Abstracting and Applying Business Modeling Patterns from RosettaNet*

Pankaj R. Telang[1,2] and Munindar P. Singh[2]

[1] Cisco Systems Inc., Research Triangle Park, NC 27709, USA
prtelang@ncsu.edu
[2] North Carolina State University, Raleigh, NC 27695-8206, USA
singh@ncsu.edu

Abstract. RosettaNet is a leading industry effort that creates standards for business interactions among the participants in a supply chain. The RosettaNet standard defines over 100 Partner Interface Processes (PIPs) through which the participants can exchange business documents necessary to enact a supply chain. However, each PIP specifies the business interactions at a syntactic level, but fails to capture the business meaning of the interactions to which they apply.

In contrast, this paper takes as its point of departure a commitment-based approach for business modeling that gives central position to interactions captured in terms of their meaning. This paper defines commitment-based business patterns abstracted from RosettaNet PIPs. Doing so yields models that are clearer, more flexible to changing requirements, and potentially enacted through multiple operationalizations. This paper validates the patterns by applying them to model the Order-to-Cash business process from the RosettaNet eBusiness Process Scenario Library.

1 Introduction

The intense competition in the global economy compels organizations to provide high-quality products and services at an attractive price. It forces organizations to innovate in how they define and deliver their services by identifying context-aware processes and activities, and operationalizing them flexibly, including by outsourcing them to other organizations that specialize in executing those processes. Important examples of such processes include human resources, workplace management, payroll, call centers, and IT infrastructure administration. By outsourcing such processes, an organization may reduce operational expenses, and at the same time gain access to any specialized resources it needs. Such outsourcing results in a network of organizations who engage in a complex set of service interactions.

RosettaNet [16], a leading industry effort, creates standards for business interactions. The RosettaNet consortium consists of over 500 organizations of various sizes, and from various industry sectors including electronics manufacturing from which domain RosettaNet began. These organizations use elements of the RosettaNet standard, named Partner Interface Processes (PIPs), to transact business that is worth billions of dollars.

* We thank the anonymous reviewers for helpful comments.

A PIP specifies two-party interactions for some specific business purpose. For example, a buyer requests a quote from a seller using PIP 3A1, and a seller requests financing from a financing processor, on behalf of the buyer, using PIP 3C2. A PIP specification includes a natural language document that informally describes the purpose of the PIP, any underlying assumptions, the intended outcome of executing the PIP, and the message structures as XML DTD or XML Schema.

RosettaNet PIPs specify the business interactions well at a syntactic level, but they fail to capture the business meanings of those interactions. For example, in PIP 3A1, a buyer sends a request for quote to a seller, and the seller responds with either a quote or a referral. RosettaNet leaves important business details unspecified. If the seller responds with a quote, does the seller commit to the buyer to selling the goods at the quoted price? As an analogy, consider the price for a book you read on an online bookseller's (e.g., Amazon's) page versus the price you read for a stock on an online stock broker's (e.g., Ameritrade's) page. You can normally buy the book but not the stock for the quoted price, meaning that Amazon commits but Ameritrade doesn't. Likewise, does the buyer's acknowledging the quote commit it to buying the goods at the quoted price?

Thus, 3A1 and other RosettaNet PIPs leave such important questions regarding the business meanings of the interactions to human interpretation. Each group of analysts and developers working in the partner organizations would negotiate such considerations between themselves, but doing so has the effect of introducing idiosyncratic constraints through which the partners become inadvertently tightly coupled. This reduces their prospects for service innovation.

In previous work [22,24], we present a (1) high-level *business metamodel* to capture models that describe, purely in terms of business meaning, how cross-organizational service engagements are carried out and (2) a method based on temporal logic model checking [7] to establish that a particular operationalization satisfies a specified business model. This paper uses the business metamodel to abstract business *patterns* from the RosettaNet PIPs, and outlines a methodology for applying such patterns to develop business models of specific service encounters based on legacy models, such as based on activity models. The patterns developed here respect our previous metamodel and consequently can be used as a basis for model checking, as in our recent work [24].

Note that the patterns we propose are not operational patterns such as the well-known enterprise integration patterns [10] or workflow patterns [18]. Such operational patterns are useful for developing implementations of processes but would not address the challenge of encoding the business meaning that we pursue in this paper. In particular, developing operational patterns for the RosettaNet PIPs might offer implementation best practices but would not answer the questions of the type we ask above: who is committed to whom, for what, and when? Instead our patterns are high-level patterns that specify the essence of a business interaction, and may potentially map to multiple operational patterns depending upon other criteria.

There are two key motivations in abstracting commitment-based patterns from RosettaNet PIPs. First, since the patterns are at a business level, business analysts can easily understand and compose them to develop a desired business model. Second, a model composed from these patterns serves as a formal specification that can be used to verify an operational model defined in any technical standard, such as sequence diagrams [24]. Organizations frequently migrate their business process implementations to newer technologies to benefit from the improvements those technologies offer. In such cases of technology migration, the high-level formal specification provides a basis for establishing the correctness of the new implementation.

