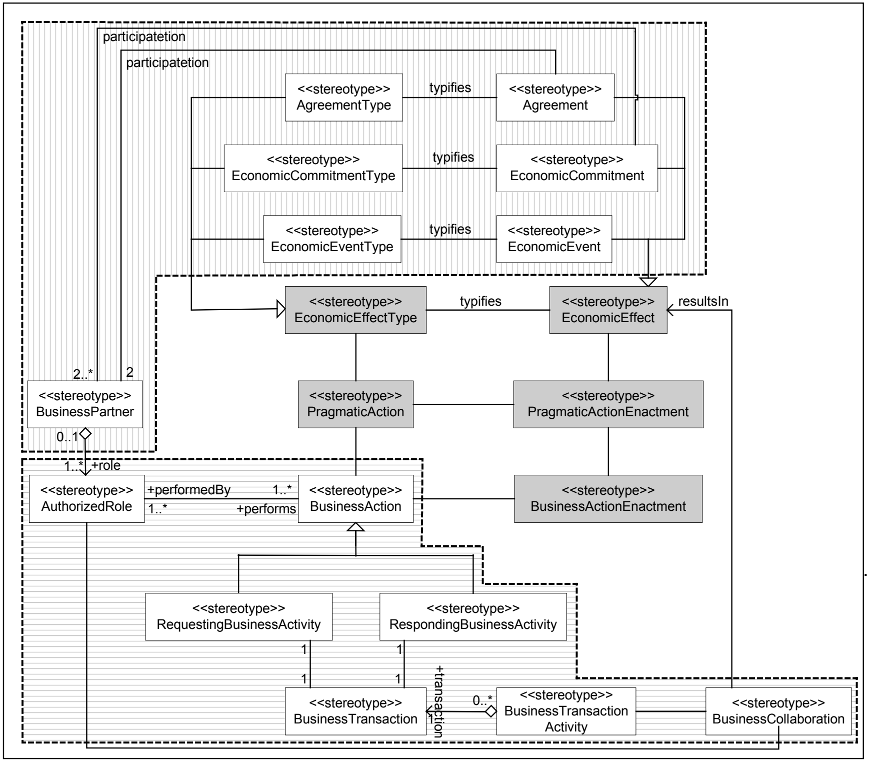
participatetion
participatetion
<<stereotype>> AgreementType
typifies
<<stereotype>> Agreement
<<stereotype>> EconomicCommitmentType
typifies
<<stereotype>> EconomicCommitment
<<stereotype>> EconomicEventType
typifies
<<stereotype>> EconomicEvent
<<stereotype>> EconomicEffectType
typifies
<<stereotype>> EconomicEffect
resultsIn
2..*
2
<<stereotype>> BusinessPartner
<<stereotype>> PragmaticAction
<<stereotype>> PragmaticActionEnactment
0..1
1..*
+role
<<stereotype>> AuthorizedRole
+performedBy
1..*
1..*
+performs
<<stereotype>> BusinessAction
<<stereotype>> BusinessActionEnactment
<<stereotype>> RequestingBusinessActivity
<<stereotype>> RespondingBusinessActivity
1
1
1
1
+transaction
1
0..*
<<stereotype>> BusinessTransaction
<<stereotype>> BusinessTransaction Activity
<<stereotype>> BusinessCollaboration

Fig. 3 Integrated Global view

4.1 Integrated View of Process and Business Models

The glue between the physical domain and the communicative domain is made up by the associations between the classes BusinessAction and PragmaticAction, and BusinessActionEnactment and PragmaticActionEnactment. 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. 3. The original BTV-parts are grouped within the area shadowed with horizontal lines, BRV-parts are grouped within the area shadowed with vertical lines and the new parts introduced in this chapter are depicted in the white area.

5 Application/Analysis of Transaction and Collaboration Patterns

In this section, a number of applications of the proposed framework with respect to business modelling patterns are introduced. 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 [14]. First, we discuss how the framework can be used for analysing the semantics of UMM business transaction patterns. Secondly, different collaboration patterns for incremental development are suggested.

5.1 Analysing UMM Business Transaction Patterns

UN/CEFACT has defined a number of business transaction patterns as part of UMM with the intention of providing an established semantics of frequently occurring business interactions. Below, we list a number of these patterns and show how they can be understood based on the framework introduced in the previous section.

