Available online at www.sciencedirect.com
ScienceDirect

Computers in Industry 58 (2007) 453–463

COMPUTERS IN INDUSTRY

www.elsevier.com/locate/compind

Dynamic workflow change in PDM systems

Z.M. Qiu [*], Y.S. Wong

Department of Mechanical Engineering, National University of Singapore, Singapore

Received 2 June 2005; accepted 28 September 2006
Available online 11 December 2006

Abstract

Current manufacturing industry requires product data management (PDM) for efficient product development and production. As an important part of effective PDM solutions, workflow management facilitates creating and executing workflow so as to streamline business processes. Unfortunately, existing workflow management solutions are designed to handle static business processes; when a workflow change occurs, these solutions usually stop the affected workflow completely and start the modified one from scratch. This over-simplified approach leads to re-execution of nodes whose work have been lost due to the restart process. This paper proposes an approach to facilitate efficient dynamic workflow change by minimising repetitive execution of finished workflow nodes. This approach also address the data integrity issue by managing various workflow data such as node properties and scripts. A case study has been carried out in a PDM system to illustrate the potential application of the approach.

Keywords: Workflow; Dynamic change; PDM; Data integrity; Case study; Implementation

1. Introduction

Current manufacturing industry is facing an increasing challenge to satisfy customers and compete in market. To stay competitive, manufacturing companies are adopting IT solutions to facilitate collaborations and improve their product development/production. Among these IT solutions, product data management (PDM) systems play an essential role by managing product data electronically. A tool to model, execute and control business processes, a workflow management tool is a key part of a PDM system.

A workflow can be either static or dynamic. A static workflow has to be finished or aborted once it is initiated, while a dynamic workflow can change during execution time. For organisations adopting workflow solutions, processes should be preferably static since (1) static processes can be applied in large scale with minimal training on personnel and (2) dynamic behaviours of business processes make involved elements, including human, equipment, and information, difficult to handle since these elements are initially designed to deal with specified processes. Hence, organisations seldom change business processes that are established cautiously. In accordance with the above considerations, most workflow management solutions are designed to cope with static business processes in three steps: (1) defining a workflow template, (2) initiating a workflow instance from the template, and (3) executing the instance till its completion. These solutions usually do not handle dynamic behaviours, such as dynamic changes on running workflow instances.

However, process change occurs frequently in business environments due to two primary reasons [1]: (1) at design time the specification of the workflow is not complete due to lack of knowledge and at run time errors happen (2) during executing of workflows, changes occur and cause various problems, such as breakdowns, reduced quality of services, and inconsistencies. Therefore, workflow management systems should handle these undesired results introduced by the dynamic aspects of workflows. The following example illustrates how dynamic workflow change happens in engineering environments.

Company A manufactures numerically controlled (NC) machines while company B is a component supplier to A. However, due to business considerations, company A decides to stop purchasing certain type of components from B and another company, C, is A's new supplier for these affected components.

[*] Corresponding author. Tel.: +65 68744856; fax: +65 67795922.
E-mail address: mpeqzm@nus.edu.sg (Z.M. Qiu).

0166-3615/$ – see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.compind.2006.09.014

Inside company A, some processes related to this change, such as procurement, after-sale service, and testing, should be changed as well since A and C might establish a different form of business relationships.

Once process changes occur, new workflow templates are defined and workflow instances are initiated accordingly. In addition, it is necessary to handle the previous workflow instances which are initiated from old workflow templates. Basically, there are four optional policies to follow [1]:

(1) *Forward recovery:* These old workflows are aborted and handled outside of the workflow management system.
(2) *Backward recovery:* These old workflows are stopped and restarted according to the new workflow template.
(3) *Proceed:* These old workflows proceed as if the change does not occur. New cases are executed based on the new template.
(4) *Transfer:* These old workflows are transferred to the new workflow template and executed all over again.

