Semantic Analysis of Business Process Executions

Fabio Casati, Ming-Chien Shan
Software Technology Laboratory
HP Laboratories Palo Alto
HPL-2001-328
December 17th , 2001*

E-mail: [casati, shan] @hpl.hp.com

workflows, data warehousing, business intelligence

Business Process Management Systems log a large amount of operational data about processes and about the (human and automated) resources involved in their executions. This information can be analyzed for assessing the quality of business operations, identify problems, and suggest solutions. However, current process analysis systems lack the functionalities required to provide information that can be immediately digested and used by business analysts to take decisions. In this paper we discuss the limitations of existing approaches and we present a system and a set of techniques, developed at Hewlett-Packard, that overcome this limitations, enabling the use of log data for efficient business-level analysis of business processes.

* Internal Accession Date Only Approved for External Publication

Semantic Analysis of Business Process Executions

Fabio Casati and Ming-Chien Shan

Hewlett-Packard Laboratories
1501 Page Mill Road, 1U-4
Palo Alto, CA, 94304 USA
[casati,shan]@hpl.hp.com

Abstract. Business Process Management Systems log a large amount of operational data about processes and about the (human and automated) resources involved in their executions. This information can be analyzed for assessing the quality of business operations, identify problems, and suggest solutions. However, current process analysis systems lack the functionalities required to provide information that can be immediately digested and used by business analysts to take decisions. In this paper we discuss the limitations of existing approaches and we present a system and a set of techniques, developed at Hewlett-Packard, that overcome this limitations, enabling the use of log data for efficient business-level analysis of business processes.

1. Introduction

Business Process Management Systems (BPMS) are tools that support the definition, execution, and management of business processes [4]. BPMSs have been traditionally used to enact administrative processes, such as travel expense reimbursement or employee relocations. More recently, they have been also used to automate the supply chain and to implement services offered via the web.

The benefits of using a BPMS, as opposed to hardcoding the business logic or executing the flow manually, include faster process executions, lower operating costs, automated exception handling, shorter time to market, and more flexibility. However, until recently, these advantages have not proven convincing enough for many companies to adopt a process management solutions. Indeed, IT architects often decide for approaches that involve hardcoding the flow, or adopting the simple process automation functionalities offered by EAI tools, application servers, or other e-business applications already in use in the organization.

Another (potential) benefit of BPMSs is that they provide functionalities for logging and subsequently analyzing process executions, in order to track operations, detect inefficiencies, and identify solutions. Indeed, this capability is often highlighted by BPMSs vendors (sometimes successfully [3]) as a key advantage with respect to both competing vendors and competing technologies.

However, BPMS analysis and monitoring capabilities are still primitive, and are by no means sufficient to enable the monitoring, understanding, and improvement of business operations. There are indeed a variety of problems in current process

analysis technologies and in the reporting tools that are provided with the BPMS, ranging from performance (speed) of the tools to the quality of the operational data logged by the system, often characterized by errors and inconsistencies. However, besides these "technical" limitations, a key problem is that it is very difficult to gain *business* insights from the reports, that are at a very low level of abstraction and do not support any kind of business-level analysis. Therefore, they do not support business manager in identifying and addressing problems.

This paper discusses the main limitations of current process analysis technologies and presents a system, recently developed at Hewlett-Packard, that overcomes many of these limitations. The system, called *HPPM intelligent Process Data Warehouse*, or PDW for short, is based on warehousing process execution data, and then on adding semantics to it, in order to provide a comprehensive view of the business operations, assess their quality, and quickly identify the critical issues that need to be addressed. While our purpose is to support both technical and business users (and we believe that the system improves the state of the art for both kinds of analysis), in this paper we particularly emphasize the PDW techniques that provide support for business users, since it involves more innovative techniques, and is therefore more suited to an industrial track of a research conference. The interested reader is referred to [5] for a detailed description of the functionalities and technical details of PDW.