Contributions. The main contribution of this paper is business patterns abstracted from a key subset of the RosettaNet PIPs. Additionally, it outlines a simple methodology for applying the patterns to a real-life business scenario. The paper evaluates its contributions via a case study using the Order to Cash process, which has been addressed by conventional approaches—in particular, by the RosettaNet consortium using conventional means—yielding a ready basis for comparison.

Organization. Section 2 provides the necessary background on the RosettaNet PIP standard and introduces our business metamodel. Section 3 presents business patterns for a subset of the RosettaNet PIPs. Section 4 applies the patterns to model the Order-to-Cash scenario. Section 5 concludes the paper with a discussion of the related work and some future directions.

2 Background

We now review some key background on RosettaNet and on our approach.

2.1 RosettaNet

The RosettaNet standard specifies over 100 PIPs for various business processes in the eCommerce supply chain. The standard classifies the PIPs using clusters and segments. A cluster represents a major business process of the supply chain and comprises segments that represent subprocesses of the cluster's business process. Each segment contains many PIP specifications. For example, Cluster 3 represents the Order Management process. Segment A of Cluster 3 represents the subprocess Quote and Order Entry. Segment A contains PIPs such as for Request Quote (3A1), Request Shopping Cart Transfer (3A3), and Request Purchase Order (3A4). RosettaNet PIPs are commonly identified through their code names such as those in parentheses above.

The RosettaNet standard employs three views to specify a PIP: the Business Operational View (BOV), the Functional Service View (FSV), and the Implementation Framework View (IFV). The BOV informally describes the element of the business process that the PIP implements. It specifies a process flow diagram that shows the participant roles, business activities, and the flow of business messages among them. For each activity, the BOV specifies performance controls, such as the need to acknowledge receipt, the nonrepudiability of the receipt, and the timeout for the acknowledgment. For a PIP, the FSV derives from the BOV, and specifies the RosettaNet services. It specifies the message exchange sequence in a business transaction dialog, and for each message, it specifies message exchange controls. These controls include the time period within which an acknowledgment is required, the time within which a response to an action is required, and whether authorization is required for an action. The IFV specifies the formats of the messages exchanged in the PIP as XML DTDs or XML Schemas. For each message, it specifies the communication requirements such as whether it needs a digital signature or secure transport.

2.2 Business Metamodel

The following discussion is extracted from our previous work [22]. A business model specifies how business is conducted. We concern ourselves with business models that involve two or more participants. The business *participants*, abstracted as *roles*, participate in a *business relationship*. The participants create, manipulate, and satisfy *commitments* in a relationship. They execute *tasks* for each other that enable them to achieve their respective *goals*.

Three distinct phases characterize business execution. First, in the *agreement* phase, participants enter into an agreement, and *create* commitments toward each other. Second, in the *assembly* phase, the participants *delegate* or *assign* commitments to others. A participant may delegate a commitment that requires the execution of a task which is not a core competency of that participant, or due to some other economic motivation. Third, in the *enactment* phase, participants execute tasks to *satisfy* their commitments.

We now describe the concepts in our metamodel.

Agent: a computational representation of a business participant. An agent has goals [23], and executes business tasks. An agent enacts one or more roles in each business relationship in which it participates.

Role: an abstraction over agents that helps specify a business relationship. Each role specifies the commitments expected of the agents who play that role along with the tasks they must execute to function in that role.

Goal: a state of the world that an agent desires to be brought about [6]. An agent achieves a goal by executing appropriate tasks.

Task: a business activity viewed from the perspective of an agent.

Commitment: a normative relationship between a debtor and a creditor: a commitment C(DEBTOR, CREDITOR, antecedent, consequent) denotes that the DEBTOR commits to the CREDITOR for bringing about the consequent if the antecedent holds [19].

Business relationship: a set of interrelated commitments among two or more roles that describe how a business interaction is carried out among the participating roles.

Since our approach is centered on commitments, we discuss them at greater length. A commitment can be in one of the following states: *inactive*, *active*, *detached*, *pending*, *satisfied*, or *violated*. Before a commitment is created, it is inactive. A commitment may remain inactive forever if it is never created. Alternatively, if the debtor creates the commitment, it becomes active.

If the antecedent of an active commitment is brought about, then the commitment is detached. It is possible for a commitment to be created as detached—if its antecedent is true at the outset. When the consequent of a commitment, whether detached or not, is brought about, it is satisfied. After a commitment is satisfied, whether its antecedent is brought about or not has no effect.

For an active commitment, an active timeout occurs if neither its antecedent nor its consequent is brought about within the specified time period. In that case, the commitment expires. Similar to an active timeout, for a detached commitment, a detached timeout may occur if its consequent is not brought about, causing the commitment to be violated.

An active commitment functions like an obligation, but key differences include that a commitment is directed from a debtor to a creditor, arises in a (here, business) context, and may be manipulated. Manipulations of commitments are particularly important to business modeling. We discussed above how a commitment may be created (making an offer), discharged (satisfied), or canceled (withdrawing an offer). In addition, a commitment may be released (rejecting an offer), delegated to a new debtor (outsourcing), assigned (a reseller directing a shipment to the actual customer site). When a commitment is delegated or assigned, in some business patterns, we can place it in the pending state to enable it being revived if necessary [21].

