Enabling Open Data Dissemination in Process Driven Information Systems

Miroslav Zarić

* University of Novi Sad/Faculty of Technical Sciences, Novi Sad, Serbia
miroslavzaric@uns.ac.rs

Abstract — Open Data movement is gaining momentum in recent years. As open data promise free access to valuable data, as well as its usage, it has become an important concept for ensuring transparency, involvement and monitoring of certain processes. As such open data is of special importance in science and in government. At the same time yet another trend is also visible - adoption of process driven information systems. In this paper an approach to systematic enabling of open data dissemination from process driven system is explored.

I. INTRODUCTION

As defined in [1] open data is data that can be freely used, re-used and redistributed by anyone - subject only, at most, to the requirement to attribute and sharealike. Open data is especially important in science, where certain information should be made available in order to promote scientific research, exchange of ideas, and maximize positive outcomes of scientific research on modern society. One other field where open data is gaining ground is open government data. Government institutions are creating vast amount of data at any given moment. Making (potentially large) portions of that data openly available can bring multiple benefits, such as transparency of government procedures, improved accountability and enhanced involvement of citizens in government. Furthermore, enabling free access to open government data can bring new business opportunities, through creation of new applications that could provide new value to the customers (citizens). In many cases, especially local, governments don't have neither human capacity nor funding to explore, evaluate innovative ways to utilize data that has been accumulated through their every-day operation.

Process driven information systems, primarily workflow management systems (WfMS - introduced in late 1980's), and later business process management systems (BPMs) have been gaining wider acceptance steadily. Though there were different languages for specifying workflow/process models in recent years systems are converging on the use of BPMN [2] as standard notation. Since these systems are increasingly found in larger enterprise suites, such as document management suites, ERPs, CRMs, there is also an ISO/IEC standard [3] governing operational view of BPM systems.

The driving force behind the idea of process driven systems is process modeling, and promise of greater efficiency achieved through workflow automation (routing, coordination between participants and task distribution). Process driven architectures are well adopted in many business suites, they also prove as a welcome tool to enhance productivity in complex multidisciplinary project environments [4], and are well suited for many administrative procedures usually implemented in government institutions [5,6].

Giving the fact that open data is becoming more interesting, and in some cases mandatory by law, or at least preferred by "good practice", and the fact that process driven systems are readily available and embedded in different enterprise systems, there should be some consistent way of specifying correlation between these two concepts.

In this paper an approach to systematically handle open data dissemination from process driven information system is presented. Although the paper discusses case of handling public procurement data, and initial deployment is performed on Activiti [7] process engine, main concepts are independent from specific usage scenario, and are mainly focused on introducing open data dissemination at process model level.

II. ACCESSING DATA IN BUSINESS PROCESS MANAGEMENT SYSTEMS

Virtually all business process management systems handles two groups of data:

- Data defined by BPM implementation environment and derived from deployed process model. This data represent the core data model needed to represent process models, process instances, activities, users, roles and other concepts related to operational process management. Although BPM implementations may display subtle differences, this data largely conforms to conceptual model given in [8]. There is almost 1:1 mapping between concepts given in [8] and entities available in [2]. Typically BPM implementations are using relational databases and object-relational mapping systems to maintain these data and track process state.

- User data - created, manipulated and maintained during process enactment. This data represent actual data processed when process instance is running. Data model is usually complex and suited for specific processing goal. This data may have it's own data storage, or can use the same persistence layer as process engine itself. This user data may be represented at process model level as Data object. BPMN specification has defined data object for process model, but support for it in implemented BPM systems is yet not standardized.
While user data (data object) is always directly exposed and available during process execution (in accordance with user roles and access permissions), it is obvious that this data is not the only source of valuable information. For process designers and management personal data recorded by process engine itself may represent valuable source of new information. During process execution, process instance state changes are recorded (during process advancement). By accessing this information process designers and managers can infer new knowledge about process behavior, possibly new process models, detect common process pitfalls, or places where process remodeling could further improve system performance. Techniques used to extract knowledge from BPM systems are commonly known as process mining [8,9]. Some data gathered by examining process execution history logs may well be suited to be exposed as open data, and may be used as valuable data set for data further analysis. Therefore not only user data objects are candidates for exposing as open data, but also process details.

