Computing and Informatics, Vol. 32, 2013, 225–250

REDUCING THE GAP BETWEEN BUSINESS AND INFORMATION SYSTEMS THROUGH COMPLEX EVENT PROCESSING

César Augusto L. Oliveira, Natália Cabral Silva
Cecília Leite Sabat, Ricardo Massa F. Lima

Center for Informatics, Federal University of Pernambuco
Cidade Universitária, 50740-560 Recife-PE, Brazil
e-mail: {calo, ncs, cls, rmfl}@cin.ufpe.br

Communicated by Patrick Brézillon

Abstract. According to the Object Management Group, a rule is a proposition that is a claim of obligation or of necessity. The concept of rule is usually employed in the context of business process to manage companies operations. While a workflow is an explicit specification of tasks' execution flow, *business rules* only impose restrictions on the tasks' execution. This provides a great deal of flexibility for the process execution, since the stakeholders are free to choose an execution flow which does not violate the rules. The execution of a task in a process can be seen as the occurrence of an event, which may enable/disable the execution of some other tasks in the process. Event-driven programming is a paradigm in which the program control-flow is determined by the occurrence of events. The capacity to handle processes that are unpredictably non-linear and dynamic makes the event-driven paradigm an effective solution for the implementation of business rules. However, the connection between the business rules and their implementation through event-driven programming has been made in an ad-hoc and unstructured manner. This paper proposes a methodology to tackle such a problem by systematically moving from business rules described in natural language toward a concrete implementation of a business process. We use complex event processing (CEP) to implement the process. CEP relies on the event driven paradigm for monitoring and processing events. The methodology allows for the active participation of business people at all stages of the refinement process. Throughout the paper, we show how our methodology was employed to implement the operations of the World Bank.

Keywords: Business rules, complex event processing, business process, event driven

Mathematics Subject Classification 2010: 68U35

1 INTRODUCTION

The construction of enterprise information systems is a complex task, involving several domains of knowledge and a variety of technologies and methodologies. It requires integrating and coordinating knowledge from many individuals that have different skills and distinct expectations about the software applications [26]. In such complex scenario, harmonizing and fulfilling the interests of different stakeholders in the development of a company's information systems is essential.

The subject of IT alignment has been one of the top 10 issues for IT executives in the past decade [20]. However, there are many companies that do not understand or underestimate the role of IT in their business planning. For example, in a survey on the pharmaceutical industry in USA, Nash [20] verified that several organizations do not adopt management practices to disseminate business objectives among the IT staff. The study also reveals a poor level of communication and partnership between business and IT people.

Peppard and Ward [26] emphasize the importance of IT managers skills to attain a sustainable competitive advantage. They state that IT managers must develop a business view and must actively participate in business discussions and decisions. Also, executives must be aware of the role of IT as a strategic tool. To accomplish that, organizations must establish processes that nurture such an integration.

The development of enterprise information systems usually employs many technologies and frameworks: informal process descriptions, UML models, data-flow diagrams, Enterprise Resource Planning (ERP) applications, databases, workflow languages, business intelligence, and so forth. Enterprises often are forced to integrate these many frameworks in an ad-hoc manner. According to Therani [19], this can introduce a series of semantic mismatches and information loss when making correspondence from one technology to the other. Furthermore, due to semantic incompatibilities, changes at one level of abstraction can possibly not be adequately propagated to other levels. Therefore, in order to achieve the sought-after IT alignment, organizations need systematic means to reduce the semantic gap between the many abstraction levels and different technologies involved in the implementation of their business processes.

According to the Object Management Group, a rule is a proposition that is a claim of obligation or of necessity [25]. Business people naturally talk about and pay attention to such rules during their daily routines. Thus, it is natural for them to write down these obligations and necessities in terms of a set of rules, introducing the notion of *business rules*. Computational engines can be developed to interpret business rules and automatically monitor and enforce their application through an information system. Due to the aforementioned semantic proximity of business rules to the level of discussion of business people, they can provide a way for organizations to reduce the gap between business needs and their implementation as an information system [2, 21, 6].

Another positive aspect of business rules is to allow a flexible execution of business process. The declarative nature of rules gives more freedom to stakeholders to determine how to execute a process. On the opposite side is the workflow imperative model, which requires a complete specification of all possible process' execution path. This is more restrictive, since the designer must anticipate all alternatives for executing a given process. However, depending on the scenario during the process execution, it may be more convenient to do something different from the process workflow specification. Unfortunately, although such new execution flow is permitted according the company's rules, it will be prohibited by the workflow. Business rules overcome such limitation, since designers are not supposed to anticipate the alternatives for the process execution. Instead, business rules allow designers to specify only "what to do" and not "how to do" in the business process description.

Although it is possible to implement computational engines to interpret business rules, in most cases, translating rules described in natural language into executable rules is still made in an ad-hoc manner [19]. Also, several management systems such as ILOG [5] and Tibco [14] force business people to adopt a programming language to express company's needs and obligations (rules). This strategy is at the same time not natural for them and error prone [32]. Moreover, converting business rules into executable rules is not simply a question of syntactical transformation. As a result, there is no guarantee that the executable rules implemented actually represent the business requirements.

The execution of a task in a business process has the effect of modifying an enterprise state that has a relevant business meaning for the business operations or for management. Such task execution can be naturally interpreted as the occurrence of an event, which enables/disables the execution of some other tasks in the process. In the event-driven programming paradigm a program control-flow is determined by the occurrence of events. Beside the natural interpretation of task executions as events, the capacity to handle processes that are unpredictably non-linear and dynamic makes the event-driven paradigm an effective solution for the implementation of business rules.

In this paper, we propose a systematic methodology for integrating business rules analysis and implementation. We use complex event processing (CEP) [18] to implement business rules. CEP relies on the event-driven architecture (EDA) [18] for monitoring and processing events. EDA has been extensively used to implement business rules, business processes, and business intelligence both in academic and commercial applications [6, 7, 21, 18].

The methodology proposed aims at supporting the implementation of business processes directly from the business rules definitions. Business people can *understand* and *actively participate* in this implementation process from a business perspective. Literature shows that closing the gap between business and software development in a systematic way and avoiding ad-hoc approaches is a critical demand today [33, 19, 26, 9].

Our methodology consists of a series of refinement steps by which the events and activities are modeled on the basis of a set of business rules provided. We argue that the active participation of business people throughout the whole development process may significantly reduce the semantic gap between the business rules and the derived process implementation.