Design patterns are defined as "descriptions of communicating objects and classes customised to solve a general design problem in a particular context" [10]. We will adopt this definition to the UMM transaction patterns and view a *transaction pattern* as a template of exactly one pair of a Requesting and Responding Business Activity customised to encode the intentions and effects of a business interaction in a context.

Definition: A *transaction pattern* (TP) is an activity diagram with two states designating the Requesting and Responding Business Activity. Every other state is either the start state or an end state. A state transition from a Requesting or Responding Business Activity is labelled by the pragmatic action(s) carried by the activity, see Fig. 4, Fig. 5 and Table 1, Table 2 below.

The analysis suggests one way to interpret the definitions of the UMM transaction patterns, but it does not make any claims to be the final, "correct" interpretation of these definitions. This is not an achievable goal as the definitions are only formulated in natural language, sometimes quite vaguely. The value of the analysis is that it provides explicit interpretations that can be judged for their validity, and thereby can help in formulating more precise and unambiguous patterns.

TP	Definition	Analysis
Commercial (Offer/ Accept)	"This design pattern is best used to model the '*offer and acceptance*' business transaction process that results in a residual obligation between both parties to fulfil the terms of the contract. The pattern specifies an originating business activity sending a business document to a responding business activity that may return a business signal or business document as the last responding message." [2]	**Request** <Propose,Create,aContract>
		Response <Accept,Create,aContract> or <Reject,Create,aContract>
Query/ Response	"The query/response design pattern specifies a query for information that a responding partner already has e.g. against a fixed data set that resides in a database. The response comprises zero or more results each of which meets the constraining criterion in the query." [2]	**Request** <Query,None,anEffectType>
		Response <Reply,None,anEffectType>or <Reject,None,anEffectType>
Request/ Confirm	"The request/confirm activity pattern shall be used for business contracts when an initiating partner requests confirmation about their status with respect to previously established contracts or with respect to a responding partner's business rules." [2]	**Request** <Query,None,aCommittment/ aContract>
		Response <Reply,None,aCommitment/ aContract>
Request/ Response	"The request/response activity pattern shall be used for business contracts when an initiating partner requests information that a responding partner already has and when the request for business information requires a complex interdependent set of results." [2]	**Request** <Query, None, anEffectType>or <Reject,None,anEffectType>
		Response <Reply,None,anEffectType>[2]
Information Distribution	"This pattern specifies the exchange of a requesting business document and the return of an acknowledgement of receipt signal. The pattern is used to model an *informal* information exchange business transaction that therefore has no nonrepudiation requirements." [2]	**Request** <Assert,None,anEffectType>
		Response Carries no pragmatic action
Notification	"This pattern specifies the exchange of a requesting business document and the return of an acknowledgement of receipt signal. The pattern is used to model a *formal* information exchange business transaction that therefore has non-repudiation requirements." [2]	**Request** <Declare,Create,aCommitment/ aContract>
		Response Carries no pragmatic action [3].

Table 1. Analysis of UMM transaction patterns in terms of pragmatic actions

Another use of the analysis is to suggest additional patterns than those already present in UMM. The Fulfilment, ContractProposal, Bilateral and Unilateral Cancellations (see Table 2) are obvious candidates.

[2] Note that the analysis fails to make a distinction between the query/response and the request/response patterns; the reason for this is that the difference between the patterns does not reside in different business effects but in different ways of computing the responses.

[3] The motivation for this analysis is that a notification results in a binding specification of business conditions for the initiating partner and thus, in a (partial) agreement.

TP	Definition	Analysis
Fulfilment	The fulfilment pattern specifies the completion of an Economic Event. Fig. 4	**Request** <Propose,Create,anEconomicEvent>
		Response <Accept,Create,anEconomicEvent>or <Reject,Create,anEconomicEvent>
Contract Proposal	The Contract Proposal Transaction Pattern is a variation of the aforementioned Offer-Accept transaction pattern where the Partners does not have to make their assertions of intentions legally binding.Fig. 4	**Request** <Propose,None,anEconomicContract>
		Response <Accept,None,anEconomicContract>or <Reject,None,anEconomicContract>
Bilateral Cancellation	The Bilateral Cancellation transaction pattern refers to the bilateral cancellation of an Economic Contract or to Commitment(s) within an Economic Contract. See the left part of Fig. 5	**Request** <Propose,Cancel,aContract/Commitment>
		Response <Accept,Cancel,aContract/Commitmntor <Reject,Cancel,aContract/Commitment>
Unilateral Cancellation	The Unilateral Cancellation transaction pattern refers to the unilateral cancellation of an Economic Contract or to Commitment(s) within an Economic Contract. See the right part of Fig. 5	**Request** <Declare,Cancel,aContract/Commitment>
		Response Carries no pragmatic action

