

Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2006 Proceedings

Americas Conference on Information Systems
(AMCIS)

December 2006

A Speech-Act Based Methodology for System Analysis

Joseph Barjis
Georgia Southern University

Follow this and additional works at: <http://aisel.aisnet.org/amcis2006>

Recommended Citation

Barjis, Joseph, "A Speech-Act Based Methodology for System Analysis" (2006). *AMCIS 2006 Proceedings*. 466.
<http://aisel.aisnet.org/amcis2006/466>

This material is brought to you by the Americas Conference on Information Systems (AMCIS) at AIS Electronic Library (AISeL). It has been accepted for inclusion in AMCIS 2006 Proceedings by an authorized administrator of AIS Electronic Library (AISeL). For more information, please contact elibrary@aisnet.org.

A Speech-Act Based Methodology for System Analysis

Joseph Barjis
Georgia Southern University
PO Box 8150
Statesboro, GA 30460, U.S.A.
JBarjis@GeorgiaSouthern.edu

ABSTRACT

In this paper a modeling methodology is discussed to prepare a basis for subsequent system design. In paving such a ground, the discussed methodology contributes to two aspects of system design: requirements definitions and process modeling. While essential requirements are captured in a series of business transactions, the process model is built through the assembly of these business transactions into a network-like structure. Each business transaction is an essential requirement; or vice versa, each essential requirement is captured as a business transaction. The paper discusses the transaction concept derived from the Speech-Act Theory for capturing requirements for the system to be built, and a type of Petri net used as a modeling technique to build business process models based on the identified transactions. In view of the fact that business processes are characterized by conditional relationships, the concept of the Norm Analysis Method is incorporated into the methodology to govern the execution of certain transactions.

Key words

Petri net, modeling technique, transaction concept, norm analysis method

INTRODUCTION

One of the challenges hindering successful IS design is the execution of proper requirements capturing and process modeling phases prior to the system design. This challenge is complicated by the increasing intensity of human interaction and involvement with information systems that by virtue are social systems. Therefore this paper is based on the Speech-Act Theory (or Language Action Perspective - LAP), which more accurately identifies with social systems, in an effort to capture essential requirements and identify business processes and the related actors.

The main premise of the LAP is that communication means doing and when people (actors) communicate they intend for certain tasks to be carried out (Dietz *et al.*, 1998). There are a number of LAP based methodologies for requirements engineering or process elicitation for the purpose of IS design. Some of these methodologies are thoroughly based on the LAP perspective while others are closely related such as organizational semiotics (Stamper, 1988; Stamper, 1997). Among the most directly based on the LAP framework are the DEMO Methodology (1994), the Action Workflow model (Denning and Medina-Mora, 1995), and the Business Action Theory (BAT) (Goldkuhl, 1996). In this paper the emphasis is put on the transaction concept as introduced within the DEMO Methodology (Dietz, 1994, 1999, 2002). The transaction concept in this paper is represented in a modeling technique using Petri net graphical notations.

While the proposed work is not a research from scratch, original works attempting to study the incorporation of the transaction concept in a modeling technique using Petri nets or similar techniques date as far back as 1999 (Barjis & Dietz, 1999; Dietz & Barjis, 1999). This paper is a further improvement of the models represented in those works. However those initial works were merely focused on the diagrammatic representation of the business transaction. More profound transformation of communication patterns into the Petri net representation was conducted by Dietz (2002) and called CAP-net; although CAP-net is not strictly a Petri net model, it maintains a parallel to Petri net graphical semantics. The current work takes the research to a nearly executable final business process model using Petri nets supported by the Norm Analysis Method. In fact, the proposed modeling technique is an integrated tool combining the transaction concept with the Petri net graphical notations and incorporating the Norm Analysis for strict governing of certain actions. Those readers interested in more reading on the DEMO methodology, the most recent developments of this methodology can be found in (Dietz, 2002). Since the first publication (Dietz and Barjis, 1999), a number of research papers addressed the idea of combining different tools and concepts to produce a more rigorous method and technique to better understand the application domain (Liu *et al.*, 2003). One of the earliest attempts to study and compare two different methods of process modeling within the LAP

community was the comparison of DEMO and BAT (Reijswoud and Lind, 1998). An important fact that was observed during these comparative analyses and integrative approaches is that many modeling techniques and methods are developed with a special interest in the authors' mind. When an approach has a sound theoretical basis, it usually lacks a sound modeling technique; or if an approach has a sound modeling technique it lacks the support of adequate computer tools support. For instance, Linderman *et al.* (2003) argue that Six Sigma, although extensively used in industry, lacks a sound theoretical underpinning. It is built on the assumption that, in order to improve, organizations must implement those processes that have been identified as "best practice" but it fails to specify this concept in more detail, or how an organization ensures that the adopted or designed process is "best practice". In solving this issue, many rigorous methodologies emerged from the combination of two or more approaches. For example, the popular UML (Unified Modeling Language) is a manifest of such a successful integrative modeling method that resulted from the efforts of three different authors combining the best of their modeling knowledge together.