3 Business Patterns from PIPs

A business pattern abstracts from operational details and, using the concepts of our business metamodel, captures the business-level interactions that underlie a PIP. We identify the business meanings that derive naturally from a PIP's description, and express them in terms of how they manipulate commitments. We now present the patterns for a selected subset of the PIPs that help us model the Order to Cash process from Section 4.

3.1 Request Quote (PIP 3A1)

Figure 1 shows the usage diagram extracted from the PIP 3A1 specification. The figure shows that a buyer desiring to purchase certain goods sends a request for a quote to a seller. If the seller is able to satisfy the requirements of the quote, the seller sends a quote response. Alternatively, if the seller is unable to satisfy the requirements of the quote request, the seller looks up and sends a referral to the buyer. In this case, the buyer at its own discretion may engage in business with the referred seller.

Figure 2 shows the business pattern we derive from PIP 3A1. Similar to the original RosettaNet specification for this PIP, the pattern specifies the two roles, the buyer and the seller. The business intent of PIP 3A1 is for the buyer to secure a price commitment from the seller for certain goods. The pattern captures this as a commitment C(SELLER, BUYER, pay, goods). Note that the PIP specifies *how* the roles interact in terms of the quote request, and the quote response. On the contrary, our pattern abstracts away from these operational details, and specifies *what* the interaction achieves in terms of the commitment that ensues from it.

At an operational level, the participants exchange messages to execute a PIP. The execution updates the state of the relevant commitments. Here, commitment C1 remains inactive if the buyer sends a request for quote, and either the seller fails to respond within certain timeout, or the seller responds with a referral. In case where the buyer requests a quote, and the seller responds with a quote, commitment C1 becomes active.

Note that PIP 3A1 requires the buyer to send a request for quote prior to the seller sending a quote. However, this is not always necessary. A seller may send an unsolicited quote as an advertisement to the buyer. Since our business pattern abstracts from operational details, it applies for the unsolicited quote scenario as well, thus offering operational flexibility beyond the traditional interpretation of the RosettaNet guidelines.

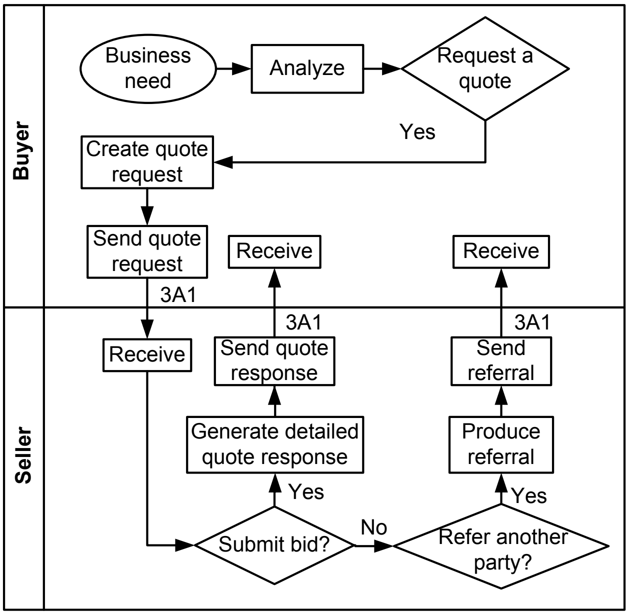
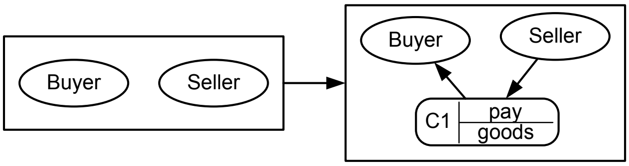


Fig. 1. Usage guideline for PIP 3A1 (verbatim from the RosettaNet specification [16])

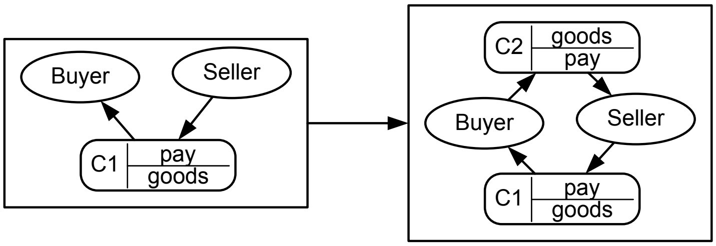


C1 C(SELLER, BUYER, pay, goods)

Fig. 2. Business pattern for PIP 3A1. Notice that the BUYER performs the task pay, and SELLER performs the task goods.

3.2 Request Purchase Order (PIP 3A4)

In PIP 3A4, a buyer sends a purchase order to a seller. The seller either accepts or rejects the order. This PIP presumes that the buyer and seller previously enacted another PIP to create a price commitment from the seller to the buyer. Figure 3 shows the business pattern for PIP 3A4. Similar to the original RosettaNet specification for this PIP, the pattern contains two roles: the buyer and the seller. The pattern presumes that a commitment C1 = C(SELLER, BUYER, pay, goods) exists from the seller to the buyer to sell the goods for certain price. In this pattern, the buyer commits to the seller to pay if the seller ships the goods. The pattern models this as the commitment C2 = C(BUYER, SELLER, goods, pay).