Table 2. Additional Transaction Patterns to UMM

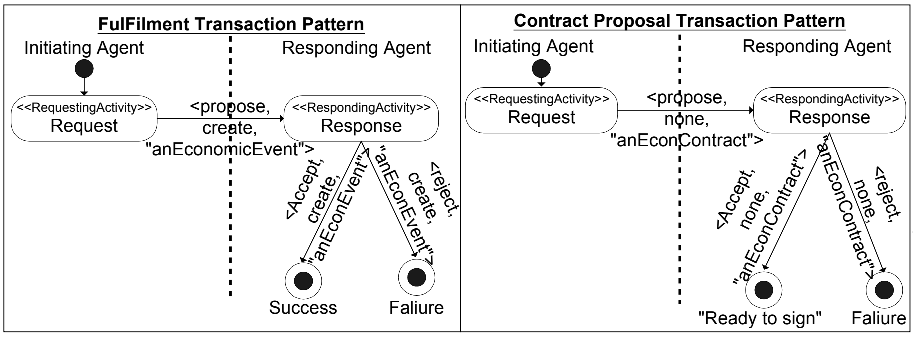

Fig. 4 Fulfilment and Contract Proposal Transaction Patterns

5.2 Collaboration Patterns

A *Business Collaboration Pattern* defines the orchestration of activities between partners by defining a set of BusinessTransaction patterns and/or more basic collaboration patterns plus the rules for transitioning from one transaction/collaboration to another [1]. The significance of a Business Collaboration Pattern is to serve as a predefined template in that it encodes business rules and business structure according to well-established best practices.

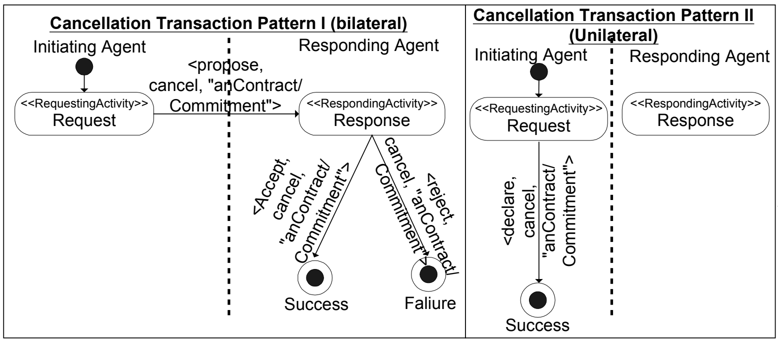
Cancellation Transaction Pattern I (bilateral)
Initiating Agent
Responding Agent
<propose, cancel, "anContract/ Commitment">
<<RequestingActivity>>
Request
<<RespondingActivity>>
Response
<Accept, cancel, "anContract/ Commitment">
<reject, cancel, "anContract/ Commitment">
Success
Faliure
Cancellation Transaction Pattern II (Unilateral)
Initiating Agent
Responding Agent
<<RequestingActivity>>
Request
<<RespondingActivity>>
Response
<declare, cancel, "anContract/ Commitment">
Success

Fig. 5 Bilateral and Unilateral Cancellation Transaction Patterns

A problem with the UMM collaboration patterns is that their complexity increases dramatically as new patterns are assembled from basic patterns, making the resulting activity diagrams hard to understand. To overcome this difficulty we use a layered approach where the transaction patterns constitute nodes in the activity diagram of a collaboration pattern. In this way the interactions between business partners within a transaction are modelled in a set of well-defined transaction patterns. In the collaboration pattern this complexity is hidden, and only the outcome of the transaction pattern is taken into consideration.

