Enterprise Information Systems
Vol. 00, No. 00, January 2014, 1–33

RESEARCH ARTICLE

Using Templates and Linguistic Patterns to define Process Performance Indicators

Adela del-Río-Ortega[a*], Manuel Resinas[a], Amador Durán[a] and Antonio Ruiz-Cortés[a]

[a]*Dpto. de Lenguajes y Sistemas Informáticos, Universidad de Sevilla, Av. Reina Mercedes s/n, 41012 Seville, Spain*;

(*v1.0 released January 2013*)

Abstract

Process performance management (PPM) aims at measuring, monitoring and analysing the performance of business processes (BPs), in order to check the achievement of strategic and operational goals and to support decision making for their optimisation. PPM is based on process performance indicators (PPIs), so having an appropriate definition of them is crucial. One of the main problems of PPIs definition is to express them in an unambiguous, complete, understandable, traceable and verifiable manner. In practice, PPIs are defined informally —usually in ad-hoc, natural language, with its well-known problems— or they are defined from an implementation perspective, hardly understandable to non–technical people. In order to solve this problem, in this article we propose a novel approach to improve the definition of PPIs using templates and linguistic patterns. This approach promotes reuse, reduces both ambiguities and missing information, is understandable to all stakeholders and maintains traceability with the process model. Furthermore, it enables the automated processing of PPI definitions by its straightforward translation into the PPINOT metamodel, allowing the gathering of the required information for their computation as well as the analysis of the relationships between them and with BP elements.

Keywords: process performance indicators; key performance indicators; business process management; process performance management; business performance monitoring; business process analysis; templates; patterns

*Corresponding author. Email: adeladelrio@us.es

ISSN: 1751-7575 print/ISSN 1751-7583 online

DOI: 10.1080/17517575.2013.867543
http://www.informaworld.com

1. Introduction

Business Process Management (BPM) aims at offering a high level managerial perspective of organisations. It can be seen as a principle to manage businesses: a company provides to the market products or services, which are the outcome of a number of activities performed. Business processes are the key instrument to organise these activities and to improve in general their relationships (Weske 2007). Nowadays, many companies are adopting a process-oriented perspective in their business, as "a way of identifying which steps really create value, who is involved in the process and which is the exchanged information; ultimately, finding out how to improve, where to increase quality, reduce waste or save time" (Alexander Grosskopf and Weske 2009).

According to Kronz (2006), collecting and analyzing process-related Key Performance Indicators (KPIs) is the first prerequisite for holistic process management and form the basis for consistent and continuous process optimization. These process-related KPIs are also known as Process Performance Indicators (PPIs) and can be defined as quantifiable metrics that allow the evaluation of the efficiency and effectiveness of business processes. They can be measured directly by data that is generated within the process flow and are aimed at the process controlling and continuous optimization (Chase *et al.* 2011).

During the definition of PPIs, several desirable properties (Franceschini *et al.* 2007, del Río-Ortega *et al.* 2010, del Río-Ortega *et al.* 2012) must be taken into account: (1) their definition, that should be unambiguous and complete, fulfilling the SMART criteria (cf. Section 2); (2) understandability, *i.e.* PPIs should be understood and accepted by process managers and employees; (3) traceability to the business process, enabling to maintain coherence between both assets, Business Process (BP) models and PPIs; and (4) possibility to be automatically analysed, allowing not only to gather the information required to compute PPI values, but also to infer knowledge to answer questions like *what are the business process elements related to certain PPI?*[1].

A notation for the definition of PPIs that fulfills the aforementioned properties is still an unresolved challenge. The main reason is they are commonly considered conflicting properties, especially understandability to non-technical users and automated analysis. This is why in practice, PPIs are defined either in an informal and ad-hoc way, usually in natural language, or from an implementation perspective, at a very low level, too formal and/or close to the technical view. As for the former, problems of ambiguity, lack of coherence/traceability with the process, incompleteness in the sense of missing information, and not amenability to automated analysis arise. As for the latter, those PPI definitions become hardly understandable to managers and users.

In this paper, we propose a novel notation to improve the definition of PPIs based on templates and linguistic patterns (L–Patterns). The use of templates for the definition of PPIs (*c.f.* Table 2) helps to structure the information in a normalised form, reduces ambiguity, promotes reuse and also serves as a guide to avoid missing relevant information. Furthermore, filling blanks in prewritten sentences, *i.e.* L–Patterns, is easier and less error-prone than writing them from scratch. Moreover, this approach has been successfully used in the past in the areas of Requirements Engineering (Durán *et al.* 1999, 2002) and Service Level Agreements (Ruiz-Cortés *et al.* 2001, 2005).

Both templates and L–Patterns are based on the PPINOT metamodel (del Río-Ortega *et al.* 2012), which is a metamodel that identifies the concepts that are necessary for

[1]Further information regarding these analysis operations amenable to be performed on PPI definitions are described in del Río-Ortega *et al.* (2012) and del Río-Ortega (2012).

defining PPIs such as the different types of measures that can be used to compute the PPI value. Equipped with these concepts, one can create unambiguous, traceable and complete models of PPIs (del Río-Ortega *et al.* 2012). This metamodel is the result a thorough review of the related literature and a study of the state-of-practice of PPIs definition in several real organisations.

An advantage of being based on the PPINOT metamodel is that the mapping from PPI-templates and L-patterns to this metamodel is straightforward (cf. Section 7). This mapping helps avoiding mistakes in the produced PPI definitions using our templates and patterns, since they are restricted to those allowed by this metamodel, *i.e.* it is not possible to define PPIs with templates and patterns without being according to it. Moreover, the mapping also allows one to leverage the techniques and tools developed around the metamodel to automate the processing of PPIs. This provides two main advantages, namely the automatic extraction of the information required to compute PPI values during BP execution, and the implementation of design-time analysis operations to provide information that can assist process analysts in the definition and instrumentation of PPIs (del Río-Ortega *et al.* 2012). All these features together with a graphical editor for the modelling of PPIs over Business Process Diagrams (BPDs) using PPINOT graphical notation were materialised in a software tool called PPINOT Tool Suite[1]. In this paper, we extend the PPINOT Tool Suite with a template editor to allow users to define PPIs using the aforementioned templates and L–Patterns.

A first version of the PPI templates and L–Patterns was presented in del Río-Ortega *et al.* (2012). This paper extends those preliminary results as follows. First, we present a refinement of those template and L–Patterns that is the result of their application to several real scenarios (*c.f.* Section 8). This application has also led to the definition of abbreviated L–Patterns for the most common cases, also described in Section 8. Further contribution of this paper includes templates and L–Patterns for other fields of the PPI template (target and scope, in Sections 5 and 6 respectively), the mapping to the PPINOT metamodel, software tool support and the evaluation conducted to test our approach.

With this contribution, we overcome the problem of providing a mechanism to define PPIs fulfilling the four properties above: PPI definitions are unambiguous and avoid missing information thanks to the use of templates and L–Patterns; they are more understandable to all stakeholders than usual approaches based on technical concepts, because they use natural language and concepts closer to the process; they are traceable to the BP since explicit reference to BP elements are included (*c.f.* Sections 3 and 4); they are amenable to automated management thanks to their underlying formal model. In addition, the results of applying our approach to three real scenarios (*c.f.* Section 8) served us to validate and refine the presented templates and patterns.

The remainder of this paper is structured as follows. In Section 2 we make a brief introduction to PPIs, presenting a real scenario that served as motivation for our research. Then in Section 3 we propose a PPI template for the definition of PPIs, using also L–Patterns. A further explanation of some of the L–Patterns used is contained in Sections 4, 5, and 6. The mapping of these PPI templates and patterns to the PPINOT metamodel as well as some details regarding the tooling support are described in Section 7. Section 8 presents the application of our proposal to real scenarios and the lessons learned. In Section 9 we present related work. Finally, Section 10 draws the conclusions from our work and outlines our future work.

[1]Accessible at `http://www.isa.us.es/ppinot`

2. PPI in a Nutshell

As stated in Section 1, a PPI can be defined as a quantifiable metric that allows the evaluation of the efficiency and effectiveness of business processes, and can be measured directly by data that is generated within the process flow. They are aimed at the process controlling and continuous optimization (Chase *et al.* 2011).

Often, the terms PPIs and KPIs are used interchangeably, although, there is no consensus in the literature regarding the relationship between PPIs and KPIs. Some authors do not establish any difference between them (Popova and Sharpanskykh 2009, 2010, Momm *et al.* 2009), while others consider PPIs as a particular case of KPIs, i.e. process-related KPIs (Wetzstein *et al.* 2008, del Río-Ortega *et al.* 2010). Finally, there are others who give different definitions to each one, placing them at different levels, KPIs nearest to the tactical and strategical level, while PPIs nearest to the operational level (Chase *et al.* 2011). We agree with the second approach and consider PPIs as a particular case of KPIs defined for measuring the performance of BPs.

2.1. *Motivating Scenario*

Let us take as an example the business process depicted in Figure 1. It belongs to one of the real scenarios to which our approach has been applied and takes place in the context of the Information Technology Department of the Andalusian Health Service. This diagram corresponds to the BP for managing Requests for Changes (RFCs) on existing Information Systems. It is modelled using BPMN 2.0 (Object Management Group (OMG) 2011), the de facto standard for modelling BPs, which is broadly extended and used nowadays in both, industry and academy. According to BPMN, activities are depicted as rounded rectangular boxes; events, which include receiving and throwing events, are depicted as circles; data objects are depicted as a sheet of paper with top right corner folded, and gateways, which control how the process flows, are depicted as diamonds.

The process starts when the *requester* submits an RFC. Then, the *planning and quality manager* must assign a priority and analyse the RFC in order to make a decision. If the RFC was in the strategic plan or pre-approved, the *requester* is asked to submit a release request and the process ends. Otherwise, according to several factors like the availability of resources, the requirements requested, and others, the RFC is either approved, cancelled, raised to a committee for further decision, paralysed or sent to the *area manager* in order for her to negotiate new requirements.

In addition, throughout the process, the RFC document[1] may pass through several states: *registered*, *in analysis*, *paralysed*, *cancelled*, *approved*, *waiting for negotiation*, *with new requirements* and *cancelled due to successful negotiation*. Furthermore, every RFC has a number of properties such as *Project*, referred to the project to which such RFC is associated; *Information systems*, referred to the number of information systems for which a change is required or to which that change affects; *Type of change*, that classifies the change required in *adaptive*, *corrective* or *perfective*, and *Priority*, referred to the importance of RFC resolution.

For the given process, a set of nine PPIs was defined. They were defined in natural language and collected in a table (an excerpt is shown in Table 1)[2].

[1]In BPMN documents are called *data objects*. Therefore, from now on we will use these terms interchangeably.
[2]Target values reflected in Table 1 are not the real ones due to the privacy policies of the organisation.