In order to show the feasibility of our approach, we modeled and implemented a real business process from the World Bank. The World Bank is an international financial institution that provides financial and technical assistance for developing countries around the world. The institution is owned by 187 countries and has more than 10 000 employees in more than 100 offices worldwide. Therefore, being able to effectively implement its business rules and business policies in all offices and to assure coherent business processes is a critical necessity.

The main contributions of this paper are threefold:

1. a systematic way for refining a set of high level business rules into low level executable rules, allowing the involvement of business and IT people in all phases of the translation process;
2. an ontology for event-driven systems that constitutes the communication interface between business people and developers;
3. a software architecture to support the implementation of business rules through complex event processing (CEP) [18] upon the adoption of the refinement steps in the proposed methodology.

1.1 Structure of the Paper

This paper is structured as follows: The background is presented in Section 2. Section 3 discusses related works in the area. Section 4 presents an ontology for event-driven systems. Section 5 presents the phases of the refinement process that support the methodology proposed in this paper. An example is shown in Section 6 in order to illustrate the application of the methodology. Finally, Section 7 discusses the conclusions of the paper.

2 BACKGROUND

Recent advances in information systems development have been directed towards the separation of business logic from software abstractions [30]. Through this separation, business people can concentrate on business issues while developers focus on providing the supportive infrastructure for implementing the business demands. Although this separation of concerns is desirable and advantageous, it may stimulate the isolation of these two perspectives and block the communication between business and IT people. In this context, tools, processes, and capabilities are required to ensure the *alignment* between business and information technology (IT) [15, 28, 23].

In the beginning of the nineties, *workflow management* [4] became a major driver of efforts to close the gap between business and IT. Business Process Management (BPM) turned to be a frequent topic at academic and industrial IT discussion forums [33, 6, 13]. In the years that followed, several technologies were developed to further improve the capabilities of enterprises to manage their operations and business processes, which, in turn, keep becoming more complex. One remarkable example of such technology is the Service-Oriented Architecture (SOA) [22, 17], which introduces the concept of *services*: small software units that provide a business functionality. The objective is to enable the construction of *flexible* information systems, i.e., systems that can be easily modified to attain new goals. The flexibility allows organizations to quickly adapt to environmental changes in order to capture maximum value from new oportunities [23, 9, 6].

In this context, there are two important frameworks developed to support the alignment between business and information systems: *Business rules* [7] and *Event-driven architectures* [18, 6].

2.1 Business Rules

Business rules is a framework with which business people can store and communicate necessities and obligations related to business operations and management functions [2]. Business rules are declarative, in the sense that they do not describe any explicit sequencing of activities but only define *what* must be done. Such rules are interpreted and implemented in information systems in different ways. The idea behind the business rules is that rules must be managed by business people while information systems are, in principle, expected to understand and incorporate these rules [7].

Business rules can be grouped into three basic classes [30]:

- **Derivation rules**: describe how data can be computed from or related to other data;
  E.g. *A customer who did not buy any product in the last one year is an inactive customer.*
- **Constraint rules**: put restrictions on the state of the organization (and, therefore, on the data stored);
  E.g. *Every driver on a car rental must be over 21 years old* [24].
- **Action/Event rules**: specify or restrict what is to be done and when.
  E.g. *It is obligatory that the insurer of the operating company is notified of each overdue rental* [25].

Several decisions must be made by the software engineer when implementing the business rules exemplified above. For example, *when will an active customer be marked as inactive*?, *how the registering of a driver with less than 21 years old will be blocked*?, and *when the database will be queried for searching for overdue rentals?*. These decisions usually depend on the technology employed to implement the process and on the experience of the engineer. Moreover, the method used to model the software artifacts for each rule is mainly ad-hoc.

Moreover, Bajec and Krispen [2] affirm that, to be able to keep information systems consistent with the business requirements, it is necessary to document how business rules evolve from their conception to their implementation. However, for most companies, as their systems evolve, the absence of methodologies for traceability of requirements spread inconsistencies accross the system.

2.2 Event-Driven Architecture (EDA)

Event-driven architectures (EDA) [18] have been attracting attention from industry and academy in last years [6]. It brings the opportunity to create information systems that are more dynamic and responsive to environmental changes. An event is any change in the state of the enterprise or its business environment that has a business meaning. EDA is commonly employed as a way to implement business rules and business intelligence.

The concept of business rules often entagles with event-driven technologies. Indeed, event-driven technologies are able to manage processes that are unpredictably non-linear and dynamic. This capacity makes such technologies an effective solution for the implementation of business rules. As a result, many frameworks employ EDA to implement business rules engines. In this regard, there are two main approaches: *Event-Condition-Action* (ECA) rules [7] and *Complex Event Processing* (CEP) [18]. ECA statements are modeled as IF/THEN production rules, while CEP statements are based on the idea of *patterns* of events and *agents* that execute actions when these patterns are recognized.

Although ECA and CEP are rich technologies for the implementation of business rules, the integration between business rules on the business level and the corresponding executable event-driven rules is still ad-hoc and made by enterprises on the basis of proprietary technologies, which reduces the interoperability and increases the costs [12, 19].

Many researches argue that event-driven technologies, such as ECA, can completely replace workflow technologies and implement the whole process [31, 11, 12, 18]. Such approach can provide great flexibility and overcome limitations of the workflow approach. The lack of flexibility of workflows has been debated at long in recent years [27, 23]. In this regard, Knolmayer et al. [10] show that ECA constructs can be successfully used to model and execute business processes. Luckham [18] also proposes a way for implementing executable business processes based on CEP.

In this work, we advocate the use of CEP as the technology for implementing executable business processes. CEP provides a rich language for describing event patterns and is recognized by its high performance, being able to process millions of events per second [18]. By using CEP we are able to implement the infrastructure for both the activity coordination and data processing required for executing business processes.

2.3 The World Bank Case

We present our approach using examples retrieved from the World Bank's operations manual. The World Bank is a recognizable international financial institution that provides financial and technical assistance for developing countries.

The Bank adopts an information disclosure policy that promotes the transparency of the Bank operations and results. One of the initiatives for information disclosure is the publication of the Bank's operations manual [3], which guides its staff in the daily operations. The operations manual includes two types of document: *Operational Policies* (OP), which define the policies, agreements, and general conditions that govern the Bank operations; and *Bank Procedures* (BP), which spell out procedures and documentation required to ensure the Bankwide consistency and quality of operations.