2. Semantics, Semantics, Semantics[1]

This section discusses the main limitations of current process reporting techniques. BPMSs today enable execution data analysis by logging process execution data into a database, that is then typically queried with either built-in or third party reporting tools such as Crystal Reports, Oracle Discoverer, or Microsoft Excel.

Fig. 1 shows a typical report provided by built-in BPMS analysis tools: It provides information on the average execution time of each node in a process. Obtaining and using these kinds of report with current BPMSs presents three major limitations:

- *Performance:* process logs are typically designed to ease (i.e, do not delay) the job of the process engine, but they are not structured for data analysis. This is reflected by the lack of indexes, partitions, and by the structure of the database schema. Due to these limitations, it may literally take days to generate even simple reports if the volume of process instances is high.
- *Data quality:* even assuming that data are retrieved in an acceptable time, it is very likely that the information shown is incorrect. In fact, very often the data logged by the operational systems include inconsistencies, and even special "codes" written in activity completion timestamps (the typical one is Jan 1, 1970) to denote failures or other events. The presence of even a single "code" like this, if not properly handled, can invalid the statistics returned by the reporting tool.
- *Semantics:* even assuming that data are retrieved with acceptable performance and that they accurately reflect the process executions, they are still useless for many

[1] "Borrowed" from an expression used by Stefano Ceri at the Asilomar Workshop on the database challenges for the XXI century [1].

analysis purposes, because they lack the abstraction level required from (business) analysts in order to understand business operations, as detailed in the following.

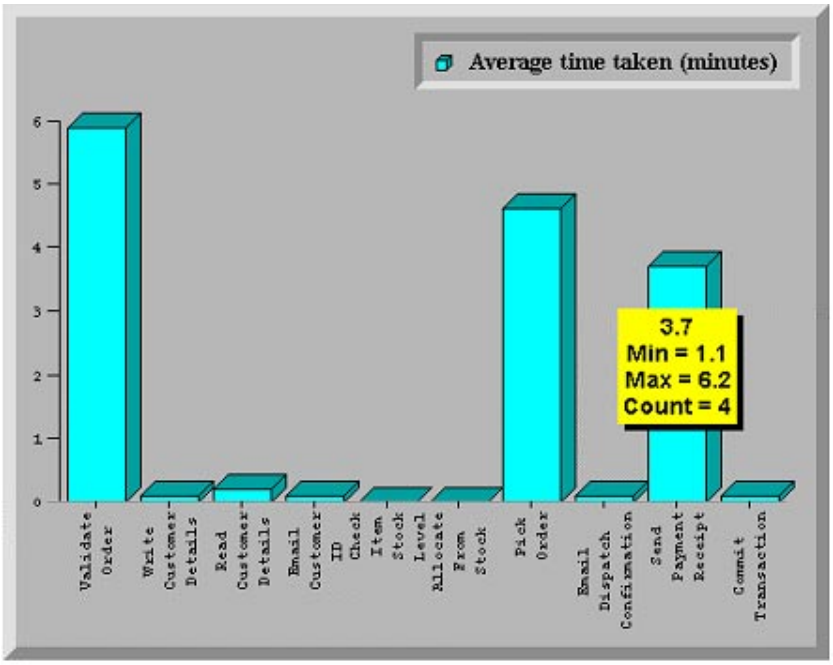

Fig. 1. A typical report than can be obtained by using BPMS reporting tools

The first two issues are "classic" log analysis problem, typically solved by using a data warehousing approach to data analysis. Indeed, PDW is also based on warehousing process execution data. While building a data warehouse for business processes does present some interesting data warehousing problems, in this paper we focus on the "semantic" aspect of the problem and of the solution we propose.

Consider the diagram of Fig. 2, representing a process executed by a supply chain hub to support negotiations of purchase order requests between buyers and sellers (this example has been taken from one of our customers). The figure shows the "business-level" view of the process. Indeed, a business analyst would benefit from statistics on this process, such as how long it takes to execute each step in average, how many orders are accepted for each week, and the like. Therefore, a report such as the one shown in Fig. 1 would indeed be helpful if it could be applied to this process.