In this paper, it is attempted to further develop and improve the combination of the transaction concept and Petri net modeling technique to produce a more rigorous tool for requirements capturing and processes modeling. It should be noted that the resulting methodology is still based on the philosophical concept of the LAP.

Summarizing this introductory section, it should be noted that the contribution of this paper is an addition of the Norm Analysis Method to make execution of certain processes more explicit and force norms to govern the behavior of actors in certain conditional situations. The *Norm Analysis Method* is one of the methods of the Semiotic Approach (Stamper, 1988). These three concepts are integrated for achieving a more structurally sound and comprehensible tool for better requirements capturing and process modeling to pave a solid ground for system (Information System) design. For the ease of reference, in this paper the proposed methodology is referred to as TOP Methodology (**T**ransaction **O**riented **P**etri nets **M**ethodology).

THE NORM ANALYSIS METHOD

Understanding of the norms and patterns of actors behavior within an organization is a foundation for designing an effective IS. In business, most rules and regulations fall into the category of behavioral norms. These norms prescribe what people *must*, *may*, and *must not* do, which are equivalent to three deontic operators "is obliged," "is permitted," and "is prohibited." There are five types of norm that influence certain aspects of human behavior. They are *perceptual norms*, *cognitive norms*, *evaluative norms*, *behavioral norms* and *denotative norms* (Stamper, 1994).

Of all the norms, particular attention is given to the behavioral norms since they are expressed as business rules, and have direct impacts on business operations. Hence, the following format is considered suitable for specification of behavioral norms:

whenever	<condition>
if	<state>
then	<agent>
is	<deontic operator>
to	<action>

The condition describes a matching situation where the norm is to be applied, and sometimes further specified with a state-clause (this clause is optional). The actor-clause specifies the responsible actor for the action. The actor can be a staff member, a customer, or a computer system if the right of decision-making is delegated to it. As for the next clause, it quantifies a deontic state and usually expresses in one of the three operators - *permitted*, *forbidden* and *obliged*. For the next clause, it defines the consequence of the norm. The consequence possibly leads to an action or to the generation of information for others to act. Incorporation of Norms into business process modeling will be illustrated later in this paper when a real life case study is discussed. The following section introduces the transaction concept.

THE TRANSACTION CONCEPT

The central theoretical concept in TOP is that of a transaction as introduced in the DEMO methodology (Dietz, 1999). The transaction concept is based on the idea that an organization and its underlying business processes can be better understood through the observation of communication between the members of the organization and the organization's interaction with its environment.

In order to introduce the concepts of a transaction and business process, consider the following, artificial, example:

A visitor calls a hotel’s reception to reserve a room. After being asked to provide the details of the required room and the dates of his intended visit, the visitor is asked to hold as the receptionist checks the availability of a room. While the visitor is on hold, a piece of Mozart is played over the phone. After a brief period, the receptionist gets back to the visitor and states that the hotel has an appropriate room available for the dates required. However, in order to confirm the reservation and give the visitor a confirmation number, the visitor must pay for the room by credit card. The receptionist therefore asks the visitor for credit card information and, after receiving the information, verifies the card and availability of sufficient funds with the credit card company. Once the payment has been approved, the receptionist is able to give the confirmation number to the visitor.

Business transactions are patterns of actions and interactions, as illustrated in Figure 1. The *action* is the core of a business transaction and represents an activity that changes the state of the world and creates a new fact. An *interaction* is either the initiation of an action or the communication of a fact as the result of the action. An example of an interaction is a request made by one actor towards another actor that leads to creation of a new fact. Other examples of interactions are clicking an “apply” button, “submit” in an electronic form, or inserting a debit card into an ATM to withdraw cash. In the hotel reservation example, the visitor’s request for a room reservation and the receptionist’s statement of the confirmation number are both *interactions*; and reserving the room by the receptionist is an *action*.

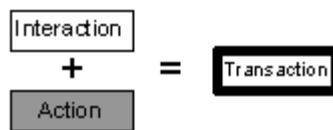


Figure 1. The business transaction concept

Each business transaction consists of three phases, the *Order phase*, the *Execution phase*, and the *Result phase*, as illustrated in Figure 2. In the order phase, an agent, the *initiator*, initiates the transaction by making a request of another agent, the *executor*, to perform some action. In the execution phase, the executor performs the action. In the result phase, the executor informs the initiator of the changes that have occurred as a result of the action. In this context, agents can be human actors, software agents or machines. Figure 21, in which the order, execution and result phases are abbreviated as O, E and R respectively, illustrates this analysis. Note that the order (O) phase and result (R) phase are interactions and the execution (E) phase an action, therefore in Figure 2 they are represented by different colors.