C1	C(SELLER, BUYER, pay, goods)
C2	C(BUYER, SELLER, goods, pay)

Fig. 3. Business pattern for PIP 3A4. Notice that the BUYER performs the task pay, and SELLER performs the task goods.

Table 1 shows the states of the commitments for some of the possible executions of PIP 3A4. Commitment C2 is not created and remains inactive if the buyer sends an order, and the timeout occurs. Further, in case where the buyer sends an order, and the seller rejects the order, the seller cancels its commitment C1 and thus C1 becomes inactive. When the buyer sends an order, and the seller accepts the order, commitment C2 is created, thereby making it active.

Table 1. Commitment state progression in alternative PIP 3A4 executions

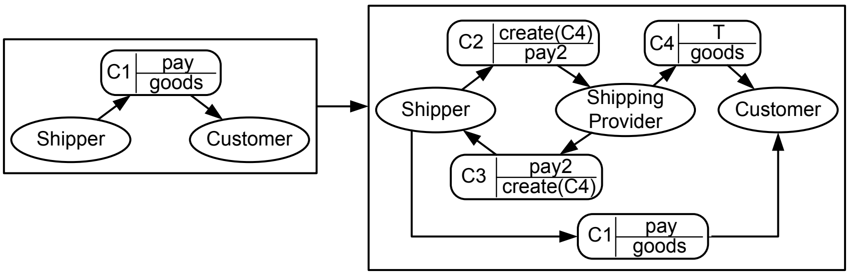
Execution	**Commitment State**	
	C1	C2
sendOrder, timeout	Active	Inactive
sendOrder, rejectOrder	Inactive	Inactive
sendOrder, acceptOrder	Active	Active

3.3 Request Purchase Order Change (PIP 3A8)

A buyer may use PIP 3A8 to request a change to an order. This PIP presumes that the buyer and the seller have already negotiated, the buyer sent an order to the seller, and the seller accepted the order. The business pattern from Figure 3 models the precondition for this pattern. This pattern updates the antecedent and the consequent of commitments C1 and C2, and does not introduce any new commitments. Since the pattern is similar to the pattern from Figure 3, and to save space, we omit the details.

3.4 Request Shipping Order (PIP 3B12)

In this PIP, a shipper sends a shipping order to a shipping provider. The shipping provider either accepts or rejects the order. The shipper is the participant that sells goods to a customer, and employs the shipping provider for shipping the goods to the customer. Figure 4 shows the business pattern we abstract from this PIP. This PIP presumes that the shipper is committed to the customer to shipping the goods, as modeled by C1. The shipper outsources this commitment to a shipping provider. C4 is the outsourced commitment that models the shipping provider's commitment to the customer to ship the goods. It is unconditional (detached) since its antecedent is true ($\top$). To set up the outsourcing, the shipper and the shipping provider create commitments C2 and C3. In C2, the shipper commits to the shipping provider to paying if the shipping provider creates the outsourced commitment C4. Notice that commitment C3 is the converse of C2. We term such commitments *reciprocal* commitments; they arise often in practice.



C1	C(SHIPPER, CUSTOMER, pay, goods)
C2	C(SHIPPER, SHIPPING PROVIDER, create(C4), pay2)
C3	C(SHIPPING PROVIDER, SHIPPER, pay2, create(C4))
C4	C(SHIPPING PROVIDER, CUSTOMER, $\top$, goods)

Fig. 4. Business pattern for PIP 3B12. Notice that the CUSTOMER performs task pay, SHIPPER performs pay2 and goods, and SHIPPING PROVIDER performs goods

Table 2 shows the progression of commitments corresponding to the possible executions of the PIP. In the case where the shipper sends a shipping order to the shipping provider, and either the timeout occurs or the shipping provider rejects the order, commitments C2, C3, and C4 remain inactive. When the shipping provider accepts a shipping order sent by a shipper, then the commitment C4 becomes active, commitment C3 becomes satisfied, and commitment C2 detaches. The original commitment C1 from the shipper to the customer becomes pending, since the outsourced commitment C4 is now

Table 2. Commitment state progression in alternative PIP 3B12 executions

Execution	**Commitment State**			
	C1	C2	C3	C4
sendShippingOrder, timeout	Active	Inactive	Inactive	Inactive
sendShippingOrder, rejectShippingOrder	Active	Inactive	Inactive	Inactive
sendShippingOrder, acceptShippingOrder	Pending	Detached	Satisfied	Active

active. Later, if the shipping provider fails to satisfy commitment C4, the original commitment C1 is reactivated. In that case, the shipper may engage with another shipping provider to bring about the shipping of the goods.

3.5 Notify of Advance Shipment (PIP 3B2) and Notify of Remittance Advice (PIP 3C6)

Using PIP 3B2, a shipper notifies a receiver about a shipment. This PIP presumes that the shipper and the receiver have negotiated, and agreed to exchanging goods for payment. At a business level, commitment C1 = C(SHIPPER, RECEIVER, pay, goods), and the commitment C2 = C(RECEIVER, SHIPPER, goods, pay) may be active prior to the shipper sending PIP 3B2. By executing this PIP, shipper satisfies commitment C1, and detaches commitment C2.

