# An Object-Based Approach to Customizable Workflow Monitoring

Ngai-Weng Tang, Yain-Whar Si
Faculty of Science and Technology, University of Macau
{wingtang311@gmail.com, fstasp@umac.mo}

*Abstract*— **Monitoring is considered as one of the key functions in workflow management systems (WfMS). Monitoring provides users with the real-time information of the business processes which are being executed in a WfMS. Information collected for monitoring is usually used by the analysts for business process improvement as well as for decision making. In this paper we propose an object-based approach to workflow monitoring. In this approach, all basic elements in WfMS are divided into "Data Objects". User can monitor the required information by combining simple data objects to form new complex objects. Specifically, fundamental data objects can be dynamically extended for monitoring complex business data. A SQL like query language is also developed to retrieve data objects for the monitoring functions. Based on the proposed approach, user can monitor the workflow status and other key performance indicators by defining different customizable views with various data objects.**

***Keywords-workflow monitoring, data object, customized views, key performance indicators***

## I. INTRODUCTION

In modern businesses, workflow systems play a critical role in streamlining the business processes. A workflow system can automate some or all the activities or behaviors in business processes. For example, an on-line leave application process within a government department can be implemented by workflow systems to increase the efficiency. A more complex and integrated workflow system is called workflow management system (WfMS). The WfMS not only accomplishes the basic requirements but also provides high level management and monitoring functions. In this paper, we will focus on the monitoring functions of a WfMS.

Workflow monitoring service is an essential component in workflow management system (WfMS). Effective workflow monitoring technique can help the organizations to pinpoint problems within the workflows. The information provided by the monitoring also serves as the evidence for refinement of the business processes. Basically, monitoring function requires frequent processing of the audit trial from the WfMS to extract necessary information. Majority of WfMS supports simple monitoring services and provides interfaces and APIs for development.

In [5], Wongchang Hur et al. proposed a customizable workflow monitoring approach. In this approach, some basic elements within WfMS are categorized into "Data Objects". Each user has a specific monitoring template. Each template includes "Objects" he/she needed to monitor, the operations act on the "Objects" and the presentation method.

In this paper, we propose an approach that extends the object-based approach for generating complex monitoring views. Our approach allows users to select objects that they want to monitor. We also allow predefined "Objects" in the system and the user can view the state of each workflow instance by querying these "Objects". A SQL like syntax is also proposed for querying. Moreover, our approach also allows users and process engineers to dynamically compose complex "Objects" for customized monitoring.

This paper is organized as follows. In section II, we briefly review the related work. Overview of our approach is presented in Section III. Section IV illustrates the object-based approach to workflow monitoring. Monitoring views are discussed in Section V. We conclude our paper in VI.

## II. RELATED WORK

Workflow monitoring standard is described by Workflow Management Coalition (WfMC) [2] in Interface 5 – Audit Data Specification [3]. The specification defines what kind of information needs to be captured and recorded during the workflow monitoring.

Wongchang Hur et al. [5] proposed a customizable workflow monitoring approach in which a number of monitoring "Data-Objects (DO)" are identified. "DO" can be used to describe all common workflow entities such as process, activity, application and participant. In Figure 1. vertical axis represents the workflow data types described by (WfMC) and the horizontal axis is based on process life cycle. The process life cycle includes build-time, run-time and archive-time. In this paper, we propose an approach to extend these DO for composing complex data objects. We denotes the DO identified in [5] as the "Primitive Data Objects". In the approach proposed [5], there can be several customized views for each user. Each view is composed by "Data Objects", "Operations" and "Presentations". The view represents a "Monitoring Template". This approach allows capturing/monitoring of all objects from data layer to presentation layer.

Minhong Wang et al. [4] proposed a design of intelligent workflow monitoring with agent technology. In [4], A flexible and dynamic architecture about monitoring system is given. A number of agents are also used in their system such as user agent, planning agent, searching agent, diagnostic agent and information agent. The agents communicate with each other based on Knowledge Query and Manipulation Language (KQML).

