Process Instance Query Language to Include Process Performance Indicators in DMN

José Miguel Pérez-Álvarez and María Teresa Gómez-López and Luisa Parody and Rafael M. Gasca
Departmento de Lenguajes y Sistemas Informáticos
Universidad de Sevilla
Seville, Spain
{josemi, maytegomez, lparody, gasca}@us.es

***Abstract*—Companies are increasingly incorporating commercial Business Process Management Systems (BPMSs) as mechanisms to automate their daily procedures. These BPMSs manage the information related to the instances that flow through the model (business data), and recover the information concerning the process performance (Process Performance Indicators). Process Performance Indicators (PPIs) tend to be used for the detection of possible deviations of expected behaviour, and help in the post-mortem analysis and redesign by improving the goals of the processes. However, not only are PPIs important in terms of their ability to measure and detect a derivation, but they should also be included at decision points to make the business processes more adaptable to the process reality at runtime. In this paper, we propose a complete solution that allows the incorporation of the PPIs into decision tasks, following the Decision Model and Notation (DMN) standard, with the aim of enriching the decisions that can be taken during the process execution. Our proposal firstly includes an extension of the decision rule grammar of the DMN standard, by incorporating the definition and the use of a Process Instance Query Language (PIQL) that offers information about the instances related to the PPIs involved. In order to achieve this objective, a framework has also been developed to support the enrichment of process instance query expressions (PIQEs). This framework combines a set of mature technologies to evaluate the decisions about PPIs at runtime. As an illustration a real sample has been used whose decisions are improved thanks to the incorporation of the PPIs at runtime.**

***Index Terms*—Business processes; Performance Indicators; Decision Model and Notation; Process Instance Query**

I. Introduction

Organizations can describe their operations by using business processes. A business process consists of a set of activities that are performed in coordination within an organizational and technical environment to achieve an objective. In order to help with the business process automation, companies usually incorporate a commercial Business Process Management System (BPMS) into their daily processes. BPMSs represent software that supports the implementation, coordination, and monitoring of the business process execution.

The use of a BPMS facilitates the monitoring of Key Process Indicators (KPIs), and, more specifically, of the Process Performance Indicators (PPIs). KPIs represent business goals that a company wants to achieve on a strategic level. These are qualitative or quantitative assessments that are the result of a comparison between an actual value and a target value over specific time periods [1]. PPIs, however, represent process goals that a company wants to achieve on a more operational level. They can be measured directly by observing the process [2].

BPMSs generate and store information about their performance, such as the number of instances of a business process execution, the duration time of each activity, who has executed each activity, the resources involved, the frequency of each activity, and the number of unfinished instances. Currently, these measurements can be included in a dashboard using Business Activity Monitoring (BAM) or Process Performance Measurement (PPM) tools, where Business Intelligence techniques can be applied [3]. This kind of information tends to be used to build PPIs [4], and are used for a post-mortem analysis and process model redesign and improvement. The monitoring taxonomy proposed by Gonzales [5] distinguishes between two types of monitoring at runtime: active and passive monitoring. Active monitoring provides information of the current state of business process instances, while passive monitoring provides status information about running business process instances upon request. The extraction of PPIs is sufficient to perform passive monitoring, since the information can be recovered when it is necessary.

BPMSs manage information that flows in the business processes. This information can be introduced by users, it may come from an external application, be derived from an internal activity, or be obtained as the result of making a decision in the process by means of decision tasks included in the work-flow model. These decisions enable the modeller to describe the set of possible alternatives by means of various executable branches, in accordance with the data-flow values at the decision points at runtime. In order to facilitate the incorporation of the decision into a business process model, such as that described using BPMN 2.0 [6] standard, Object management group (OMG) has defined *Decision Model And Notation (DMN)*[7] switch is designed to be usable alongside the standard BPMN. The primary goal of DMN is to provide a common notation that is readily understandable by all business users: from the business analysts, needing to create initial decision requirements and then decision models in greater details to the technical developers, responsible for automating the decisions in processes; and finally, to the business people, who will manage and monitor those decisions. DMN creates a standardized bridge for the gap between the business decision design and implementation, and permits the inclusion of decision tasks into the process model. The way in which the data that flows in the process can influence decision-making has been the focus of study of several papers and technologies, however our contribution lies in the fact that we also consider the importance of including the performance data at runtime in the model. Unfortunately, to the best of our knowledge, there are no solutions that permit the process performance indicators to be incorporated into the decision points of the models.

PPIs are highly related to the process instances that are being executed at any moment, and therefore the description of PPIs implies the description of the instance data. The incorporation of PPIs into the business process execution can be crucial: for example, when the assignment of a task to one particular person or another depends on the number of activities executed by each of them in the past, or when the time associated to a clock event depends on the number of instances that are being executed. For this reason, we have defined a *Process Instance Query Language* to extract the necessary information to build the PPIs and ascertain their values at run-time.

To illustrate our proposal, a real example about a platform for football bets, called Tutiplay^{TM} [8], is used. The example presents the necessity to incorporate information about the execution of other processes in order to improve the profits during the prognostic time, for example, by enlarging the open platform time to establish a bet for the most promising instances.