However, when the process is implemented, additions and modifications are needed, to take care of many detailed issues that must be considered in real-life operations, such as getting information from the database, checking permissions, or managing deadlines. In the case at hand, the process that was actually implemented is shown in Fig. 3[2]. The reporting tools will provide statistics based on the actual process of Fig. 3, that include cryptic node names and no apparent relation to the

[2] The figure is small both because of space constraints but also because we were not allowed to show the details of the process.

conceptual process. Therefore, the report is virtually useless (or at least very hard to interpret) for a business user.

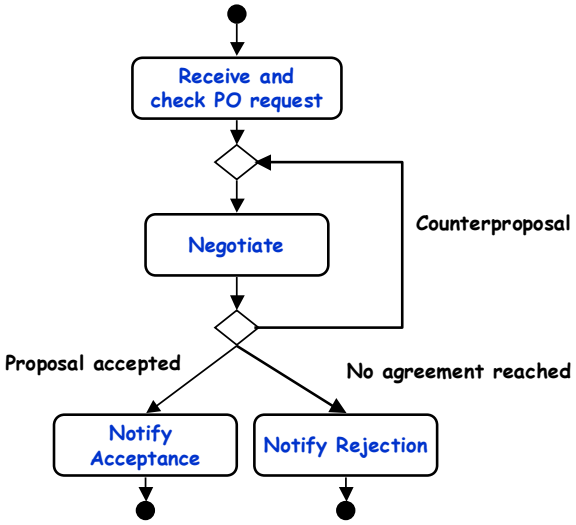

Fig. 2. Supply Chain Hub process (conceptual view)

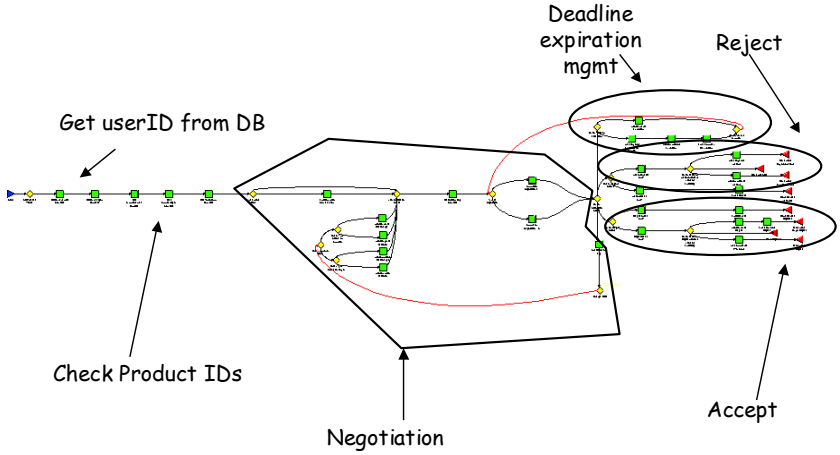

Fig. 3. Supply Chain Hub process (actual implementation)

Typically, the relation between the conceptual and the actual process is only known to the IT personnel who implemented the process, and this knowledge is often spread among different persons. In principle, the use of *subprocesses* in the implementation could help preserve the structure of the conceptual process. However, this does not happen in practice for a variety of reasons, ranging from detailed operational issues that require a deviation from the clean and simple conceptual structure, to the technical characteristics (limitations) of the process model provided by the BPMS, and to the fact that implementers are very focused on the challenging

task of making things work correctly and with the required performance, leaving little room for a "design for analysis".

Besides the problem of mapping between the conceptual and actual process, existing approaches have many other limitations. For example, analysts typically want to classify processes according to taxonomies of their choice (such as "Fast/Slow", or "Accepted/AcceptedWithNegotiation/Rejected", etc.) and examine data and trends depending on the classes to which processes belong (e.g., see the number of accepted purchase orders by week). Current BPMSs cannot offer these kinds of high-level reports, since they do not provide mechanisms to associate semantic information to process executions in order to facilitate their analysis.