A buyer sends a remittance advice to a seller using PIP 3C6. Similar to PIP 3B2, this PIP presumes that the buyer and the seller earlier agreed to exchanging goods for payment. At a business level, commitment C1 = C(SELLER, BUYER, pay, goods), and commitment C2 = C(BUYER, SELLER, goods, pay) may be active prior to the buyer sending PIP 3C6. By executing this PIP, the buyer satisfies commitment C2, and detaches commitment C1.

3.6 Distribute Inventory Report (PIP 4C1), Notify of Shipment Receipt (PIP 4B2), and Notify of Invoice (PIP 3C3)

Using PIP 4C1, an inventory owner reports the inventory status to an inventory information user. We view this PIP as merely supporting information exchange between the participants. At a business level, such a PIP does not create or manipulate any commitments by either party to perform a tangible task for the other. In principle, we can create commitments for such notifications as well if necessary, such as Xing et al. [25] have done, but for simplicity we do not do so in this paper. Similarly, we view PIPs 4B2 and 3C3 as information exchange, and they do not create or manipulate any commitments.

4 Evaluation: Order to Cash Process

This section applies the above business patterns to model the Extended Order to Cash process, which is specified as an important scenario in the RosettaNet eBusiness Process Scenario Library [17].

Figure 5 shows the Order-to-Cash business process scenario. The participants of this process are a supplier, a customer, and a shipper. The customer orders products from the supplier. Later, the customer may request a change to the order. The supplier engages a shipper for shipping the goods to the customer. Additionally, the shipper periodically sends an inventory report to the supplier. Prior to shipping the goods, the shipper notifies the customer. The customer notifies the supplier upon receiving the goods. Subsequently, the supplier sends an invoice to the customer. The customer validates the invoice, and thereafter sends a remittance advice to the supplier. Figure 5 shows the RosettaNet PIPs that the participants employ for the above interactions.

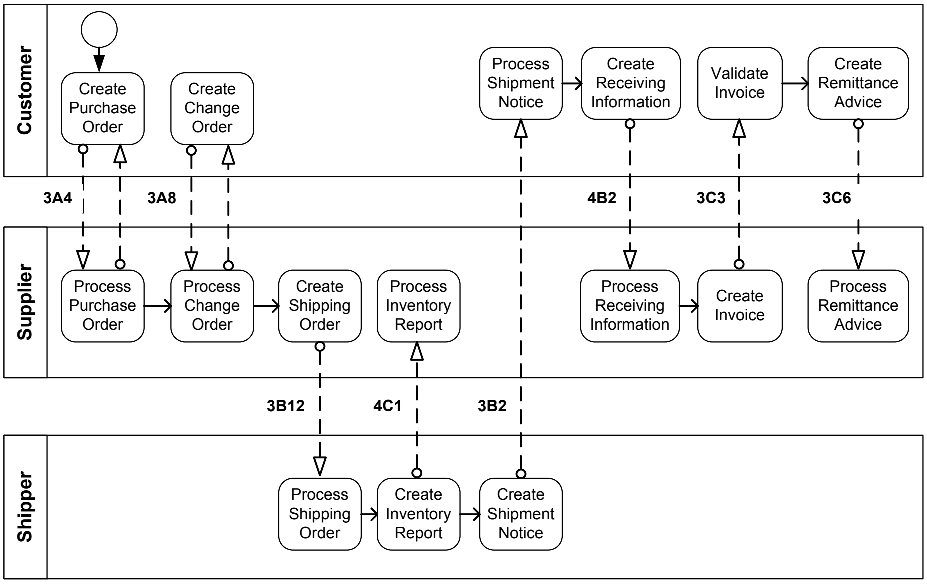
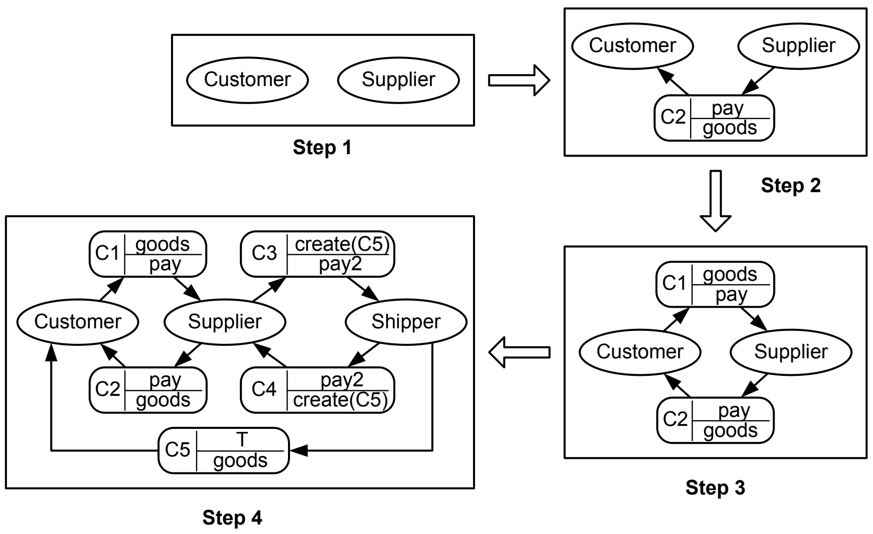