However exposing process data in this way, by accessing data storage and/or execution history data tends to be completely decoupled from any given process model, and hence completely out of its control. In process driven environment such thing is rarely desirable. It would probably be beneficial if at least user data object exposure as open data is controlled by process model.

BPMN provides enough concepts to allow embedding open data dissemination as an integral part of relevant process. In fact, there are several ways to implement this, with different impact on process model. Furthermore, embedding data dissemination as integral part of process models, where such dissemination is desirable, may lower the burden of later data gathering, formatting and preparation for publishing as open data.

As presented in [10], one of the problems with overall strategy to make more government data accessible as open data, comes from the fact that most of the data is produced by local levels of government, and with no additional funding for IT services for the task of preparing open data, this tends to create problems.

To explore possibilities of embedding open data dissemination as part of business process model, the rest of the paper will concentrate on case of exposing public procurement data. Main reason for this is that public procurement is well-defined procedure, yet subject to changes in legal requirements and as such is a good candidate to be implemented in BPM systems. Furthermore, most of the readers understand the basics of such procedures, although details can be quite complex. However, full model of public procurement process is quite detailed, large, and will not be presented here, as process model specifics are not of prevailing relevance.

Simplified, procurement procedure requires public entities to publicly declare their procurement need in a public tender. Interested parties pickup tender documentation and decide if they are willing to participate. To participate they need to send a bid to the public entity before the stated deadline. Public entity then performs selection to award the contract to the party with best bid (the scoring system should be also clearly stated in the tender documentation). After the selection procedure, bidders have certain rights to challenge the selection result in a defined period of time. Depending on the results of this challenge, public entity either sign the contract with winning bidder, or the procedure should be annulled. Actual process has much more details, but for basic understanding this simplified description is sufficient.

Currently, there is an international data standard in development for public procurement purposes, named Open Contracting Data Standard [11]. The standard specifies data needed in any phase of public contracting process: planning, tender, award, contract, implementation. Hence, contracting process is more exhaustive than procurement process as it also includes contract implementation monitoring and conclusion. In the procurement process data represented by this standard would be data object during process execution.

III. ENABLING OPEN DATA IN BUSINESS PROCESS MODELS

As stated earlier both working data object, relevant in process executions, and process execution details may be offered as open data. However, since process business management system aims at enforcing timely and coordinated execution of business activities, time aspect of data exposure should be taken into consideration.

To illustrate this, we can analyze public procurement data. Although it is of public interest to have insight in public procurement procedures, not all the data, and not at all times should be readily available as open data. In fact if some data (such as details of a bid) were exposed prior to tender deadline, whole procurement procedure would have to be aborted since this constitutes violation of law. Hence, it is not only important which data will be exposed as open data, but also when it will be exposed.

In scenarios where no attention has been given to the process model, data gathering and exposure (export to open data storage) will be performed outside of the process control. In that case, the logic of selecting data for "harvesting" from process data storage, is entirely in an outside entity (export module). In this case, one solution is to gather, and deliver as open data, only the data gathered from completed process instances.

However, this solution also has its shortcomings - although this approach can guarantee that no data will be exposed before process instance is completed, it will also prevent some data, that should already be readily available, from being visible. In the case of procurement procedure, tender details should at available as soon as it is approved. It is obvious that timing of publicly opening data is important, and furthermore directly influenced by process model.

In other words some data should be exposed as open data as soon as process execution reaches certain point in process ("milestone").

Since process model can change (sometimes even frequently), rules that apply to data exposure may also change, but in this scenario no such change is directly conveyed to the data export module. Additional effort is therefore needed to reconcile behaviors of process management system and data export module. Obviously such approach is prone to errors, or at least requires constant monitoring and adjustment.
There are two domains in which action may be taken to put data exposure under process control.