3. PDW Concepts

This section describes the support that PDW provides for adding semantics to process execution data, in order to overcome the limitations of existing approaches and enable business-level analysis.

3.1 PDW Data Structure

This section briefly introduces the PDW warehouse structure, as a background for the description of the following subsections. The PDW database is structured according to a star schema design, where data are described in terms of "*facts*", i.e., happenings of interest to be analyzed, and "*dimensions*", i.e. perspectives under which the facts are analyzed. A design based on a star schema enables the analysis of facts seen from different perspectives and allows the use of many query optimization techniques. The PDW includes the following facts: *Process instance executions*, *Node instance executions*, and *Behaviors*. These facts can be analyzed based on the following dimensions:

- *Processes and process groups*, to focus on facts related to a specific process definition, or to processes within a process group.
- *Nodes*, to focus on facts related to a specific process node or set of nodes.
- *Resources*, to focus on processes started by, nodes assigned to, or node executed by a specific human or automated resource or group of resources.
- *Time*, to focus on facts occurred in a certain (fiscal or calendar) time window, or on specific days, weekdays, or hours of the day.
- *Behaviors*, to focus on instances that exhibited a user-defined behavior of interest (details on behaviors are provided below).

In addition, PDW includes a large set of views on top of facts and dimension that make it easy for users to retrieve key statistics and performance metrics, without having to write complex queries. PDW also includes pre-packaged configuration files for many commercial reporting tools, so that users can access business reports without writing any code. Details are provided in [5].

3.2 Behaviors

One of the most significant and innovative feature of PDW is *behavior analysis*. In fact, a frequent analysis need is that of identifying process instances that exhibit specific behaviors, and to understand the *causes* of such behaviors. Examples of behaviors of interest are *Supply Chain* process instances that last more than 20 days or include more than 3 negotiation cycles, *Claim Management* instances in which node "Examine expenses" was executed by a manager, or processes instances related to order for goods over 20,000$.

The PDW approach is agnostic about the behaviors that users may be interested in analyzing. Indeed, it allows users to define the behaviors to be monitored. PDW will then take care of identifying which processes instances exhibit a specific behavior and of analyzing them. As shown throughout this paper, we use the behavior concept for a variety of purposes. This approach allows us to simplify the user interaction with the system since, by getting familiar with the notion of behaviors, users can configure a variety of different analysis and monitoring functionalities.

Behaviors are defined by instantiating *behavior templates*. A template is a parametric definition of a behavior, such as "*Instances of process **P** that takes more than **N** days to complete*". In order to define a behavior of interest for a specific process definition, users simply need to instantiate the template, i.e., provide values for the parameters. No coding is needed. Multiple specific behaviors to be monitored (on the same or different processes) can be defined for each behavior type, and a process can be analyzed for multiple behaviors.

Behavior templates are (conceptually) defined by Boolean conditions over process and node execution data available in the warehouse. Templates are implemented by means of SQL statements, that detect behaviors of interest when data are loaded into the warehouse. PDW includes a large set of predefined behavior templates, to account for the most common monitoring and analysis needs. Users can add new behavior templates by downloading them from template libraries, made available on the web[3]. If users need to monitor a kind of behavior that is neither among the predefined ones nor downloadable from web template libraries, they can still specify the behavior template they need, although in this case they would need to define the corresponding condition (and consequently the SQL statement that detects whether an instance has a behavior of a given type). The occurrence of a behavior is stored as a fact in the warehouse, so that processes can be also analyzed from the behavior perspective.

By detecting behaviors of interest, analysts can perform multidimensional analysis to understand the causes of "good" and "bad" process executions. In particular, a very useful analysis consists in examining *correlations* among behaviors, i.e., in examining which other behaviors occur when a process instance has a behavior B. In this way, the effects of B on the process can be analyzed. For example, the analyst can define B as processes being "started by John Smith" and B2 as processes being "too slow". Behavior analysis can be used to first examine how many processes are "too slow" (say, 15%), and then to examine how many processes among those "started by John Smith" are "too slow" (say, 55%), thereby indicating a cause-effect relationship between John Smith and the process being slow.