Saphira Heijens [6] has proposed a design of administration and monitoring (AdMo) tools in YAWL environment [9]. The paper defined the functionality of the AdMo tools. AdMo tool can do simple task such as adding resources and role management.

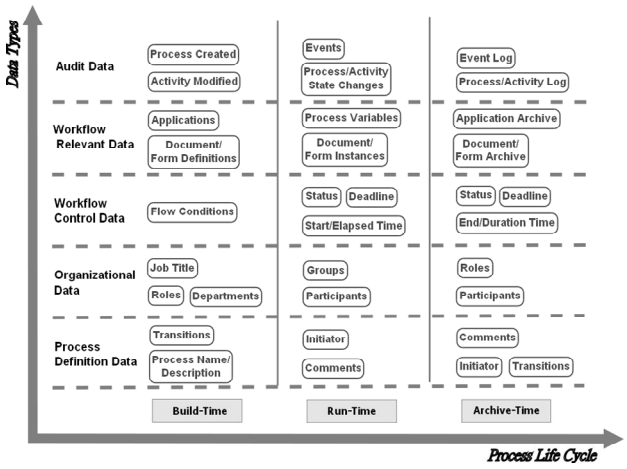


Figure 1. Data Objects proposed by Wongchang Hur et al.[1]

The Business Process Analytics Format (BPAF) [10] is an implementation-independent data format for monitoring workflows. BPAF allows the aggregation and correlation of audit data across multiple platforms. BPAF can be used to capture the information related to the multiples events occur during the lifetime of a process. These events include instantiations, completion, operation related to process engine and other functions.

## III. SYSTEM OVERVIEW

In Figure 2. we depict the architecture of the proposed monitoring system. The system can be decomposed into three layers structure: workflow engine layer, monitoring layer and presentation layer. The workflow engine layer provides database direct access, workflow run-time API and event triggers.

For each type of workflow engine, an "Adaptive Monitoring Interface" can be implemented. This interface which is in the second layer (monitoring layer) generates the corresponding "Data Objects" for monitoring and exports them to the "Data Object Repository (DOR)".

The "Query Module (QM)" is responsible for executing the queries on DOR and returns the required content (the real time monitoring information) to the "View Rendering Module (VRM)". The VRM renders different kinds of views such as Overall View, Resource View, Case View, Exception View and Customized View for the users. The system use XML format for each view generated. Before the user can view the data, XML data is converted to HTML by "HTML Converter". The user can view the final result via a web browser.

In the proposed approach, Data Object Repository controls and maintains all types of "Data Object (DO)" which is the complete set including all groups of data objects. Three groups of DO(s) are defined namely: "Primitive Data Group", "Complex Data Group" and "Meta Data Group". Every identified DO belongs to one of the groups.