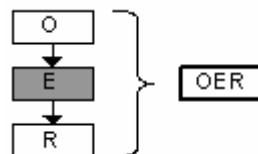


Figure 2. The Order, Execution, Result phases in a business transaction

As it becomes obvious, we must now distinguish between simple and composite transactions. Business processes typically consist of numerous transactions that are chained together and nested into each other. *Simple* transactions do not involve, i.e. trigger or cause, other transactions during their execution. In *composite* transactions, on the other hand, one or more phases will trigger further, nested, transactions. Thus, in the hotel reservation example above, the receptionist can only make the reservation after she has received a credit card payment from the visitor. However, this transaction can only be completed once the receptionist has made the necessary checks with the credit card company. Thus, the “receive payment” transaction can only be completed once the receptionist initiates a transaction with a credit card company and this transaction maybe nested inside a larger transaction.

The hotel reservation example also illustrates that each transaction involves two actors. The actor that initiates the transaction is called the *initiator* of the transaction, while the actor that executes the transaction is called the *executor* of the transaction.

This leads to the following series of statements of the concept of a business transaction:

A *business transaction* is a generic pattern of activity carried out between two actors, an initiator and an executor. The activity is carried out in three phases, the order phase, the execution phase, and the result

phase, where the first and last phases are interactions and the execution phase is an action. The activity creates a new fact and changes the state of the world.

A *business transaction* comprises two types of activities: communicative activities, which represent an *interaction*, and productive activities, which represent an *action*. An *interaction* is coordinating or negotiating the essence of an *action* and/or communicating the result of an action. Thus, a business transaction involves two interactions: one before an action is carried out and one after an action is carried out and a result is achieved.

A *business transaction* can embed one or more business transactions. If so, the primary transaction is called a *composite* (or nesting) transaction and the embedded transactions are called *nested* transactions.

A *business transaction* has a single start point and a single end point. A transaction starts with a request by an actor and ends with a result accepted by the same actor.

As the example above illustrated, the initiation, execution, or completion of a business transaction may lead to the initiation and execution of new transactions. In this way transactions are chained into arbitrarily large structures, called *business processes* (Dietz, 1999).

This then leads to the following definition of the concept of a business process:

A *business process* is a network of interrelated business transactions that delivers value to an external agent through the production of products (goods or service)

TRANSACTION-ORIENTED PETRI NETS

The limited space of the paper forces us to assume that readers are well familiar with Petri nets, therefore discussion of Petri nets are skipped in this paper; however, novices in Petri nets or interested readers are referred to (Peterson, 1981; Agostini and Michelis, 1998; Aalst and Hee, 2002). In this section we only discuss the extensions made to Petri nets.

Although especially hierarchical Petri nets are eminently suitable for business process modeling, we add a few minor extensions. The primary extension derives from the fact that basic Petri nets do not allow one to model the fact that different parts of the process being modeled take place in units that are somehow distinguishable. Clearly, the ability to do is potentially of great importance in business process modeling, as one needs to be able to distinguish between departments, and indeed organizations involved in the same business process. We therefore provide a simple extension to Petri nets that allows us to do so. Table 1 gives the set of basic elements that we use in building Transaction-Oriented Petri Nets.











Element	Notation	Description
Start		A start represents a starting point for a process that leads to the creation of a new result or a previously unknown fact.
Interaction		An interaction represents the transfer of some information from one actor to another, e.g. a request to perform some action, or a communication of the result of some action. In terms of the OER model, both the Order and the Result phases are interactions.
Action		An action represents a production act when a certain result is created. The Execution phase in the OER model is an action.
Transaction		A transaction encompasses the entire OER cycle. The introduction of a symbol for an entire transaction shows that our notation is an example of a hierarchical Petri net. The reason for the introduction is that it allows the modeler to abstract away from certain details that he or she considers irrelevant to the model.
Composite Transaction		A composite transaction is a set of transactions, and thus summarizes a sub-net. It is useful if the modeler wants to hide certain details.
Sequence Flow		A sequence flow indicates the order in which interactions and actions are initiated and performed.
Optional		An optional link represents a link that is neither a condition for proceeding nor essential, but one that usually takes place. E.g., before applying for a policy, customers usually request a quote, but customers can apply for a policy without first requesting a quote.
Boundary		A boundary illustrates the boundary of a business process (department or organization). It is this feature that allows the modeling of intra- and inter-departmental and intra- and inter-organizational processes.
Intermediate state		An intermediate state represents a result or state achieved after an interaction or action.
End		An end notation represents a termination point for a process, and thus represents the final result of the process.