---

[3] The PDW web site is currently available only on the HP internal web site.

PDW analysts can also associate a *value* (or *cost*) to behaviors, to denote the associated benefit or cost. For example, it is possible to say that the fact that when a certain node in a process execution is performed by a unit manager, then a value (cost) of –3 is assigned to the process instance. When the same node is performed by a department manager, then a value (cost) of –2 is assigned, and so on. In this way it is possible to get reports about the combined value (cost) of a set of process executions. Fig. 4 shows a chart, obtained by accessing a PDW view with Oracle Discoverer, that shows the total process value (i.e, the sum of the values or costs of the individual process instances). Data are aggregated based on the week in which the process instances started.

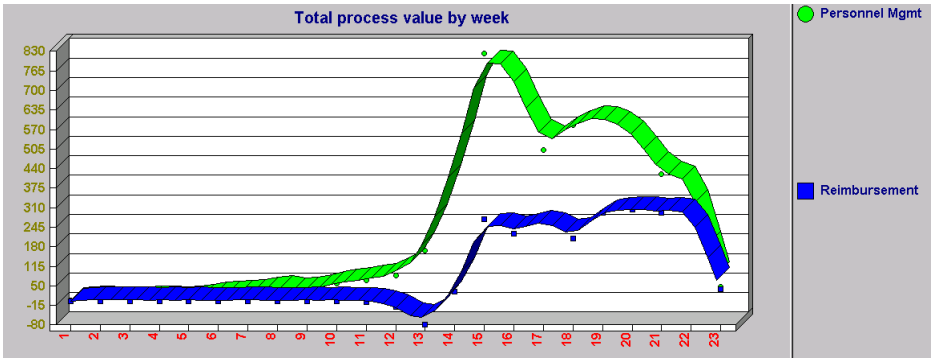

Fig. 4. A chart depicting the total process value, shown by week of the year

3.3 Taxonomies

Another concept that allows PDW users to add semantics to process execution data is that of process *taxonomies*. A taxonomy is a user-defined criterion to classify instances of a process depending on their characteristics. Many taxonomies can be defined for the same process. Each taxonomy can have several *categories*, and for each taxonomy a process instance can be in one and only one category. For example, a taxonomy *outcome* may include the categories *accepted* and *rejected*, while the taxonomy *duration* may include the categories *fast, acceptable, slow*, and *very slow*.

Taxonomies can be defined by specifying the categories that compose the taxonomy. Each category is then associated to a behavior, with the meaning that the process instance is classified in a given taxonomy if the process instance has the corresponding behavior. Taxonomies are flat, that is, there is no hierarchy among categories. Two categories, *Other* and *Error*, are automatically defined by PDW for each taxonomy. *Other* contains instances that do not fall in any other category within the taxonomy, while *Error* includes instances belonging to more than one category.

Once taxonomies have been defined, then business analysts can access reports that can immediately provide information which is easy to consume. For example, Fig. 5. shows, for each week, the distribution of process instances within the categories of the user-defined "duration" taxonomy, described above.