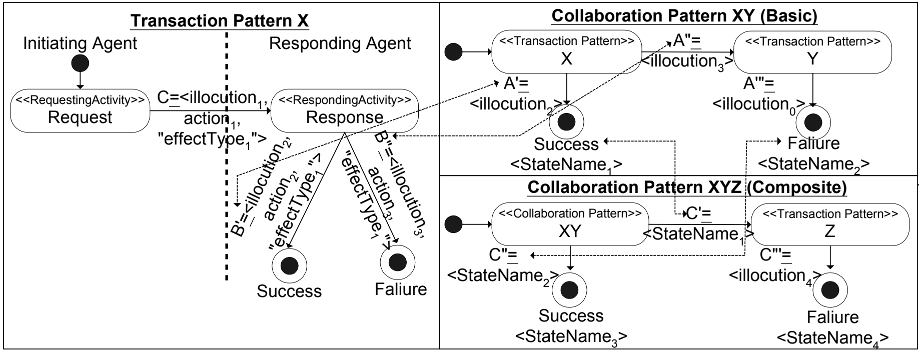

Fig. 6 Generic Transaction Pattern, Basic and Composite Collaboration Patterns

Definition: A *collaboration pattern* is a state chart over transaction and collaboration pattern(s). A collaboration pattern has exactly two end states representing success or failure of the collaboration, respectively. A transition *A* (see A′/A′′ of Fig. 6) from a transaction pattern must correspond to a transition *B* (see B′/B′′ of Fig. 6) to an end state in that transaction pattern. Furthermore, *A* is labelled (see illocution$_2$/illocution$_3$ of Collaboration Pattern XY of Fig. 6) by the illocution of *B*. A transition *C* (see C′/C′′ of Fig. 6) from a collaboration pattern is labelled by the names of the end states (see StateName$_1$/StateName$_2$ of Collaboration Pattern XYZ of Fig. 6) of the corresponding collaboration pattern.

5.2.1 Fulfilment Collaboration Pattern

The Fulfilment collaboration pattern specifies relevant transaction patterns (see the right most part of Fig. 7) and the rules for transitioning among these within the completion of an EconomicEvent. The pattern is assembled from the Fulfilment and Unilateral Cancellation transaction patterns defined in the previous section.

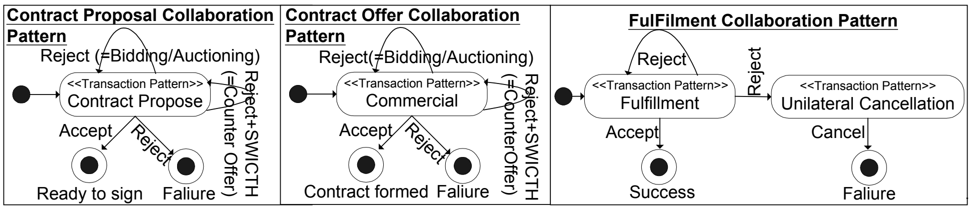

Fig. 7 Contract Proposal, Contract Offer and Fulfilment Collaboration Patterns

5.2.2 Contract Proposal and Contract Offer Collaboration Patterns

Two basic collaboration patterns for business negotiation for contract formation are given in the Proposal and Offer collaboration patterns [2]. The Contract Proposal collaboration pattern models the non-legally binding negotiation phase in a contract formation, whereas the Contract Offer collaboration pattern expresses the formal creation phase of a contract, see the left most and the middle parts of Fig. 7. These patterns are assembled from the Contract proposal transaction pattern and Commercial transaction pattern (described in Section 5.1), respectively.

The two recursive paths when a contract offer/proposal has been rejected have a natural correspondence in the business concepts 'Counter Offer' and 'Bidding' (or 'Auctioning'), respectively. 'Counter Offer' models the situation when the responding agent has rejected the requesting agent's offer and makes a new offer of her own. (The label 'SWITCH' indicates that the roles of the agents are reversed – the agent who received an offer becomes the one who proposes a new offer.) 'Bidding' models the situation where the responding agent has rejected the requesting agent's contract offer, and the latter then initiates a new Transaction with a new (changed) offer.

5.2.3 Composite Collaborations

More complex modelling and assembly of commitments, contracts and fulfilments are expressed in the example patterns found in [1]: a) Business Negotiation pattern, b) Order-Fulfilment-Settlement pattern, c) Long-term contract pattern with periodic releases, d) Escalating commitments pattern, e) Customer order direct delivery pattern.

We will apply our framework for analysing the first collaboration pattern in the list above: the Business Negotiation pattern. This pattern is composed of, in turn, the previously defined transaction patterns: Query-Response transaction pattern, Contract Proposal and Commercial transaction patterns, see the left part of the Fig. 8.