Table 1. Elements in Transaction-Oriented Petri Nets

As Table 1 illustrates, the Transaction-Oriented Petri nets contain distinct graphical elements for actions and interactions. Moreover, they include short-cut notations for complete transactions and composite transactions, consisting of a subset of transactions. Transaction-Oriented Petri nets are therefore an example of hierarchical Petri nets. The reason for introducing short-cut notations for complete and composite transactions is to enable the modeler to (temporarily) hide details of the model under construction. This feature is useful both when it comes to constructing the model (modelers can temporarily ignore the details of certain transactions as they concentrate on other transactions within the business process) and when seeking feedback on a model under construction from a stake-holder (hiding irrelevant information from stake-holders makes it easier for them to provide feedback on those aspects of the model that are of more direct relevance to them.) Since these are transitions in the sense in which the term is used in Petri nets, all are represented by rectangles.

A further extension is the inclusion of two special places, a *start place* and an *end place*, to indicate where a process starts and terminates respectively. Because the start and end place are special places, we represent them by two different types of circles.

A third extension is the inclusion of the *boundary* element. It is this element that allows us to distinguish between intra-organizational and inter-organizational processes. This element allows an analyst to model the interaction of one process with another process within an organization or with the environment. This interaction can be modeled with a set of places between the process boundaries.

The final extension concerns the introduction of an *optional link*. In standard Petri nets, all input places must hold in order for a transition to be executed, that is, execution of a transition is guaranteed only whenever all its input places have a token in them. Optional links weaken this assumption and therefore allow the analyst to represent situations where the transition is

executed, even when a state represented by its input places does not hold. For instance, prior to applying for a policy, customers usually request a quote. However, it is entirely possible for customers to apply for a policy without first requesting a quote. Thus, the relationship between requesting a quote and applying for a policy is an optional one.

In concluding this section, Figure 3 represents a business transaction (and its three phases known as OER) using elements of TOP that we just introduced both in detailed and compact forms.

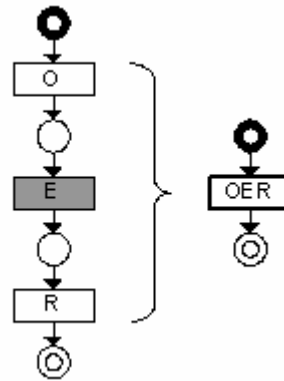


Figure 3. The OER diagram using Petri net (detailed and compact)

An Example of a Transaction-Oriented Petri Net

In order to illustrate Transaction-Oriented Petri nets, we build a partial model of the hotel reservation example. Figure 4 represents the hotel reservation process at a very high level, and simply shows the process as having a starting place and an end place, and being conducted within the confines of the hotel, at least as far as the initiator is concerned.

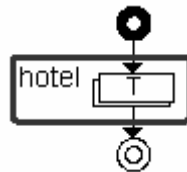


Figure 4. The high-level “Room Reservation” model

However, a small degree of thought will quickly show that the analysis is too simple. While the visitor will initiate the business process, in order for the process to be completed, the hotel must receive payment from the visitor. Paying is an action initiated by the receptionist, who asks for the payment, but executed by the visitor. We therefore have two transactions:

Transaction 1:	Reserving a room
Initiator:	Visitor
Executor:	Receptionist
Fact:	A room is reserved
Transaction 2:	Making payment
Initiator:	Receptionist
Executor:	Visitor
Fact:	Payment is made

Since the second transaction is triggered while the first transaction is being completed, we have to expand our representation of the first transaction to make its three phases explicit. Figure 5 illustrates:

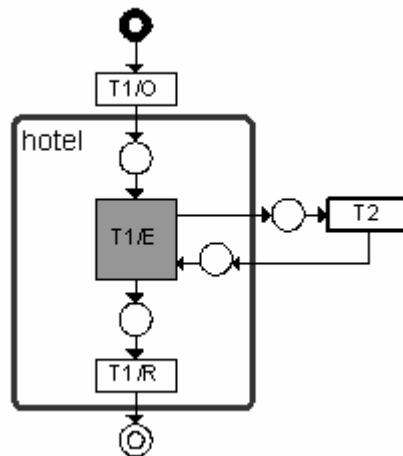


Figure 5. The detailed “Room Reservation” model

There is one further clarification that needs to be made regarding Figure 4. At first glance, it may seem that the model in this figure is defective because it does not represent a clear sequence of actions. After all, in order for the process to be completed successfully, transaction 2 must be completed before transition T1/E can be completed. We therefore assume that if any transition calls a nested transaction and the result of the nested transaction is needed for the completion of the nesting transaction, the nested transaction must be completed before the transition itself can finish. Figure 6 illustrates the sequence of transitions through the model. According to this model, the execution phase of Transaction 1 (T1/E) starts and before completing this phase, it requires the result of Transaction 2.