For these reasons, we consider that the execution of a process can be enriched by incorporating information concerning the business execution, and therefore we wish to include data obtained from the business process performance into the decision rules executed at the decision points of the process. The incorporation of PPIs into the decision at run-time permits continuous improvements to be added, thereby building a more flexible and adaptable model. In particular, this incorporation provides a way to combine the process data and behaviour of various processes and instances at the same time. Unfortunately, the decision rules supported by commercial BPMSs fail to support the incorporation of this information both in the model and during the execution.

We propose an extension to the Decision Model and Notation (DMN)[7] to model PPIs and their introduction into the decision tables, thereby enriching the types of decisions and the managed data at decision points, and shielding the user from unnecessary details on how these PPIs are obtained. In order to extract the PPIs, we have defined a Process Instance Query Language (PIQL) that allows business experts to describe the PPIs.

The proposed business rule engine and a Domain Specific Language (DSL) are completed with an implementation of an entire framework that combines a set of mature technologies.

The paper is organized as follows: Section II introduces an illustrative example Section III describes the necessary grammar and a DSL associated to decision points. Section IV explains the architecture of the solution. Section V shows an implementation of the described architecture and technologies. Section VI analyses an overview of related work found in the literature. And finally, conclusions are drawn and future work is proposed.

II. AN ILLUSTRATIVE EXAMPLE

In order to illustrate our proposal, a real-world example is used. It consists of a collaborative platform to play a football pool, called Tutiplay^{TM} [8]. Using the platform, the participants try to predict the outcomes of 14 football matches, where the alternatives are "1" to forecast the local team as the winner, "x" to draw, and "2" to forecast the visiting team as the winner.

Figure 1 shows two business process models implemented to support the platform. The first model (*a. New bet creation*) shows how a bet is managed by the person who administers the platform, from the creation to the final formalization of the bets. The second model (*b. Forecast an outcome*) shows the steps that a player must follow to forecast a specific outcome.

As shown in Figure 1, the process to manage a bet (*a. New bet creation*), is divided into three parts.

In the first stage, the person who administrates the platform creates a new bet, and configures the parameters, such as *open time* (date from which users can forecast), *close time*, (date from which no more forecasts are allowed), and *extended time* witch is used to grant extra time, if necessary.

The second stage consists of monitoring the players' forecasts, and therefore starts when the bet is opened, and predictions are made by users. For each bet, the aim of the platform is to formalize as many forecasts as possible in order to maximize the profits. For this reason, the number of formalized forecasts can be considered as a PPI to be maximized. In order to improve the aforementioned PPI, three different actions can be executed, performed by means of the three condition flows shown in the process, plus one more in case of do nothing:

- **email:** consists of sending a reminder email to the people who have yet to make a forecast.
- **tweet:** consists of sending a tweet with the aim of alerting followers that they have yet to make a forecast. This tweet is not a personal reminder, like the email: it is tweeted with any content to produce an alert to connect players and followers.
- **time extension:** consists of extending the *open time*, thereby providing users with more time to make a forecast.

There are certain business policies that need to be taken into account to determine the actions that can be executed. These policies are established by the business experts with the aim of improving the PPIs, such as "number of forecasts", but not of worsening others. For example, if too many emails or tweets are sent, the risk of being considered a spammer At arises, with the consequence of losing players and followers. In that sense, in the case of the "email" branch, the business policies of Tutiplay establish that it is not possible to send more than one reminder email for each bet, and it must be sent within 24