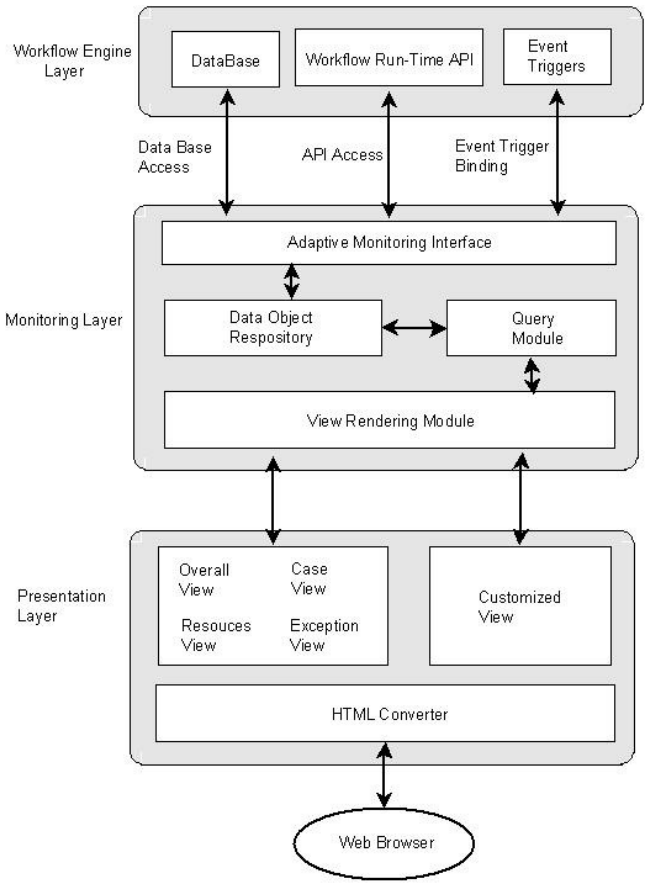


Figure 2. System architecture of object-based workflow monitoring

*Primitive Data Object*: Primitive Data Objects are used to represent predefined basic data objects that provide basic information about the workflow.

*Complex Data Object*: Complex Data Objects are composed by two or more objects (Primitive or Complex) and provides more complete workflow information.

*Meta Data Object*: Meta Data Objects are system objects that provide Meta information about the Data Object Repository. For Example, the user can query the number of all "Complex Data Object" within the "Data Object Repository".

In this paper, we define a SQL like Query Language to define and filter what to be retrieved from each "Data Object". When monitoring requests are sent from Presentation Layer, the "View Rendering Module (VRM)" will submit queries to the "Query Module (QM)". The QM will return the result set to the VRM and then views are created. The syntax for the SQL like query is depicted in the following expression:

*Query expression*:
**Select** *Projection of attributes* **from** *Group.Data-Object* **where** *Filter Conditions*

In the above expression, "Select", "from" and "where" are key words.

- *Projection of attributes*: Each Data Objects has its own attributes and can be retrieved in the same way as columns from tables in Databases.
- *Group*: Prefix of the Data Objects. The symbol can be in an abbreviated form of p, c and m representing primitive, complex and Meta data.
- *Data-object*: Name of the data-object.
- *Filter Conditions:* These conditions can be added to filter the attribute values in order to restrict the size of the result set.

Query for listing the basic information of running cases in the workflow management system can be written as follows:

```
Select case_id,spec_name,spec_id from c.aw where case_id > 2000
```

Query1. Query for monitoring the basic information of running cases in the workflow management system.

TABLE I. EXAMPLE RESULT SET OF QUERY 1

| Case_id | Spec_name | Spec_id |
|---|---|---|
| 2001 | Purchase Order | spec01 |
| 2002 | Purchase Order | spec01 |
| 2003 | Complain | spec02 |

After submitting the Query 1 to the system, a projection of attributes is selected from a data-object of a group. The query also filters the result set by using conditions. Suppose that there is a Data Object "Active-Workflow" (aw) in "Complex Data Group" (C). That is, (aw) $\in$ c. Within (aw), the full attribute set of (aw) is {case_id, spec_name, spec_id, spec}. The example query only projects three required attributes and after the projection, final result is filtered by condition "case_id > 2000".

The SQL like query supports several comparison operators. Data types supported include string, integer, long and date time. TABLE I. shows the example result after executed the query. According to the result, there are three running cases from two different workflow specifications where "case_id" is greater than 2000.

## IV. OBJECT-BASED WORKFLOW MONITORING

Workflow monitoring tools can help us to analyze the behaviors of workflow systems. Although majority of commercial workflow management systems provide general monitoring tools, they cannot meet the specific requirements of the clients. One of the reasons is due to the complexity and relationship among the data within the workflow management system. In this paper, we propose a data object model for describing the required information for monitoring. The proposed data object model is depicted in Figure 3.