- First, process model could be supplied with information about which data should be exposed. As stated earlier, BPMN specifies the data object to convey information about data needed for process execution. BPMN allows for describing data object state in each step of the process. However, implementation of data object concept is varying between BPMN systems. If BPM system has support for data object, a logical solution would be to tie the information whether data object (or its constituents) should be made open data. In this case, a special flag property would be assigned to the data object marking it for exposure as open data. However this solution is inadequate from timing perspective.

- Second dimension is timing of exposing data as open data. Process models are composed of activities, gateways and control flow objects (nodes of process graph). Each of these objects has unique id – usually assigned by process designer in order to have some meaningful value. If not assigned, these unique values are created during process model deployment. During process execution, process engine tracks the nodes that have been reached. Some engines are using tokens for this purpose, while others are using open node list approach to achieve this. Using the node id, it is possible to detect if certain point in process is reached. If data object is marked for exposure as open data, besides the simple flag it may also has assigned id of the node that must be reached before this data is made openly accessible. Furthermore, since all process engines provide timer functions it would also be possible to specify moment in time when data should be published.

Therefore, in order to achieve controlled publication of open data, any standard data object, representing process working data, would be extended by special openData property, marking it for open data publication, and optional openDataCondition property stating the node id and/or moment in time when the data should be accessible. Using these markers it is now possible to enforce data publication as open data using the process engine itself.

Since process execution data (such as logged user, performed activity and other) are available at any moment to the process engine, we can treat it in the same manner as user data, practically creating another data object (as process variable), populated with process execution data.

IV. IMPLEMENTATIONAL ISSUES

For test implementation Activiti BPM is used. It is a Java based open source BPM solution. It is BPMN conformant, and provides both implementation engine (process engine), as well as graphic editor for process modeling.

Although Activiti supports data object at model level, this support is limited. Data object may be defined at model level for the process model, but not at node level. Furthermore data objects may be only of simple types. To circumvent this situation it is possible to create xml schema describing complex data objects. But, since no direct support for data object is available at node level of the model (for example at task nodes), object representing procurement data, extended by openData marker and openDataCondition is created as standard POJO and used as process variable. In this manner this object is readily available to process engine and to relevant task instances. Therefore, this approach solves the problem of identifying which data needs to be exposed as open data.

The exact moment during process execution in which data will be published may be determined and expressed in model in several ways:

- Explicitly using automatic (service) tasks
- Using the event listeners on certain elements in model
- Explicitly using intermediate or boundary events

Each of these approaches have some benefits and shortcomings.

Using explicit service tasks in the model – in this approach process model is amended by specific service tasks, fig.1. In this case data export process is becoming a main activity in certain phases of the process. It is easily understandable from the model.

![Figure 1. Data publication as service task](image)

As stated in [11], task nodes in process model should represent main activities needed to advance process toward its end state. In the case of publication of procurement tender documentation the use of service task is justifiable since public availability of this documentation is a prerequisite for process continuation. If service task is used, complete specification of component needed to accomplish the task must be specified. In this case it ammounts to specifying JavaDelegate class responsible for performing this task. JavaDelegate class is then able to access process instance, and its variables. As added flexibility, it also possible to use expression to specify delegate class. In test case this JavaDelegate class was responsible for accessing procurement tender data object and transferring it to external data storage. Most common option for achieving any communication with external systems is through web services. In this case since the place of the service task is defined by the model, data export to public domain (open data) will happen as soon as process execution reaches this task. Downside of this approach is that if used unproportionally, for non-principal activities, it tends to clutter the model, making it hardly readable. Additionally, it is tightly coupled with implementation class.

Using event listeners on certain elements of the model – this approach does not add any visible elements
to the process model. Although this may be the positive side, since the model is not expanded by additional elements, it is at the same time also it drawback. This implementation is not comprehensible just from viewing the model, and for unfamiliar process designer/developer it may take more time to identify all points at which certain actions are performed. Additional annotation on the model may help remedy this deficiency (fig 2).