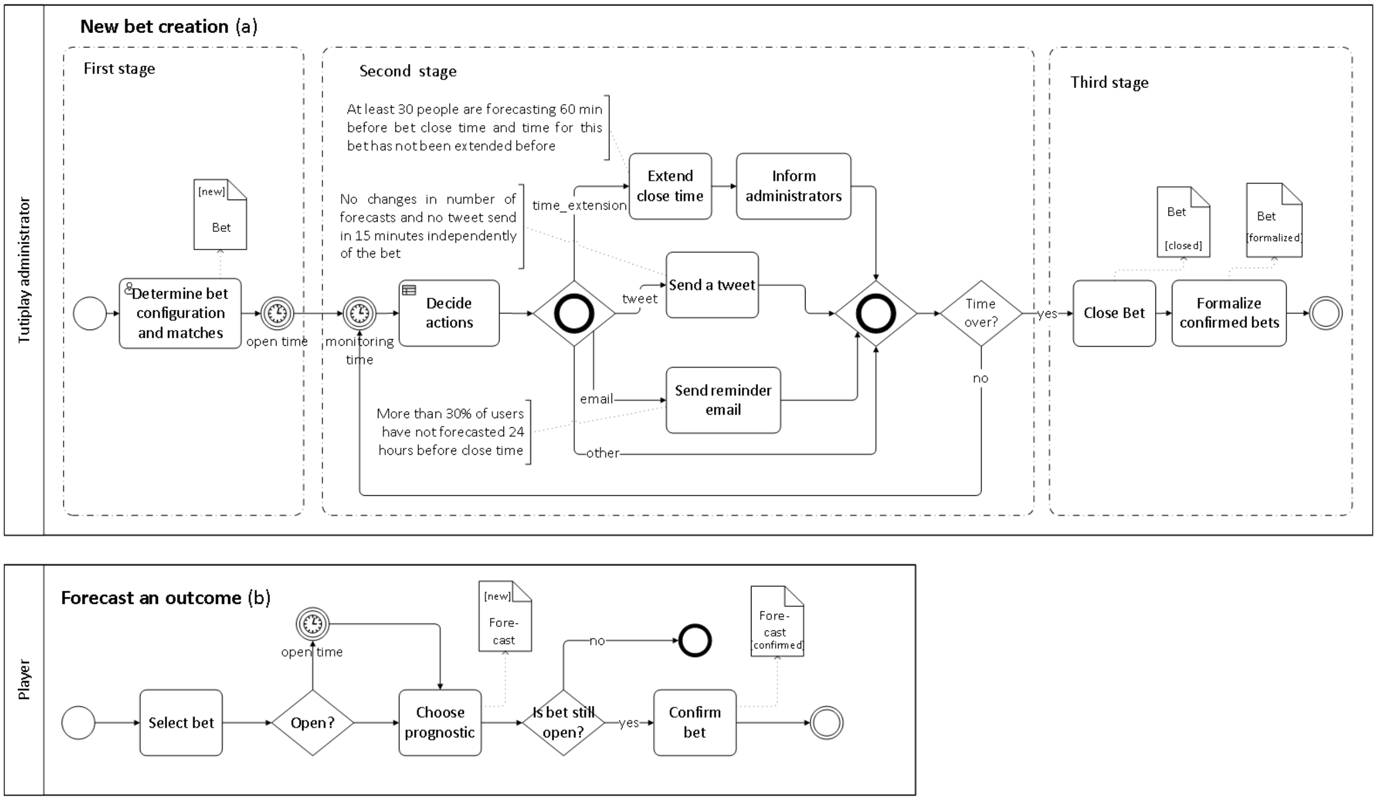

Fig. 1. Tutiplay business process

hours of closing time, if and only if at least 40% of the players have yet to make a forecast. The policy about "tweet" action does not allow more than one tweet to be sent per 15 minutes. Moreover the tweets are sent if the number of forecasts has not been incremented within the last hour. The "extended" alternative can be executed only once for each bet, and it will take place if, during the last 30 minutes before close time, at least 30 players are still forecasting.

As explained before, the presented PPI enables the quality of the process to be measured in accordance with the number of finished instances. The *Tutiplay* example needs to tailor each bet at runtime in accordance with to the value of this PPI, thereby rendering it unnecessary to redesign the model in accordance with to a PPI analysis. Therefore, the idea of the proposal needs to cover the instance adaptation in each case to improve the PPI. The adaptation of each case at runtime makes the model more flexible and agile. It does not contradict the improvement of the process redesign following the life-cycle proposed in [9], since this approach permits the deductions derived from business intelligence to be incorporated out design time, and each instance to be adapted to optimize the PPIs at runtime.

The third stage starts once the forecast time ends. At that moment the, forecast is closed, and the final tickets are printed and formalized in the lottery administration.

On the other hand, process *b. Forecast an outcome* of Figure II describes how users access the platform to play. First of all, a bet is selected, in case there is any open bet. The task "choose prognostic" can then be performed for the participant to decide his/her prognostic. Once this task is finalized, a new *Forecast* object is created to store the forecast. In the case when the bet is still open, the user can confirm the forecast and the *Forecast* object is set to "forecast" state. Notice that not every participant completes the whole process at once; in real life the players can access to process several times to play a bet (as many bets as they want).

The architectures proposed for the current BPMSs present an isolation between the information of the execution logs (through administration and monitoring tools) and process modelling tools [9]. For this reason, it is not a minor task to incorporate the information, that is normally obtained in the monitoring phase, into the process model (as is needed in the second stage of Process *New bet creation*), which needs to include the PPI values in the decisions at runtime.

Various solutions are available to solve this problem, but they are very complex and ad-hoc methods that are time consuming, and cannot be used in a general way, due to the necessary for specific implementations created by designer experts to transform the business decision policies into executable policies ones.