Most workflow management systems, such as IBM Domino, iPlanet, Fujisu iFlow, TeamCenter (a PDM system with built-in workflow management capability), and Epaf, are able to implement the first three policies in various degrees. However, the fourth policy, transfer, is not effectively supported. Before applying the new workflow template, the old instance has to be stopped and restarted again. Restarting a workflow might lose key runtime information since the state of the workflow instance is refreshed to the original value. More importantly, some completed tasks have to be carried out unnecessarily after restarting the workflow instance. If the affected workflow instance is complex and involves a lot of external collaborators, substantial business cost will be incurred.

To address the above problem, dynamic workflow change management might be brought in as a potential solution. A dynamic change can happen on a single workflow instance or a set of instances under a common workflow template. If a workflow management system supports dynamic workflow change, it can either directly modify the affected instance, or restart it based on the new workflow template while minimising repetitive execution of affected nodes. The first method is instance based while the second is template based (schema evolution).

This paper introduces an approach to address template-based dynamic workflow changes instead of instance-based changes. The motivation behind this paper is based on a common industrial practice that each workflow instance is initiated from its template. Besides supporting template-based dynamic change management, this approach also addresses issues such as transfer efficiency and data integrity. It has been also realised in SmarTeam, a leading PDM system with built-in workflow capabilities. The main contribution of this paper is the in-depth analysis of dynamic workflow change and the solid implementation on a real PDM platform. An efficient non-recursive procedure is developed to identify bypassable nodes in workflow instances. Several implementation issues, such as login management and web-enabled deployment, are addressed. To the best knowledge of the authors, this is the first piece of work ever done to realise dynamic workflow change in commercial PDM systems.

The paper is organised in the following way: Section 2 discusses related work on dynamic workflow change; Sections 3 and 4 present the preliminaries and mechanism of the proposed approach, respectively; Section 5 details the implementation issues; a case study is conducted in Section 6; finally Section 7 gives the summary.

2. Related work

Workflow is a broad topic in computer-supported cooperative work (CSCW) as researchers in various disciplines address workflow-related aspects from their own perspectives. Their work can be classified into the following categories: (1) modelling, (2) analysis and verification, (3) design and implementation, and (4) workflow change.

Workflow modelling has been extensively studied and most modelling techniques are based on Petri Net (PN) and directed network graph (DNG). PN-based modelling [6–8] is widely studied due to its formal nature and the ability to support correctness verification. Further information is given in a survey paper [9]. In addition, there are other modelling tools such as SEAM [12] and UML activity graph [11,10].

Graph reduction techniques were used in [2] to detect structural conflicts in DNG-based workflow models. Verification complexity was addressed at conceptual level to identify fundamental problems [15]. There is also research done to address workflow verification [6,13].

Compared with the above well-studied topics, dynamic workflow change remains an unsolved problem. Casati et. al. [23] proposed a method to facilitate change of workflow schemata by applying a complete, minimal and consistent set of modification primitives. Their analysis was performed at theoretical level and did not address implementation-related aspects. Van der Aalst adopted the concept of workflow inheritance to handle dynamic workflow change [24,25]. His approach requires the inheritance relationship between old and new workflow templates. Shingo defined number of node changes as the cost indicator of dynamic workflow change to evaluate change performance [26]. Ellis presented a Petri Net based approach to handle dynamic changes in workflow systems [27]. He also reported ML-DEWS as a modelling language to support dynamic workflow changes [28]. Reichert et. al. developed ADEPTflex, a method to facilitate dynamic changes of workflow [29,30]. In his work, three kinds of dynamic change were discussed: insertion of tasks, deletion of tasks, and change of task sequences.

All above research on dynamic workflow change is focused on theoretical aspects, while present commercial solutions are not capable of applying existing methods to manage workflow change effectively. Therefore, it is necessary to investigate how to realise dynamic workflow change management in practical PDM systems. Dedicated workflow systems are not good platform for this study since (1) current workflow management tools need to work with various applications to support