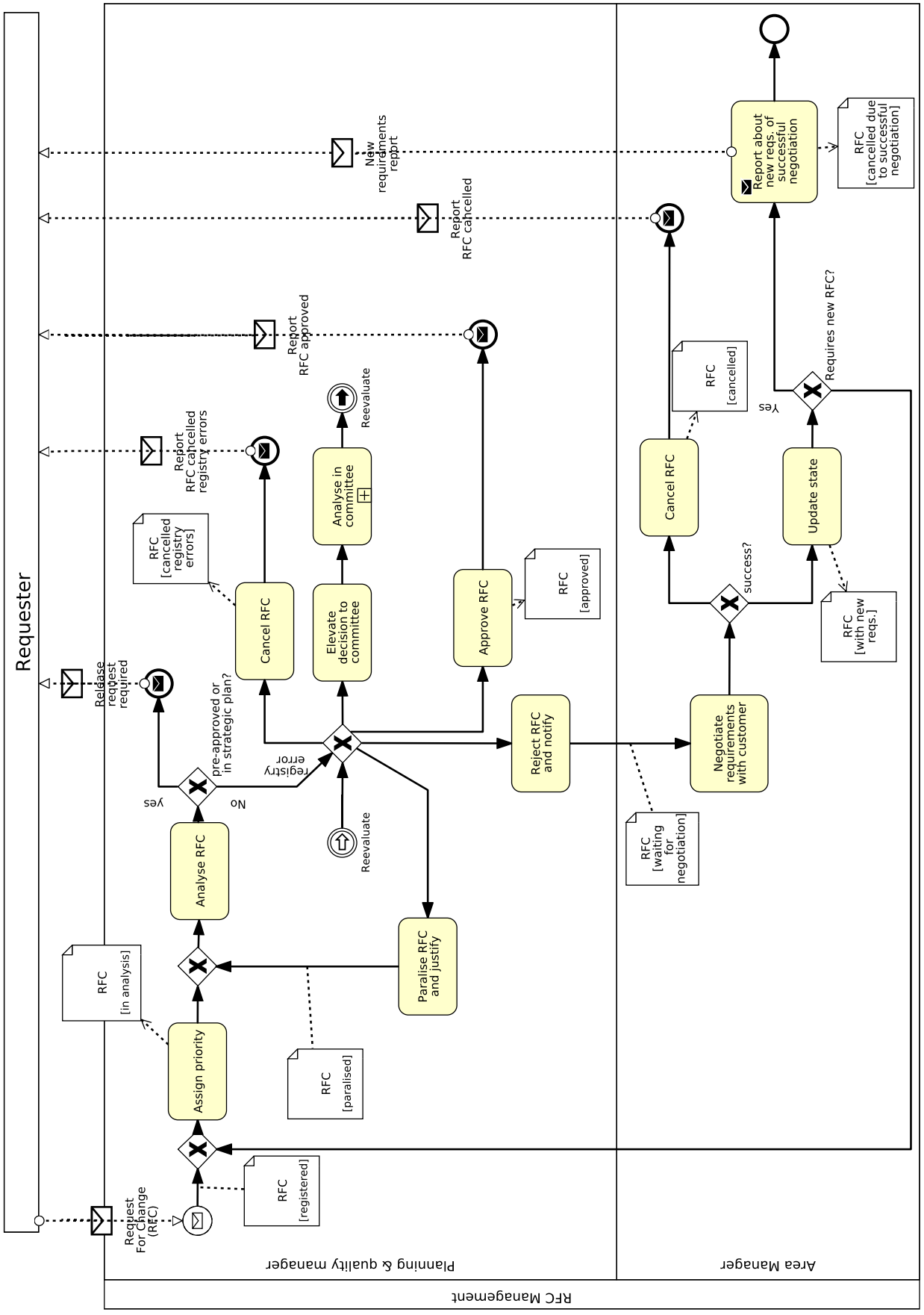

Figure 1. Process of the Request For Change (RFC) management

Table 1. PPIs defined for the RFC management process in Figure 1, of the IT Department of the Andalusian Health Service. All these PPIs have as responsible the *planing and quality manager* role

ID	Description	Target value	Scope
PPI1	RFCs cancelled out of RFCs registered	$\leq 4\%$	weekly
PPI2	Average time of committee decision	≤ 1 working day	weekly
PPI3	Corrective RFCs out of approved RFCs	$\leq 2\%$	weekly
PPI4	Perfective and adaptive RFCs from approved RFCs	$\geq 4\%$	weekly
PPI5	Average time of RFC analysis	≤ 2 working days	weekly
PPI6	Number of RFCs under analysis	≤ 2 RFCs	weekly
PPI7	Number of RFCs per type of change	≤ 20 corrective RFCs ≤ 30 adaptive RFCs ≤ 20 perfective RFCs	monthly
PPI8	Number of RFCs per project	≤ 50 for project RR.HH ≤ 60 for project Diraya ≤ 1 for project Pharma	monthly
PPI9	Average lifetime of an RFC	≤ 3 working days	monthly

2.2. *Defining PPIs with PPINOT*

Like other KPIs, PPI definitions are recommended to satisfy the SMART criteria (Doran 1981, Meyer 2003, Shahin and Mahbod 2007). SMART is a mnemonic that broadly conforms to the following words: *Specific*, it has to be clear what the PPI exactly describes; *Measurable*, it has to be possible to measure a current value and to compare it to the target one; *Achievable*, it makes no sense to pursue a goal that will never be met; *Relevant*, it must be aligned with a part of the organisation's strategy, something that really affects its performance; and *Time-bounded*, a PPI only has a meaning if it is known the time period in which it is measured.

The definition of SMART PPIs requires a detailed specification of several characteristics of a PPI such as how it is measured or which is its target. The identification of the concepts that are necessary for defining such characteristics of SMART PPIs is the main purpose of the PPINOT metamodel proposed in del Río-Ortega *et al.* (2012). This metamodel is the result of a thorough review of the related literature and the study of the current picture in several real organisations. In particular, three real scenarios served to validate its expressiveness, namely the IT Department of the Andalusian Health Service, the Information and Communication Service of the University of Seville and a part of the administration of the Andalusian Regional Government. In summary, PPINOT metamodel allowed the definition of all of the 54 PPIs for 10 different process models ranging from a Request For Change (RFC) process to user management process and evaluation and certification management process.

A simplified view of the PPINOT metamodel is shown in Figure 2. It depicts the attributes that are usually provided while defining a PPI. Some of them are present in Table 1, namely: an *ID*, a *description*, a *target value*, and a *scope*. A detailed description of them is provided in Sections 3 and subsequent.

Concerning, the *measure definitions* associated to a PPI can be divided into three different types, namely: *base measures*, *aggregated measures* and *derived measures*. These

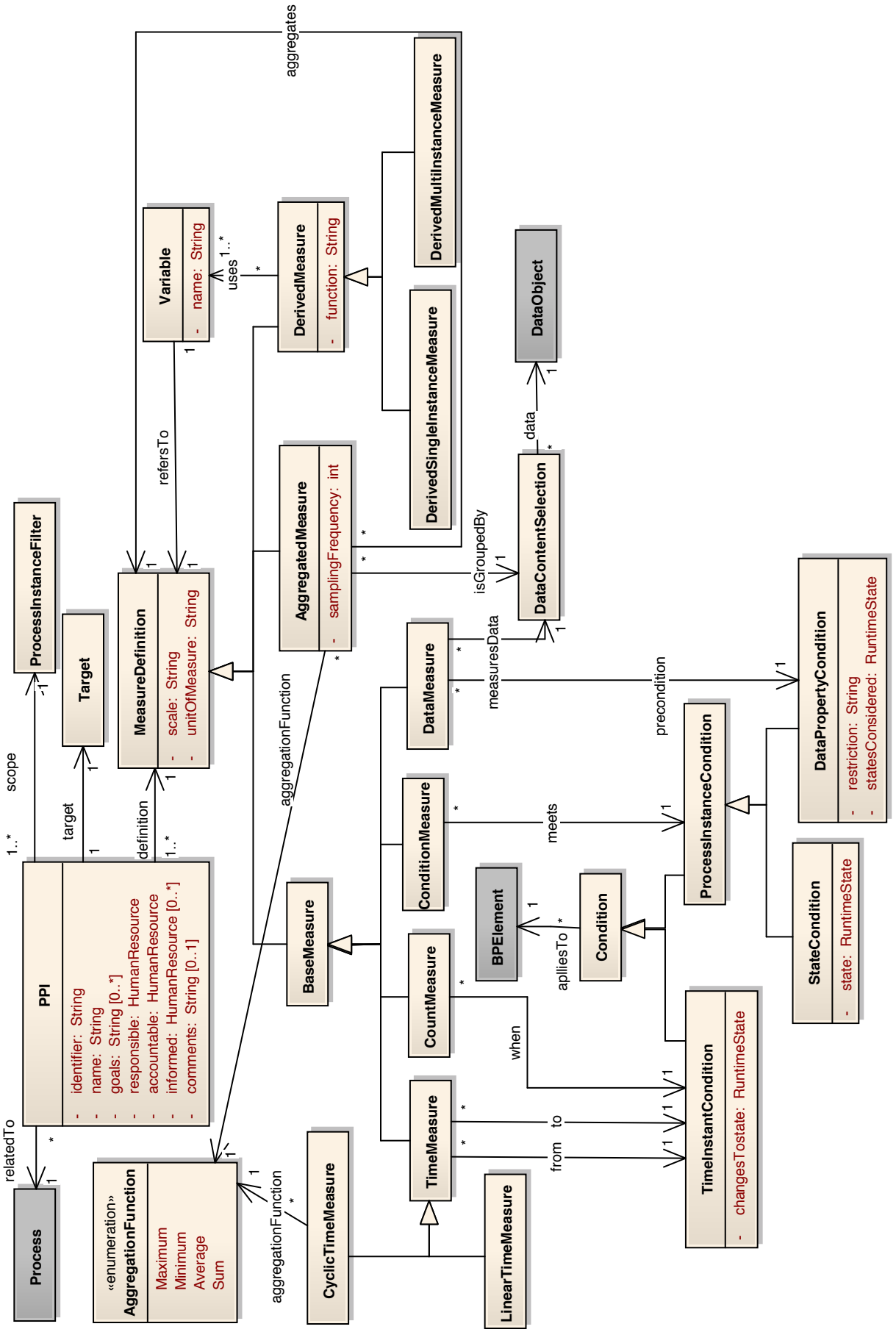

Figure 2. PPINOT metamodel overview

types can be classified according to two different dimensions (cf. Figure 3): *The number of process instances necessary to compute the measure value*, and *the nature of the measure*. As for the former, the measure definitions are *single-instance measures* if a single process instance is used to take the measure, and *multi-instance measures* if the measure value is calculated using a set of process instances. As for the latter, measures can be classified as: *time measure* to reflect the duration between two time points in the process, for instance, the duration of activity analyse in committee, being the instants the start and the end of the activity; *count measure* to count the number of times certain condition is satisfied, such as activity analyse in committee changes its state to completed; *condition measure* to check if certain condition is (for running instances) or has been (for finished instances) met, *e.g.* if data object RFC has high priority; *data measure* to take the value of a property of certain data object, for instance the value of property *information systems* of data object RFC; and *derived measure*, when it is calculated by performing a mathematical function over any number of measures previously defined, *e.g.* the percentage of the duration of activity analyse in committee out of the duration of the whole process.