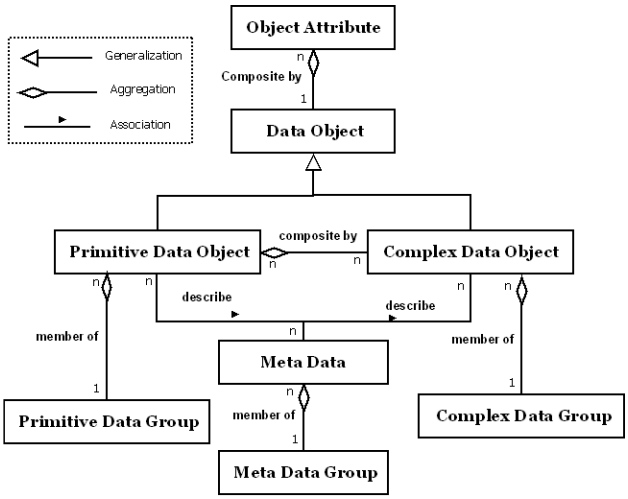


Figure 3. Data Object Model

### A. *Data Object*

Figure 3. describes the data object model of the proposed monitoring system. The most basic item for monitoring is called "Data Object" (PDO). The content in each DO is called "Object attribute". The basic Data Object is called "Primitive Data Object" (PDO). A "Complex Data Object" (CDO) can be built from different combination of PDO.

All data objects are handled and stored in a component called Data Object Repository (DOR). It stores all system default and user defined DO(s).

The information that describes the content of each PDO and CDO is called "Meta Data" (MD). The schema of MD is shown in TABLE II. Every MD has its own identifier and value pair. Each DO should have only one copy of instance in the DOR. The scope of each DO is across the entire workflow instance.

TABLE II. META DATA SCHEMA

| Meta Data Identifier | Description |
|---|---|
| **ID** | Identify ID of the DO |
| **Name** | Meaningful name of the DO |
| **Type** | Primitive or complex object type |
| **Process Life Cycle** | Build-Time(BT), Run-Time(RT) and Archive-Time(AT) |
| **Description** | Text description about the DO |
| **Object attributes** | Schema of attributes that the DO contains |
| **Reference Key Set** | Key set that is available to join with another object |
| **Join Definition (available in Complex Data Object only)** | The link of definition that contains how this object can be formed. |

There are 6 kinds of attribute types for Data Object including "Char", "Int, "Bigint", "Double", "Timestamp" and "Duration" (see TABLE III. ). Some of them can be converted to each other.

TABLE III. ATTRIBUTE DATA TYPES

| Type Identifier | Description |
|---|---|
| Char | Variable length of characters. |
| Int | 4 bytes whole number |
| Bigint | 8 bytes whole number |
| Double | 8 bytes real number |
| Timestamp | Date and time presentation type |
| Duration | Value between two timestamps |

### B. *"Portfolio" of Data Object*

0shows the "Portfolio" of a DO named "Process State Changes". Data Object can be in one of the states of the Process Life Cycle (PLC), Build-Time, Run-Time or in Archive-Time. Although the ID of each DO is unique, the name of the DO can be same across different PLC. Therefore, a proper way to distinguish the PLC of the Data Object is supported by condition filtering.

TABLE IV. PORTFOLIO OF DATA OBJECT "PROCESS STATE CHANGES"

| Process State Changes | | |
|---|---|---|
| **ID** | DO1 | |
| **Name** | process_state_changes | |
| **Type** | Primitive | |
| **Process Life Cycle** | Run-Time(RT) | |
| **Description** | state changes of process instances(cases) | |
| **Object attributes** | case_id<bigint>, previous_state<char>, current_state<char> | |
| **Reference Key Set** | {Case ID} | |
| **Join Definition** | N/A | |
| **Example query** | select * from p.process_state_changes where process_life_cycle = 'RT' | |
| **Result** | the tuples of all object attributes of data object process_state_changes | |
| **Example Data** | ***Case ID*** | 1001 |
| | ***Previous state*** | Ready |
| | ***Current State*** | Running |

### C. *Joining of Data Objects*

Recall that type of DO can be either "Primitive" or "Complex". However, Primitive Data Object (PDO) only contains limited information. Therefore, in order provide sufficient information to user, PDO can be combined to form different Complex Data Objects (CDO). This approach uses the mechanism like SQL join to compose CDO from PDO. In this approach, each object can use their reference key set for joining. However, only one reference key can be used in joining two PDO(s). The join information of each CDO (which is not available in PDO) is defined in a set of "Join Definition" which is stored in the "Meta Data Group".

