TOWARDS FLEXIBLE DISTRIBUTED REAL-TIME MONITORING AND MANAGING OF WORKFLOWS

Andre Baumgart
Institute of Computer Engineering,
Automation Laboratory
University of Mannheim
andre.baumgart@ti.uni-mannheim.de

Abstract

The unpredictability of business processes requires workflow systems to support monitoring functions with the ability to flexibly adapt to the changing environment, in which the activities of the workflow are distributed and the status of the process has to be presented in real-time fashion. In this context flexibility refers to the following main requirements: (a) Adaptation to changes of the different dimensions of the workflow (process flows, resources and specific cases) (b) Context awareness of the monitoring systems such that the graphical user interface displays information depending on the situation of the different workflow entities. Previous approaches to workflow monitoring focused on automated routing, distributed monitoring or various repetitive requests to the workflow system. These approaches did neither consider real-time data exchange from user interface clients to the server nor the separation of the process logic from the representational logic of the graphical user interface.

In this paper flexible distributed real-time workflow monitoring is realized by: (1) middleware components supporting real-time monitoring; (2) different controllers for controlling workflows definitions, the specification of the graphical user interface and the data that is managed by the user interfaces. The prototypical software system was implemented as a client-server application in Java using a real-time CORBA middleware component. The server component manages the registration of the user interface clients and controls the data exchange with the workflow management system. The definition of the user interface view, the process data and the managed data is stored in XML files and transferred as XML streams between the software components. The prototypical implementation shows that the separation of process models, data models and view models can lead to flexible monitoring and managing of workflows in real-time and thus guarantee valuable information of business processes at runtime.

Keywords *— Workflow Monitoring, Workflow Management, Real-Time CORBA, Flexible Service, Context-Aware User Interface.*

1 Introduction

In manufacturing and logistic processes workflows show complex and dynamic behavior that is hard to monitor and control. Moreover the unpredictability of the involved business processes require workflow systems to support monitoring functions with the ability to adapt flexibly to the changing environment, where the activities of the workflow are distributed and the status of the process have to be presented in real-time fashion.

Workflow monitoring is used to monitor the state and performance of a process instance during its execution. Figure 1 [14] shows the run time context of the monitoring system. During enactment of the process, the WfMS interacts with users and application tools, e.g. the monitoring service. The WfMS automates the continuous collection of the workflow relevant data at runtime by tracing, storing and transmitting the audit data on basis of events to users and applications. Applications and services are thereby exposed to a specific technological context, and users are situated in a certain environment mostly defined by their role and organizational unit. Both contexts impose restrictions on the data that is exchanged between users, applications and the workflow engine. As a result, the infrastructure of a proposed monitoring system has to scope with the adaption of the different dimensions and perspectives of the workflow, the context awareness of the workflow entities and users and the application specific environment of software tools.

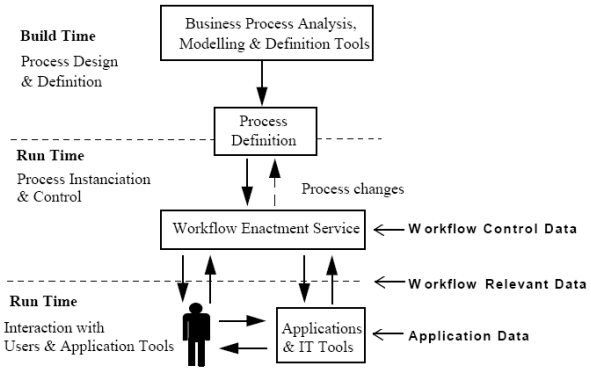

Figure 1: Overview of Data Structures

In this paper we focus on presenting a prototypical distributed graphical user interface (GUI) application and service for displaying process-relevant data in real time. The prototype addresses most of the above outlined challenges. Accordingly, in section 2, we give a more detailed outline of the related work. We subsume and relate workflow monitoring, real time systems and model driven development to our system realization. Section 3 introduces the general architecture of the system and a detailed description of its components. The prototypical implementation is given in section 4 by presenting simulation results of the software system. Section 5 concludes the paper by giving some remarks in future research directions.

2 Related Work

This section explains the related research that is covered in the proposed architecture and prototype. Moreover we review the previous work on workflow monitoring and man-

aging from a technical perspective for developing software tools.