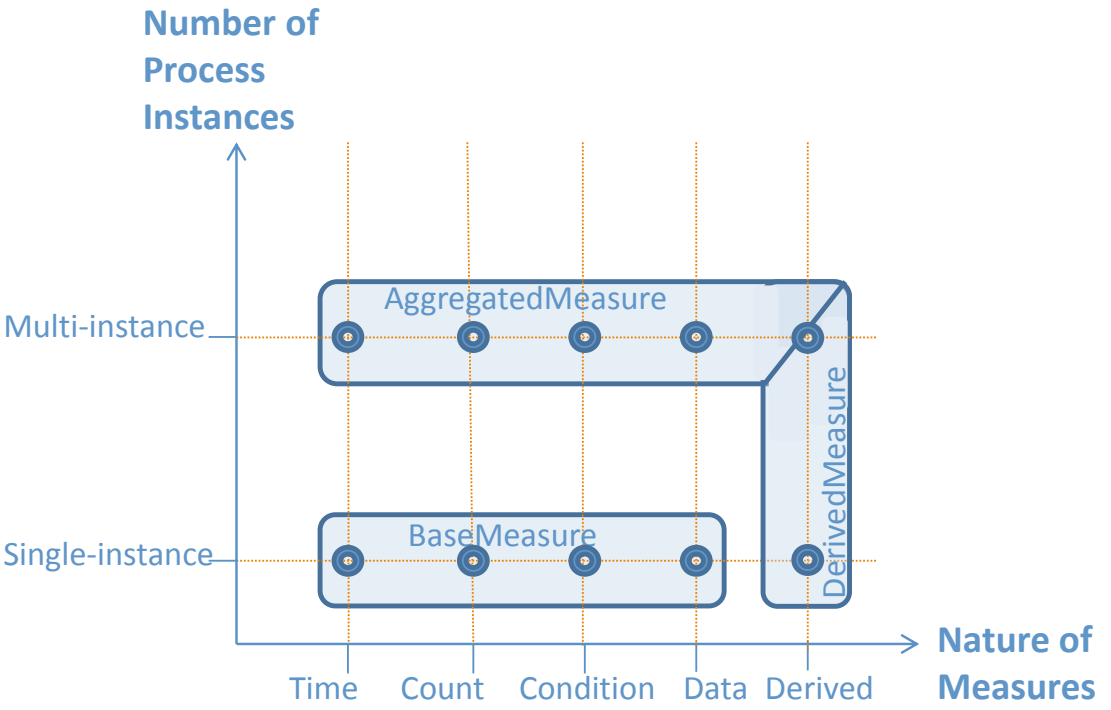

Figure 3. PPI dimensions

Consequently, according to these dimensions, *base measures* group all the single-instance measures that can be calculated without using any other measure. They include *time*, *count*, *condition* and *data* measures. *Aggregated measures* are multi-instance measures calculated by applying an aggregation function on different instances of a single-instance measure. Finally, *derived measures*, as stated above, are those whose value is calculated by performing a mathematical function over other measures. Derived measures can be both, single-instance and multi-instance measures, depending on the measures used to calculate its value.

Note that *aggregated measures* and *derived measures* are adjoining in Figure 3 because a derived multi-instance measure of an aggregated measure and an aggregated measure that aggregates a derived single-instance measure are two different types of measures.

For instance, let $\Delta_t AC$ be the duration of activity analyse in committee, and $\Delta_t P$ be the duration of the whole process, a derived multi-instance measure of an aggregated measure could be defined as $\frac{max(\Delta_t AC)}{max(\Delta_t P)}$, whereas an aggregated measure of a derived single-instance measure could be something like $max(\frac{\Delta_t AC}{\Delta_t P})$.

3. Template for the Definition of PPIs

In this section and the following three (4, 5 and 6), templates and linguistic patterns are proposed for the definition of PPIs and their associated characteristics. These templates are inspired by the requirements templates originally proposed by Durán *et al.* (1999). As commented in that work, the use of templates helps to structure the information in a fixed form, reduces ambiguity, promotes reuse, and also serves as a guide to avoid missing relevant information. Furthermore, following Durán *et al.* (1999, 2002), L-patterns are also integrated in the proposed templates because filling blanks in prewritten sentences is easier, faster and less error–prone than writing whole paragraphs from scratch.

Table 2. Template for PPI specification

PPI–<*id*>	**<*PPI descriptive name*>**
Process	<*process name (process id)*>
Goals	<*strategic or operational goals the PPI is related to*>
MeasureDefinition	The PPI value is calculated as <*Measure*>
Target	The PPI value <*Target constraint*>
Scope	The process instances considered for this PPI are ◦ all ◦ [those in] <*Scope (S–x)*>
Source	<*source from which the PPI measure can be obtained*>
Responsible	{ <*role*> | <*department*> | <*organization*> | <*person*>}
Informed	{ <*role*> | <*department*> | <*organization*> | <*person*>}
Comments	<*additional comments about the PPI*>

Template for the definition of PPIs is shown in Table 2. It has been defined in order to fulfill the SMART criteria for the definition of PPIs and is based on the PPINOT metamodel (del Río-Ortega *et al.* 2012). The notation used in the template is the following: words between "<" and ">" are placeholders for either literals (lower case) or L-patterns (upper case first letter); words between "{" and "}" and separated by "|" are one-only options; words between "[" and "]" are optionals. The meaning of the template fields is the following:

- **Identifier and descriptive name**: every PPI must be uniquely identified by a number and a descriptive name in order to allow traceability. In order to help rapid identification, PPIs identifiers start with PPI.
- **Process**: this field refers to the process for which the PPI is defined.

- **Goals**: this field allows the user to explicitly state the strategic or tactical goal/s that the PPI is related to. Associating a set of goals with a PPI highlights its relevance, thus connecting to the *Relevant* characteristic of the SMART criteria. This field can be filled with an expression in natural language. A more formal definition of the relationship between the PPI and the organisational goals is out of the scope of this paper. Some approaches regarding this issue can be found in (Popova and Sharpanskykh 2010, 2011, Barone *et al.* 2010).
- **MeasureDefinition**: this field indicates the measure associated to this PPI and is defined by means of one of the L-patterns described in Section 4. With this field, the two first characteristics of the SMART criteria, *Specific* and *Measurable*, are fulfilled.
- **Target**: this field defines the aimed-for value for this PPI. The target of a PPI must be aligned with the goals the PPI is related to so that it helps to determine the degree of goal fulfilment. Furthermore, in order to fulfill the *Achievable* characteristic of the SMART criteria, this target value must be reasonable, based on previous experiences and/or simulations when possible. This field is defined by means of one of the L-patterns described in Section 5.
- **Scope**: this field is used to select the process instances that must be considered to compute the PPI value. It is defined either considering every existing process instance (*i.e.* the whole set of process instances for which execution data is available), or by means of a scope template, described in Section 6. This scope template is usually based on the moment in which the process instance started, *e.g.*, those process instances started last month. This field contributes to achieve the *Time-Bounded* characteristic of the SMART criteria.
- **Source of information**: this field references the source from which the required information to obtain the PPI measure is gathered, for example event logs.
- **Responsible**: this field specifies the human resource in charge of the PPI. This human resource can be a person, a role, a department or an organisation.
- **Informed**: this field specifies the human resources that must be informed about the state of this PPI. These human resources can be persons, roles, departments or organisations.
- **Comments**: other information about the PPI that cannot be fitted in previous fields can be recorded here.

An example of use of this PPI template is shown in Table 3. It represents the definition of the PPI5 from Table 1, and contains a reference to scope *Last 100 instances (S–5)*, whose definition can be found in Appendix A, Table A7. The whole set of PPI and scope templates for each PPI from Table 1 are listed in Appendix A.

4. Catalogue of Measure L–Patterns

These L–Patterns provide the traceability required between PPIs and the BP elements thanks to the explicit references to the BP elements. They are organised according to the two dimensions described in Section 2: the number of process instances and the nature of measures. Therefore, L–Patterns for time, count, condition, data, derived and aggregated measures are described.

Table 3. PPI template example

PPI–005	**Average time of RFC analysis**
Process	Request for change (RFC)
Goals	• BG–002: Improve customer satisfaction • BG–014: Reduce RFC response time
MeasureDefinition	The PPI value is calculated as the average of the duration of activity *RFC Analysis*
Target	The PPI value must be less than or equal to *1 working day*
Scope	The process instances considered for this PPI are *Last 100 instances (S–5)*
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	*Most RFCs are created after 12:00.*

4.1. *L–Patterns for Time Measures*

A time measure can be intuitively defined as the difference between the instants when two events occur, considering as events not only BP event triggerings, but also BP element transitions. If the notation for the modelling of BPs is BPMN 2.0 (Object Management Group (OMG) 2011), the BP elements considered are *activities*, *processes*, *events* and *data objects*; and the states that may be used for them together with the rest of possible placeholders used along the whole set of L–Patterns presented in this section are summarised in Table 4. However, although this definition is enough for the usual cases, i.e., those in which events occur just once in each instance, it does not consider those cases in which the time measure is taken between elements located within a loop.

To take both cases into account, it is necessary to distinguish between two kinds of time measures, namely: *Linear*, which measures the difference between the first occurrence of the first event and the last occurrence of the second event; and *Cyclic*, in which the elapsed time between the pairs of event occurrences of each iteration is aggregated by using one aggregation function (e.g. sum, maximum or average). Having said that, the *Time Measure* L–Pattern can be defined as:

$$\textit{TimeMeasure} \quad ::= \quad \textit{LinearTimeMeasure} \mid \textit{CyclicTimeMeasure}$$

where the *LinearTimeMeasure* and *CyclicTimeMeasure* L–Patterns can be defined as:

$$\begin{aligned} \textit{LinearTimeMeasure} \quad &::= \quad \textit{the duration between the [first] time instant[s] when} \\ &\quad\; <\text{Event}_1> \textit{and [the last time instant] when} <\text{Event}_2> \\ \textit{CyclicTimeMeasure} \quad &::= \quad \textit{the } \{ \textit{ total } \mid \textit{ maximum } \mid \textit{ minimum } \mid \textit{ average } \mid \ldots \\ &\quad\; \} \textit{ duration between the pairs of time instants when} \\ &\quad\; <\text{Event}_1> \textit{and when} <\text{Event}_2> \end{aligned}$$

Finally, the *Event* L–Pattern can be defined as:

Table 4. Definition of placeholders in measure L–Patterns

Placeholder	**Description**
<BP element type>	One of the different types of elements of the BP referred. In the case of BPMN 2.0, they can be: *activity*, *data object*, *event* and *process*.
<BP element name>	The name of one of the existing BP elements in the process referenced in the PPI. *e.g.* Assign priority for the BP element type activity in our RFC process example.
<BP element state>	Possible states for each type of BP element. For instance, in BPMN 2.0, the possible states for activities are *ready, active, withdrawn, completing, completed, failing, failed, terminating, terminated, compensating* and *compensated*.
<BP event name>	The name of one of the existing events in the process referenced in the PPI, *e.g.* receive RFC in our RFC process example.
<Data object name>	The name of one of the existing data objects in the process referenced in the PPI, *e.g.* RFC in our RFC process example.
<Data object state>	Possible states for each data object, defined by the user during the definition of the process, *e.g. registered* for the data object RFC in our RFC process example.
<Data object property name>	The name of one of the defined properties for the existing data objects in the process referenced in the PPI, *e.g. affected departments* for the data object RFC in our RFC process example. Note that the way these data object properties are defined can vary, for instance, if the data object is defined as an XML document, the property could be an XPath expression pointing to a specific part of the XML.

$$\begin{aligned} \textit{Event} \quad ::= \quad & \textit{<BP element type> <BP element name> becomes} \\ & \textit{<BP element state>} \\ | \quad & \textit{event <BP event name> is triggered} \end{aligned}$$

Note that whenever a time measure is taken between elements that are not located within a loop, the *Linear Time Measure* L–Pattern will be used, but suppressing the elements in square brackets.