For example, we can construct a new data object called "My Process" from primitive data objects. This new object can be constructed from "case_id", "process_name", "process_status", "start_time" and "deadline". These primitive data objects are already available in the workflow management system. First, "Process Name" can be joined with "Case-Process Name Mapping" with reference key process_id. Then "Case-Process Name Mapping" can be joined with the other three with reference key "case_id". The joined model is shown in Figure 4.

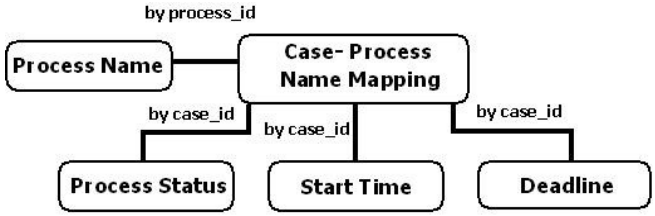


Figure 4. Joined Data Object Model

TABLE V. shows the "Join Definition Table" of the CDO "MyProcess". The final result CDO will be formed after completing the join steps from top to the bottom. There are 6 columns in each step:

TABLE V. JOIN DEFINITION TABLE OF CDO "MYPROCESS"

| Left DO | Right DO | Reference Key | Join Type | Join Filter | Result CDO |
|---|---|---|---|---|---|
| Case-Process Name Mapping | Process Name | process_id | Inner | -- | TDO1 |
| TDO1 | Process Status | case_id | Inner | status = 'running' | TDO2 |
| TDO2 | Start Time | case_id | Inner | -- | TDO3 |
| TDO3 | Dead line | case_id | Inner | -- | My Process |

The new CDO "MyProcess" is constructed from 5 DO(s). As each join step combines two DO(s) (Left and Right) to form a new CDO, the construction of "MyProcess" can be completed within 4 steps. During the construction, 3 temporary DO(s) will be formed (TDO1, TDO2 and TDO3). Attributes from TABLE V. are:

*Left DO* – Left side DO in each join step.

*Right DO* – Right side DO in each join step.

*Reference Key* – Keys for comparison and matching.

*Join Type* – There are four types: "Left", "Right", "Inner" and "Full". The difference among these types is the scope of joining.

In Figure 5. , Content Set "A" (circle on the left) represents the Left join result. Content Set "B" (circle on right) represents the Right join result. Content Set "C" (The intersection part) represents the Inner join result. Content Set "A" plus Content Set "B" minus duplicate Content Set "C" represents the Full join.

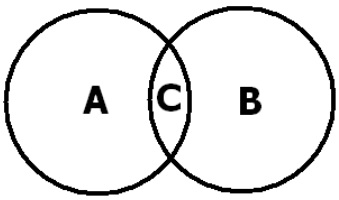


Figure 5. Content Set presentation of DO Joining

*Join Filter* – Conditions for filtering the joined content. Different condition can be defined in each join step.

*Result CDO* – New CDO after joining two DO(s).

## V. MONITORING VIEWS

After CDO(s) are prepared, user can monitor the state of the workflow state via different views. In the work of Michael zur Muehlen [8], three kinds of views are provided: process view, resource view and object view. In this paper, we extend these views into five categories: Overall View, Resources View, Case View, Exception View, and Customized View. The first four views are defined by the system in order to monitor the fundamental information of the WfMS. The last view is a kind of personalized view for monitoring information tailored for individual user. Each user can have multiple customized views as required. In addition, the queries for the first four views can be changed to provide customized relevant contents. In fact, all views provided by the monitoring system can be considered as sets of query. Therefore, a customized view is functionally equivalent to each or a combination of the first four views in terms of content. However, the presentation of the customized view could be different from these views.