2.1 Process Monitoring and Controlling

In order to reduce the complexity, which is related to process monitoring and managing, it is appropriate to introduce different dimensions of workflows [11]. The fundamental property of a workflow process is that every piece of work is executed for a specific case or workflow instance. The basis of the enactment of the workflow is the workflow process definition which specifies in what order tasks are executed. A work item which is executed by specific resource is defined as activity. Detailed knowledge of the allocation of resources to work items, the duration of activities, and the timing characteristics of events are decisive factors when monitoring workflow at runtime and analyzing the performance of a workflow [12].

The workflow dimensions lead to several perspectives for workflow modeling and execution [12]. The dimensions and perspectives of workflow processes help us in identifying the workflow relevant data stated in section 1. Moreover it raises the need for flexible adaption. The need for flexibility raises many challenging and technical questions to workflow management and monitoring that will lead to our proposed architecture in chapter 3. [15]

The workflow monitoring system is one component of the more general workflow management system. The WfMC identified a reference model that describes the characteristics, terminology and components of the workflow system [2]. In this framework the workflow monitoring system interacts with the workflow enactment subsystem through an API, called Interface 5, for audit functions of processes and activity instances, remote operations, as well as process definitions [13]. McGregor extended this approach by how data can be used for better reporting and performance monitoring [7]. This approach proposes workflow monitoring in the context of a decision support system (DSS) where a user interface interacts with the workflow model subsystem and the data warehouse of the DSS. The data warehouse receives data through the Interface 5 API. We use the extended approach of McGregor to emphasize the importance of flexible and adaptive user interfaces for process monitoring. Nevertheless, our architecture is more flexible and not restricted to this view of a WfMS. Next we present the necessary middleware services for distributed real time monitoring.

2.2 Distributed Real-time Systems

Many software systems today require the control of distributed real-time (DRE) systems, including manufacturing plants or workflow monitoring systems, e.g. in hospitals. Software controllers are added to mechanical and human controllers. Thus we need technologies that can ensure the quality and cost-effectiveness of DRE systems [4]. In WfMS monitoring of large-scale distributed real-time processes require stable applications, latency and dependability requirement if workflow activities are late, and additionally have to run on a wide variety of computer platforms. Therefore adaptable, robust and platform independent middleware that resides between applications and the operating system, network protocol stack, and hardware.

2.3 Model Driven Software Development

In order to built flexible adaptive software systems, engineers need a development approach that separates business and application logic from underlying platform technology. OMG's Model Driven Architecture (MDA) provides an open, vendor-neutral approach to the challenge of technology changes [5]. In MDA, you develop a model of your application that is platform-independent, e.g. in the Unified Modeling Language (UML). This platform independent model (PIM) is then mapped into one or a set of appropriate infrastructure and implementation environments, such as CORBA or Java. The standard mapping is often done by automated translation tools that map PIM to the target application and platform into a more detailed platform-specific model or application (PSM) [3].

2.4 Previous Approaches

Here we give a short review of scientific work done on workflow monitoring systems. Previous approaches to workflow monitoring focused on automated routing of workflows, distributed monitoring in distributed workflow systems or various repetitive requests to the workflow system and its entities. These approaches did neither consider real-time data exchange from user interface clients to the server nor the separation of the process logic from the representational logic. Our perspective concentrates on technical aspects of the graphical user interface and its necessary services for workflow monitoring and managing.

Workflow monitoring, managing and controlling have been under investigation for several years. Due to the various perspectives and dimensions workflow modeling monitoring was discussed under different circumstances. Zachary, Ryder, Ross and Weiland (1992)[8] introduced user models for human computer interaction in real-time workflow monitoring system. Aiello(2004)[1] discussed workflow monitoring and controlling in the context of performance evaluation and timing behavior of workflow performance in real-time. Wang and Wang (2002) and Savarimuthu, Purvis and Fleurke (2004) [9] proposed monitoring systems in a multi-agent based context. All these systems lack workflow perspective support and flexible adaption. A comprehensive overview of workflow monitoring is given in [16], where the reviewed monitoring systems are classified by several perspectives.

3 GUI Prototype for Monitoring

This section describes the GUI Prototype that includes the described requirements (section 1) and methodologies (section 2) for flexible adaption in interactive process workflow monitoring.