For example, in order to measure the duration of the Analyse RFC activity, the *Linear Time Measure* L–Pattern can be instantiated as:

the duration between the time instants when activity RFC analysis becomes active *and when* activity RFC analysis becomes completed

Since activity Analyse RFC is located within a loop, it is also possible to measure its average duration for one instance, by instantiating the *Cyclic Time Measure* L–Pattern as follows:

the average duration between the pairs of the time instants when activity RFC analysis becomes active *and when* activity RFC analysis becomes completed

4.2. *L–Patterns for Count Measures*

A count measure for PPIs counts the number of times a specific *event*—as considered in previous section—happens. Therefore, its corresponding L–Pattern can be as simple as

CountMeasure ::= *the number of times* <Event>

For example:

the number of times activity Analyse RFC becomes completed

4.3. *L–Patterns for Condition Measures*

A condition measure takes boolean values depending on either the state of a BP element or a condition specified on a data object. The two corresponding L–Patterns can be:

ConditionMeasure ::= <*StateCondM*> | <*DataPropertyCondM*>
StateCondM ::= <*BP element type*> <*BP element name*> [*that*] { [*is*] [*not*] *currently* | *has* [*not*] *finished* } [*in state*] <*BP element state*>
DataPropertyCondM ::= [<*data object state*>] <*data object name*> *that satisfies*: <*condition on data object properties*>

For example, for each of them respectively:

Activity Analyse in committee *that is currently* active

approved RFC *that satisfies*: type of change = perfective

4.4. *L–Patterns for Data Measures*

A data measure takes the value of a specific property of a data object. The L–Pattern can be as simple as:

DataMeasure ::= *the value of* [*property*] <*data object property name*> *of* [*data object*] <*data object name*>

For example, assuming the RFC data object has a property indicating the affected departments:

the value of [property] affected departments *of* [*data object*] RFC

4.5. *L–Patterns for Derived Measures*

A derived measure is a function defined over other measures. In this case, the L–Pattern includes the expression of the function and a mapping from function variables to the L–Patterns of other measures:

$$\begin{array}{lll} \textit{DerivedMeasure} & ::= & \textit{the function } <\textit{expression over } x_1 \ldots x_n>, \textit{ where } \{ \\ & & <x_i> \textit{ is } <\text{MeasureForDer}_i> \}_{i=1..n} \end{array}$$

$$\begin{array}{lll} \textit{MeasureForDer} & ::= & \textit{TimeMeasure} \\ & | & \textit{CountMeasure} \\ & | & \textit{ConditionMeasure} \\ & | & \textit{DataMeasure} \\ & | & \textit{AggregatedMeasure} \end{array}$$

For example, a derived measure for the ratio of RFCs cancelled out of registered ones could be defined as:

the function $\frac{c}{r} * 100$, *where* c *is* the sum of the number of times data object RFC becomes cancelled and r *is* the sum of the number of times data object RFC becomes registered

Table 5 shows a second example of a PPI defined using the previous derived multi-instance measure.

Table 5. Specification of a PPI defined over a derived multi-instance measure

PPI–001a	**Percentage of RFCs cancelled out of registered during holidays**
Process	Request for change (RFC)
Goals	• BG–002: Improve customer satisfaction
Measure	The PPI value is calculated as *the function* $\frac{c}{r} * 100$, *where* c *is the number of* cancelled RFCs and r *is the number of* registered RFCs
Target	The PPI value must be greater than or equal to *90 %*
Scope	The process instances considered for this PPI are those in *Holidays period (S-1)*
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	*First values of this PPI are dated in 2007*

4.6. *L–Patterns for Aggregated Measures*

In a similar way to derived measures, aggregated measures are defined over one of the previous measures by applying one aggregation function, i.e. sum, maximun, minimum, average, etc. If the measure aggregated is a condition measure, whose value can be 0 or 1 (boolean value), only the aggregation function *sum* can be applied (the rest of possibilities makes no sense in this case). Furthermore, it is possible to group aggregated measures

according to certain property of a data object. In such case, the PPI value is a map (several pairs (key, value)). The corresponding L–Pattern is the following:

AggregatedMeasure ::= *the* { *sum* | *maximum* | *minimum* | *average* | ... } *of* <MeasureForAgg> [*grouped by [property]* <*data object property name*> *of* [*data object*] <*data object name*>]

MeasureForAgg ::= *TimeMeasure*
| *CountMeasure*
| *ConditionMeasure*
| *DataMeasure*
| *DerivedMeasure*

Two examples of this aggregated measure L–Pattern appear in previous subsection, where they are used in the function of the derived measure.

Another example of this L–Pattern, where values are grouped, is the following one, corresponding to PPI7 from Table 1:

the sum of the number of times data object RFC *becomes* registered *and is grouped by property* type of change *of data object* RFC

A possible value of the previous PPI could be: *3 corrective RFCs, 2 adaptive RFCs, 5 perfective RFCs.*

5. Catalogue of Target L–Patterns

As stated in Section 3, every PPI has a target that defines the aimed-for value, or more generally, the aimed-for range of values of the PPI. Two kinds of targets can be defined: *simple* and *composed*. In the following subsections the different L–Patterns proposed for them are described. Furthermore, both of them include the definition of the *unit of measure* that is used in the target.

5.1. *Simple Target Value*

A simple target value is used to specify the lower bound and/or upper bound that make up the range within which the PPI value should be. If only an upper bound is defined, it acts as a maximum, and the corresponding L–Pattern is:

must be less than [*or equal to*] <*upper bound*> <*unit of measure*>

For example, the target value of PPI5, defined in Table 3 is specified as:

must be less than or equal to 1 working day

If only a lower bound is defined, it acts as a minimum, and the corresponding L–Pattern is:

must be greater than [*or equal to*] <*lower bound*> <*unit of measure*>

For example:

must be greater than 5 %

Finally, if both bounds are set, they define a range within which the PPI value must be. The L–Pattern in this case is

must be between <lower bound> and <upper bound> <unit of measure> [*inclusive*]

For example:

must be between 15 and 30 RFCs

5.2. *Composed Target Value*

The composed target allows one to define several target values or ranges, for those cases where the value of the PPI is composed by more than one value, *i.e.* when an aggregated measure *is grouped by* a property of a data object (*e.g.* PPI7 and PPI8 from our scenario, see Table 1). The L–Pattern corresponding to this kind of target value is the composition of the previous ones; it must define one target value per each value of the property of the data object used to group the PPI value, as follows:

for <property> of <data object name>
when $<value_1>$ $<simple\text{–}target_1>$,
...
when $<value_n>$ $<simple\text{–}target_n>$ [,
otherwise <default-simple–target>]

For example, the target value for PPI7:

for type of change *of* RFC
when corrective *must be less than or equal to* 20 RFCs,
when evolutive *must be less than or equal to* 30 RFCs,
when perfective *must be less than or equal to* 20 RFCs

6. Scope Template and Catalogue of Scope L–Patterns

The scope can be defined as a filter that selects the process instances that are considered for computing a PPI according to different types of conditions. If it is defined in the context of a single-instance PPI, it establishes the process instances for which such single-instance PPI will be computed. For example if we want to measure the time of RFC analysis, but only for those instances within the holidays period. Instead, if it is defined in the context of a multi-instance PPI, it delimits the set of process instances whose data will be used. For example, to measure the average time of RFC analysis during holidays, we would aggregate the corresponding value for all the instances within the holidays period. Table 6 depicts the template for the definition of scopes. The meaning of the template fields is the following:

- **Identifier and descriptive name**: the same as in PPI template, except that scope identifiers start with S.
- **Conditions**: this field allows defining a given number of instances, as well as state and/or temporal conditions over process instances to be considered when calculating

Table 6. Template for scope specification

S–<*id*>	***<S descriptive name>***
Conditions	This scope includes ◦ [not] <*NumberOfInstancesCondition*> [and | or] ◦ [not] <*ProcessInstanceStateCondition*> [and | or] ◦ [not] <*TemporalCondition*>
Periodicity	The set of process instances is re–calculated ◦ daily { every <*d*> days | every day } ◦ weekly [on <*days of week*>] ◦ monthly [on { <*day of month*> | the <n^{th}> <*day of week*> }] ◦ yearly on { <*month*> [, <*day of month*>] | the <n^{th}> <*day of week*> of <*month*> }
Comments	<*additional comments about the S*>

the PPI value. These conditions can be combined using *and*, *or* and *not*. The L–Patterns used depending on the different conditions are presented in Subsections 6.1, 6.2 and 6.3.

- **Periodicity**: this field indicates the periodicity with which the PPI is calculated. It is defined using an L–Pattern where the user has to chose between a daily, weekly, monthly and yearly periodicity. If a weekly periodicity is selected, the day of the week may be completed, by default it is Monday. For the case of monthly periodicity, whether to take into account the day of the month (e.g. 3rd January), by default the first day of month, or the day of the week (e.g. third tuesday of the month), by default the first monday of the month, may be selected. For the yearly periodicity, the month must be indicated but it can be also added the day of month, by default the first one. The user can also indicate the frequency of such periodicity (e.g. every 2 months, for a monthly periodicity) and when to finish taking such measure (e.g. ends 31-12-2014).

The following subsections describe the L–Paterns defined for each of the three possible conditions for defining scopes.

6.1. *Number of Instances Condition*

In this case the scope is defined by establishing the number of process instances to be considered to compute the PPI value. This scope only makes sense for multi-instance PPIs . The L–Pattern used is the following one:

the last <number> *process instances*

For example:

the last 10 process instances

6.2. *Process Instance State Condition*

In this case the scope is defined by establishing the state that process instances must be (or not be) in to take them into account. This scope can be defined for both, single- and multi-instance PPIs. The L–Pattern used is the following one:

process instances [*in state*] <*state*>

where <*state*> values can be active, finished or any other interesting state for users.

6.3. *Temporal Condition*

In this case the scope is defined by establishing a temporal condition over process instances. This condition can be related to the start or the end of the process instance, or to both of them, composing two conditions. The user can select instances that started "before", "before or at", "after" or "after or at" certain point in time. This point in time can be a concrete date or a relative time window (defined by completing the time from now and the unit). The generic L–Pattern used is the following one:

TemporalCondition ::= *process instances* {*started* | *finished*} <*Moment*>

Moment ::= *before* [*or at*] <*date*>
| *after* [*or at*] <*date*>
| *more than* <*t*> <$unit_T$> *ago*
| *in the last* [<*t*>] <$unit_T$>

For example, to define a scope for holidays period (including Christmas and Summer) during course 2012–2013:

process instances started after or at 23-12-2012 and finished before or at 04-01-2013 or started after or at 01-08-2013 and finished before or at 31-08-2013

An example of use of this scope template is presented in Table 7. It represents the holidays period previously described, but with the year as a parameter.

Table 7. Example of a scope definition

S-1	**Holidays period (<*year*>)**
Conditions	This scope includes process instances in state completed and started after or at 23-12-<*year*> and finished before or at 04-01-<*year+1*> or started after or at 01-08-<*year+1*> and finished before or at 31-08-<*year+1*>
Periodicity	yearly on september

7. Enabling the Automated Management of PPIs

Since both templates and L–patterns are based on the concepts identified in the PPINOT metamodel, their mapping to the metamodel is straightforward. To proof its usefulness, we have developed a software tool called PPINOT Templates Editor that allows users to model PPIs according to the templates and L–Patterns. Then, we have used the mapping to integrate it into the PPINOT Tool Suite, which implements a variety of techniques and tools developed around the metamodel to automate the management of PPIs. In particular, the values of PPIs defined following the PPINOT metamodel can be automatically computed during BP execution in a Business Process Management System and a variety of design-time analysis operations can be applied on PPI definitions to assist process analysts during their definition and instrumentation (del Río-Ortega *et al.* 2012).