For the case study conducted in this work, we collected business rules and procedures from the *development policy lending* operation [3, OP/BP 8.60]. Development policy lending aims to help a borrower achieve sustainable reductions in poverty through a program of policy and institutional actions that promote growth and enhance the well-being and increase the incomes of poor people. The World Bank provides development policy lending in the form of loans or grants to help a borrower address actual or anticipated development financing requirements [3].

We also use the rules from the *Signing of Legal Documents and Effectiveness of Loans and Credits* policies [3, OP/BP 13.00], which affect all lending operations of the Bank.

We analyzed the operations and policies and transcribed the guidance statements as a number of business rules written according to the RuleSpeak standard [29]. The aim of RuleSpeak is to reduce the ambiguities and improve the preciseness and quality of business rules through a number of best practicies. Thus, the business rules presented along this paper are a RuleSpeak version of the guidance statements found in the operations manual [3, OP/BP 8.60 and 13.00].

3 RELATED WORK

Although many works have investigated the integration of business rules and business process implementation, there is still a lack of a methodological background for improving the alignment of business and IT.

Therani [19] proposes an ontology for designing flexible business processes. The work proposes a two-layer framework for the description of business processes that aims at bridging the communication between domain abstractions and software abstractions. The first layer corresponds to domain semantics, which is extracted from the real world, observing the user point of view. The second layer corresponds to the technology-specific abstractions, which is constructed from a developer's point of view. The authors argue that managing the relationship between these two layers in a consistent manner is the key for developing reliable process-management systems. Nevertheless, no systematic mechanism for performing such management is presented. The mapping from tasks, states, and agents to software objects is completely ad-hoc. Also, no methodology for defining the tasks, states, and agents from a business analysis is provided.

Knolmayer et al. [10] propose an approach for modeling workflow using business rules. These rules are represented by Events, Conditions, and Actions (ECA). The work proposes that textual business rules should be structured in the form of Event, Condition, and Action descriptions. Such structure should be incrementally detailed in order to achieve an executable process specification. Unfortunately, the work do not present any methodology for modeling the ECA rules. Also, refinement from the high-level business specification to the corresponding low-level implementation is ad-hoc.

Kovacic et al. [16] discuss *business renovation*, which is the effort for redesigning business processes and information systems on the basis of a critical examination of current business policies and practices. They state that business rules should be described in *natural language* and business processes should be modeled *only* at the level of detail that is sufficient to achieve the rules' objectives. They also propose that textual rules should be incrementally detailed into lower-level abstractions. Unfortunately, they do not provide any framework for methodologically deriving software models from business rules descriptions. They argue that, in small cases, the manual revision is more economic than the use of current tools.

Several other works propose the use of event-driven technology (e.g. ECA [10] and CEP [18]) to implement flexible business processes using business rules [18, 31, 11]. However, none of these works provide means for integrating the *business rules side* and the software abstraction level provided by event-driven frameworks.

In this paper, we aim at overcoming the limitations found on these related works. We define a methodology for systematically integrating the concepts of high-level business rules and the concepts of complex event processing. Through the proposed methodology one can systematically refine a high-level business rules into an implementation based on complex event processing technology.

4 ONTOLOGY OF EVENTS

This section defines an ontology regarding event-driven systems concepts. An ontological framework is essential for a good communication between business people and developers [19].

The fundamental concepts regarding event-driven systems are defined as follows.

Definition 1 (Event)**.** Any change in the enterprise state that has a relevant business meaning for the business operations or for management.

Definition 2 (Activity)**.** Any action that significantly changes the state of the enterprise (causing the generation of events).

Definition 3 (Constraint)**.** A restriction on the systems' states.

Definition 4 (Rule)**.** A statement that govern the occurrence of events and relate them to the execution of activities.

Definition 5 (Agent)**.** The application/person responsible for executing activities when events are recognized.

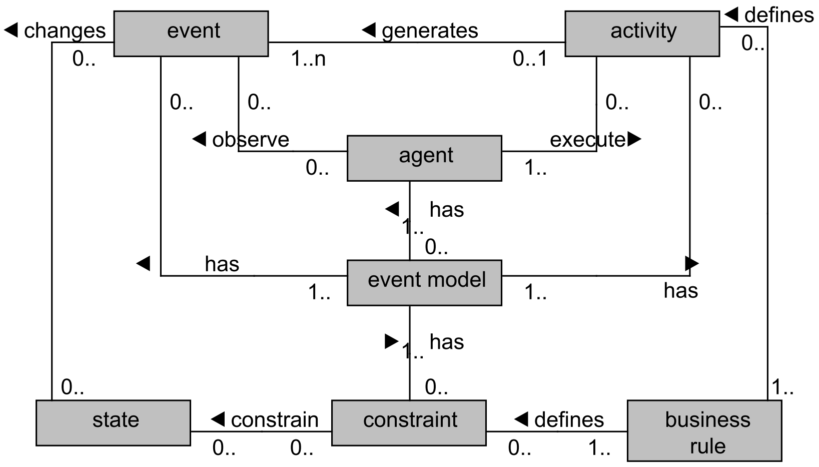

Fig. 1. Conceptual UML diagram for the event ontology proposed in this paper

An organization's events, activities, and rules build up its event model. Figure 1 depicts the relationship between the elements we have described.

An event model must be *complete* and *consistent*. The concepts of *event model*, *completeness*, and *consistency* are defined as follows:

Definition 6 (Event Model)**.** A set of events, activities, agents, and relationships between them (rules and constraints) that interact for building up the behavior of a dynamical system.

Definition 7 (Complete Event Model)**.** An event model is complete if, for a set of events and activities, the following statements are true:

1. there is no activity generating events outside the model, and there is no event triggering the execution of activities outside the model;
2. all structural constraints that affect these events and activities in the real world are present in the model.

Definition 8 (Consistent Event Model)**.** An event model is consistent if it correctly reproduce the system's behavior, regarding only the events from the system that are present in the model.

An event model is a representation of a dynamical system (e.g., an enterprise) in terms of events, activities and so on (Definition 6). The criteria for its *completeness* and *consistency* assure the correctness of the model.