The following sections explain the proposed grammar designed to enlarge the description of the decision rules, and how these rules can be evaluated at instantiation time for each case. This is carried out using the so-called Process Instance

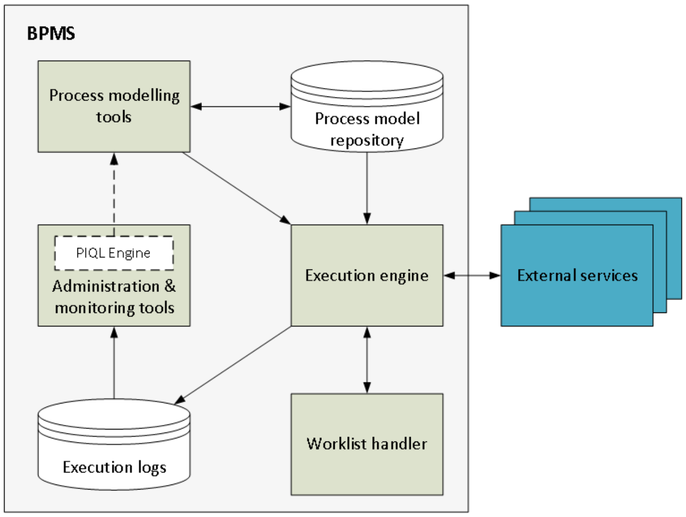

Fig. 2. Proposed architecture of a BPMS

Query Expressions, which support the incorporation of PPIs into the decisions.

III. Process Instance Query Language (PIQL)

The BPM life-cycle [9] defines how to improve processing based on the knowledge of historical executions. This information is extracted from the BPMSs by using administration and monitoring tools 2, and constitutes major support to the redesign phase. Since these improvements are carried out in a manual way, basic BPMS architecture includes of administration and monitoring tools. However, these tools remain disconnected from the process modelling tools, since it is the business expert who introduces the necessary modification into the model.

To solve the isolation between the model and the PPIs generated for the process engine at run time, this paper presents a proposal for the values of the PPIs at runtime to be used also in the DMN rules. This implies creating a union between the process modelling tool and the monitoring tool, as presented in Figure 2. This new module connects the data of the execution logs to the modelling phase by allowing the modeller to make queries for the evaluation of the status of the BPMS execution. Our proposal is an improvement of the typical architecture of BPMSs [9].

The adaptation of the architecture facilitates the business experts the use of PPIs aligned in the process execution. Also, we propose the use of Process Instance Query Expressions (PIQEs) using a Process Instance Query Language (PIQL), to extract information about the process instances and PPIs with a defined grammar and a friendly Domain Specific Language. A PPI can be described by using arithmetical combinations with the information extracted from the process instances. These expressions are a combination of one or more constants, variables, operators, and functions, with the peculiarity that a PIQE also includes the capacity to incorporate information on the instances of processes and information concerning the activities of various instances in more detail. Inspired in the Process Query Language (APQL) [10], we define a new grammar to describe and evaluate the sentences written in PIQL, to obtain the PPIs at runtime. APQL is a process oriented query language that allows to select process models, while PIQL is a process instance oriented query languaje that allows to select process instances.

In order to formalize the expressions incorporated in the PIQL, we need to introduce the elements that can be included in the queries: Process instances (PI) and Task Instances (TI).

Process instances PI are described by the tuple $\langle$*CaseId*, *Process_Name*, *Start*, *End*, *Cancelled*, *Who*, *List Of Global Data*$\rangle$, where the attributes mean:

- *CaseId*: Identification that describes the instance in an univocal way. It is assigned by the BPMS when an instance is created.
- *Process_Name*: Name of the process model.
- *Start*: Date when the instance has started.
- *End*: Date when the instance has finished, or *null* otherwise.
- *Cancelled*: Date when the instance has been cancelled, or *null* otherwise.
- *Who*: Person or rol that has started the execution of the instance.
- *List Of Global Data*: Represents the global variables specific for each process model, that can be interesting to be known out of it during each instantiation.

The task instances TI represent the various tasks executed for each instance, and are described by the tuple $\langle$*CaseId*, *Task_Name*, *Process_Name*, *Start*, *End*, *Cancelled*, *Who*$\rangle$, where the attributes mean:

- *CaseId*: Identification that describes the instance in an univocal way. It is assigned by the BPMS when an instance is created.
- *Task_Name*: Name of the task.
- *Process_Name*: Name of the process model associated to the activity.
- *Start*: Date when the task has started, or *null* otherwise.
- *End*: Date when the task has finished, or *null* otherwise.
- *Cancelled*: Date when the execution of the task has been cancelled, or *null* otherwise.
- *Who*: Person or rol that has started the execution of the task in this instance.

By using Set Theory over these two types of elements, and by applying filters over their attributes, it is possible to ascertain:

- **The number of instances of Processes** finished, unfinished, started by a specific user, started after or before a specific time, finished after or before a specific moment in time, that contains a specific variable with a determined value, etc.
- **The number of Activities executed in a Process Instance** started in a specific process instance, assigned to a specific user, finalized, not finalized, cancelled, started

after or before a specific moment in time, finished after or before a specific moment in time, etc.

Based on the above description of PI and TI, certain predicates can be defined to count the number of process or task instances in a determined status. This predicates allow to select "processes or tasks finalized", "process or task not finalized", "processes or tasks started", "processes or tasks cancelled", "processes or tasks not cancelled", "processes or tasks executedBy *someone*", "processes or tasks that start before *date*" and "processes or tasks that end after *date*".

