Pergamon

Information Systems Vol. 26, No. 2, pp. 93–120, 2001

Printed in Great Britain
0306-4379/01 $20.00

PII: S0306-4379(01)00012-6

WEB INTERFACE-DRIVEN COOPERATIVE EXCEPTION HANDLING IN ADOME WORKFLOW MANAGEMENT SYSTEM

DICKSON K.W. CHIU[1], QING LI[2], and KAMALAKAR KARLAPALEM[3]

[1]Department of Computer Science, University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
[2]Department of Computer Science, City University of Hong Kong, Tat Chee Avenue, Kowloon, Hong Kong
[3]Indian Institute of Information Technology, Gachibowli, Hyderabad, India

(Received 30 September 2000; in final revised form 14 February 2000)

Abstract — Exception handling in workflow management systems (WFMSs) is a very important problem since it is not possible to specify all possible outcomes and alternatives. Effective reuse of existing exception handlers can greatly help in dealing with workflow exceptions. On the other hand, cooperative support for user-driven computer supported resolution of unexpected exceptions and workflow evolution at run-time is vital for an adaptive WFMS. We have been developing ADOME-WFMS as a comprehensive framework in which the problem of workflow exception handling can be adequately addressed. In this article, we present an adaptive exception manager and its web-based interface for ADOME-WFMS with procedures for supporting the following: reuse of exception handlers, thorough and automated resolution of expected exceptions, effective management of Problem Solving Agents, cooperative exception handling, user-driven computer supported resolution of unexpected exceptions, and workflow evolution. © 2001 Elsevier Science Ltd. All rights reserved

Key words: Workflow Management, Exception Handling, Reuse, Workflow Evolution, ECA Rules, Meta-Modeling, Object-Orientation, Workflow Recovery, Web Interface

1. INTRODUCTION

Workflow management system technology, though recent, has been regarded as one of the main types of advanced information systems. It is perceived that workflow technology not only requires the support for complex data model functionality, but also flexibility for dynamically modifying the workflow specifications, especially in cases of exception handling. Because of unanticipated possibilities, special cases, and/or changes in requirement and operation environment, exceptions may occur frequently during the execution of a business process. An exception is an event (i.e., something that happens) that deviates from normal behavior or may prevent forward progress of a workflow. Upon unexpected exceptions, a comprehensive WFMS should be able to support the users to reallocate resources (data / object update) or to amend the workflow, such as adding alternatives (workflow evolution). Further, frequent occurrences of similar exceptions have to be incorporated into workflow specifications as expected exceptions. Such workflow evolution can help avoid unnecessary exceptions by eliminating error-prone activities, adding alternatives, or by enhancing the operation environment. This can lead to a WFMS that supports workflow adaptation through exceptions.

In contrast with traditional software systems, workflows usually evolve more frequently, making reuse a vital issue. Reuse of workflow definitions and exception handlers are very important for the smooth operation of a flexible WFMS. Support for workflow evolution at run-time is vital for an adaptive WFMS. There have been a few WFMSs designed to address these two problems (viz. reuse issues and workflow evolution) effectively and adequately. However, none of them is based on a meta-modeling exception-centric approach.

With a meta-modeling approach, we can have a simple but expressive core data dictionary (meta-schema). From this meta-level schema, users can define all other classes for the WFMS, including activity schemas, exceptions and handlers. Further, from these schemas WFMS objects (in particular, activity instances) can be instantiated. This contributes a substantial improvement to WFMS modeling based on relational models because the entity modeling and implementation is tied together in a straightforward manner. Extensive reuse can also be facilitated as discussed in Section 5.

We use an integrated, event-driven approach for execution, coordination, and exception handling in our WFMS. Events (such as database events / exceptions, or external inputs) trigger the WFMS Execution

Manager to start an activity. The WFMS Execution Manager uses events to trigger execution of tasks, while finished tasks will inform the Execution Manager with events. A task is executed by a problem solving agent (PSA) which is a hardware/software system or a human being. Upon an exception, exception events will trigger the WFMS Exception Manager to take control of resolutions. These resolutions can trigger a software solution or will involve a human to cooperatively resolve an exception.

An effective user interface is vital to the execution of the above features. We choose to use a web-based user interface because workflow and agents tends to be widely distributed, even involving other organizations across the Internet. Mobile clients can be supported (alerted by ICQ or electronic mail) by the Internet, and they can access the WFMS with a web-browser. Direct message passing between clients and remote data sharing can be facilitated. On the other hand, web-based tools are a prevailing technology that has a wide range of utilities and off-the-shelf applications, supporting wide ranges of hardware and software platforms at a relatively low-cost.

In this regard, we have developed ADOME-WFMS, based-on an Advanced Object Modeling Environment (ADOME [45]), with a novel exception centric approach. The key features are as follows:

1. Effective coordination of PSAs and an object-oriented capability-based approach to match tasks and agents;
2. Automatic resolution of expected exceptions and exception driven workflow recovery;
3. Dynamic binding of exception handlers to activities with scoping, and to classes, objects and roles;
4. Addition, deletion and modification of exception handlers at run-time through workflow evolution;
5. Specifying and reusing exception handlers upon unexpected exceptions and system-assisted exception handling; and
6. Application of workflow evolution and workflow recovery in exception handling.

Thus, adding a web-based user interface allows ADOME-WFMS to effectively support distribution of PSA and workflow execution even with a centralized control design. In this article, we present a framework for flexible workflow enactment and online workflow evolution in an advanced object environment (active OODBMS with role and dynamic schema support), with reference to ADOME-WFMS. More details regarding classification of exceptions and handlers, and modeling aspects for ADOME-WFMS are given in [22]. In addition, ADOME-WFMS exception driven workflow recovery has been presented in [26]. The objectives and contribution of this article include: (i) the mechanism of ADOME-WFMS and resolution of expected exceptions, (ii) support of reuse for workflow definitions, constraints, exception types and handlers in ADOME-WFMS, (iii) an augmented solution for exception handling based on workflow evolution, (iv) management of distributed PSA and allow for distributed users, (v) user-driven computer supported resolution of unexpected exceptions, and (vi) demonstrating the use of ADOME-WFMS in supporting exception handling through effective web interface facilities.

The rest of our article is organized as follows. Section 2 presents a meta-modeling approach to activity modeling, which facilitates reuse and workflow evolution. Section 3 presents the architecture of ADOME-WFMS with web-based PSA coordination and general mechanisms. Section 4 discusses how ADOME-WFMS resolves for expected exceptions. Section 5 explains how reuse can be facilitated. The ADOME-WFMS Human Intervention Manager is detailed in Section 6 to illustrate how unexpected exceptions are handled in ADOME-WFMS, with novel web-based workflow evolution functions. Section 7 compares related work. Finally, we conclude the article with our plans for further research in Section 8.

2. FLEXIBLE ACTIVITY META-MODELING

In this article, we model a business process as a workflow (an activity) executed by a set of problem solving agents. We use the terms *activity* and *workflow* interchangeably. A *Problem Solving Agent* (PSA) is a hardware/software system or a human being with an ability to execute a finite set of tasks in an application domain. Typically an activity is recursively decomposed into *sub-activities* and eventually down to the unit level called *tasks* (as illustrated by the example in Figure 1). A task is usually handled by a *single* PSA. The WFMS selects the PSAs for executing the tasks. We match the tasks with PSAs by using a capability-based *token/role* approach, where the main criterion is that the set of capability tokens of a chosen PSA should be matched to the requirement of the task. A *token* embodies certain capabilities of a PSA to execute certain functions / procedures / tasks, e.g., programming, database-administration,