Pergamon

Information Systems Vol. 25, No. 1, pp. 43–69, 2000

Printed in Great Britain
0306-4379/00 $20.00 + 0.00

VERIFICATION OF WORKFLOW TASK STRUCTURES: A PETRI-NET-BASED APPROACH†

WIL M.P. VAN DER AALST[1] and ARTHUR H.M. TER HOFSTEDE[2]

[1]Department of Mathematics and Computing Science, Eindhoven University of Technology, GPO Box 513, NL-5600 MB Eindhoven, The Netherlands

[2]Cooperative Information Systems Research Centre, Queensland University of Technology, GPO Box 2434, Brisbane, Qld 4001, Australia

(Received 15 December 1998; in final revised form 23 February 2000)

Abstract — While many workflow management systems have emerged in recent years, few of them provide any form of support for verification. Consequently, most workflows become operational before they have been thoroughly checked. This frequently results in runtime errors which need to be corrected on-the-fly at, typically, prohibitive costs. This paper shows how verification of a typical process control specification, which is at the heart of most workflow specifications, can benefit from state-of-the-art Petri-net based analysis techniques. To illustrate the applicability of the approach, a verification tool has been developed. This tool can download and verify the correctness of process definitions designed with Staffware, one of the leading workflow management systems.

Key words: Verification, Task Structures, Petri Nets, Workflow

1. INTRODUCTION

Recent years have seen the proliferation of workflow management systems developed for different types of workflows and based on different paradigms (see e.g. [2, 21, 20, 22, 27, 30, 31, 41, 50]). Despite the abundance of such tools, the critical issue of workflow verification is virtually neglected. Few tools provide any form of verification support.

This lack of support can be explained from the fact that the verification of workflows is hard from a computational as well as an algorithmic point of view (see e.g. [1, 7, 26]). The consequences, however, are that few workflows are thoroughly checked before they are deployed in practice, which often results in errors having to be corrected in an ad hoc fashion often at prohibitive costs.

Workflow specifications address many issues including data flow, exception handling, recovery etc. Hence, verification of a full workflow specification is typically not feasible. However, typically the specification of *process control* is at the heart of most workflow specifications. Workflow specification languages need to support the specification of moments of choice, sequential execution, parallelism, synchronization, and iteration.

In this paper we focus on Task Structures (see e.g. [24]) which is a powerful language for the specification of process control. Task structures can be seen as a good general representative of process control specification languages used in workflow management. The specification language as used in [15] is essentially the same as Task Structures. In [10, 9] Task Structures were extended with advanced workflow concepts and used for a real-life workflow application involving road closures in Queensland. There are also workflow management systems that use a language close to Task Structures. In fact, we show that there is a one-to-one correspondence between Task Structures and the diagramming technique used in Staffware [45]. Staffware is one of the leading workflow management systems with more than 550,000 users worldwide. In fact, according to the Gartner Group, Staffware is the market leader with 25 percent of the global market [16].

Petri nets have been around since Carl Adam Petri's PhD thesis in the early sixties [37] and have found many applications in computer science. Petri nets have a rigorous mathematical foundation and a substantial body of theory for their formal analysis has been developed. In this paper this

†Recommended by Gottfried Vossen

theory is exploited and state-of-the-art Petri-net based techniques are used for the verification of Task Structures. The results provide an important impetus for the further automation of workflow verification, in particular as many sophisticated automated tools for the analysis of Petri nets exist. One such tool, Woflan [5], and its application, will be briefly discussed in this paper. In particular, it will be demonstrated how Woflan can be used for the verification of workflow specifications in Staffware.

The organization of this paper is as follows. In section 2 the various perspectives of workflow modeling are discussed. In Section 3, Task Structures are introduced. In Section 4 Task Structures are first mapped to Petri nets and then an extension of this mapping is described to a particular form of Petri nets, WF-nets, particularly suitable for workflow modeling. Section 5 then applies formal Petri net analysis techniques to the results of such mappings. In Section 6 we describe the functionality of Woflan and in particular the implementation of the link between Staffware and Woflan. Section 7 provides a concrete case study highlighting the main aspects of the approach presented. Section 8 gives pointers to related work and Section 9 provides conclusions and some topics for future research.

2. WORKFLOW PERSPECTIVES

The primary task of a workflow management system is to enact case-driven business processes by joining several perspectives. The following perspectives are relevant for workflow modeling and workflow execution: (1) *control flow* (or process) perspective, (2) *resource* (or organization) perspective, (3) *data* (or information) perspective, (4) *task* (or function) perspective, (5) *operation* (or application) perspective. (These perspectives are similar to the perspectives given in [27].) In the control-flow perspective, *workflow process definitions* (workflow schemas) are defined to specify which *tasks* need to be executed and in what order (i.e., the routing or control flow). A task is an atomic piece of work. Workflow process definitions are instantiated for specific *cases* (i.e., workflow instances). Examples of cases are: a request for a mortgage loan, an insurance claim, a tax declaration, an order, or a request for information. Since a case is an instantiation of a process definition, it corresponds to the execution of concrete work according to the specified routing. In the *resource* perspective, the organizational structure and the population are specified. The organizational structure describes relations between roles (resource classes based on functional aspects) and groups (resource classes based on organizational aspects). Thus clarifying organizational issues such as responsibility, availability, and authorization. Resources, ranging from humans to devices, form the organizational population and are allocated to roles and groups. The data perspective deals with *control* and *production data*. Control data are data introduced solely for workflow management purposes, e.g., variables introduced for routing purposes. Production data are information objects (e.g., documents, forms, and tables) whose existence does not depend on workflow management. The task perspective describes the elementary operations performed by resources while executing a task for a specific case. In the operational perspective the elementary actions are described. These actions are often executed using applications ranging from a text editor to custom build applications to perform complex calculations. Typically, these applications create, read, or modify control and production data in the information perspective.

This paper addresses the problem of workflow verification. Although each of the perspectives is relevant, we focus on the control flow perspective. In fact, we focus on the life cycle of one case in isolation. In the remainder of this section, we will motivate why it is reasonable to abstract from the other perspectives when verifying a workflow.

The resource perspective can only restrict the routing of cases, i.e., it does not enable execution paths excluded in the control flow perspective. Therefore, it suffices to focus on deadlocks as a result of restrictions imposed by the resource perspective. A potential deadlock could arise (1) when multiple tasks try to allocate multiple resources at the same time, or (2) when there are tasks imposing such demanding constraints that no resource qualifies. The first type of deadlock often occurs in flexible manufacturing where both space and tools are needed to complete operations thus potentially resulting in locking problems [43]. However, given today's workflow technology, such