Fig. 5. The Order to Cash business process (verbatim from RosettaNet documentation [17])

We propose a simple bottom-up methodology to develop a business model for this scenario. In this methodology, we begin with a process flow diagram, such as that based on RosettaNet PIPs. The flow specifies the participating roles as swim lanes, and the interactions between the participants in terms of the PIPs that they execute. The methodology progressively composes the business patterns abstracted from the PIPs. The roles in the patterns are substituted by the corresponding roles in the modeled scenario. Further, the commitment labels in the composed patterns are updated to prevent spurious identification of distinct commitments. This methodology considers only those PIPs that pertain to the agreement and the assembly phase, that is, the PIPs that introduce new commitments.

Figure 6 applies the above methodology to the Order-to-Cash process, as described in Figure 5. Step 1 introduces the primary participant roles: the customer and the supplier. Notice that PIP 3A4 presumes that the buyer has previously secured a price commitment from the seller using a PIP such as PIP 3A1. However, the traditionally specified flow in Figure 5 fails to show such an interaction. Step 2 applies the business pattern of PIP 3A1, which introduces commitment C2 into the model. We substitute the buyer role with the customer role, and the seller role with the supplier role. Step 3 applies the business pattern of PIP 3A4, which introduces commitment C1. Step 4 applies the business pattern of PIP 3B12, and introduces commitments C3, C4, and C5 into the model. This step substitutes the pattern roles shipper, shipping provider, and customer with the scenario roles supplier, shipper, and customer, respectively. Since the remaining PIPs in the process flow do not pertain to the agreement or the assembly phases, the methodology stops here.



C1	C(CUSTOMER, SUPPLIER, goods, pay)
C2	C(SUPPLIER, CUSTOMER, pay, goods)
C3	C(SUPPLIER, SHIPPER, create(C5), pay2)
C4	C(SHIPPER, SUPPLIER, pay2, create(C5))
C5	C(SHIPPER, CUSTOMER, $\top$, goods)

Fig. 6. The Order-to-Cash business process expressed as a business model

The model in Figure 5 specifies the messages, in the form of PIPs, that the participants exchange and the temporal ordering of those messages. In this model, a participant complies with the model only if it exchanges messages in the prespecified temporal order. Such an operational restriction limits the flexibility that the participants have in executing the business. For example, the operational model mandates that the customer pays (3C6) only after receiving the shipment (3B2) and the invoice (3C3). At a business level, such a restriction may not be appropriate: e.g., for tax purposes, a customer may wish to pay prior to receiving the shipment.

Instead of imperatively specifying how the participants conduct business in terms of their messages and orderings, the business model in Figure 6 declaratively specifies how the business executes in terms of commitments. A participant complies with the business model if it satisfies all the detached commitments of which it is a debtor. Consider commitment C1(CUSTOMER, SUPPLIER, goods, pay). As per the operational semantics of a commitment, a customer can pay either before or after receiving the goods, thus benefiting from flexibility. A customer violates this commitment only if the supplier ships the goods, and the customer never pays.

Unlike RosettaNet's PIP-based flow model, our business model is based on the notion of commitments with a declarative semantics. A business model can formally detect if a given business execution is complete. An execution is complete if it leaves no commitments in detached state. Consider Table 3, which shows the commitment

Table 3. Commitment state progression (top to bottom) in the Order-to-Cash business process

PIP	C1	C2	C3	C4	C5
3A4 (sendOrder, acceptOrder)	Active	Active	Inactive	Inactive	Inactive
3A8 (sendChgOrder, acceptChgOrder)	Active	Active	Inactive	Inactive	Inactive
3B12 (sendShipOrder, acceptShipOrder)	Active	Detached	Satisfied	Detached	
4C1 (sendInventory)	Active	Active	Detached	Satisfied	Detached
3B2 (notifyShip)	Satisfied	Detached	Detached	Satisfied	Satisfied
4B2 (notifyGoodsReceipt)	Satisfied	Detached	Detached	Satisfied	Satisfied
3C3 (sendInvoice)	Satisfied	Detached	Detached	Satisfied	Satisfied
3C6 (sendRemitAdvice)	Satisfied	Satisfied	**Detached**	Satisfied	Satisfied

progression corresponding to the process flow of Figure 5. In the end, the process flow execution leaves commitment C3 in a detached state. That is, the shipper commits to shipping the goods, but is never paid by the supplier. This means that the flow shown in Figure 5 is incomplete. It lacks a PIP that enables the supplier to pay the shipper.

At runtime, a participant may violate a commitment. In such a case, a pattern that creates a penalty commitment [13] may apply. For example, if the supplier ships the goods but the customer fails to pay $10 within 15 days to the supplier, that is, the customer violates commitment C1, then the model can activate a new penalty commitment for the customer to pay $15 within 30 days to the supplier.