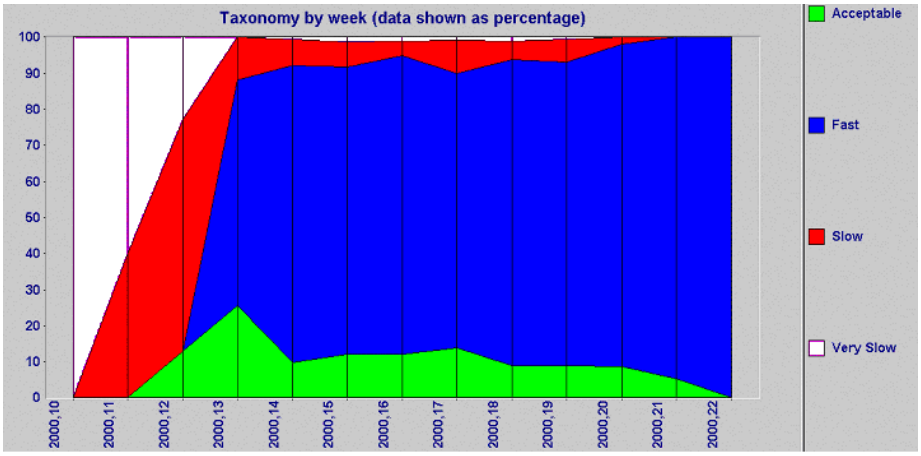

Fig. 5. Distribution of process instances within the categories of the "duration" taxonomy. For each week, the areas describe the percentage of instances started in the specified week that fall in each category of the taxonomy.

Analogously to behaviors, PDW users can also examine correlations among categories of different taxonomies. This kind of analysis is very powerful, since it allows users to easily understand the cause-effect relationships among the categories of the two taxonomies. For example, Fig. 6 shows the correlation between the categories of taxonomies *duration* and *deadline* (that describes whether the deadline for node "approve" within process "reimbursement" has expired at least once in process instance execution). Users can examine the process performance distribution, depending on whether the deadline for the "approve" node has expired or not[4]. The two bars in the Figure correspond to instances that are in the categories "expired" (left) and "not expired" (right) of the *deadline* taxonomy. Each bar is then divided according to the percentage distribution of instances within the categories of the *duration* taxonomy. The figure shows for instance that less then half of the instances in which the node deadline expired are "very slow", while no instance is "very slow" if the deadline is "not expired".

Note that, while taxonomy correlations can typically produce reports that are at a higher level of abstraction with respect to behavior correlations (and therefore easier to interpret), they do not replace it. Indeed, users often need to define behaviors that do not belong to specific taxonomies (possibly because they do not generate a partition in the process instance space), and to analyze correlation among these behaviors. Hence, both taxonomy and correlation analysis are useful and needed.

3.4 Process regions

Another PDW concept that supports semantic analysis is the *process region*. A process region is a subgraph of a process that can be treated as a unit from a business

[4] Note therefore that we are comparing the duration of the *process* with the deadline expriation of a given *node* in the process.

analysis perspective. The purpose of process regions is to bridge the gap between the conceptual process and the actual (implemented) process, which typically include many nodes to implement a single conceptual step. A region R is defined by selecting two nodes *s* and *e* in an underlying process *P*, such that:

1. *s* has only one input arc, and this arc comes from a node outside R;
2. *e* has only one output arc *a*, that connects *e* to a node that does not belong to R;
3. Every other arc in R must only connect nodes in R.

All the nodes between *s* and *e* (included) belong to the region. Once a region has been defined, PDW can provide reports that include region-level statistics. For example, with respect to the example of Figures 2 and 3, it is possible to define process regions to analyze steps "Notify Acceptance" and "Notify Rejection" and obtain reports about these regions. However, the nodes that implement steps "Check PO" and "Negotiation" are intermingled, so that a clear division into process region was not possible for those steps.