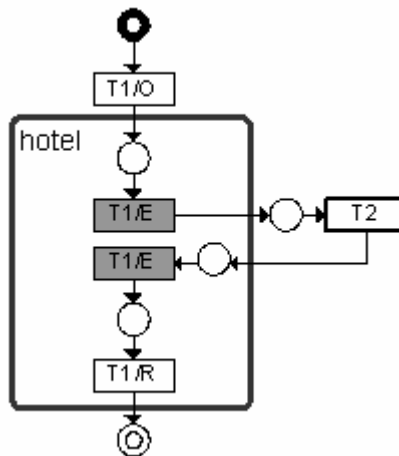


Figure 6. Flow of action in the detailed “Room Reservation” model

In order to provide additional clarification of TOP, we further complicate the hotel example. There are cases in which a receptionist may have to cancel an existing reservation before requesting payment for the new reservation. For example, many hotel chains allow employees to reserve rooms but only on the understanding that the reservation may be cancelled if there is no demand for that room. In order to cancel an existing reservation, the receptionist must interact with the reservation system (e.g., database). This inherently changes the list of the transactions involved in the “Room Reservation” process:

Transaction 1:	Reserving a room
Initiator:	Visitor
Executor:	Receptionist
Fact:	A room is reserved
Transaction 2:	Canceling a reservation
Initiator:	Receptionist
Executor:	Database
Fact:	Cancellation is confirmed
Transaction 3:	Making payment
Initiator:	Receptionist
Executor:	Visitor
Fact:	Payment is made

The modified model of the “Room Reservation” incorporating the new transaction is illustrated in Figure 7. In this figure both the high level TOP model (a) and the lower-level Petri net model (b) are illustrated.

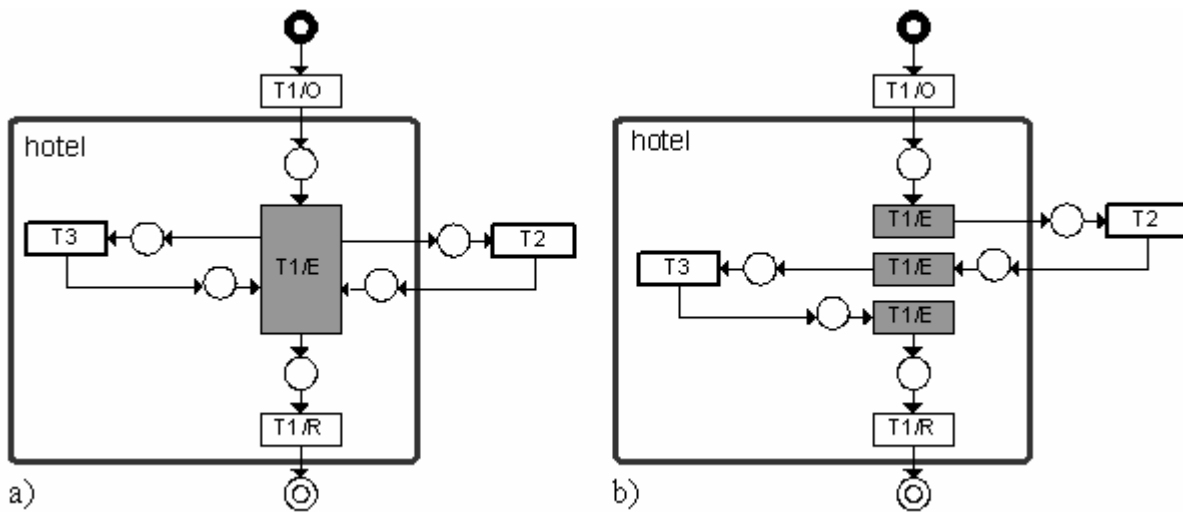


Figure 7. a) Modified model of “Room Reservation”; b) and its low level Petri net model

In the above modified model we stated that the cancellation of existing reservation should take place first before the receptionist can request for a payment. However, it is possible for these actions to take place in parallel. Figure 8 illustrates this situation.