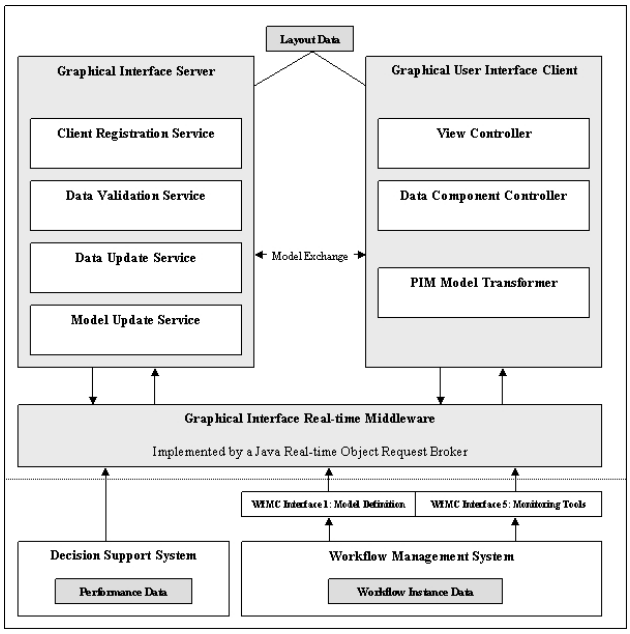

Figure 2: Proposed Architecture

3.1 Overall Architecture

The entire architecture of the GUI Prototype is presented in Figure 2. The main components are the Graphical Interface Server, the Graphical User Interface Client and the Graphical Interface Real-time Middleware components. These components are separated from the workflow management system and the decision support system (DSS). The exchanged models are presented in more detail in section 3.2.

3.1.1 The Graphical Interface Server

The server acts as a multicast sender that handles the exchanged models and data and registers the clients. It consist of the following basic services:

- Client Registration Service: De-/Registers the clients from the server services; handles the client specific data exchange according to the security restrictions set out by clients role and rights.
- Data Validation Service: Validates the process and performance data and assured data consistency for all clients.
- Model Update Service: If models are added, changed or removed this service handles the updating for the clients and offers all necessary information for model updating with regard to user interface presentation.
- Data Update Service: The most frequent service sends new process and performance data to the clients, e.g. if a process instance is monitored in real-time activity state changed by the process participants are transmitted to the registered clients.

Generally, the server should act as the heavyweight component and serves the clients as service pool and computationally expensive tasks.

3.1.2 The Graphical User Interface Client

The client is responsible for displaying the relevant information to the specific user. The data models are received by a specific PIM transformer and then controlled by specific control modules. The main components are:

- PIM Tranformers: Tranform PIMs into PSMs, extract the necessary information for displaying and forwards the data to the component controllers.
- Data Components Controllers: The different controllers depend on the models they are processing.
- View Controller: Handles user context information, the general GUI view and controls the top level container of the GUI.

The client is the lightweight component of the GUI subsystem. The workflow participants face the software for monitoring, controlling and managing workflow processes.

3.1.3 Real-time Middleware

A real-time CORBA-ORB middleware component was used to address the need for timely behavior. The middleware transmits the XML streams to the clients and guarantees the consistency of the exchanged data by adequate time coordination.

3.2 Used Meta-Models

The models used to exchange information are stored in an platform independent manner as XML-Schemata for the meta models and XML for their instances. The information stored in the XML files are transferred via the middleware components. This kind of describing models brings the needed flexibility for the external tools that transfer the needed information to the GUI components and the GUI interface components to exchange valid information. The models used in our implementation are:

- Application Specific Data Model: The Model consists of the process workflow model, the workflow entities and the workflow logic in order to display the necessary workflow information.
- User context model: The user context defines what, when and how workflow information is displayed.
- Performance model: The performance model hold the performance specific information, such as the used performance metric, e.g. activity based costing.
- View Model: The central model of the GUI. It maps the other models (PIM) into the platform specific view of the Java implementation and its data structures.

The view model handles the actual transformation into the platform specific view. In our implementation we used the Java GUI libraries for a platform independent realization.

3.3 Simulation and evaluation

In order to evaluate the performance of the proposed architecture and implementation we carried out a simulation study with 10 registered GUI clients per server. The

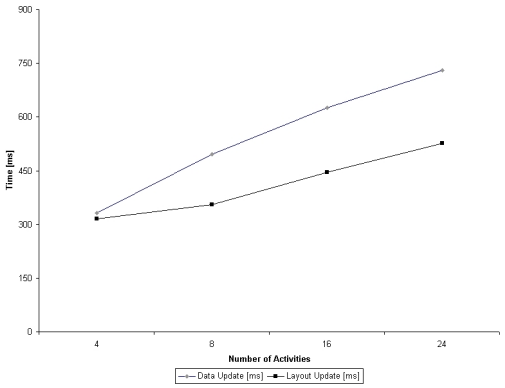

Figure 3: Simulation Study

simulation server sent multicast messages containing different amounts of workflow activities (4, 8, 16, 24 activities). Moreover we tested view and layout model updating for the received and at runtime stored workflow processes and activities at the clients.