The following abstract grammar has been defined to describe the PPI in combination with PIQL expressions. This grammar contains the most common elements of a grammar with numerical operations, extended with a special construction that allows the business expert to make queries over the environment where the process is running. When the PIQL expressions (PIQE) are evaluated at runtime, an Integer is obtained that represents the count of selected instances that match the specified criteria, can be combined with other values to obtain the desired PPI values.

PPI $\triangleq$ EXPR (ARITHMETIC_OP PPI)?
EXPR $\triangleq$ PIQE | Number
PIQE $\triangleq$ CONTEXT ListOfAttributes
CONTEXT $\triangleq$ P | T
ListOfAttributes $\triangleq$ AttributeComp BOOLEAN_OP ListOfAttributes
AttributeComp $\triangleq$ Attribute COMPARATOR_OP Attribute
Attribute $\triangleq$ String | Number | Date | Boolean | Null | Variable | ExternalVariable | Numerical_Expression

Different specific syntax can be used to describe the previous grammar. In order to facilitate the creation of the PPI description by a business expert, we propose a Domain Specific Language (DSL) closer to the natural language. Table I shows the patterns allowed, which help instantiate the grammar previously presented.

Table II shows the allowed predicates with the transformation in a DSL pattern. Furthermore DSL patterns and predicates permit the separation by using the words **with** or **that**. Predicates and separation words make the grammar more used friendly to business experts.

An application of this grammar is shown in the following section.

IV. Process Aware Performance Indicators

The definition of PPIs that uses the DSL described for the PIQEs facilitates the incorporation of the PPIs into the business process model by an expert. We propose their incorporation as information for the evaluation of the business rules defined using the DMN standard [7], since it provides a human decision-making model. Since the primary goal of DMN is to provide a common notation that is readily understandable by all business users, a friendly DSL is crucial. If the information on the instances can be included easily in the decision of the process, then the process model will be more adaptable and flexible to each instance. DMN provides a way to incorporate the decision rules into a decision task for routing the workflow

TABLE I
Patterns allowed

Grammar Component	**DSL Syntax**	
Context of Processes	The number of instances of processes	
Context of Tasks	The number of instances of tasks	
Attributes Defined for IP	**Attributes**	**DSL Syntax**
	idCase	*with a case id*
	Process_Name	*with a name*
	Task_Name	*with a name*
	Start	*with a start date*
	End	*with an end date*
	Cancelled	*cancelled*
	Who	*executed by the user*
ARITHMETIC_OP	**Operator**	**DSL Syntax**
	+	*plus*
	-	*minus*
	*	*multiplied by*
	/	*divided by*
BOOLEAN_OP	**Operator**	**DSL Syntax**
	$\wedge$	*and*
	$\vee$	*or*
COMP_OP	**Operator**	**DSL Syntax**
	$=$	*is equal to*
	$\neq$	*is not equal to*
	$<$	*is less than*
	$>$	*is greater than*
	$\leq$	*is less than or equal to*
	$\geq$	*is greater than or equal to*

TABLE II
Predicates allowed

Predicated	**Transformed pattern**
are finalized	end date is not equal to Null
are not finalized	end date is equal to Null
are cancelled	cancelled is not equal to Null
are not cancelled	cancelled is equal to Null
executed by {name}	the user is equal to {name}
start before {date}	a start date is less than {date}
end before {date}	an end date is less than {date}
start after {date}	a start date is greater than {date}
end after {date}	an end date is greater than {date}

in accordance with the evaluation of the decision. The DMN standard includes two components that need to be enlarged to support the incorporation of PIQEs into the decisions:

- **Decision table** defines a set of input variables used to make the decisions. These variables might be obtained from the data-flow. In our proposal, input variables can include PPIs related to process instance information. The grammar of the description of the variables is enlarged by using the PIQEs described above.
- **Business knowledge model** denotes a function encapsulating business knowledge (such as business rules, a decision table, or an analytic model). In our case, we use the tables to describe the business rules, thereby relating the obtained output (email, tweet and time_extension) in concordance with the input values (emails_bet, percentage_finalized_forecast_bet and tweet_after_15min). The expressions permitted in business knowledge tables are in Friendly Enough Expression Language (FEEL) and they are also described in the DMN standard. In the corner of this tables, there is necesarry to specify a hit policy. A hit

policy specifies how many rules of a decision table can be satisfied and which of the satisfied rules are included in the decision table result [7].

In order to illustrate the use of the grammar and the syntax of the DSL proposed, Figure 3 shows the DMN applied to the Tutiplay example. The PIQEs are used as input values and are defined in the Decision table (Figure 3.a), and the decision rules are included in the table of Decision knowledge (Figure 3.b), which are obtained by means of a transformation from each PIQE into a specific value. The Decision table and business knowledge are associated to the task "Decide action" of Figure 1.

The decisions described in the business knowledge are stored in a database and associated with an identifier, called *Decision Identifier*. When a decision is made, the business rule task requests the evaluation of the business rules by using this identifier, together with other input data needed by the engine to evaluate the associated PIQE, such as *Case id* and information from the data-flow. Once the rules have been evaluated, the decision result is returned to the decision task. This task is responsible for the incorporation of the results into the data-flow of the process, such as putting it into the data-flow in order to route the process execution.