We can consider three levels of abstraction at which an event model is realized:

Business level: events and activities are informally present on business rules, business documents, and on the executives' vocabulary;

Abstract level: events and activities are identified and well-documented, but not mapped into software code;

Software level: the event model is implemented through software artifacts.

Our approach defines of a sequence of phases to refine the event model from the highest level (business level) until the lowest level (software level). These phases are described in Section 5. The *business level* is the field of discussion of the business community. The *software level* is the field of discussion of the IT community. The *abstract level* is the communication interface between these two communities. Both business and IT staff are involved in elaborating the event model at the abstract level.

5 SYSTEMATIC REFINEMENT OF EVENT MODELS

In this section we present our refinement method for implementing event-driven business processes.

5.1 Analysis Phase

Input: Natural language business rules.

Output: Business Rules Analysis (BRA).

Level: Business.

At this phase, a document called Business Rules Analysis (BRA) is constructed. The BRA contains all relevant information extracted from the rules at the business level and includes complementary information assessed from different alternative sources, such as reports, process specifications, interviews, and others.

The construction of the BRA is made by answering the following questions for each individual business rule:

1. which enterprise's *product* or *service* the rule affects or contributes to?
2. which organizational *roles* the rule affects?
3. which *resources* are mentioned or needed to evaluate the rule?
4. which *events* are mentioned by the rule (directly or indirectly)?
5. which *events* are produced by trigerring the rule?
6. which *activities* are required by the rule or are necessary to evaluate the rule?

To avoid biased answers, these questions must not rely on existing implementations of the system. To exemplify, let us answer the BRA questions for the Rule DRAW-LOAN (BP8.60).

[BP8.60] Rule – DRAW-LOAN: A borrower is allowed to draw on the loan only if all the follow conditions are true:

a) according to IMF, the borrower's macroeconomic framework is adequate;
b) the borrower continues adhering to the overall program (according to the Program Document – PD).

The BRA answers should be objectively filled in a *Rule Description Form (RDF)*. In this particular case, the answers are based on the Rule itself and on the World Bank Procedures (BP), available at [3, OP/BP 8.60]. The BRA is composed of a set of RDFs. Table 1 displays an example of RDF, filled with answers for the Rule DRAW-LOAN (BP8.60).

Rule	DRAW-LOAN (BP8.60)
Prod./Service	This rule affects the lending policy.
Roles affected	The task team assigned to the project verifies the conditions required by this rule.
Resources	The task team needs: the *IMF's macroeconomic indicators*; and the *Program Document*.
Events observed	The rule is applied when *the borrower requests to draw on the loan*.
Events produced	After the rule's evaluation, the draw is either allowed or denied.
Activities	An *evaluation of the borrower's condition to draw on the loan*.

Table 1. Example of Rule Description Form (RDF)

5.2 Event Modeling Phase

Input: Business Rules Analysis(BRA).

Output: Event Definitions (ED).

Level: Abstract.

Notice that events are mentioned in the BRA in an informal manner. The purpose of this phase is to identify and register all events found during the previous phase. Each event found receives a unique name and is registered in a document called *Event Definitions (ED)*. The *ED* is a glossary of event names. As such, it may include a brief description of the event, synonymous names and, when necessary, examples of situations when it occurs. This phase is where we begin to move from the business level to the abstract level of the event model.

During this phase, we classify events according to their causal relations regarding the phenomena that created them in the real world. Two situations are recognized:

Event After Action (EAA): the event indicates the occurrence of a past fact;

Event Before Action (EBA): the event indicates that something is about to occur.

This classification is important because the implementation of each type of event is different. EAA events are not controllable, since they indicate events that already occurred in the real world. EBA events, on the other hand, are controllable. Therefore, one can define rules to impose restrictions on its execution.

For example, the World Bank procedures state that:

> **[BP8.60] Rule** – PD-PUBLICATION: A Program Document (PD) may be made available to the public only if it is ready to publish.

This rule says that the Program Document cannot be published if it is not considered ready for publication. The bank procedures describe a number of requirements that must be true for the PD to be considered ready to publish.

The rule PD-PUBLICATION (BP8.60) imposes a constraint to the occurrence of the event of publishing the Program Document, which we can call the *PD Publication* event. Once event-driven frameworks are naturally reactive, in the sense that they can only process an event after its occurrence, it is difficult to define rules for prohibiting the occurrence of an event. Therefore, how can a prohibition rule be effectively triggered?

The solution we provide for handling prohibition rules is as follows: *All* EBA events must be issued in two steps. Firstly, an advice event (*Event Attempt*) is generated. Rules that could prohibit *Event* are triggered on the occurrence of this advice event. If a condition that prohibits the event is found, a denying event (*Event Denied*) is generated. Otherwise, *Event* is issued normally. A single entity in the system is responsible for verifying the conditions that prohibits the execution of a particular event. Only it is allowed to issue the *Event* event. This avoids issuing both the event and a denying of the same event.

The Event Definitions (ED) document would contain the following entries:

> **Event Name:** PD Publication
>
> **Event Type:** Event-Before-Action
>
> **Description:** it makes the Program Document (PD) available to the public
>
> **Associated data:** date/time of publication; PD identification; identification of the lending process associated to this PD.
>
> **Synonymous:** *PD made public* (notice: “PD” may appear as “Program Document” in mentions to this event).
>
> **Software Name:** `org.iadb.events.PDPublication`.

5.3 Pattern Definition Phase

Input: Business Rules Analysis (BRA), Event Definition (ED).

Output: Event Pattern Definition (EPD).

Level: Abstract/Software.

This phase has the purpose of identifying the conditions that should be observed in order to trigger a given rule. These conditions are recognized by the definition of event patterns. Patterns are defined by data, environment, and time constraints that recognize the states when the conditions hold. In this phase, IT people and business people must be involved in order to formally define the patterns through a pattern language. The use of a pattern language is necessary at this point to ensure that the rule has no ambiguity and that both business and IT people understand and agree about its semantics.

Patterns identified are registered in a document called Event Pattern Definition (EPD). Each pattern is associated with a rule in the BRA. The EPD also classifies the corresponding rules according to three generic classes:

Reactive Rule defines that some events must be triggered when a give condition is satisfied;

Prohibition Rule defines that certain events can not happen if a give condition is satisfied;

Communication Rule specifies that someone must be notified when a given condition holds.

The objective of this classification is to assure better comprehension on how the rules should be modeled during the implementation phase.