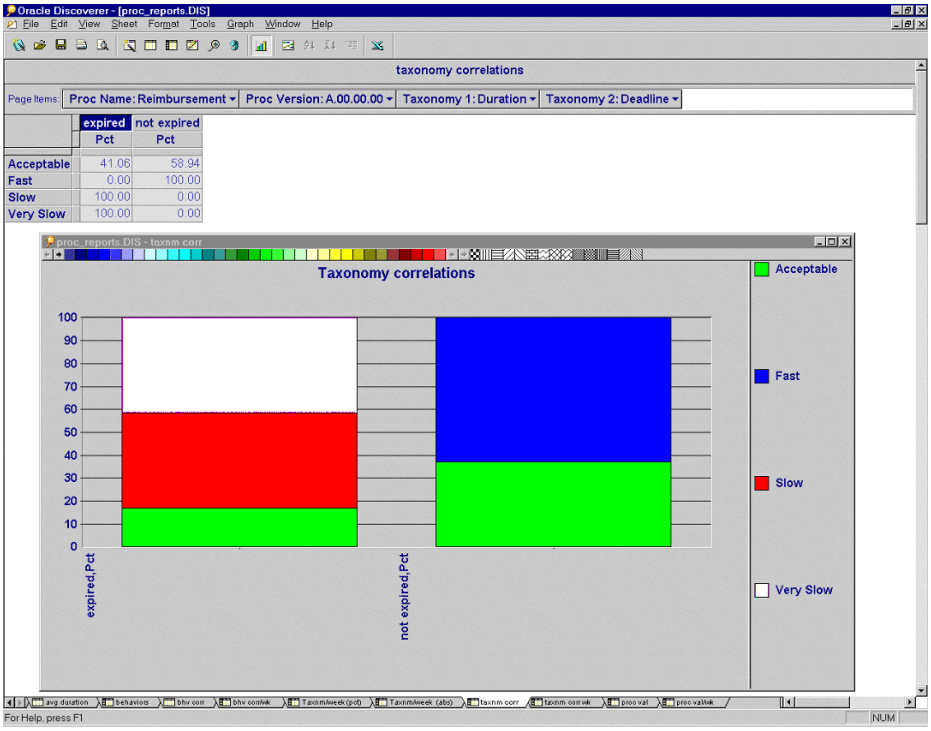

Fig. 6. Correlations among categories of different taxonomies

3.5 Alerts

Besides providing data and statistics on completed processes, PDW can also provide information on active processes, to enable process monitoring and management, and to provide high (business) level as well as detailed information

about the health not only of the operational system, but also of the overall business. In addition, PDW allows users to define *alerts*. An alert is a condition on active process data that is of particular interest to the analyst, possibly because it corresponds to an undesired situation that needs to be addressed. Like behaviors and taxonomies, alerts allow users to add semantics to information stored in the warehouse, to allow higher-level business process analysis (or monitoring, in this case).

In order to keep the PDW configuration and usage simple, alerts (like taxonomies) are defined by reusing the notion of behaviors: in fact, an alert is characterized by a name and by an associated behavior. When data on active processes are refreshed, PDW detects behaviors (alerts) of interest and stores them into tables that can be queried to get the number and kind of alert. Value information in behavior definition can be used to indicate the perceived importance of the alert.

4. Concluding remarks

This paper has discussed the limitations of existing process analysis systems, and has described the approach that we took in order to address them. In particular, we have emphasized that a major challenge consists in adding semantics to process execution data, in order to make the monitoring and analysis meaningful for business managers. The system described in this paper has been fully implemented at HP. It is intended to augment the capabilities of HP Process Manager, the BPMS provided by HP. We are currently working to add more capabilities to the PDW. In particular, we plan to add a prediction module, that derives prediction models and applies them on running processes, in order to make predictions on behaviors of interest (such as the probability of deadline expirations or of a purchase order being accepted). In particular, we plan to extend initial studies, described in [2], with the goal of designing a fully automated approach that does not require human input in building prediction models and that automatically understands if that data allows building a "reasonably" accurate prediction model (consistently with the idea of keeping the system easy to use for non-technical people).

References

[1] P. Bernstein et al. The Asilomar Report on Database Research. *Sigmod Record 27(4)*, Dec 1998.

[2] F. Casati, U. Dayal, D. Grigori, M.C. Shan. Improving Business Process Quality through Exception Understanding, Prediction, and Prevention. *Procs. of VLDB'01*, Rome, Italy. Sept. 2001

[3] Hewlett-Packard. Let'sBuyIt.com Case Study. Aug. 2000. Available from www.hp.com/go/e-process

[4] F. Leymann, D. Roller: *Production Workflow*. Prentice-Hall, 2000.

[5] F. Casati. Intelligent Process Data Warehouse for HPPM 5.0. HP Labs Technical Report. Available from www.hpl.hp.com. Nov. 2001.