**View 1 (Overall View)**

Overview view displays the overall statistics of the workflow system. This view can be initiated by the System Administrator or General Officer of the workflow. In TABLE VI. , two basic Key Performance Indicators of the system is displayed in an overall view. They are "Total number of running cases" and "Total number of available resources (R1)". The query for retrieving the example data is defined in Query 2 and Query 3. Multiple queries can be used within a view.

TABLE VI. EXAMPLE OF AN OVERALL VIEW

| Key Performance Indicator | Value |
|---|---|
| Total number of running cases | 120 |
| Total number of available resources (R1) | 10 |

```
Select sum('ca.cases_id') as 'Total number of running cases'
from complex.cases where ca.status = 'running'
```

Query 2. Query for monitoring KPI (Total number of running cases) of the system

```
Select sum('re.resources_id') as 'Total number of available
resources (R1)' from complex.resources re where
re.resource_type = 'R1'
```

Query 3. Query for monitoring KPI (Total number of available resources (R1) of the system)

**View 2 (Resource View)**

Resource view displays the status and run-time statistics of the resources in the workflow. A resource view may be initiated by a CEO, Human Resource Manager, or Department heads. TABLE VII. lists the information available for resource monitoring. The query used for generating the resource view in TABLE VII. is depicted in Query 4.

TABLE VII. EXAMPLE OF RESOURCE MONITORING VIEW

| Name | Position | Role in workflow | Status | Work item | Duration |
|---|---|---|---|---|---|
| Jack | OD clerk | POManager | idle | N/A | N/A |
| Fredo | Head of OD | POManager | Task allocated | Create Purchase Order | 32min(s) |

```
Select 're.resources_name' as 'Name', 're.position' as 'Position',
're.role' as 'Role in workflow', 're.status' as 'Status',
're.work_item' as 'Work item', 're.duration' as 'Duration' from
complex.resources re
```

Query 4. Query for monitoring available resources.

**View 3 (Case View)**

Case view displays the status and run-time statistics of a workflow case. Case view may be generated by the system administrator or general manager. An example of a case view for monitoring the information for cases is shown in TABLE VIII. The query used for generating the case view in TABLE VIII. is listed in Query 5.

TABLE VIII. EXAMPLE OF A CASE VIEW

| ID | Start User | Status | Start time | End time | Current task |
|---|---|---|---|---|---|
| 10 | Mary | running | 21/7/10 10:20 | N/A | Produce Shipment Notice |
| 11 | Jack | finish | 19/7/10 9:30 | 25/7/10 15:50 | N/A |

```
Select 'ca.case_id' as 'ID', 'ca.initiator' as 'Start User', 'ca.status'
as 'Status', 'ca.start_time' as 'Start time', 'ca.end_time' as 'End
time', 'ca.current_task' as 'Current task' from complex.cases ca
```

Query 5. Query for monitoring all available cases.

**View 4 (Exception View)**

Exception view displays the information about exceptions occurred in the workflow. Exception view may be initiated by the system administrator. TABLE IX. lists the information available for system exception monitoring. The result set in TABLE IX. is generated by executing Query 6.

TABLE IX. EXAMPLE OF EXCEPTION VIEW

| ID | Exception Type | Case ID | Status | Raise work item |
|---|---|---|---|---|
| 20105 | Time out | 2004 | Cancelled | Carrier Timeout |
| 20106 | Constraint Violate | 2030 | No action | Produce Shipment Notice |

```
Select 'ex.exception_id' as 'ID', 'ex.exception_type' as 'Exception Type', 'ex.case_id' as 'Case ID', 'ex.status' as 'Status', 'ex.raise_work_item' as 'Raise work item' from meta.exception ex where ex.timestamp > '2011-1-1' and ex.timestamp < '2011-12-31"
```

Query 6. Query for monitoring exception information.