An example of pattern can be found in the Rule AGREEMENT-SIGNING-DELAY (BP13.00). It was retrieved from the Bank Procedures BP 13.00, which guides on the signing and effectiveness of loans and credits. This rule also affects the development policy lending.

> **[BP13.00] Rule** – AGREEMENT-SIGNING-DELAY: If a loan legal agreement is not signed six months after loan approval, the loan must be considered a signing delay loan.

From this we can model the following pattern, using Esper's *Event Query Language* (EQL) [8].

> **[BP13.00] Pattern** – AGREEMENT-SIGNING-DELAY:
> **Type:** Communication rule
> **Description:** Six months after the *Loan Approved* event, no event of *Loan Agreement Signed* was issued for that same loan number yet.
> **EQL expression:**

```
every (a = LoanApproved -> (timer:interval(6 month)
  and not LoanAgreementSigned(loanNumber = a.loanNumber)));
```

5.4 Activity Modeling Phase

Input: Business Rules Analysis (BRA), Events Definition (ED).

Output: Activities Definition (AD), Events Definition (ED) (updated).

Level: Abstract.

Business rules are the main source for discovering what activities are executed by the enterprise. Rules either require the execution of activities or affect how these activities are executed. There are two aspects concerning what the rule requests to be done:

1. implicit activities – some action must be executed to verify the applicability of the rule;
2. explicit activities – the rule may explicitly require some action to be taken.

For example, a rule that states "*If a package arrives with defect, it can not be accepted.*" does not explicitly require any action. However, to figure out if the package has any defect, an activity *Verify Package* must be executed. On the other hand, a rule that states "*Any package arriving must be verified before being accepted.*" explicitly defines the action to be taken before accepting the package.

The activities are registered in a document named *Activity Definitions (AD)*, which contains unique names for each activity and their descriptions. It also describes which events are related to the activity, which roles are responsible for executing it, and what resources are necessary.

Also, for each activity A, we include events to indicate the activity start (A *Started*), finalization (A *Finished*), cancellation (A *Canceled*), and an event that requests its execution (A *Request*). Notice that the *ED* must be updated with these new events.

Below, we give an example of the document AD related to the Activity "Evaluate Borrower's Conditions", recognized in the Rule DRAW-LOAN (BP8.60).

Activity – *Evaluate Borrower's Conditions*: Verify if the borrower has the conditions necessary to draw on the loan at a given moment.

Role – task team;
Request event – Evaluate Borrower's Conditions Request;
Begin event – Evaluate Borrower's Conditions Start;
Conclusion event – Evaluate Borrower's Conditions Finished;
Cancel event – Evaluate Borrower's Conditions Canceled;
Resources – IMF macroeconomic indicators, Program Document, other documents generated during the program implementation.

5.5 Implementation Phase

This section describes an architecture by which the event model can be implemented. It presents the fundamental components of the event-driven system. Naturally, several supporting technologies must be employed, such as databases, network, and graphical interfaces. We abstract the necessity of these technologies and focus on the specific concerns of the event-driven part of the system.

The objective is to allow the flexibility and modularity in the implementation of event models, improving the ability to add and exclude rules during the system's lifetime without compromising its execution.

The architecture has six components: Data Model; Event Objects; CEP Engine; Agents, Worklist Handler; and Rules Manager.

Figure 2 displays a graphical view of the architecture elements and their relationship.

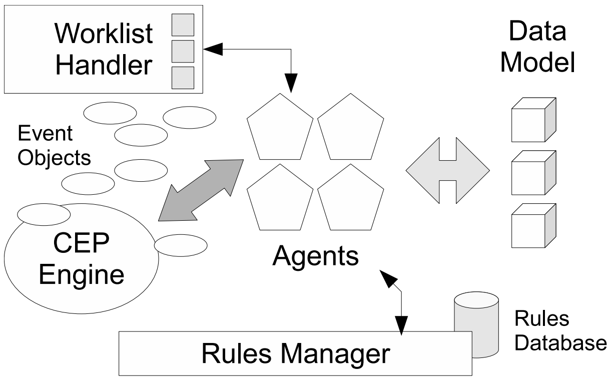

Fig. 2. Graphic view of the architecture

5.5.1 Data Model

This component contains all classes that represent business entities. It corresponds, for example, to the model layer in a Model-View-Controller (MVC) architecture.

In the World Bank event model, the data model contains classes such as *Borrower*, *Loan*, and *BoardSubmissionForm*.

5.5.2 Event Objects

The ED document provides the definition of all events that must be present in the event model. Every event described in the ED is modeled by a class that contains the data about its occurrence. For example, the "*Loan Draw Performed*" event is modeled by a class `LoanDrawPerformed` which contains fields such as date/time of the draw, borrower's identification, amount drawn etc.

5.5.3 CEP Engine

The responsibility of the CEP Engine is to process the events generated by the application. Traditional engine implementations can handle millions of events, recognize complex patterns of events, and trigger actions in the application, when necessary. Each implementation of CEP uses different approaches for achieving this and different languages for expressing the event patterns.

We chose EsperTech's Esper framework [8] for implementing the World Bank's event model. However, any framework that has the necessary features could be used as well. The main requirement is that the language provided by the engine is powerful enough to express the conditions necessary for recognizing the business rules that will be implemented.

5.5.4 Agents

Agents are objects responsible for taking actions when patterns are recognized. Each agent detects a set of patterns that have common characteristics. Different agents run in parallel, processing different sets of events, therefore improving the performance and scalability of CEP [18].

The structure of an agent is composed of the set of patterns that it recognizes and the code that it will execute upon pattern matching. These patterns are retrieved from the Event Patterns Definition (EPD) document. Possible actions taken by the agents can be the execution of automated business operations, database updates, issuing of new events or interaction with other applications. This behavior must be codified by the IT staff according to the definition of the rules.

5.5.5 Worklist Handler

Worklist is a list of tasks that are assigned to an employee. The goal of the worklist handler is to manage the interaction between the system and the employees, by submitting tasks to the employee's worklist and getting the notification upon task completion. An agent is the bridge between the CEP engine and the worklist handler. It converts the *request events* action into tasks in the worklist and notifications of completion into *conclusion events*.

5.5.6 Rules Manager

As the system evolves, it is necessary to manage its execution, change, add, or remove rules. The rules manager provides the funcionality for managing the business rules stored in the system. It enables the communication between the user, the rules database and the CEP engine.