The way in which the PIQEs are evaluated in the decision process is detailed in the following. Figure 4 shows the proposed architecture (denoted as DMN Extension) that uses the PIQEs obtained from a BPMS.

The DMN Extension is formed of two modules to support the description and evaluation of the PIQEs:

- **PIQE Engine Module** evaluates then PIQEs by using data received from invocation and data extracted from the BPMS. This is one of our proposals in the paper.
- **DMN Engine Module** evaluates the DMN rules. In Figure 4, this module is marked with *, since some BPMSs contain this engine and can be used as a part of the BPM services, although, as started earlier they do not support the inclusion of the PPIs.

The sequence of steps shown in Figure 4 that are executed to evaluate a decision routing, and involve the PPIs includes: (1) when a decision needs to be made, the business task calls the engine to communicate the identifier of decision to select the decision rules involved and the required data-flow of the instance, such as the case_id; (2) the decision process starts by managing the PIQE contained within the DMN decision table, and evaluates the PIQEs in accordance with the information obtained from the BPMS, if necessary; (3) once the PIQE Engine Module has the PIQEs resolved, then (4) the business knowledge is evaluated and the output that represents the decision taken is communicated to the PIQE Engine Module, and finally; (5) to route the execution, these variables are incorporated into the process data-flow.

V. Implementation of the proposal

In order to enable the incorporation of the decision concerning PPIs described by using PIQL in a real scenario, the architecture of Figure 4 has been implemented by using a set of mature technologies.

The BPMS used is *Camunda*™, since this is an open-source platform that includes other components necessary to conform the proposed architecture, such as a set of Application Programming Interfaces (APIs) used to extract the PPI values. The sample shown in previous section have been modelled in this platform.

The 'DMN Extension' has been implemented as an out-of-the box application, containing a set of APIs that allow the interaction with the BPMS and also containing a web interface that permits users manage the PIQEs. 'PIQL Engine' is the module responsible for the resolution of the PIQEs, to this end an Eclipse Modelling Framework (EMF) [11] is instantiated by using a grammar implemented using xText [12]. *xText* is an open-source framework for the development of programming languages and domain-specific languages, with features to describe the grammar and the parsing from the model description as an EMF model via text-to-model transformation.

The 'DMN Engine' have been implemented by using external libraries provided also by *Camunda*™.

VI. Related Work

A performance indicator is a quantitative or qualitative indicator that reflects the state of the processes of an organization. The importance of the measurement of key process indicators (KPIs) has been studied in several papers [13][14]. These indicators are fundamental in achieving strategic goals of the companies.

Indicators constitute a major element of business modelling as they offer criteria for the determination of whether an organization is fulfilling its objectives. They can involve strategic goals, quality requirements, or production targets. However, the problem not only consists of ascertaining whether the objectives are reachable, but it also involved reaching the objectives through the modification of behaviours to deviate the KPIs [15].

Various dimensions have been catalogued concerning KPIs (time, cost, quality, and flexibility), that can be refined into a number of PPI measures [16]. Furthermore, the way in which those measures can be described at design time has been studied in depth [4] [2].

In order to ascertain the PPI values, the process monitoring is fundamental. The importance of monitoring the processes is known and implemented in most commercial BPMSs [17]. Typically, monitoring has been used to validate the correct order of activity execution [18] and the compliance of the data-flow values [19]. Weske [20] distinguishes between Monitoring at the enactment stage and Business Process Administration (BPA) at the evaluation stage of the business process life-cycle. In order to extract the information, relevant Business Activity Monitoring (BAM) tools have been developed [21]. BAM enables the continuous real-time performance measurement of business processes based on PPI. On the other hand, BAM is employed to apply analysis techniques, such as

Decide action	
Decide action rules	
emails_bet	**The number of instances** of task 'Send reminder mail' **with** betId equal to $id
percent_finalized_forecast_bet	**(The number of instances** of process 'Forecast an outcomet' **that** are finalized and Id is equal to $id) divided by $num_users
tweet_after_15min	**The number** of instances of task 'send a tweet' that started before $current_time minus 900
forecasts_in_60min_by_bet	**The number of instances of process** 'Forecast an outcomet' **that** are finalized and betId equal to $id - **The number of instances** of process 'Forecast an outcomet' **that** end before $current_time minus 60 and betId equal to $id
extensions_by_bet	**The number of instances** of task 'Extend close time' **with** betId equal to $id
people_forecasting_by_bet	**The number of instances** of process 'Forecast an outcomet' **that** are not finalized and Id is equal to $id
date_to_close	$currentTime minus $close_time

(a) Decision about Actions

Decide action rules										
CL	emails_bet	percent_finalized_forecast_bet	tweet_after_15min	forecasts_in_60min_by_bet	extensions_by_bet	people_forecasting_by_bet	date_to_close	email	tweet	time_extension
								X,-	X,-	X,-
1	0	< 0.6	-	-	-	-	-	X	-	-
2	-	-	0	0	-	-	-	-	X	-
3	-	-	-	-	0	> 30	< date(00:30)	-	-	X

(b) Action Rules about Actions

Fig. 3. DMN Model applied to Tutiplay example

Process Mining [22] in the logged data, to check, for instance, the quality of models or the accuracy of the execution.