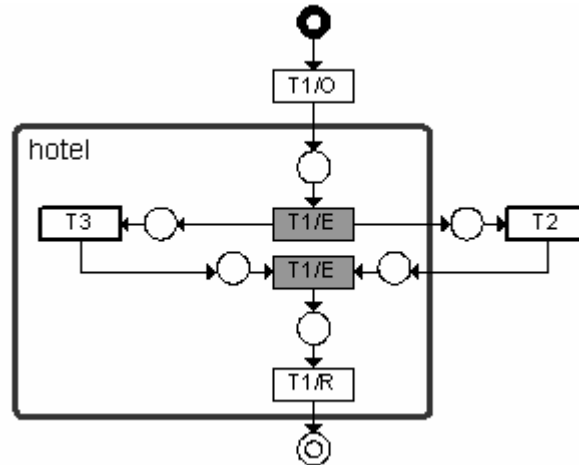


Figure 8. Modified low level Petri net model of “Room Reservation”

The following summarizes the concept of a transaction oriented Petri net:

The *Transaction-Oriented Petri net* (TOP) is a nested model that can be mapped into (or represented as) a standard place-transition Petri net. The TOP models both communicative activities (interactions) and productive activities (actions). The main building blocks of TOP are business transactions. Each transaction is represented as a set of three transitions whereas each transition corresponds to one of the three transaction phases (OER). Two of these three transitions represent an *interaction* and one of these three transitions represents an *action*. Overall, TOP can be classified as a hierarchical Petri net.

The *Transaction-Oriented Petri net* (TOP) is a process oriented model capturing essential atomic processes, whereas underlying detailed actions are kept encapsulated. Therefore, TOP does not illustrate decision points as it would be anticipated from a detailed low-level Petri net. In other words, each transition of TOP is subject to further decomposition to capture true and false conditions for each atomic action.

In the following section we demonstrate application of the TOP Methodology.

APPLICATION OF THE TOP METHODOLOGY

This section illustrates the TOP methodology through a case study conducted in a major insurance company identified by the fictitious name “SSM Insurance” currently having 80,200 employees and 17,000 agents servicing more than 70 million policies in the US and Canada. Although SSM Insurance offers many different types of insurance, we restrict ourselves to auto insurance.

The major business processes within SSM Insurance are *reviewing new applicants*, *generating contracts*, *reviewing claims*, *reviewing current applicants*, and *servicing customers daily*.

Description: Applying for Insurance Process

If a person wants SSM Insurance Company to be his/her insurance agency he/she contacts an SSM Insurance Company agent. These agents then offer plans and the prices to best fit the person. The premium offered is based on the customer’s driving record. If the person wishes to purchase the insurance he/she then fills out an application for insurance with the agent. The agent then sends the application to one of the regional offices. The regional office checks the background of the applicant. If approved, the regional office notifies the agent who informs the client that they were approved and a contract is generated. The contracts are produced at the regional office and are sent to the customer’s agent, who adds the type of coverage the customer has. The Agent then explains what is and is not covered under the plan and asks the client to sign the contract. By signing the contract, the customer pays an initial payment to start the coverage when approved.

Identification of Business Transactions and Actors

Now, based on the description of the business process in regard to the policy issuance, business transactions and their relevant actors are identified using the transaction concept. As mentioned, each business transaction entails two distinct actors and brings about a new fact or result.

Following this concept, the first business transaction can be identified as “obtaining a quote” or “requesting a quote.”

Transaction 1:	Requesting for a quote
Initiator:	customer
Executor:	agent
Fact:	a quote is given

After completion of this transaction, the customer may decline or consider applying for an insurance policy. If the customer decides to apply for a policy, this will constitute the second transaction:

Transaction 2:	Applying for a policy
Initiator:	customer
Executor:	agent
Fact:	a policy is issued

In order for the agent to proceed, the agent must request approval of the regional office, which checks the customer’s background:

Transaction 3:	Requesting regional approval
Initiator:	agent
Executor:	regional office
Fact:	approved/declined

If the approval is given, the agent asks the regional office to generate a contract based on the inputs of the customer:

Transaction 4:	Generating a contract
Initiator:	agent
Executor:	regional office
Fact:	a contract is generated

Now the agent reviews and explains the prepared contract to the customer (agent and customer). If the customer agrees to the terms and conditions of the contract, the agent requests the customer to sign the contract and make the initial payment:

Transaction 5:	Paying for the policy
Initiator:	agent
Executor:	customer
Fact:	policy is paid

Identification of Norms

We just illustrated how essential business transactions can be identified and the relevant actors defined. As the transaction concept is concerned about the essential business transactions, many behavioral norms simply remain undisclosed; therefore in analyzing the constructed models, it is not clear how conditional transactions should be dealt with, especially when conducting simulation of the models. To that extent, the Norm Analysis Method is used as a suitable complement to deal with behavioral norms specification. Recall the format we discussed above for specification of behavioral norms:

whenever	<condition>
if	<state>
then	<agent>
is	<deontic operator>
to	<action>

Now using this format along with identified business transaction, an analyst can build a more complete model of business processes that may be used as a significant input for IS design or IT Application development. Addition of behavioral norms to the models will eliminate execution ambiguity. The following are some norms specified based on the SSM case example introduced above. We will use a numbered capital “N” for norms.