5 Discussion

This paper presents business patterns abstracted from a small subset of the RosettaNet PIPs. It outlines a simple methodology to apply these patterns to a real-life business scenario. The paper highlights the flexibility that our commitment-based model offers. For example, the business pattern for the Request Quote PIP 3A1 applies to the unsolicited quote scenario as well. The paper shows how our model detects incomplete executions. Our model allows several alternative executions so long as the execution leaves no commitments in the detached state.

RosettaNet PIPs can be divided into two broad categories: PIPs that create or manipulate commitments, and PIPs that merely enable the participants to exchange information. Section 3 describes how we abstract patterns from the PIPs that create or manipulate commitments. We can naturally expand our set of patterns to include PIPs used for information exchange, e.g., as described by Xing et al. [25], which would be suitable for capturing the PIPs focused on notifications. With this expansion, our approach will model all of the RosettaNet PIPs.

5.1 Related Work

Traditional business process modeling approaches, and several related industry standards, are based on low-level concepts of data, and control flows. For example, BPMN

[13] is a leading standard that expresses business processes in terms of sequence flow, and message flow. Such specifications do not capture the business meaning of the interactions, which is better described in terms of how the participants create and manipulate their commitments. Since traditional models ignore business intent of interactions, they over-constrain business behavior by mandating the exchange of a predetermined sequence of messages.

Existing works that aim at creating a catalog of reusable patterns for business interactions, such as the patterns that Zdun et al. [26] propose, concentrate on low-level abstractions. Although they are valuable for characterizing best practices of operationalizations, they do not specify the business relationships between the participants. It would be interesting to combine the patterns from this paper founded on the business relationships with Zdun et al.'s and other such approaches.

Singh et al. [21] propose a set of commitment patterns for business service interactions. They describe a pattern using a statechart that shows relevant elements of the life cycles of the commitments involved. In contrast, this paper describes patterns using a graphical language based on our business metamodel, which functions at a higher level of abstraction than statecharts, and indicates the design-time business relationships among the parties. Our graphical language emphasizes the roles and tasks in addition to showing the commitments. Singh et al. additionally outline a graphical notation based on business relationships: our approach can be thought of as applying in the same spirit, though offering additional refinement in terms of business relationships.

Nitto et al. [12] agree with us that systems in open environments need to be highly dynamic and self-adaptive. Toward that end, they identify the need for natural, high-level design abstractions for modeling such systems. Our research aims at developing such abstractions while satisfying the need for formalization. This paper extracts business patterns from RosettaNet PIPs in terms of the high-level abstractions from our business metamodel.

Kotinurmi et al. [11] and Haller et al. [8] incorporate semantics at the lower-level of data in RosettaNet PIPs. They develop an ontology using the Web Service Modeling Language (WSML) for the PIP payloads and choreographies. In contrast, our work identifies the business-level meaning of the PIPs in terms of the commitments.

Hofreiter et al. [9] present UN/CEFACT's methodology UMM for modeling global choreographies, that is, the business interactions realized by B2B scenarios. Similar to our approach, UMM intends to specify a choreography at a business level, independently of the underlying implementation technology. UMM's business domain view and business requirements view can benefit from our business metamodel by naturally modeling the collaborations in terms of commitments.

Redding et al. [15] propose an artifact-centric approach for flexible business process modeling called FlexConnect. This approach models a process as communicating state machines of the relevant business objects. In contrast, our approach models a business scenario more naturally in terms of business relationships founded upon commitments.

Work on agent-oriented approaches for services is also relevant here. We mentioned the notion of goals above. Although goals are not central to this paper, they are important in understanding why agents, i.e., independent parties, participate in a business interaction. At the design level, the work on Tropos [5] is relevant. At the implementation level, approaches such as Jadex [14] and others for agent programming [4] are crucial. It would be important to relate our approach to such works. With respect to design, we have taken some preliminary steps [23]. With respect to enactments, Avali and Huhns [2] show how to relate commitments to agents who reason based on representations such as beliefs, desires, and intentions.

Research on obligations, norms, and other deontic concepts is relevant here. Section 2.2 compares commitments with obligations. Other normative concepts, such as permissions and prohibitions, can be useful in describing high-level relationships among autonomous parties [19]. Although they are not as central to business transactions as commitments, it would be worth expanding our approach to accommodate them.

Research on temporal logic for representation and reasoning [7] is also relevant. Most temporal approaches for describing processes, however, apply at a lower level of abstraction [20,1]. Our recent effort [24] shows how to express a business model in a temporal language for model checking sequence diagrams. Baldoni et al. [3] draw deeper connections between commitments and time, which it would be interesting to synthesize with this paper.

5.2 Future Directions

This work opens up several interesting directions. Of these, we are pursuing the development of formal techniques that involve formalizing our business patterns so as to verify the compliance of a low-level operational model with respect to a specified business model. We expect also to develop a catalog of well-defined reusable patterns for business modeling, including those for notifications.

The methodology outlined above is a good start on an approach for developing business models based on traditional models. However, a more extensive and complete methodology to specify business models in high-level terms would also be crucial to the greater success of this effort. We hope to pursue such a methodology in future work.