In this case information about what needs to be done is attached as specification of listener attached to an element in the model or the process model as a whole.

In this case we have different events to listen for:
- for process, automatic tasks, gateways: start, end
- for user tasks: start, assignment, complete, all
- for transitions: take

As with the service task implementations, listener class needs to be specified.

Positive side of this approach is that it is very flexible in regarding the moment in which action will be performed. In our case

Using explicit events

BPMN defines events as basic concept. Events are concept used to enable process to communicate with its environment. Process may be listening for an event (in this case process model will contain catching event), meaning the event will be triggered by outside source, or process may be the one responsible for creating the event (throwing event). Events may be start, end and intermediate. Furthermore there are different types, in correspondence with the nature of event.

In this approach process model is amended by intermediate throwing events - fig 4, and possibly boundary events. When process execution reaches the intermediate event it will generate event to the execution environment. Other processes or parts of the same process may be registered to listen for certain type of events. This approach provides additional flexibility since it may be possible to allow multiple listeners to react to the same event.

BPMN allows different throwing events, but for this purpose signal and message throwing event may be of interest. Difference between message and signal event is subtle but important, while message events must be specifically aimed at certain receiver, signaling event is more general purpose, signal is emitted to anyone capable of receiving it, making it more versatile.

Activiti modeler does not support using message throwing event – its implementation is covered by signaling throwing event.

Usage of event based triggering for publication of data is appropriate whenever the task is not considered as principal task required for process model.

Another approach would allow for using combination of intermediate events and boundary events. Boundary events allow for task or sub-process to listen and react to some events that may happen during its execution. It allows for alternative (or additional path, if event is non canceling) of process. This approach proves to be useful when multiple outcomes may result during the sub-process execution.

Similar to previous solutions a listener class must be registered to receive certain type of signal.

All the solutions use delegation principle to accomplish their goals. Basic principle is relaying on process engine to signal the data export module that it is appropriate moment to export certain data.

In test model all principles has been used, but in later refinement, model was streamlined to use service tasks for critical activities, and signaling events for non-critical, while event listeners attached to tasks and transitions has been removed. Primary reason for such decision was to make model explicit in regards to the points of data export. Data export was implemented by exporting to external database. Nevertheless, any other data export is easily achievable, once the data is extracted form process engine at appropriate moment.

By implementing these steps, data export from the running process has been brought under its control. Changes introduced to the model had no affect on the way process is executed form the users point of view. This way introduction of data export was transparent to the users. And since BPM system is used to control the process execution, it was later easy to adapt model and rearrange elements related to data export to better fit the process model.

One obvious improvement may be implemented - creation of generalized process definition specifically aimed at data export. This process definition could then be used as called activity in different processes as a standardized pattern for data export.

V. CONCLUSIONS

Business process management systems are common solution in enterprise systems. Their primary goal is to enable deployment, enactment and monitoring of processes and enhance process productivity. They accomplish this goal by implementing explicit process
Open data may be sourced from different kind of information systems, but it is almost inevitable that some data will be extracted from BPM systems deployed in various enterprise systems. Gathering data and publishing it to open domain is not always straightforward task, it often requires additional effort from IT departments. Often special care needs to be given not only to the problem of data that should be published, but also to ensure that data is published in appropriate moment. If BPM systems are used as data source for open data, it is only natural to embed data export capabilities in the process models. This way process may control which data is only natural to ensure that data is published in appropriate moment.

This approach, although it requires additional effort to adjust process models, simplifies later tasks of data exposure by reducing need to gather data on different storage, and in process execution history tables. Additionally it ensures that required data is available when certain conditions in the process execution are reached.

In this paper an approach to embedding data extraction features as integral part of process model has been discussed. Several different options, possible in BPMN, and available in current BPM systems have been employed and discussed.

Further enforcement of this principle (of process controlling data export) may be achieved by creating a standard called activity, that would be used whenever data export is needed from running process.