N1:

whenever	<a new member applies for a policy>
if	<the member is approved by the regional office>
then	<agent>
is	<permitted>
to	<generate a contract>

N2:

whenever	<a new member applies for a policy>
if	<the regional office does not approve>
then	<agent>
is	<prohibited>
to	<award a contract>

N3:

whenever	<buying a new policy>
if	<the member agrees with the terms and conditions of the policy>
then	<agent>
is	<obliged>
to	<request initial payment from the new member>

These behavioral norms can be attached to the corresponding business transactions. For example, behavioral norms N1 and N2 can be attached to Transaction T4 - generating a contract; that is, a conditional transaction can be governed by norms that dictate whether or not to execute it. Similarly, behavioral norm N3 can be attached to Transaction T5 – paying for the policy. These attachments are shown in the business process model developed in the next section.

The above identified behavioral norms are based on purposefully simplified scenarios. A more detailed and realistic scope will reveal considerably more norms. However, the objective here is to simply illustrate how norms can be specified and attached to business transactions. As a general rule, all optional or conditional transactions will be complemented by one or more behavioral norms that will control whether the transaction takes place or not. Now that we have briefly learned how behavioral norms can be specified, it is time to implement all the obtained results into one concise representation – a business process model.

Constructing a Business Process Model of SSM Insurance

After having all the business transactions identified in the previous section, the TOP graphical elements are applied to construct a business process model based on these transactions by first drawing the boundary of the two departments – the agent office and the regional office. Second, transactions that are executed within the agent’s office, regional office and environment are placed within the according boundaries. Figure 9 shows all the transactions in a sequential order and their relation in regard to the departmental boundaries.

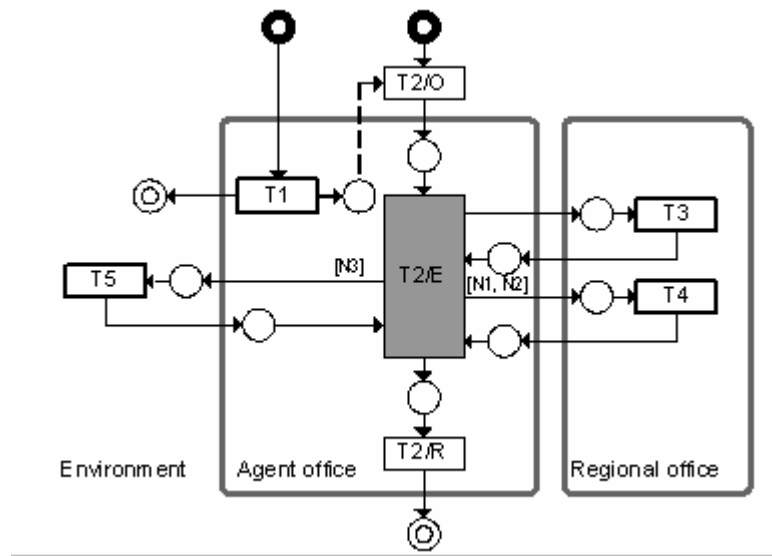


Figure 9. Transactional model of “policy issuing process”

The figure above should be read in the following manner. The big grey rectangles represent boundaries of processes (e.g., agent office, regional office). The two circles with black dots (tokens) represent the start point; correspondingly, the two circles with the holes represent the end point of the processes, where the processes terminate. Transaction T1 starts in the environment and the result is also communicated to the environment. This is a simple transaction and is, therefore, executed completely at once; its result can serve as a conditional link for the initiation of Transaction T2; therefore a dotted arrow is connecting the result of transaction T1 to Transaction T2. Now, Transaction T2 is a composite transaction that nests Transactions T3, T4, and T5; therefore, this transaction is split into three phases showing that for the completion of Transaction T2, Transactions T3, T4, and T5 are initiated and should be completed before proceeding with Transaction T2. As for Transactions T3 and T4, they are initiated in the Agent Office, executed in the Regional Office, and the results are communicated back to the Agent Office. Transaction T5, payment for the policy, is initiated in the Agent Office, executed in the environment, and the result is communicated back to the Agent Office. A final aspect of the above figure is the distinction between intra-organizational and inter-organizational processes. For example, Transactions T3 and T4 are inter-organizational processes. In this case the actors involved are not persons but rather organizations; however, detailed discussion of this matter will be left for a future opportunity.

CONCLUSION AND FUTURE RESEARCH

This paper has presented the TOP methodology for capturing essential requirements in the form of business transactions and constructing business process models for subsequent system (Information System) design. The TOP methodology combines a sound theoretical grounding in the concept of a transaction as developed within the Language Action Perspective with a powerful, yet easy to use graphical notations in the form of Transaction Oriented Petri nets. We have tested the TOP methodology through a variety of case studies, one of which we presented in this paper, although not completely due to space limitations.