An example of a more complex collaboration pattern, composed of basic collaboration patterns only is also given below. In this example, the Business Negotiation pattern is a part of more complex collaboration patterns, see the right part of Fig. 8, where we envisage a larger collaboration pattern named, Contract Fulfillment. As can be seen the collaboration pattern is formed of a number of collaboration patterns where the two base patterns that make up are the Business Negotiation collaboration pattern and the Fulfilment collaboration pattern.

The three basic collaborations patterns in composite Business Negotiation and Fulfillment Collaboration in Fig. 8 can easily be mapped into typical phases in eContracting process. For instance those introduced in [4] namely, Information Phase, Pre-Contracting Phase, Contracting Phase and finally Enactment Phase.

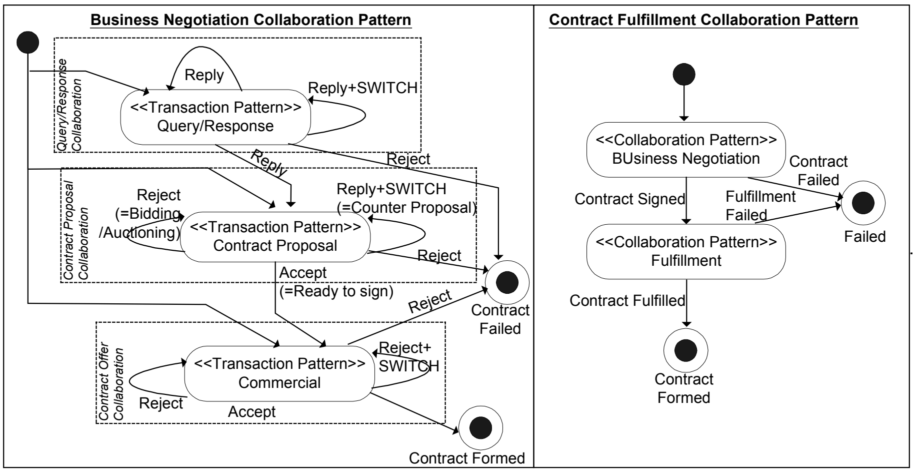
Business Negotiation Collaboration Pattern
Query/Response Collaboration
Reply
Reply+SWITCH
<<Transaction Pattern>> Query/Response
Reply
Reject
Contract Proposal Collaboration
Reject (=Bidding /Auctioning)
Reply+SWITCH (=Counter Proposal)
<<Transaction Pattern>> Contract Proposal
Reject
Accept (=Ready to sign)
Reject
Contract Failed
Contract Offer Collaboration
<<Transaction Pattern>> Commercial
Reject+ SWITCH
Reject
Accept
Contract Formed
Contract Fulfillment Collaboration Pattern
<<Collaboration Pattern>> BUsiness Negotiation
Contract Failed
Contract Signed
Fulfillment Failed
Failed
<<Collaboration Pattern>> Fulfillment
Contract Fulfilled
Contract Formed

Fig. 8 Business Negotiation and Contract Fulfillment Collaboration Pattern

6 Choreography of Transactions and Collaborations

Patterns evolve from structures and/or interactions that occur frequently in a certain context or domain. An issue is how to combine the patterns, i.e. how to avoid combining them in an incorrect way that diminishes their usefulness in solving problems. In this section, we propose rules governing the choreography, i.e. the sequencing of business transactions and business collaborations.

6.1 Ordering of Transactions

When a designer constructs a choreography for a collaboration, it is helpful to consider the dependencies that exist among the transactions of the collaboration. Two kinds of dependencies occur across many domains: trust dependencies [12] and flow dependencies [15].

A *trust dependency* is an ordered pair of transactions <A, B>, which expresses that A has to be performed before B as a consequence of limited trust between the initiator and the responder. As an example, it is possible to require that a product be paid before it can be delivered.

A *flow dependency* is an ordered pair of transactions <A, B>, which expresses that A has to be performed before B because the Economic Resources obtained in A are needed for carrying out B.

We now define two partial orders, Flow and Trust, whose members are ordered pairs of BusinessTransactions between whom a trust or flow dependency holds. Furthermore a BusinessTransaction is classified according to the Economic EffectType of the pragmatic action it targets, i.e. the EconomicContract, Economic-Commitment, or EconomicEvent to be affected. Fulfilment transactions targets EconomicEvents, commitment transactions targets EconomicCommitments and contract transactions target EconomicContracts. Cancellation transactions refer to all types of pragmatic actions, where the Action is of type ‘Cancel’. The signatures of the partial orders are given below where *Ful*, *Com*, *Ctr* and *Can* refer to the sets of fulfilment, commitment, contract, and cancellation transactions, respectively.