The recovery of KPIs and PPIs is the first step for the strategic business models. Decision-making processes can incorporate the values that the variables can take in the future [23][24]. Unfortunately, however to the best of our knowledge, they cannot be included in the evaluation and decision-making at run-time. The importance of the incorporation of decisions of a more complex nature during process execution remains an open problem, since PPIs cannot yet be easily included.

Several process areas can be improved by incorporating PPIs during process execution, not only for purposes of recovery, but also to incorporate how their values can affect the process execution at instantiation time. One of the most relevant examples is that of security applied to business processes [25] or to aspects of risk [26]. In both cases, the process analysis deductions are included by means of an analysis of the information and they are obtained in previous instances of the process, although they can be included in the model in a static way.

VII. Conclusions and Future work

This paper proposes an extension of the DMN standard, which allows business experts to automatize decision-making processes, by taking into account the process performance indicators available. To this end, we have developed a Process Instance Query Language (PIQL) that permits the extraction of information from the instances and from tasks executed in the instances. This information, related to the PPIs of the process, is incorporated in the decision knowledge through the Process Instance Query Expressions (PIQEs). The extraction of PPIs and alignment with the process decisions have been completed with the definition of an architecture and the implementation of a framework where a set of technologies has been combined to produce an usable solution. In order to validate our proposal, a real example has been used where the incorporation of the PPIs in the decisions is fundamental.

Thanks to our proposal, business experts can model decision-making processes by taking into account the current and past status of the business with a friendly language. In addition, the business instances become more agile and adaptive in terms of the rest of instances executed at any moment. Furthermore, the use of Process Instance Query Expressions enables business experts to include who, when and what instances are being executed at any moment.

From our point of view, future contributions could include the extension of PIQEs to enrich the information that can be included, in the same way as other concepts are considered, such as the use of resources, execution times, business load, and security aspects. We also consider the use of PIQL to be a major interest in the building of dashboards by the user experts, these dashboards can help in the business process governance.

VIII. Acknowledgement

This work has been partially funded by the Ministry of Science and Technology of Spain (TIN2015-63502-C3-2-R) and the European Regional Development Fund (ERDF/FEDER). Thanks to Lesley Burridge for revision of the English version of the manuscript.

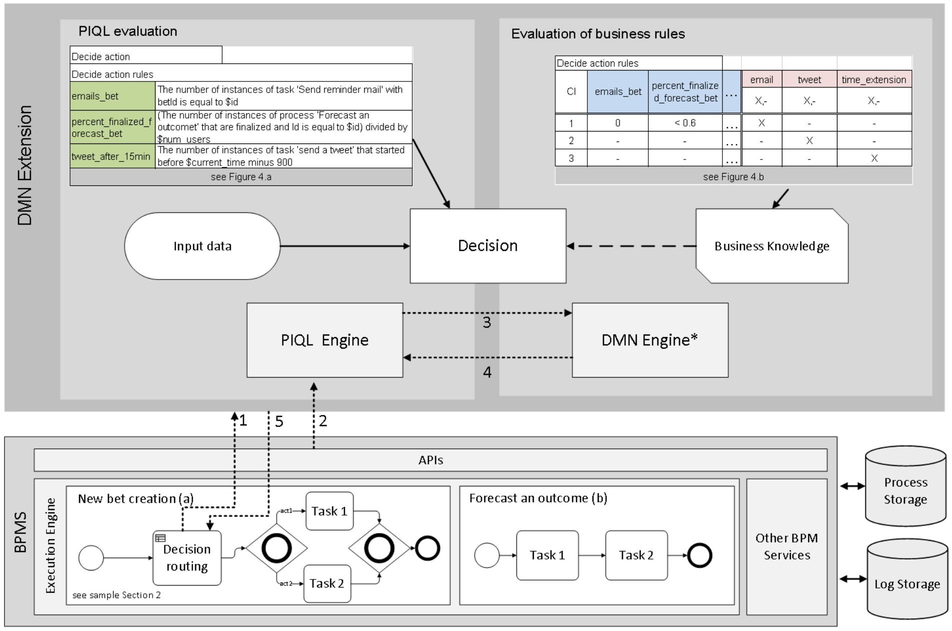

Fig. 4. Decision-Process Architecture

REFERENCES

[1] A. Maté, J. Trujillo, and J. Mylopoulos, "Conceptualizing and specifying key performance indicators in business strategy models," in *Conceptual Modeling - 31st International Conference ER 2012, Florence, Italy, October 15-18, 2012. Proceedings*, 2012, pp. 282–291.

[2] A. del-Río-Ortega, M. Resinas, and A. R. Cortés, "Defining process performance indicators: An ontological approach," in *On the Move to Meaningful Internet Systems: OTM 2010 - Confederated International Conferences: CoopIS, IS, DOA and ODBASE, Hersonissos, Crete, Greece, October 25-29, 2010, Proceedings, Part I*, 2010, pp. 555–572.