There are a number of avenues for future research that we intend to explore. First, we will continue to use the TOP methodology to build additional models and implore others to do so as well. As more models are constructed, we will gain a better understanding of the strengths and weaknesses of the current notation. While we believe it is relatively easy to learn (a belief that is partly based on the fact that some of our students have been able to use the TOP methodology to build business process models), it is useful to put this belief to an empirical test. Further testing of the methodology may also reveal weaknesses in our graphical notation. It may, for example, be necessary to explicitly include the initiator and executor of a transaction. Also, continuous development of models, certainly for organizations in a similar line of business, will allow us to build a library of business processes. This in turn can be used to identify best practices.

Second, until now our work has primarily concentrated on building static models. However, one of the attractions of Petri nets is that a model, once constructed, can be used to simulate a business process as well, and we are currently working on software that allows us to make our models “active” and run simulations. Since our main graphical tool is based on the widely accepted Petri net notation, we expect to be able to use an existing simulation package

This paper then merely presents a first step towards a comprehensive set of tools and techniques that should help a business analyst capture requirements, model, simulate and analyze business processes. However, we believe this to be an important and promising first step.

REFERENCES

1. Aalst, W. van der; Hee, K. van (2002). *Workflow Management: Models, Methods, and Systems*, MIT Press.
2. Agostini, A.; Michelis, G. De. (1998). Simple Workflow Models. In the proceedings of the Workshop on Workflow Management: Net-based Concepts, Models, Techniques and Tools (WFM'98), Lisbon, Portugal, June 22.
3. Barjis J., & Dietz J.L.G. (1999). Business Process Modeling and Analysis Using GERT Networks. In the proceedings of the First International Conference on Enterprise Information Systems (ICEIS'99), Setubal, Portugal, March 27-30, 1999, volume II, pp. 748-756. ISBN: 972-98050-0-8.
4. Deiters, W. (1998). Information Gathering and Process Modeling in a Petri Net Based Approach. In Aalst, W. van der, Desel, J., Oberweis, A. (ed.) *Business Process Management: Models, Techniques and Empirical Studies*. Springer-Verlag Berlin Heidelberg New York
5. Denning, P.J. and R. Medina-Mora (1995). Completing the loops. *Interfaces*, 25 (3).
6. Dietz J L G (1994) Business Modeling for Business Redesign. Proceedings of the 27th Hawaii International Conference on System Sciences, IEEE Computer Society Press. Los Alamitos, pp. 723-732.
7. Dietz J.L.G, Goldkuhl G., Lind M., Reijswoud V.E. van (1998). The Communicative Paradigm for Business Modelling – A Research Agenda. Proceedings of the Third International Workshop – The Language Action Perspective on Communication Modelling, Jönköping International Business School, Sweden
8. Dietz, J.L.G., Barjis, J. (1999). Supporting the DEMO Methodology with a Business Oriented Petri Net. In the proceedings of the Fourth CAiSE/ IFIP8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'99), Heidelberg, Germany, June 14-15.
9. Dietz, J.L.G. (1999). Understanding and modeling business processes with DEMO. In the Proceedings of the Annual International Conference on Conceptual Modeling (ER'99), Paris, November.
10. Dietz, J.L.G. (2002). The Atoms, Molecules and Matter of Organizations. In the Proceedings of the Seventh International Workshop on the LAP, Delft, Netherlands, ISBN: 90-9015981-9.
11. Goldkuhl, G. (1996). Generic business frameworks and action modeling. In: Proceedings of Conference Language/Action Perspective '96, Springer Verlag.
12. Linderman, K., Schroeder, R.G., Zaheer, S., & Choo, A.S. (2003). Six Sigma: a goal-theoretic perspective. *Journal of Operations Management*, Volume 21, Issue 2, pp. 193-203
13. Liu, K., Sun, L., Barjis, J., Dietz, J.L.G. Modeling Dynamic Behavior of Business Organizations: extension of DEMO from a semiotic perspective. *Knowledge-Based Systems Journal*, Volume 16, Issue 2 (March 2003), pp. 101-111.
14. Peterson, J. L. (1981) *Petri net theory and the modeling of systems*. Prentice-Hall, Inc., Englewood Cliffs, NJ.
15. Reijswoud V.E. van, Lind M. (1998). Comparing Two Business Modeling Approaches in the Language Action Perspective. Proceedings of the Third International Workshop – The Language Action Perspective on Communication Modeling, Jönköping International Business School, Sweden.
16. Stamper, R.K. (1988). *MEASUR*. University of Twente. Enschede, The Netherlands.
17. Stamper, R.K. (1994) Social Norms in Requirements Analysis, in Jirotko, et al.
18. Stamper, R.K. (1997). *Organizational Semiotics*. In Mingers J. & Stowell F. (eds.) *Information Systems: An Emerging Discipline*. Mc Graw Hill, London