Trust is a partial order over $\{\mathrm{Ful} \cup \mathrm{Com} \cup \mathrm{Ctr}\}$ X $\{\mathrm{Ful} \cup \mathrm{Com} \cup \mathrm{Ctr}\}$.

Flow is a partial order over Ful X Ful.

A set of rules that govern the orchestration of activities (as defined of a pair of Requesting/Responding Business Activities in a Transaction) can now be defined.

Rule1: If A and B are nodes in a choreography C, and <A,B> $\in \{\mathrm{Flow} \cup \mathrm{Trust}\}$ then there must exist a path from A to B in C.

Furthermore, we observe that the establishment of a commitment or contract must precede the cancellation of the same, which gives rise to the following rule:

Rule 2: If A and B are nodes in a choreography C and A $\in \{\mathrm{Com} \cup \mathrm{Ctr}\}$ and B $\in$ Can where B is cancelling the contract or commitment established by A, then there must exist a path from A to B in C.[4]

Returning to the relationships between EconomicCommitment, EconomicContract and EconomicEvent, as stated in [17], we observe that Economic Contracts are subtypes of Agreements carrying Economic Commitments that some actual economic exchange will be fulfilled in the future. Thus, we identify the following rule:

Rule 3: If A and B are nodes in a choreography C and A $\in \{\mathrm{Com} \cup \mathrm{Ctr}\}$ and B $\in$ Ful, where B is establishing the economic event that fulfils the commitment established by A, then there must exist a path from A to B in C.

6.2 Inter Collaboration Sequencing

Sequencing of transactions make up business collaborations, which in their most basic form are expressed as state charts over transactions. We now turn to the sequencing within the next layer, i.e. between the business collaborations.

We introduce two new partial orders, analogous to the previously defined Trust and Flow. The signatures of the partial orders are given below where *FulC*, *ConP, ConO* and *BusiNeg* refer to the sets of Fulfilment-, ContractProposal-, ContractOffer- and Business Negotiation collaborations as defined above.

Trust^C is a partial order over

$\{\mathrm{FulC} \cup \mathrm{ConP} \cup \mathrm{ConO} \cup \mathrm{BusiNeg}\}$ X $\{\mathrm{FulC} \cup \mathrm{ConP} \cup \mathrm{ConO} \cup \mathrm{BusiNeg}\}$.

Flow^C is a partial order over FulC X FulC.

Rule 4 below is analogous to rule 1 and states that trust and flow dependencies should be respected. Rule 5 expresses that fulfilments always follow upon negotiations. Rule 6 states that in a business negotiation, a contract proposal comes before a contract offer. Finally, rule 7 states that transactions and collaborations that do not change social/institutional relationships can be placed anywhere in a larger collaboration.

Rule 4: If A and B are nodes in a choreography C, and <A,B> $\in \{\mathrm{Trust}^C \cup \mathrm{Flow}^C\}$, then there should exist a path from A to B in C, i.e. when a trust or flow dependency holds between the two collaborations A and B, A must precede B.

[4] A fulfilment transaction can be performed or not performed but it cannot be cancelled.

Rule 5: If A and B are nodes in a directed graph C, A ∈ BusiNeg and B ∈ FulC, where B is establishing the economic event(s) that fulfils the commitment(s) within a given contract established by A, then there must exist a path from A to B in C.

Rule 6: If A and B are nodes in a choreography C, A ∈ ConP and B ∈ ConO, where A is the non legally binding negotiation about the terms to be established in B, then there must exist a path from A to B in C.

Rule 7: If a transaction or collaboration pattern carries only pragmatic actions of type <_, None, _>, i.e. where the 'Action' part of the pragmatic action triplet is equal to 'None', then these transactions and collaborations can be included optionally in any collaboration irrespective of order.

Rules 1 - 7 can be used to guide and restrict the design of a choreography, i.e. give suggestions for possible paths between different transactions and collaborations, as well as sequences of these, and rule out incorrect paths.

7 Concluding Remarks

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 fulfil obligations, while business models depict the social/institutional world where economic relationships such as 'ownership' hold and actions such as transfer of economic resources occur.