Figure 3 summarizes the mean times needed for data updating and model updating. The results clearly shows that the client performance scales reasonably for the transmitted data and model changes. Thus the architecture and implementation shows promising behavior for further exploration.

4 Conclusions and Future Work

We realized a flexible distributed real-time workflow monitoring by: (1) middleware components supporting real-time monitoring; (2) different controllers for controlling workflows definitions, the specification of the graphical user interface and the data that is managed by the user interfaces. The prototypical software system was implemented as a client-server application in Java using a real-time CORBA middleware component. The prototypical implementation shows that the separation of process models, data models and view models can lead to flexible monitoring and managing of workflows in real-time and thus guarantee valuable information of business processes at runtime.

The promising approach for flexible user interface design for workflow monitoring can be extended from technical perspective to several directions. Currently we work on:

1. the integration of mobility in workflow monitoring by ubiquitous and pervasive technologies. Many workflows are executed in mobile environments. Thus monitoring in such an environment becomes much more complex.
2. extended real time support with regard to quality of service requirement. Middleware components should inherit robust behavior in dynamic environments.
3. the extension to various Workflow performance methods have to be integrated into the Workflow Monitoring System to guarantee valuable support for the WfMS users.

References

[1] R. Aiello, "Workflow performace evaluation," Ph.D. dissertation, Universita di Salerno, March 2004.

[2] W. M. Coalition, "http://www.wfmc.org," 2005.

[3] B. P. Douglass, *Real-Time Design Patterns: robust scalable architecture for Real-time systems.* Phearson Education, Inc., 2003, vol. 1.

[4] C. D. Gill, C. Andrews, C. Cross, R. Natarajan, and S. J. Fern, "Towards dependable real-time and embedded CORBA systems," Mar. 22 2002.

[5] O. M. Group, "Mda specification," Object Management Group, Tech. Rep. formal/01-09-67, 2001.

[6] O. O. M. Group, "http://www.omg.org," 2005.

[7] C. McGregor, "The impact of business performance monitoring on wfmc standards," in *Workflow Handbook 2002*, L. Fischer, Ed. Lighthouse Point (FL): Future Strategies, 2002.

[8] J. Ryder, L. Ross, M. Z. Weiland, P. D, and W. Zachary, "Intelligent computer-human interaction in real-time, multi-tasking process control and monitoring systems."

[9] B. T. R. Savarimuthu, M. Purvis, and M. Fleurke, "Monitoring and controlling of a multi-agent based workflow system," in *Australasian Workshop on Data Mining and Web Intelligence (DMWI2004)*, ser. CRPIT, M. Purvis, Ed., vol. 32. Dunedin, New Zealand: ACS, 2004, pp. 127–132.

[10] C. Schuler, R. Weber, H. Schuldt, and H.-J. Schek, "Scalable peer-to-peer process management - the osiris approach." in *ICWS*, 2004, pp. 26–34.

[11] W. M. P. van der Aalst and K. van Hee, *Workflow Management. Models, Methods, and Systems.* Cambridge, MA: MIT Press, 2002.

[12] H. M. Verbeek, "Verification of wf-nets," Ph.D. dissertation, University of Eindhoven, 2004.

[13] WfMC, "Interface 1: Process definition interchange process model," Workflow Management Coalition," Document Number WfMC-TC-1016-P Version 1.1 Final, 1999.

[14] ——, "Terminology and glossary, 3rd edition," Workflow Management Coalition," Document Number WFMC-TC-1011, 1999.

[15] M. W. W.M.P. van der Aalst and G. Wirtz, "Advanced topics in workflow management: Issues, requirements, and solutions," *Journal of Integrated Design and Process Science*, vol. 7, no. 3, pp. 40–77, 2003.

[16] M. zur Muehlen, *Workflow-based Process Controlling. Foundation, Design, and Implementation of Workflow-driven Process Information Systems.*, ser. Advances in Information Systems and Management Science. Berlin: Logos, 2004.