7.1. *Mapping to the PPINOT Metamodel*

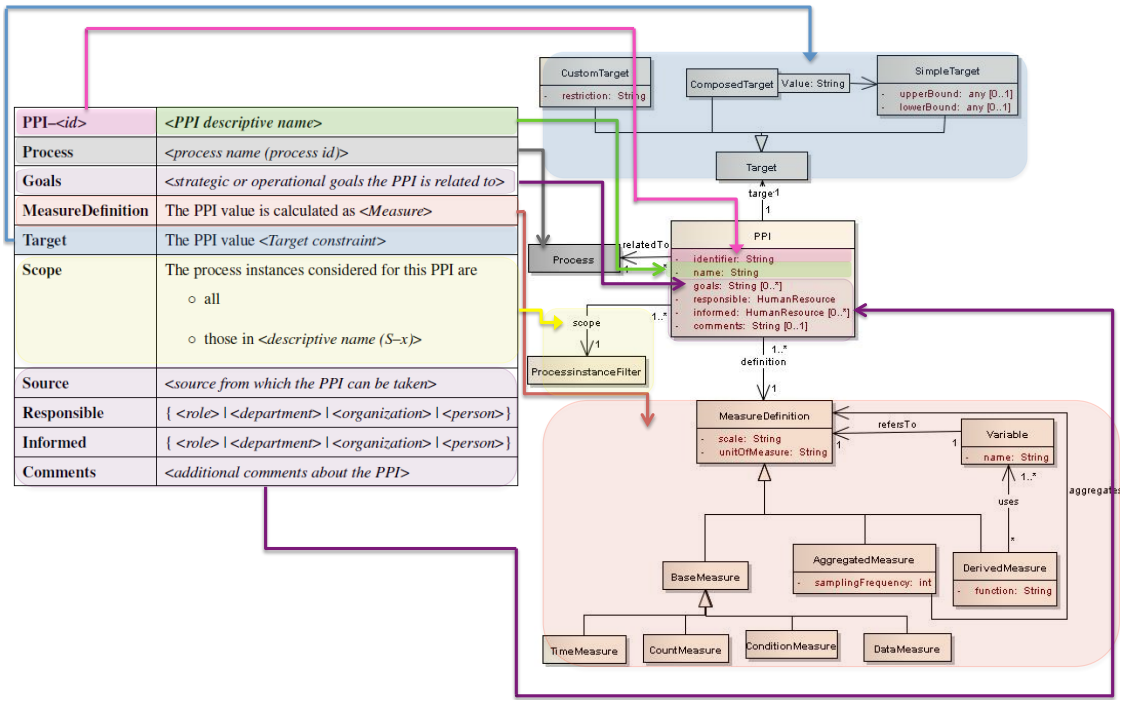

Figure 4. Mapping of PPI template to PPINOT metamodel

Figures 4 and 5 depict the mapping from the PPI and Scope templates to PPINOT metamodel, respectively. Regarding the mapping from the PPI-template to PPINOT metamodel, Figure 4 shows that there exists a 1:1 correspondence between the fields *identifier*, *name*, *goals*, *responsible*, *informed* and *comments* and the corresponding attributes with the same names in the class *PPI*. Furthermore, the *target* and the set of L-patterns defined for it can be directly mapped to the *target* class and its subclasses *SingleTarget*, *ComposedTarget* and *CustomTarget* in PPINOT metamodel. Note that the attributes defined in these subclasses (*lowerBound*, *upperBound*, *value*, and *restriction*) correspond to the information contained in the corresponding L–patterns defined in Section 5. Regarding the mapping of the *Measure* field, and the set of L-patterns presented

in Section 4, there is also a direct correspondence to the *MeasureDefinition* class and its subclasses and associations. We do not delve into this detail for the sake of simplicity and readability[1].

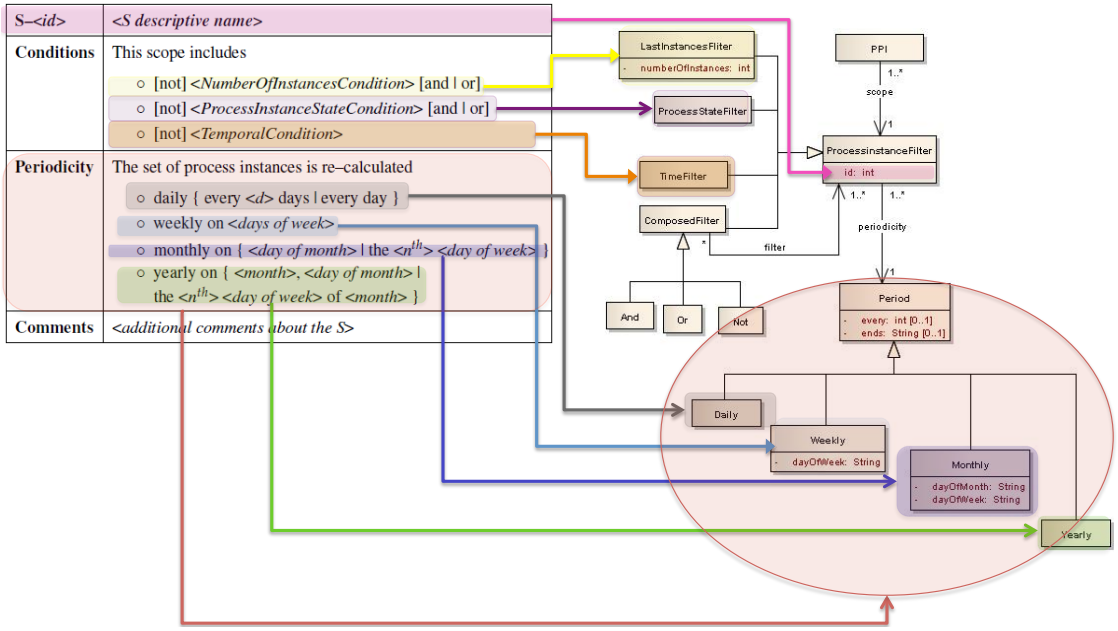

Figure 5. Mapping of scope template to PPINOT metamodel

Regarding the definition of the scope (Figure 5), on the one hand, the conditions in the scope template are translated into the corresponding classes in the metamodel; concretely, the first condition corresponds to the class *LastInstancesFilter*, and the information fulfilled by the user is mapped to the attribute *numberOfInstances*; the second condition corresponds to the class *ProcessStateFilter* and the information fulfilled by the user is mapped to the attribute *processState*; finally, the third one corresponds to the class *TimeFilter*. Regarding the information fulfilled by the user, in this case, there are several aspects to be fulfilled, that are mapped to a set of subclasses and associations of *TimeFilter*. Again, for the sake of simplicity and readability, we do not detail all these correspondences. It is worth noting that every condition starts with an optional *not* and ends (except the last one) with an optional *and* / *or*. This options are mapped to the class *ComposedFilter* and allow to form any combination of the different filters (options) provided by using *and*, *or* and *not*.

On the other hand, with respect to the periodicity, it is mapped to the class *Period*. Concretely, each option is mapped to one of its subclasses *Daily*, *Weekly*, *Monthly* and *Yearly*, respectively, and the information fulfilled by the user is mapped to the corresponding attributes.

7.2. *Tooling support*

A PPINOT Templates Editor has been developed as part of the PPINOT Tool Suite, whose whole structure is depicted in Figure 6, in which dashed lines connect input/output

[1]For further information we refer the reader to (del Río-Ortega *et al.* 2012).

data with the corresponding components while solid lines represent the communication and its direction between the different components. The PPINOT Tool Suite includes: (1) two editors, namely, the aforementioned template-based editor (PPINOT Templates Editor), and a graphical editor (PPINOT Graphical Editor), implemented as an extension of the Oryx platform (Decker *et al.* 2008) to allow the depiction of PPIs together with their corresponding BP; (2) a design-time analyser of PPIs (PPINOT Analyser), that uses the DL formalisation of PPINOT metamodel to implement several analysis operations regarding the relationship between BP elements and PPIs and between PPIs themselves, allowing questions like *what are the business process elements involved in the definition of PPI1?* or *does PPI1 depends on PPI4?* to be answered (del Río-Ortega *et al.* 2012); (3) an instrumenter (PPINOT Instrumenter) to configure a Complex Event Processor (CEP) to compute the values of the defined PPIs from the events generated by Activiti (an open source BPMS[1]) during business process execution and store them into a database (PPI Database), and (4) a reporter (PPINOT Reporter) to present the user these values[2]. Furthermore, PPINOT Tool Suite is BPMN 2.0 compliant, since PPIs can be defined over BP diagrams (BPDs) previously defined using this specification.

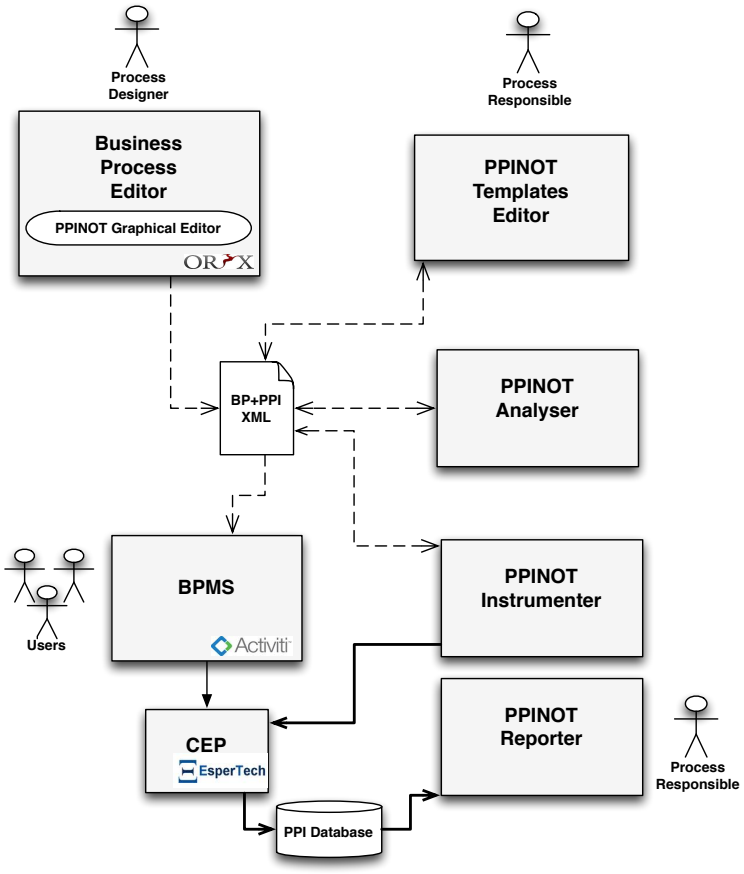

Figure 6. PPINOT Tool Suite general structure

[1]`http://activiti.org`.

[2]In its current version, this reporter provides a simple list of values. We plan to extend it to improve the GUI and provide an enriched report.