The main contribution of this paper 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 agents that establish, modify, cancel and fulfil commitments and contracts. In carrying out these activities, agents 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 two main 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 [2], ebXML BPSS[9], BPEL4WS [3], etc). Thus, pragmatic actions provide a clean interface to collaboration protocols.

References

1. "eBTWG – Business Collaboration Patterns/Business Commitment Patterns Technical Specification (BCP^2) – "Monitored Commitments"", accessed 20030401, http://www.collaborativedomain.com/standards/.
2. "UN/CEFACT Modelling Methodology (UMM-N090 Revision 10), accessed 20030328, http://webster.disa.org/cefact-groups/tmg/doc_bpwg.html.
3. Andrews Y., et al., "Business Process Execution Language for Web Services", accessed 20030404, http://www-106.ibm.com/developerworks/webservices/library/ws-bpel/.
4. Angelov S., Grefen P., "A Conceptual Framework for B2B Electronic Contracting", in *Proc of 3rd IFIP Working Conf.e on Infrastructures for Virtual Enterprises*, 2002.
5. Bergholtz M., et al., "Reconciling Physical, Communicative and Social/Institutional Domains in Agent Oriented Information Systems - a Unified Framework", *Int. WorkShop on Agent Oriented Information Systems*, ER 2003, Springer-Verlag, LNCS 2814.
6. Bergholtz M., et al., "Business and Process Models – a Unified Framework", in *Proc. of eCOMO'2002, 21th Int. Conference on Conceptual Modeling (ER'2002),* Springer-Verlag.
7. Dietz J., "Modelling Communication in Organisations", in *Linguistic Instruments in Knowledge Engineering*, Ed. R.v.d.Riet, pp 131-142, Elsevier Science Publishers, 1992
8. Dignum F. and Weigand H., "Modelling Communication between Cooperative Systems", in *Proc. of the 7th Conference of Computer Advanced Information System Engineering (CaiSE)*, LNCS, Springer Verlag, 1995
9. ebXML Deliverables, *Electronic Business eXtensible Mark-up Language (ebXML),*Valid on 20030328, http://www.ebxml.org/specs/index.htm
10. Gamma E., Helm R., Johnson R., and Vlissides J., "*Design Patterns: Elements of Reusable Object-Oriented Software*", Addison Wesley. 1995
11. Gordijn J., Akkermans J. M. and Vliet J. C., "Business Modelling, is not Process Modelling", *Int. Workshops on Conceptual Modeling Approaches for E-Business and The World Wide Web and Conceptual Modeling*, Springer-Verlag, LNCS 1921, 2000.
12. Jayaweera P., Johannesson P., Wohed P., "Collaborative Process Patterns for e-Business", *ACM SIGGROUP Bulletin 2001/Vol 22, No. 2*
13. Kolp M., Giorgini P., Mylopoulos J., "Information Systems Development through Social Structures", Int. Conf. on Software Engineering and Knowledge Engineering, 2002.
14. Larman C., "*Applying UML and patterns: an introduction to object oriented analysis and design*", ISBN 0-13-74880-7
15. Malone et al.: "Towards a handbook of organisational processes", *MIT eBusiness Process Handbook*, Valid on 20030404, http://ccs.mit.edu/21c/mgtsci/index.htm.
16. Martin, J., Odell, J.: "*Object-Oriented Methods. A Foundation*", Prentice Hall 1994
17. McCarthy W. E., "*REA Enterprise Ontology*", accessed 20030404, http://www.msu.edu/user/mccarth4/rea-ontology/.
18. Moore Scott A., "A foundation for flexible automated electronic communication", *Information Systems Research*, 12:1, 2001
19. Searle J. R., "*A taxonomy of illocutionary acts*", K. Gunderson (Ed.), Language, Mind and Knowledge, Minneapolis: University of Minnesota, 1975.
20. Taveter K. and Wagner G., "Agent-Oriented Enterprise Modeling Based on Business Rules", *Proc. of 20th Int. Conf. on Conceptual Modeling*, Springer-Verlag, LNCS, 2001.
21. van den Heuvel W. and Weigand H., "Coordinating Web-Service Enabled Business Transactions with Contracts", In Eder J. and Missikoff M. (eds). Conf. of Information Systems Engineering (CAiSE'03), Springer-Verlag, LNCS 2681, pp. 568-583, 2003.
22. Wagner G., The Agent-Object-Relationship metamodel: towards a unified view of state and behavior. *Information. Systems. 28* (5): 475-504 (2003)
23. 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