References

1. van der Aalst, W.M.P., Pesic, M.: DecSerFlow: Towards a truly declarative service flow language. In: Bravetti, M., Núñez, M., Zavattaro, G. (eds.) WS-FM 2006. LNCS, vol. 4184, pp. 1–23. Springer, Heidelberg (2006)
2. Avali, V.R., Huhns, M.N.: Commitment-based multiagent decision making. In: Klusch, M., Pěchouček, M., Polleres, A. (eds.) CIA 2008. LNCS (LNAI), vol. 5180, pp. 249–263. Springer, Heidelberg (2008)
3. Baldoni, M., Baroglio, C., Marengo, E.: Commitment-based protocols with behavioral rules and correctness properties of MAS. In: Proc. Intl. Wkshp. Declarative Agent Languages and Technologies (DALT), pp. 66–83 (2010)
4. Bordini, R.H., Dastani, M., Dix, J., Fallah-Seghrouchni, A.E. (eds.): Multi-Agent Programming: Languages, Platforms and Applications. Springer, Heidelberg (2005)
5. Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: An agent-oriented software development methodology. Autonomous Agents and Multi-Agent Systems 8(3), 203–236 (2004)
6. BRG: The business motivation model (2007)
7. Clarke, E.M., Grumberg, O., Peled, D.A.: Model Checking. MIT Press, Cambridge (1999)

8. Haller, A., Kotinurmi, P., Vitvar, T., Oren, E.: Handling heterogeneity in RosettaNet messages. In: Proc. 22nd ACM Symposium on Applied Computing, pp. 1368–1374 (2007)
9. Hofreiter, B., Huemer, C., Liegl, P., Schuster, R., Zapletal, M.: UN/CEFACT's Modeling Methodology (UMM): A UML Profile for B2B e-Commerce. In: Proc. 2nd Intl. Wkshp. Best Practices of UML (ER), pp. 19–31 (2006)
10. Hohpe, G., Woolf, B.: Enterprise Integration Patterns. Addison-Wesley, Boston (2004)
11. Kotinurmi, P., Vitvar, T.: Adding semantics to RosettaNet specifications. In: Proc. 15th Intl. Conf. World Wide Web, pp. 1059–1060 (2006)
12. Nitto, E.D., Ghezzi, C., Metzger, A., Papazoglou, M.P., Pohl, K.: A journey to highly dynamic, self-adaptive service-based applications. Automated Software Engineering 15(3-4), 313–341 (2008)
13. OMG: Business process management initiative (August 2009), http://bpmn.org/
14. Pokahr, A., Braubach, L., Lamersdorf, W.: Jadex: A BDI reasoning engine. In: [4], pp. 149–174 (2005)
15. Redding, G., Dumas, M., ter Hofstede, A.H.M., Iordachescu, A.: Modelling Flexible Processes with Business Objects. In: Proc. IEEE Conf. Comm. & Ent. Comp, pp. 41–48 (2009)
16. RosettaNet: Overview: Clusters, Segments, and PIPs (2008), http://www.rosettanet.org
17. RosettaNet: Extended order to cash (2010), http://www.rosettanet.org/
18. Russell, N., ter Hofstede, A.H.M., van der Aalst, W.M.P., Mulyar, N.: Workflow control-flow patterns: A revised view. Tech. rep., BPMcenter.org (2006)
19. Singh, M.P.: An ontology for commitments in multiagent systems: Toward a unification of normative concepts. Artificial Intelligence and Law 7(1), 97–113 (1999)
20. Singh, M.P.: Distributed enactment of multiagent workflows: Temporal logic for service composition. In: Proc. 2nd Intl. Joint Conf. Autonomous Agents and MultiAgent Systems (AAMAS), pp. 907–914 (July 2003)
21. Singh, M.P., Chopra, A.K., Desai, N.: Commitment-based service-oriented architecture. IEEE Computer 42(11), 72–79 (2009)
22. Telang, P.R., Singh, M.P.: Business modeling via commitments. In: Kowalczyk, R., Vo, Q.B., Maamar, Z., Huhns, M. (eds.) SOCASE 2009. LNCS, vol. 5907, pp. 111–125. Springer, Heidelberg (2009)
23. Telang, P.R., Singh, M.P.: Enhancing Tropos with commitments. In: Borgida, A.T., Chaudhri, V.K., Giorgini, P., Yu, E.S. (eds.) Conceptual Modeling: Foundations and Applications. LNCS, vol. 5600, pp. 417–435. Springer, Heidelberg (2009)
24. Telang, P.R., Singh, M.P.: Specifying and verifying cross-organizational business models: An agent-oriented approach. TR 12, North Carolina State University (May 2010)
25. Xing, J., Wan, F., Rustogi, S.K., Singh, M.P.: A commitment-based approach for business process interoperation. IEICE Trans. Info. & Syst. E84-D (10), 1324–1332 (2001)
26. Zdun, U., Hentrich, C., Dustdar, S.: Modeling process-driven and service-oriented architectures using patterns and pattern primitives. ACM Trans. Web 1(3), 14 (2007)