As for the PPINOT Templates Editor, it provides a template of PPI (cf. Figure 7) where, depending on the selection that the user performs in the different fields, the corresponding linguistic patterns are shown, and within these patterns, the user must fill the blanks. Once the PPI template is defined, it can be saved as a file that can be analysed or used by the instrumenter to compute PPI values. The PPINOT Templates Editor has been developed on top of *Concrete*[1], which is a lightweight, web-based model editor that can be configured for different DSLs (Domain Specific Languages), and is available at `http://www.isa.us.es/ppinot`.

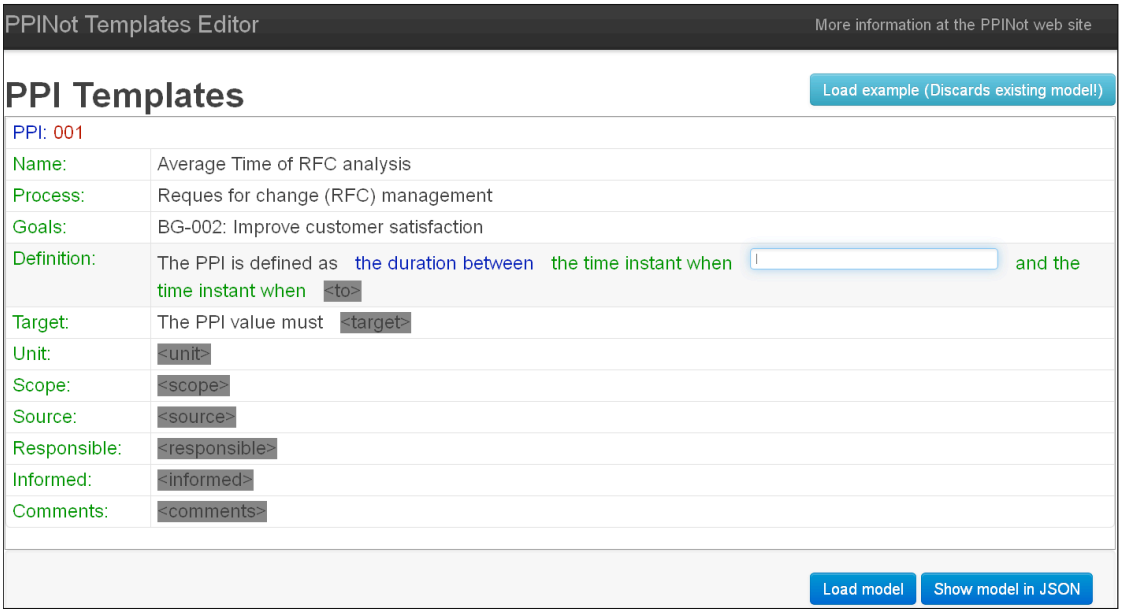

Figure 7. PPINOT template editor caption

8. Application scenarios

To validate the feasibility and usefulness of these templates and L–Patterns, we have used the PPINOT Templates Editor, framed within the PPINOT Tool Suite, to define PPIs in three real scenarios. They helped us to test and improve the templates and L–Patterns, as well as to identify abbreviated L–Patterns and specific scopes. In the following, we present the details of each of them.

8.1. *IT Department of the Andalusian Health Service*

As described in del Río-Ortega *et al.* (2012), one of the motivating scenarios for the PPINOT metamodel is the one presented in Section 2, belonging to the IT Department of the Andalusian Health Service. This department engaged some years ago in an initiative to adopt the set of good practices proposed by ITIL[1]. This entailed, among others, defining business process models as well as performance indicators for them.

[1] `http://concrete-editor.org`.
[1] Information Technology Infrastructure Library

After accomplishing this goal, they had two kinds of documents: the business process models (modelled in BPMN), produced by the roles in charge of the modelling and execution of business processes; and the documents containing the definition of their associated PPIs in natural language, produced by the roles in charge of the definition of goals and its associated indicators. This situation led to several problems. First, the ambiguity, inherent in natural language, and incompleteness in PPI definitions, in the sense of missing information required to instrument business processes for the PPI values computation. Second, the lack of traceability between the two kinds of documents, that makes it really complicated to maintain the coherence across them, since changes in one document had to be reflected in the other by hand and vice-versa. As a result of the coexistence of these two worlds unable to communicate to each other, inconsistencies between them appeared and it was necessary the human intervention to solve them, which was quite tedious and error-prone.

After the application of the PPINOT metamodel and the analysis operations described in del Río-Ortega *et al.* (2012), most of the problems mentioned above were solved, but a new one showed up: they lacked a representation for PPIs that were understandable for all kind of users (managers and employees) without loosing the unambiguity, completeness, traceability and automated support offered by the PPINOT metamodel. This was the motivation for developing our templates and patterns. We used them for the definition of the nine PPIs of table 1. In particular, within these 9 PPIs, there were 3 *time aggregated measures*, 6 *count aggregated measures*, two of them using *isGroupedBy*, 1 *state condition aggregated measure*, 2 *data property condition aggregated measures*, and 2 *derived multi-instance measures*. Regarding the patterns for target values, 7 were *simple target values* and 2 were *composed target values*. In addition, 6 scope templates were also used, from which 2 were *process instance state conditions*, 3 were *temporal conditions* and 1 was *number of instances conditions*.

The set of templates produced for these PPIs was reviewed by the quality manager of the department, who approved them and found them very useful, but considered that some of the patterns could be simplified or shortened. The feedback obtained from this scenario together with the other two helped and led us to define the abbreviated L-Patterns and specific scope presented in the Section 8. The final result is the set of templates contained in Appendix A

8.2. *Company for training health professionals*

Our second scenario takes place in a company for training health professionals[1] that works within a project for the Department of Health and Social Prosperity, a part of the Andalusian Regional Government.

The project is defined within the context of the management for the acquisition of transversal competences by health professionals. Each province in Andalusia used to carry out this task independently, with different organisational models, operational processes, *et cetera*. The main goal of the project was unifying this task and optimising it, leveraging the existing resources. To achieve this goal, one of the steps was to define a common set of business processes for all of the provinces, as well as to define and evaluate the corresponding PPIs.

During the first phase, two processes were defined using BPMN, with their corresponding subprocesses, whose size ranged between 28 and 44 tasks in total. These processes were

[1]The name and concrete information about the company and its processes and PPIs are not provided due to privacy reasons.

aimed at the planning and closure of the corresponding set of courses (called modules). For these processes, 10 PPIs were defined by employees from the company. They used our approach with templates and patterns for their definition without great difficulties and the result was quite satisfactory. In particular, within these 10 PPIs there were 1 *time aggregated measure*, 6 *count aggregated measures*, 4 *state condition aggregated measures*, 1 *data property condition aggregated measure*, and 2 *derived multi-instance measures*. As for the target values, all of them (10) were *simple target values*. Besides, 4 scope templates were also defined and in all of them a combination of *process instance state conditions* and *temporal conditions* was used.

By the end of our collaboration with this company, the set of abbreviated L–Patterns had not been defined yet; this is the reason why they were not applied neither verified within this scenario. Nevertheless, we used the information collected in this scenario, together with that from the others, as input for the definition of abbreviated L-Patterns and specific scopes.

8.3. *Service Level Agreement for Software Maintenance*

The third scenario where our approach was applied was related to the service level agreement (SLA) between a software development company and one of its most relevant customers, a very big company in the energy sector. The SLA specified not only a number of indicators and a set of penalties and rewards depending on them, but also the software maintenance processes that had to be followed. The SLA, which was specified completely in a textual manner some years ago, did not use either any graphical notation or any specific templates for PPIs.

Since our approach requires a BP model in order to specify PPIs, the first step was the modelling of the software maintenance processes as a set of BPMN processes. The results of this first step were three BPMN models, namely Project Development Service, Small-Size Evolutive Maintenance Service and User Support and Corrective Maintenance Service. The number of tasks identified in each project was respectively 44, 30 and 37. Based on these BPMN processes, the 25 PPIs originally included in the SLA were specified using the templates and patterns presented in this article. All of the PPIs were based on derived measures (in particular, percentages), 16 of them were related with on-time delivery of documents or software and 9 of them with quality assurance. In all PPIs, the scope was the current month, mainly because penalties and rewards were applied in a monthly billing basis; this means all of them used a scope template with a temporal condition. All the translation work from the original form of the SLA into BPMN and PPINOT models was performed by one of the employees of the software development company with more than 15 years of experience in software maintenance under the conditions specified in the SLA. One of the authors assisted the employee on issues related with PPINOT notation and templates during the scenario development.

With respect to the on-time delivery measures, a limitation not only of templates and patterns, but also of the PPINOT metamodel itself, was found. It was not possible to measure the time instant when an activity had started or finished, hindering the possibility to measure if a specific task had been started or finished on time with respect to a due date contained in a document like a project plan or similar. As a first solution, a new type of time measure was defined that measures the date and time in which an event takes place instead of the duration between two events. However, this led to long and unreadable definitions that involved a *derived instance measure*, a *data measure* and one of these new *time measures*. This is the reason why a new abbreviated L-Pattern for this specific case was defined.

Table 8. Summary of PPIs defined within the application scenario (measures, scopes and targets)

	Scn 8.1	**Scn 8.2**	**Scn 8.3**	**Sum**
Measure L–Patterns				
TimeMeasure	3	1	28	32
CountMeasure	6	6	0	12
StateCondMeasure	1	4	23	28
DataPropCondM	2	1	7	10
DataMeasure	0	0	32	32
AggMeasure	12	12	62	86
DerivedMeasure	2	2	61	65
Target L–Patterns				
SimpleTarget	7	10	22	39
ComposedTarget	2	0	3	5
Scope L–Patterns				
NumInstCondition	1	0	0	1
ProcInstCondition	2	4	0	6
TempCondition	3	4	25	32

Scn 8.1 - IT Department of the Andalusian Health Service
Scn 8.2 - Company for training health professionals
Scn 8.3 - Service Level Agreement for Software Maintenance

Another interesting feedback was the use of composed target values. In this case, there were 3 quality related PPIs that had different values depending on the priority of the software under corrective maintenance. For example, for the PPI Percentage of reopen corrective maintenance issues the target value was specified as follows:

For priority of Corrective Maintenance Issue
when 1 must be less than or equal to 2%
when 2 must be less than or equal to 2%
when 3 must be less than or equal to 4%

Table 8 summarises the occurrences for the different types of L-Patterns used in the three application scenarios for measures, targets and scope conditions. Note that all of the patterns are used at least once. However, there are patterns such as *Derived Measure* that are much more frequent than others such as *NumInstCondition*.

8.4. *Discussion*

Several conclusions and lessons learned can be drawn from the three application scenarios. First, the templates and L–Patterns were able to specify all the information that was necessary to define the PPIs used in the application scenarios except for that minor adjustment that was necessary to do with the time measure pattern in the last scenario. Therefore, we can conclude that the templates and L–Patterns were expressive enough to define the PPIs used in the scenarios.

As for the understandability, our users (the employees of the different organisations), technical and non-technical ones, were able to read and/or validate every PPI template-based definition. Furthermore, in two out of the three scenarios (the training and the

software development companies), the employees of the organisation were able to specify the PPIs using the templates and L–Patterns by themselves, with minor assistance from one of the authors in one case. These results are encouraging and having defined our templates using structured natural language, we can state that they tend to be more understandable than other approaches that use implementation level or more formal definitions. Nevertheless an experiment is planned to be conducted in order to prove understandability.