[3] R. Tardío and J. Peral, "Obtaining key performance indicators by using data mining techniques," in *Advances in Conceptual Modeling - ER 2015 Workshops, AHA, CMS, EMoV, MoBiD, MORE-BI, MReBA, QMMQ, and SCME Stockholm, Sweden, October 19-22, 2015, Proceedings*, 2015, pp. 144–153.

[4] A. del-Río-Ortega, M. Resinas, C. Cabanillas, and A. R. Cortés, "On the definition and design-time analysis of process performance indicators," *Inf. Syst.*, vol. 38, no. 4, pp. 470–490, 2013.

[5] O. González, R. Casallas, and D. Deridder, "Monitoring and analysis concerns in workflow applications: from conceptual specifications to concrete implementations," *Int. J. Cooperative Inf. Syst.*, vol. 20, no. 4, pp. 371–404, 2011.

[6] OMG, *Object Management Group, Business Process Model and Notation (BPMN) Version 2.0*. OMG Standard, 2011.

[7] O. M. Group, *Decision Model and Notation. Reference Manual*. OMG Standard, 2014.

[8] Tutiplay. (2015) Tutiplay. [Online]. Available: http://www.tutiplay.com

[9] M. Dumas, M. L. Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management.* Springer, 2013.

[10] A. H. M. ter Hofstede, C. Ouyang, M. L. Rosa, L. Song, J. Wang, and A. Polyvyanyy, "APQL: A process-model query language," in *Asia Pacific Business Process Management - First Asia Pacific Conference, AP-BPM 2013, Beijing, China, August 29-30, 2013. Selected Papers*, 2013, pp. 23–38.

[11] D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, *EMF: Eclipse Modeling Framework 2.0*, 2nd ed. Addison-Wesley Professional, 2009.

[12] *xText Documentation*, 2015. [Online]. Available: https://www.eclipse.org/Xtext/

[13] V. Popova and A. Sharpanskykh, "Modeling organizational performance indicators," *Inf. Syst.*, vol. 35, no. 4, pp. 505–527, Jun. 2010.

[14] L. Sánchez-González, F. García, F. Ruiz, and J. Mendling, "Quality indicators for business process models from a gateway complexity perspective," *Information & Software Technology*, vol. 54, no. 11, pp. 1159–1174, 2012.

[15] D. Barone, L. Jiang, D. Amyot, and J. Mylopoulos, "Reasoning with key performance indicators," in *The Practice of Enterprise Modeling - 4th IFIP WG 8.1 Working Conference, PoEM 2011 Oslo, Norway, November 2-3, 2011 Proceedings*, 2011, pp. 82–96.

[16] A. Baumgraß, N. Herzberg, A. Meyer, and M. Weske, "BPMN extension for business process monitoring," in *Enterprise modelling and information systems architectures - EMISA 2014, Luxembourg, September 25-26, 2014*, 2014, pp. 85–98.

[17] K. Goser, M. Jurisch, H. Acker, U. Kreher, M. Lauer, S. Rinderle, M. Reichert, and P. Dadam, "Next-generation process management with adept2," in *BPM*, 2007.

[18] L. T. Ly, S. Rinderle-Ma, and P. Dadam, "Design and verification of instantiable compliance rule graphs in process-aware information systems," in *CAiSE*, 2010, pp. 9–23.

[19] M. T. Gómez-López, R. M. Gasca, and J. M. Pérez-Álvarez, "Compliance validation and diagnosis of business data constraints in business processes at runtime," *Inf. Syst.*, vol. 48, pp. 26–43, 2015.

[20] M. Weske, *Business Process Management: Concepts, Languages, Architectures*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2007.

[21] O. B. A. Monitoring. (2016) Oracle. [Online]. Available: http://goo.gl/PFexkD

[22] W. M. P. van der Aalst, *Process Mining - Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.

[23] M. T. Gómez-López, R. M. Gasca, and J. M. Pérez-Álvarez, "Decision-making support for the correctness of input data at runtime in business processes," *Int. J. Coop. Info. Syst.*, vol. 23, no. 2, p. 29, 2014.

[24] M. T. Gómez-López, R. M. Gasca, L. Parody, and D. Borrego, "Constraint-driven approach to support input data decision-making in business process management systems," in *Information Systems Development, Reflections, Challenges and New Directions [Proceedings of ISD 2011, Heriot-Watt University, Edinburgh, Scotland, UK, August 24 - 26, 2011]*, 2011, pp. 457–469.

[25] A. J. Varela-Vaca and R. M. Gasca, "Formalization of security patterns as a means to infer security controls in business processes," *Logic Journal of the IGPL*, vol. 23, no. 1, pp. 57–72, 2015.

[26] R. Conforti, M. de Leoni, M. L. Rosa, W. M. P. van der Aalst, and A. H. M. ter Hofstede, "A recommendation system for predicting risks across multiple business process instances," *Decision Support Systems*, vol. 69, pp. 1–19, 2015.