6 IMPLEMENTING THE WORLD BANK EVENT MODEL

In this section we present the implementation of the World Bank's event model.

As we explain in Section 5.5, we chose EsperTech's Esper framework [8] to implement the CEP Engine. This is a well documented framework that runs under Java and has support for integration with most technologies available for enterprise information systems, such as Web Services, XML, XPath, and Java Messaging Service (JMS).

In order to explain how the World Bank event model is implemented, we detail the implementation of three rules. For each rule, we present the events, the agents, how the pattern is described and what the agent will process. These aspects are the essential characteristics to describe the rule implementation. The implementation of all rules follows the same principles.

Consider Rules 1, 2, and 3, described below.

Rule 1 – PD-PUBLICATION: A Program Document (PD) may be made available to the public only if it is ready to publish.

Rule 2 – AGREEMENT-SIGNING-DELAY: If a loan legal agreement is not signed six months after loan approval, the loan must be considered a *signing date delay loan.*

Rule 3 – EFFECTIVENESS-CONDITIONS: A signed loan legal agreement becomes effective if, and only if, the borrower of the loan has provided satisfactory evidences of compliance with the conditions of effectiveness.

6.1 Phase 1: Analysis

The first step for implementing theses rules is filling the Rule Description Form (RDF). Tables 2, 3 and 4 display these forms.

Rule	PD-PUBLICATION (BP8.60)
Prod./Service	This rule affects the development policy lending.
Roles affected	The publication of the Program Document (PD) is handled by SECBO, IDU, and the InfoShop.
Resources	The staff needs access to the Program Document.
Events observed	The rule is applied when a member of the task team or the Bank staff wants to publish the Program Document.
Events produced	After evaluation of the rule, the publication is either allowed or denied.
Activities	An evaluation of the PD's readiness to publish must be conducted before applying this rule.

Table 2. RDF for Rule 1

6.2 Phase 2: Activity Modeling

After the rules are analyzed, we need to register the activities recognized. We can recognize the following activities for the Rules studied:

Rule	AGREEMENT-SIGNING-DELAY (BP13.00)
Prod./Service	This rule affects all lending processes.
Roles affected	The task team leader is responsible for monitoring delays in signing and taking appropriate actions.
Resources	The task team leader needs access to the legal agreements.
Events observed	The rule is applied when it has passed six months after the *loan approved* event and no event of *legal agreement signed* was issued for that same loan yet.
Events produced	After application of this rule, an event indicating that the loan has been marked as a signing date delay loan must be issued.
Activities	An evaluation of the delay time must be performed. Notice that the bank procedures also have rules that define what activities must be performed when a loan is marked as a signing date delay loan.

Table 3. RDF for Rule 2

Evaluate Readiness to Publish PD: evaluate the conditions for publication of the Program Document;

Check Agreements Signing Delay: check the time elapsed since loan approval to compute the signing delay;

Analyze Evidence of Compliance: evaluate the evidences of compliance to effectiveness conditions that were provided by the borrower.

These activities are registered at the Activities Definition (AD) document, informing about the events related to them and the roles responsible for their execution.

6.3 Phase 3: Event Definition

The next step is the events description. Observing the RDF of each rule, we can define several events:

PDPublicationAttempt: one attempts to publish a Program Document;

PDPublication: the Program Document is published;

PDPublicationDenied: the publication of the Program Document is denied;

Loan Approved: the Bank staff approves the loan;

LegalAgreementsSigned: the authorized representatives of the borrower and the Bank sign the legal agreements of the loan;

Rule	EFFECTIVENESS-CONDITIONS (BP13.00)
Prod./Service	This rule affects all lending processes.
Roles affected	The task team leader, the country director, and the lawyer are responsible for verifying the legal aspects of compliance with the conditions of effectiveness.
Resources	The participants involved need access to the documents provided by the borrower for evidence of its compliance to the conditions of effectiveness.
Events observed	This rule is applied after the loan legal agreements are signed.
Events produced	After application of this rule, the team leader informs that the evidences of compliance provided by the borrower are either accepted or refused. If evidences are accepted, the leader informs that the loan is effective.
Activities	An analysis of the evidences of compliance provided by the borrower must be performed.

Table 4. RDF for Rule 3

LoanMarkedAsSigningDelay: the loan has been marked as a signing delay loan because it has not been signed a certain time (six months in our example) after it has been approved.

AnalyzeEvidenceOfComplianceRequest: an analysis of the evidences of the borrower's conditions of effectiveness is required;

AnalyzeEvidenceOfComplianceConcluded: the analysis of the conditions is concluded;

EvidenceOfComplianceAccepted: the evidences that the borrower furnishes to the Bank are accepted;

EvidenceOfComplianceRefused: the evidences that the borrower furnishes to the Bank are not accepted;

LegalAgreementsEffective: the legal agreements become effective;

There is also an event for the *Cancelling* of the activity *Analyze Evidence of Compliance*, as also as events for "request", "conclusion", and "cancellation" of other activities recognized during the activity modeling, which we have ommitted from the list above.

6.4 Phase 4: Patterns Definition

Once the events are recongnized, it is necessary to define the patterns that identify each rule. According to the classification we have proposed, Rule 1 is a prohibition rule, Rule 2 is a communication rule, while Rule 3 is a reactive rule. In the case of Rule 1, whenever a participant attempts to publish the PD, the conditions must be verified. Therefore, the pattern monitored is every occurrence of the *PDPublicationAttempt* event.

In Esper's Event Querly Language (EQL), this pattern is expressed by:
"`select * from PDPublicationAttempt;`"

Observe that we need to insert a `select` command before the pattern, because EQL has a syntax based on the traditional SQL database query language. The `select` command in EQL provides most of the powerful aggregation techniques present in SQL.

For Rule 2, we need to recognize a time dependent condition. CEP provides constructors for expressing time intervals within a pattern definition. We want to recognize the situation in which six months have passed after the moment the loan was approved and no legal agreement signing was observed in the mean time. In EQL, it is expressed by:

```
select * from pattern
[every (a = LoanApproved -> (timer:interval(6 month) and not
LegalAgreementsSigned(loan.loanNumber = a.loan.loanNumber)))];
```

Notice that, to assure that both events correspond to the same loan, we use EQL's field access capabilities to check whether the loan numbers in each event are the same.