**View 5 (Customized View)**

Customized view allows business level monitoring for all roles in the workflow. An example of a customized view for monitoring the information on ordering details of the manager 'Peter Chan' is listed in TABLE X. . This example also illustrates that a business level monitoring which is available only on some specific running cases. The result set in TABLE X. is produced by executing Query 7.

TABLE X. EXAMPLE OF CUSTOMIZED VIEW

| Company Name | Cost | status | Task |
|---|---|---|---|
| IBM | 1008000 | running | Payment |
| NEC | N/A | running | Ordering |
| SONY | N/A | running | Carrier Appointment |
| TOYOTA | 6000000 | complete | End_overall |

```
Select 'o.company_name' as 'Company Name', 'o.cost' as 'Cost', 'o.status' as 'Status', 'o.task' as 'Task' from complex.Ordering o where o.POManager = 'Peter Chan'
```

Query 7. Query for listing the information for a customized view.

## VI. CONCLUSION

In this paper, we propose an object-based approach for monitoring workflows. Our approach allows users and process engineers to dynamically compose complex objects for customized monitoring. In this approach, all monitoring criteria are converted to object-based presentation. As a result, the proposed approach enables high level of abstractions for generating customized views. A prototype monitoring system was implemented with YAWL workflow management system [9]. As for the future work, a monitoring framework will be extended to minimize the degradation of the system performance when there are proliferations of monitoring views. We are also planning to address the issues in controlling of user rights (security issue).

## ACKNOWLEDGMENT

This research is funded by the University of Macau.

## REFERENCES

[1] Saphira Heijens, "Support for Workflow Administration and Monitoring in the YAWL Environment," in Information Science, QUT, Brisbane, Australia, August 5, 2005, http://yawl-system.com/documents/Master_Thesis_Saphira_Heijens.pdf, last accessed 27 Nov 2010.

[2] Workflow Management Coalition (WfMC), http://www.wfmc.org , last accessed 27 Nov 2010.

[3] WfMC, "Workflow Management Coalition Audit Data Specification, Workflow Management Coalition," Document Number WFMC-TC-1015, 1998, http://www.wfmc.org/standards/docs/TC-1015_v11_1998.pdf, last accessed 27 Nov 2010.

[4] Minhong Wang, Huaiqing Wang, Dongming Xu, "The design of intelligent workflow monitoring with agent technology," in Knowledge-Based System - Volume 18, Issue 6, page 257-266, 2005.

[5] Wonchang Hur, Hyerim Bae and Suk-Ho Kang, "Customizable Workflow Monitoring," in Concurrent Engineering: Research and Applications - Volume 11, number 4, page 313-325, Sage Publication, 2003.

[6] Saphira Heijens, "Support for Workflow Administration and Monitoring in the YAWL Environment," in Information Science, QUT, Brisbane, Australia, August 5, 2005, http://yawl-system.com/documents/Master_Thesis_Saphira_Heijens.pdf, last accessed 27 Nov 2007.

[7] W.M.P. van der Aalst, L. Aldred, M. Dumas, and A.H.M. ter Hofstede, "Design and implementation of the YAWL system," in Technical Report FIT-TR-2003-07, Center for IT Innovation, QUT, 2003, http://yawlfoundation.org/documents/yawls.pdf, last accessed 27 Nov 2010.

[8] Michael zur Muehlen, Michael Rosemann, "Workflow-based Process Monitoring and Controlling – Technical and Organizational Issues," in Proceedings of the 33rd Hawaii International Conference on System Sciences – Volume 6, page 6032, 2000.

[9] YAWL official website, http://www.yawl-system.com, last accessed 27 Nov 2010.

[10] The Workflow Management Coalition Workflow Standard, Business Process Analytics Format(BPAF), http://www.bpm-research.com/wp-content/uploads/2009/02/2009-02-20-wfmc-tc-1015-business-process-analytics-format-r1.pdf