Finally, also related with understandability is the feedback obtained from the users regarding measure L–Patterns and scopes. On the one hand, due to the precision required for PPI definitions, the *measure L–Patterns* become quite verbose and complex to read, specially when they involve derived measures. On the other hand, the scope of most PPIs is limited to a few periods, chiefly weekly, monthly and yearly. Therefore, it was tedious to define new scope templates for such common periods of time. This feedback led us to complement the L–Patterns with abbreviated L–Patterns and specific scopes as detailed in the next section.

8.5. *Reacting to Feedback from Scenarios: Abbreviated L–Patterns and Specific Scopes*

The application of our approach to the three real scenarios above showed us that apart from the set of "*generic L–Patterns*" defined for the different types of measure or scope, it is also possible, and even desirable, to define a set of *abbreviated L–Patterns*, *i.e.*, simplified versions of L–Patterns, for a subset of them. The reason for doing so is to avoid the aforementioned issues and improve the understandability and readability by all type of stakeholders. In the following, we present a set of *abbreviated L–Patterns* and *specific scopes* identified from the experience gained within our application scenarios.

8.5.1. Abbreviated Measure L–Patterns

This set of abbreviated L–Patterns simplify the generic ones, attending to the common use of natural language, so that they appear more concise while precision is maintained. Table 10 lists them, however, this list can be enlarged and customised according to the specific needs of a concrete organisation and depending on the language.

Table 10 shows examples for some of the abbreviated L–Pattern of Table 9, as well as their corresponding generic ones. In addition, more examples can be found in Appendix A, where all the PPI templates presented are defined using abbreviated L–Patterns.

8.5.2. Specific Scopes

As in the previous case, to ease the use of scopes and as a result of the feedback from the scenarios, we propose to provide a set of predefined templates with the most common ones. In tables 11 and 12 we present two examples for monthly and Y/Y (year on year) periods. In the same way, it is possible to define scopes for yearly, weekly, quarterly or biannual periods. Again, this list can be enlarged and customised according to the specific needs of a concrete organisation and the language.

9. Related Work

In the context of process performance management (PPM), a number of languages and architectures for describing and monitoring PPIs have been proposed. Soffer and Wand (2005) propose a set of formally defined concepts to enable the incorporation of metrics

Table 9. Some abbreviated measure L–Patterns identified

Abbreviated L–Pattern	**Generic L–Pattern**
the duration of <activity> *activity*	*the duration between the time instants when activity <activity> becomes* active *and when activity <activity> becomes* completed
the duration of <process> *process*	*the duration between the time instants when process <process> becomes* active *and when process <process> becomes* completed
the number of <BP element state> <BP element name in plural>	*The sum of the number of times <BP element name> becomes <BP element state>*
The percentage of <measure1> out of <measure2>	*The function (a/b)*100 where a is <measure1> and b is <measure2>*
<BP element type> <BP element name> has {started / finished} on time according to the <data object property> of the <data object name>	*The function (a {<= / >=} b) where a is the time instant when <BP element type> <BP element name> becomes {active / completed} and b is the value of <data object property name> of <data object name>*

Table 10. Some examples of abbreviated measure L–Patterns and their corresponding generic ones

Abbreviated L–Pattern Example	**Generic L–Pattern Example**
the duration of Analyse in Committee *activity*	*the duration between the time instants when activity* Analyse in Committee *becomes* active *and when activity* Analyse in Committee *becomes* completed
the duration of RFC management *process*	*the duration between the time instants when process* RFC management *becomes* active *and when process <process>* RFC management *becomes* completed
the number of cancelled RFCs	*the sum of the number of times* RFC *becomes* cancelled
activity Software Installation *has finished on time according to the* softwareInstallationDate *of the* Project Plan	*the function (a<=b) where a is the time instant when activity* Software Installation *becomes* completed *and b is the value of* softwareInstallationDate *of the* Project Plan

and indicators (referred to as criterion functions for soft-goals) into process modeling and design methods. Popova and Sharpanskykh (2010) present a framework for modeling per-

Table 11. Specific Scope definition (monthly)

S-2	**Monthly period (*month*)**
Conditions	This scope includes process instances started after or at 1-<*month*> and finished before 1-<*month+1*>
Periodicity	monthly on the first monday

Table 12. Specific Scope definition (year on year)

S-3	**Year on Year period**
Conditions	This scope includes process instances started in the last 1 year and finished in the last 1 year
Periodicity	yearly

formance indicators within a general organisation modeling framework; Pedrinaci *et al.* (2008) present a metric ontology to allow the definition and computation of metrics integrated in SENTINEL, a Semantic Business Process Monitoring Tool; Castellanos *et al.* (2005) propose the use of templates provided by a graphical user interface (integrated in the iBOM platform) to define business measures related to process instances, processes, resources or the overall business operations; the approach of Momm *et al.* (2007) to develop process monitoring systems is built on the principles of the Model Driven Architecture (MDA), they present a metamodel for the specification of the PPI monitoring, as well as an automated generation of the required instrumentation and monitoring infrastructure; Wetzstein *et al.* (2008) introduce a framework for BAM as part of the semantic business process management, they describe a KPI ontology using WSML (the Web Service Modeling Language) (W3C 2005) to specify KPIs over semantic business processes.

Nevertheless, all the aforementioned researchers have ignored or undervalued issues of understandability for non-technical users focusing always on semantics and implementation and leaving disregarded this high-level representation. Still, some authors have already identified this problem and make some proposals.

Korherr and List (2007) extend the BPMN and EPC metamodels to define business process goals and performance measures. However, the expressiveness of these metamodels is quite limited, since they only allow the definition of cost, quality and cycle time measures, from which only cycle time measure are explicitly connected to the business process elements. Furthermore they ignore all the information required to define such measures and to compute them. Finally, the supposed high level definitions support by means of a graphical notation is quite limited, only cycle time measures can be modelled.

Costello and Molloy (2009) propose a model to include the definition of PPIs into process models. They define events associated to what they call process (business activity). These events are mainly intended to the computation of what they call cycle time PPIs. They also present a mapping of this event-based model to an ontology developed using OWL (the Web Ontology Language) (Motik *et al.* 2009). This ontology serves as a basis for defining rules for the computation of PPI values. The main weakness of this proposal is that the PPI definition provided is quite restrictive in the sense of expressiveness, since

it is focused only on time measures.

González *et al.* (2009) claim to present "a language for high-level monitoring, measurement data collection and control of business processes", called MMC. This is a declarative language where each specification contains three main blocks: data, event and rule blocks. In order to provide the information required for these three blocks, EBNF specifications need to be defined. In this way, the purpose of providing a high level of abstraction to the monitoring concerns is somehow ignored, since the specifications to be defined require certain level of technical knowledge.

Barone *et al.* (2010) present the Business Intelligence Model (BIM), whose main goal is to allow business users to conceptualise business operations and strategies, and performance indicators, so that they can be connected to enterprise data through automated tools. They propose to define a global view of a company's workflows and define indicators on its activities and resources. Nevertheless, they do neither provide any mechanism to facilitate these definitions of PPIs to non-technical users, nor describe any formal foundation for them. To overcome this problems, this proposal could be complemented with our template–based approach presented in this paper.

To the best of our knowledge, no approach exists similar to the one presented in this paper for the definition of PPIs in a user-friendly manner, taking advantage of a formal foundation and providing the benefits described in Section 1.

10. Conclusions and Future Work

A mechanism to define PPIs should have four properties (Franceschini *et al.* 2007, del Río-Ortega *et al.* 2010, del Río-Ortega *et al.* 2012), namely (1) easy to understand and learn by non-technical and untrained personnel while maintaining its preciseness; (2) standard and as reusable as possible, in order to facilitate the task and reduce the time to define PPIs; (3)traceable to the BP to maintain the coherence between BPs and PPIs; and (4) amenable to be automatically processed.

In this paper we have introduced a notation based on templates and L–Patterns that exhibit these properties. Regarding the first one, the templates and L–Patterns were able to specify all the information that was necessary to define the PPIs used in the application scenarios. Furthermore, this information could be mapped to the PPINOT metamodel in all cases. This means that no ambiguous or uncomplete definitions of PPIs were made using them. In addition, our templates also helps the user to define PPIs that meet the five characteristics of the SMART criteria. In particular, the L–Patterns used in the measure definitions contribute towards defining PPIs that are specific and measurable, whereas the L–Patterns used in the scope contribute towards its time-boundedness. Regarding the achievable and the relevant characteristics, templates and L–Patterns cannot assure their fulfillment since it is domain-dependent. Nevertheless, the target field and the goal field of the template should serve as a reminder to the user to consider each of these characteristics, respectively.

As for the understandability, the employees of the different organisations that participate in the application scenarios, technical and non-technical ones, were able to read and/or validate every PPI template-based definition. Furthermore, in two out of the three scenarios, the employees were able to specify the PPIs by themselves. Therefore, although more experiments need to be done in order to prove understandability, at least we can claim that the understandability of the notation has not been an issue by any of the stakeholders that participated in the application scenarios.

Concerning the traceability, the templates and L–Patterns also provide the traceability required between PPIs and the BP elements thanks to the explicit references to the BP elements as part of the L–Patterns defined for measure definitions. This traceability enables identifying relationships between PPIs and BP elements as done by the design-time analysis operations implemented by the PPINOT Analyser.

Finally, the mapping to the PPINOT metamodel and the implementation of PPINOT Tool Suite clearly shows the possibility of automating the management of template-based PPI definitions, enabling their values computation and their automated analysis.

As far as we know, this is the first work that explores the use of templates and linguistic patterns for the definition of PPIs, and thus more work can be done in four directions: to extend abbreviated L–Patterns and predefined scopes when more feedback from real scenarios is available; to discover more patterns, specially for the definition of PPIs related to the human resources that participate in the business process (Cabanillas *et al.* 2012, see); to conduct an experiment in order to prove the understandability and ease of use of the presented templates and L–Patterns, and to extend the L–Patterns used to define the target of the PPI.

Regarding the last one, an L–Pattern called *Custom Targets* could be defined to offer the possibility of defining constraints different than those included in simple and composed targets. This L–Pattern could be something like this:

must fulfill the following constraint <custom target constraint> <unit of measure>

In this way, the user could define customised targets, for instance by defining a utility function or using a metamodel of preferences like the one presented by García *et al.* (2010). Furthermore, another extension we plan to do for simple target values is to consider expressions on sets and logical expressions. This requires not only extending the L–Pattern, but also the PPINOT metamodel, since its current version does not contemplate this possibility.

Acknowledgments

We would like to thank to the Quality Office of the Information Technology Department of the Andalusian Health Service for kindly providing us their internal business process models and its PPIs.

This work has been partially supported by the European Commission (FEDER), Spanish Government under the CICYT projects SETI (TIN2009-07366) and TAPAS (TIN2012-32273); and projects THEOS (TIC-5906) and ISABEL (P07-TIC-2533) funded by the Andalusian local Government.

References

Alexander Grosskopf, G.D. and Weske, M., Mar 2009. *The Process. Busisness Process Modeling using BPMN*. Megan-kiffer Press.

Barone, D., *et al.*, 2010. Enterprise Modeling for Business Intelligence. *In*: *Proc. of the 3rd IFIP WG 8.1 Working Conference on the Practice of Enterprise Modeling (PoEM)*, 31–45.