The last rule is applied for all occurrences of *LegalAgreementsSigned*:
"`select * from LegalAgreementsSigned;`"
when a request for analysis of compliance is issued.

This rule is reactive. It requests that the loan legal agreements must become effective when they are signed and the effectiveness conditions are met. This produces a second pattern to be recognized by this rule:

```
select * from pattern
[every (a = LegalAgreementsSigned ->
EvidenceOfComplianceAccepted(loan.loanNumber = a.loan.loanNumber)))];
```

When this pattern is recognized, the agent is programmed to issue the event *LegalAgreementsEffective*.

6.5 Phase 5: Implementation

Firstly, the event objects must be modeled through the definition of *classes*. It is necessary to define what data each event will store. On the case of the World Bank, we observed that for most events, only the loan identification is enough for processing them. Therefore, most events have a similar structure as exemplified by the *LegalAgreementsEffective* event, below:

```
public class LegalAgreementsEffective {
 Loan loan;
}
```

The `Loan` is a class of the *Data Model* that stores all information about the loan being processed, as shown below.

```
public class Loan {
  private int loanNumber;
  private Borrower borrower;
  private boolean effective;
  ...
}
```

Next, several agents are implemented. A good criteria for defining agents is to group rules that correspond to the same organizational role and create an agent for each role. So, the World Bank model has the following agent objects: *TaskTeam, TaskTeamLeader, SECBO, CountryDirector*, amongst others.

All agents implement a common interface `IAgent`, defined as:

```
public interface IAgent {

  public void uploadToEngine
               (CEPEngine engine);
  public void stop();

}
```

The `uploadToEngine` method is responsible for registering, in the CEP engine, what patterns the agent will listen to. Every time the Esper engine recognizes one of these patterns, the agent object will be called to execute a method corresponding to the processing of that pattern. Each implementation of `IAgent` will define its own patterns and processing methods.

Considering the Rule 1 above, the `SECBOAgent` is responsible for, at every time the *PDPublicationAttempt* occurs, verifying if the PD is ready to publish. `SECBOAgent` is also responsible for monitoring other patterns of events related to the Program Document, which are not covered by the rules exemplified in this paper.

Rules 2 and 3 are handled by the `TeamLeaderAgent`, which listens for all patterns related to the legal agreements signing and perform or request the execution of the activities necessary.

We implemented simulation features in order to test the correctness of the World Bank event model implementation. The simulator generates loan requests with exponential or normally distributed inter-arrival times and with loan data retrieved from a collection of hypothetical loan requests stored in a database.

In all simulations, the system presented the output expected regarding the rules that were implemented, which includes several rules from World Bank Procedures 8.60 and 13.00.

7 CONCLUSIONS

Despite the advances on business rules theory and the increasing number enterprises doing efforts to model their business rules, there is still a lack for a meaningful integration between business analysis and process implementation activities. Part of this problem is due to the abscence of a common framework to integrate business people and developers.

Aiming at overcoming these difficulties, we propose a new approach for implementing *event-driven* business processes by employing a stepwise refinement from the business level to the software level. We show how a set of business rules described in natural language can be used as the specification for the development of an event-driven management system in a way that business people can actively participate.

The need for a better alignment between business and information systems has been extensively discussed in the literature [20, 2, 1, 26, 9]. Many works have tackled this problem by creating tools that help developers to separate business requirements from technical issues. However, as far as we know, no previous work has provided means by which business people can be integrated in this process. We propose a methodology that begins in the field of discussion of business people, at the level of business rules in natural language. Then, we propose a number of phases into which these business rules are incrementally modeled as an event model, which expresses the activities and events involved. Finally, our methodology shows how this event model can be implemented using Complex Event Processing. This process produces a number of documents that are useful for tracing initial business rules. At the same time, such documents are specifications of the software requirenents. Therefore, the documents are an effective channel of cummunication between business and IT.

The contributions of our methodology are fourfold:

- allowing for the traceability between business needs and software artifacts;
- keeping business people and stakeholders involved along the implementation process, which reduces the semantic gap between business and software levels of abstraction;
- providing a number of documents that help in the system specification, maintaining a vision towards *events* since the analysis phase;
- allowing for the coordination of activities based on rules. This improves the alignment between business processes and business rules and also provides a great deal of flexibility for the process execution.

Furthermore, we present an ontology for event-driven systems that constitutes a valuable library of concepts, useful to improve the communication between business people and developers.

The possibility of changing the behavior of the system simply by changing business rules is essential for achieving greater flexibility and business alignment [2]. By proposing the *event models*, we allow for the implementation of more declarative, *rules-directed* business processes (as oposed to *workflows*) using technologies that already showed their value and effectiveness for implementing real world applications [18, 7].

We validated our approach by implementing the scenario of the World Bank. The experiments showed that the event model is an effective approach for implementing business processes, providing the desired flexibility while maintaining robustness and improving alignment and manageability.

As future works, we intend to implement tools for supporting the development of event models by following a model-driven approach and for the simulation and performance evaluation of the system. A stochastic CEP simulator is already under development.

Acknowledgements

This work was partly supported by the Brazilian Research Council (CNPq), grants PQ 314539/2009-3 and GD 140512/2009-8, and FACEPE (APQ-1202-1.03/10).

REFERENCES

[1] Avison, D.—Jones, J.—Powell, P.—Wilson, D.: Using and Validating the Strategic Alignment Model. Journal of Strategic Information Systems, Vol. 13, 2004, No. 3, pp. 223–246.

[2] Bajec, M.—Krisper, M.: A Methodology and Tool Support for Managing Business Rules in Organisations. Inf. Syst., Vol. 30, 2005, No. 6, pp. 423–443.

[3] World Bank – The World Bank Operations Manual. Technical report, World Bank 2009.

[4] Workflow Management Coalition. WfMC standards: The workflow reference model, version 1.1., 1995.

[5] IBM Corp. Ibm ilog. `http://www-01.ibm.com/software/websphere/ilog`, November 2011.

[6] Cummins, F. A.: Building the Agile Enterprise with SOA, BPM, and MBM. Morgan Kauffman Publishers, Burlington, MA, USA 2009.

[7] Debevoise, T.: Business Process Management with a Business Rules Approach: Implementing The Service Oriented Architecture. BookSurge Publishing 2007.

[8] EsperTech. Esper – complex event processing. `http://esper.codehaus.org/`, 03 2010.

[9] Fink, L.—Neumann, S.: Exploring the Perceived Business Value of the Flexibility Enabled by Information Technology Infrastructure. Information & Management, Vol. 46, 2009, No. 2, pp. 90–99.

[10] Endl, R.—Knolmayer, D.—Pfaher, M.: Modeling Processes and Workflows by Business Rules. Business Process Management: Model, Techniques and Empirical Studies, pp. 16–29, 2000.

[11] Gong, Y.—Janssen, M.—Overbeek, S.—Zuurmond. A.: Enabling Flexible Processes by ECA Orchestration Architecture. In ICEGOV '09: Proceedings of the 3rd International Conference on Theory and Practice of Electronic Governance, New York, NY, USA 2009, pp. 19–26, ACM.

[12] Graml, T.—Bracht, R.—Spies, M.: Patterns of Business Rules to Enable Agile Business Processes. 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), pp. 365–365.

[13] Harmon, P.—Wolf, C.: The State of Business Process Management: February 2008. Technical report, Business Process Trends, 2008.

[14] TIBCO Software Inc. Tibco business events. `http://www.tibco.com/products/business-optimization/complex-eventprocessing/businessevents/businessevents.jsp`, November 2011.

[15] Kang, D.—Lee, J.—Kim, K.: Alignment of Business Enterprise Architectures Using Fact-Based Ontologies. Expert Syst. Appl., Vol. 37, 2010, No. 4, pp. 3274–3283.

[16] Kovacic, A.—Groznik, A.: Business Renovation: From Business Process Modelling to Information Systems Modelling. In Proceedings of the 24th International Conference on Information Technology Interfaces 2002, pp. 405–409.

[17] Lawler, J. P.—Howell-Barber, H.: Service-Oriented Architecture: SOA Strategy, Methodology, and Technology. Auerbach Publications, Boca Raton, FL, USA 2008.

[18] Luckham, D.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Professional 2002.

[19] Madhusudan, T.: Ontology Development for Designing and Managing Dynamic Business Process Networks. IEEE Trans. Industrial Informatics, Vol. 3, 2007, No. 2, pp. 173–185.

[20] Nash, E. M.: IT and Business Alignment: The Effect on Productivity and Profitability. IT Professional, Vol. 11, 2009, No. 6, pp. 31–36.

[21] Nelson, M. L.—Peterson, J.—Rariden, R. L.—Sen, R.: Transitioning to a Business Rule Management Service Model: Case Studies from the Property and Casualty Insurance Industry. Inf. Management Vol. 47, pp. 30–41, January 2010.

[22] Newcomer, E.—Lomow, G.: Understanding SOA with Web Services. Addison Wesley Professional, Hagerstown, Maryland 2004.

[23] Nurcan, S.: A Survey on the Flexibility Requirements Related to Business Processes and Modeling Artifacts. In HICSS '08: Proceedings of the 41st Annual Hawaii International Conference on System Sciences, Washington, DC, USA 2008, p. 378.

[24] Object Management Group. Business Motivation Model (BMM), v1.0. Technical report, Object Management Group 2008.

[25] Object Management Group. Semantics of business vocabulary and business rules (SBVR), v1.0. Technical report, Object Management Group 2008.

[26] Peppard, J.—Ward, J.: Beyond Strategic Information Systems: Towards an Is Capability. The Journal of Strategic Information Systems, Vol. 13, 2004, No. 2, pp. 167–194.

[27] Pesic, M.: Constraint-Based Workflow Management Systems: Shifting Control to Users. Ph. D. thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands 2008.

[28] Radhakrishnan, A.—Zu, X.—Grover, V.: A Process Oriented Perspective on Differential Business Value Creation by Information Technology: An Empirical Investigation. Omega, Vol. 36, December 2008, No. 6, pp. 1105–1125.

[29] Ross, R. G.: Principles of the Business Rule Approach. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA 2003.

[30] Shao, J.—Pound, C. J.: Extracting Business Rules from Information Systems. BT Technology Journal, Vol. 17, 1999, No. 4, pp. 179–186.

[31] van Eijndhoven, T.—Iacob, M. E.—Ponisio, M. L.: Achieving Business Process Flexibility with Business Rules. In Proceedings of the 12th International IEEE Enterprise Distributed Object Computing Conference (EDOC '08), Los Alamitos, pp. 95–104.

[32] Weigand, H.—van den Heuvel, W.-J.—Hiel, M.: Business Policy Compliance in Service-Oriented Systems. Inf. Syst., Vol. 36, 2011, No. 4, pp. 791–807.

[33] zur Muehlen, M.: Workflow-Based Process Controlling: Foundation, Design, and Application of Workflow-driven Process Information Systems. Logos Verlag, Berlin 2002.

Cesar Augusto L. Oliveira is a Ph. D. student in the Center for Informatics at Federal University of Pernambuco, Brazil. He received his M. Sc. degree in computer engineering in 2008 from the Computing Systems Department at University of Pernambuco, Brazil. He has participated in several research and development projects in manufacturing and energy industries; he also gave consultancy to government institutions. His main research area is in strategic application of information technology and in alignment between business and IT. He is also active in other research lines related to business process management, formal methods, modelling and simulation, and service-oriented computing.

Natália Cabral Silva is a M. Sc. student in the Center for Informatics at Federal University of Pernambuco, Brazil. Her main research area is in modeling and enactment of flexible and adaptable business processes.

Cecília Leite Sabat is an undergraduate student of production engineering in the Center of Technology and Geosciences at Federal University of Pernambuco, Brazil. She has participated in research and Development projects in the manufacturing industry, having experience with business process optimization and business rules.

Ricardo Massa F. Lima is an Associate Professor at Federal University of Pernambuco. He is the vice-coordinator of UFPE's computer science postgraduate program. He received his Ph. D. degree in computer science from Federal University of Pernambuco, Brazil, in 2000. He was a post-doc in the Formal Methods Group at Chalmers University of Technology, Sweden, in 2001. His main research interests include compiler construction and optimization; performance evaluation of discrete-event dynamic systems using Petri nets, with projects sponsored by the Brazilian Petroleum Industry (Petrobras), São Francisco's Hydroelectric Company (Chesf), and National Council for Scientific and Technological Development (CNPq). He received the "IBM Faculty Award 2005" for his work with system analysis with Petri nets (EZPetri Project). He is a member of the ACM and the Brazilian Computer Society.