Cabanillas, C., Resinas, M., and Ruiz-Corttés, A., 2012. Automated Resource Assignment in BPMN Models using RACI Matrices. *In*: *Proc. of the 20th International Conference on Cooperative Information Systems (CoopIS). OTM 2012, Part I*, 56–73.

Castellanos, M., *et al.*, 2005. iBOM: a platform for intelligent business operation management. *In*: *Proc. of the 21st International Conference on Data Engineering*, 1084–1095.

Chase, G., *et al.*, 2011. *Applying Real-World BPM in an SAP Environment*. SAP Press Galileo Press.

Costello, C. and Molloy, O., 2009. Building a Process Performance Model for Business Activity Monitoring. *In*: W. Wojtkowski, G. Wojtkowski, M. Lang, K. Conboy and C. Barry, eds. *Information Systems Development*. Springer US, 237–248.

Decker, G., Overdick, H., and Weske, M., 2008. Oryx - An Open Modeling Platform for the BPM Community. *In*: *Proc. of the 6th International Conference on Business Process Management (BPM)*, 382–385.

del Río-Ortega, A., 2012. On the Definition and Analysis of Process Performance Indicators. Thesis (PhD). University of Seville.

del Río-Ortega, A., *et al.*, 2012. On the Definition and Design-time Analysis of Process Performance Indicators. *Information Systems*, 38 (4), 470–490.

del Río-Ortega, A., *et al.*, 2012. Defining Process Performance Indicators by Using Templates and Patterns. *In*: *Proc. of the 10th International Conference on Business Process Management (BPM)*, 223–228.

del Río-Ortega, A., Resinas, M., and Ruiz-Cortés, A., October, 2010. Defining Process Performance Indicators: An Ontological Approach. *In*: *Proc. of the 18th International Conference on Cooperative Information Systems (CoopIS). OTM 2010, Part I*, 555–572.

Doran, G.T., 1981. There's a S.M.A.R.T. way to write management's goals and objectives. *Management Review*, 70 (11), 35–36.

Durán, A., *et al.*, 1999. A Requirements Elicitation Approach based in Templates and Patterns. *In*: *Proc. Workshop on Requirements Engineering (WER)*, 17–29.

Durán, A., *et al.*, 2002. Supporting Requirements Verification using XSLT. *In*: *Proc. IEEE Joint International Conference on Requirements Engineering (RE)*, 165–172.

Franceschini, F., Galetto, M., and Maisano, D., 2007. *Management by Measurement: Designing Key Indicators and Performance Measurement Systems*. Springer Verlag.

García, J.M., Ruiz, D., and Ruiz-Cortés, A., 2010. A Model of User Preferences for Semantic Services Discovery and Ranking. *In*: *Proc. of the 7th Extended Semantic Web Conference (ESWC), Part II*, 1–14.

González, O., Casallas, R., and Deridder, D., 2009. MMC-BPM: A Domain-Specific Language for Business Processes Analysis. *Business Information Systems*, 21, 157–168.

Korherr, B. and List, B., 2007. Extending the EPC and the BPMN with Business Process Goals and Performance Measures. *In*: *Proc. of the 9th International Conference on Enterprise Information Systems, (ICEIS), 3*, 287–294.

Kronz, A., 2006. Managing of Process Key Performance Indicators as Part of the ARIS

Methodology. *Corporate Performance Management: Aris in Practice*. Springer Berlin Heidelberg, 31–44.

Meyer, P.J., 2003. *In*: *What would you do if you knew you could not fail? Creating S.M.A.R.T. Goals*. The Meyer Resource Group.

Momm, C., Gebhart, M., and Abeck, S., 2009. A Model-Driven Approach for Monitoring Business Performance in Web Service Compositions. *In*: *Proc. of the 4th International Conference on Internet and Web Applications and Services (ICIW)*, 343–350.

Momm, C., Malec, R., and Abeck, S., 2007. Towards a Model-driven Development of Monitored Processes. *In*: *Proc. 8th Internationale Tagung Wirtschaftsinformatik (WI), 2*, 319–336.

Motik, B., Patel-Schneider, P.F., and Grau, B.C., 2009. OWL 2 Web Ontology Language Direct Semantics. [online] Available from: http://www.w3.org/TR/2009/REC-owl2-direct-semantics-20091027/ [Accessed 12 August 2012].

Object Management Group (OMG), 2011. Business Process Model and Notation (BPMN) Version 2.0. [online] Available from: http://www.omg.org/spec/BPMN/2.0/PDF [Accessed 08 August 2012].

Pedrinaci, C., *et al.*, 2008. SENTINEL: a semantic business process monitoring tool. *In*: *international Workshop on Ontology-Supported Business Intelligence (OBI)*, 26–30.

Popova, V. and Sharpanskykh, A., 2009. Formal analysis of executions of organizational scenarios based on process-oriented specifications. *Applied Intelligence*, 34, 226–244.

Popova, V. and Sharpanskykh, A., 2010. Modeling organizational performance indicators. *Information Systems*, 35 (4), 505–527.

Popova, V. and Sharpanskykh, A., 2011. Formal modelling of organisational goals based on performance indicators. *Data and Knowledge Engineering*, 70 (4), 335–364.

Ruiz-Cortés, A., *et al.*, 2001. Automated support for quality requirements in web-services-based systems. *In*: *Proc. of the 8th IEEE Workshop on Future Trends of Distributed Computing Systems (FTDCS)*, 184–200.

Ruiz-Cortés, A., *et al.*, 2005. Improving the Automatic Procurement of Web Services Using Constraint Programming. *International Journal of Cooperative Information Systems*, 14 (4), 439–468.

Shahin, A. and Mahbod, M.A., 2007. Prioritization of key performance indicators: An integration of analytical hierarchy process and goal setting. *International Journal of Productivity and Performance Management*, 56, 226 – 240.

Soffer, P. and Wand, Y., 2005. On the Notion of Soft-Goals in Business Process Modeling. *Business Process Management Journal*, 11, 663–679.

W3C, 2005. Web Service Modeling Language (WSML). [online] Available from: http://www.w3c.org/Submission/WSML/ [Accessed 6 November 2012].

Weske, M., 2007. *Business Process Management: Concepts, Languages, Architectures*. Springer.

Wetzstein, B., Ma, Z., and Leymann, F., 2008. Towards Measuring Key Performance Indicators of Semantic Business Processes. *Business Information Systems*, 7, 227–238.

Appendix A. PPI Definitions for the RFC Management Process using Templates and L–patterns

Table A1. PPI template for PPI1 from Table 1

PPI–001	**RFCs cancelled out of RFCs registered**
Process	Request for change (RFC)
Goals	• BG–002: Improve customer satisfaction
MeasureDefinition	The PPI is calculated as *the function (c/r)*100 where c is the number of cancelled RFC and r is the number of registered RFC*
Target	The PPI value must be less than or equal to *4 %*
Scope	The process instances considered for this PPI are Finished Instances (S-4)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	

Table A2. S-4 scope definition

S-4	**Finished instances**
Conditions	This scope includes process instances in state finished
Periodicity	weekly on Friday

Table A3. PPI template for PPI2 from Table 1

PPI–002	**Average time of committee decision**
Process	Request for change (RFC)
Goals	• BG–014: Reduce RFC time–to–response
MeasureDefinition	The PPI is defined as *the average of the duration of activity Analyse in Committee*
Target	The PPI value must be less than or equal to *1 working day*
Scope	The process instances considered for this PPI are Finished Instances (S-4)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	

Table A4. PPI template for PPI3 from Table 1

PPI–003	**Corrective RFCs out of RFCs approved**
Process	Request for change (RFC)
Goals	• BG–002: Improve customer satisfaction
MeasureDefinition	The PPI is defined as *the function (a/b)*100 where a is the sum of approved RFC that satisfies: type of change = corrective and b is the sum of the number of times event Report RFC approved is triggered*
Target	The PPI value must be less than or equal to *2 %*
Scope	The process instances considered for this PPI are Finished Instances (S-4)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	*values up to 3 % could beaccepted*

Table A5. PPI template for PPI4 from Table 1

PPI–004	**Perfective RFCs out of RFCs approved**
Process	Request for change (RFC)
Goals	• BG–002: Improve customer satisfaction
MeasureDefinition	The PPI is defined as *the function (a/b)*100 where a is the sum of approved RFC that satisfies: type of change = perfective and b is the number of approved RFCs*
Target	The PPI value must be less than or equal to *4 %*
Scope	The process instances considered for this PPI are Finished Instances (S-4)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	

Table A6. PPI template for PPI5 from Table 1

PPI–005	**Average time of RFC analysis**
Process	Request for change (RFC)
Goals	• BG–014: Reduce RFC time–to–response
MeasureDefinition	The PPI is defined as *the average of the duration of activity Analyse RFC*
Target	The PPI value must be less than or equal to *2 working days*
Scope	The process instances considered for this PPI are Last 100 instances (S-5)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	*Most RFCs are created after 12:00.*

Table A7. S-5 scope definition

S-5	**Last 100 instances**
Conditions	This scope includes the last 100 process instances
Periodicity	weekly on Friday

Table A8. PPI template for PPI6 from Table 1

PPI–006	**Number of RFCs in analysis**
Process	Request for change (RFC)
Goals	• BG–002: Improve customer satisfaction • BG–014: Reduce RFC time–to–response
MeasureDefinition	The PPI is defined as *the sum of activity Analyse RFC that is currently in state active*
Target	The PPI value must be less than or equal to *2 RFCs*
Scope	The process instances considered for this PPI are Active instances (S-6)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	

Table A9. S-6 scope definition

S-6	**Active instances**
Conditions	This scope includes process instances in state active
Periodicity	weekly on Friday

Table A10. PPI template for PPI7 from Table 1

PPI–007	**Number of RFCs per type of change**
Process	Request for change (RFC)
Goals	• BG–005: Improve customer proactivity • BG–010: Reduce faulty deliverables
MeasureDefinition	The PPI is defined as *the number of registered RFCs grouped by type of change of RFC*
Target	The PPI value *for type of change* *when corrective must be less than or equal to 20 RFCs* and *when evolutive must be less than or equal to 30 RFCs* and *when perfective must be less than or equal to 20 RFCs*
Scope	The process instances considered for this PPI are those in Monthly period (S-2)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	*the ideal situation is that corrective RFCs tend to 0*

Table A11. PPI template for PPI8 from Table 1

PPI–008	**Number of RFCs per project**
Process	Request for change (RFC)
Goals	• BG–008: Optimise resource allocation
MeasureDefinition	The PPI is defined as *the sum of the number of times dataObject RFC becomes registered grouped by project of RFC*
Target	The PPI value *for project* *when RR.HH must be less than or equal to 50 RFCs* and *when Diraya must be less than or equal to 60 RFCs* and *when Pharma must be less than or equal to 1 RFCs*
Scope	The process instances considered for this PPI are those in Monthly period (S-2)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	

Table A12. PPI template for PPI9 from Table 1

PPI–009	**Average lifetime of an RFC**
Process	Request for change (RFC)
Goals	• BG–014: Reduce RFC time–to–response
MeasureDefinition	The PPI is defined as *the average of the duration between time instants when event Receive RFC is triggered and when process RFC management becomes completed*
Target	The PPI value must be less than or equal to *3 working days*
Scope	The process instances considered for this PPI are those in Monthly period (S-2)
Source	Event logs
Responsible	*Planning and quality manager*
Informed	*CIO*
Comments	*Most RFCs are created after 12:00.*