Future Generation Computer Systems 38 (2014) 61–68

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Developing security-aware resource management strategies for workflows

Ligang He *, Nadeem Chaudhary, Stephen A. Jarvis

Department of Computer Science, University of Warwick, Coventry, CV4 7AL, United Kingdom

HIGHLIGHTS

- Allocating resources for a type of workflow in pervasive computing.
- Authorization policies are considered when designing the resource allocation strategy.
- Developing resource allocation strategies for both human tasks and computing tasks.

ARTICLE INFO

Article history:
Received 4 December 2012
Received in revised form
15 May 2013
Accepted 14 September 2013
Available online 16 October 2013

Keywords:
Security
Authorization
Workflow
Resource management

ABSTRACT

This paper investigates the resource allocation problem for a type of workflow in pervasive computing. These workflows are abstracted from the enterprise-level applications in the business or commerce area. The activities in these workflows require not only computing resources, but also human resources. Human involvement introduces additional security concerns. When we plan/allocate resource capacities, we often assume that when a task is allocated to a resource, the resource will accept the task and start the execution once the processor becomes available. However, the security policies impose further constraints on task executions, and therefore may affect both application- and system-oriented performance. Authorization is an important aspect in security. This paper investigates the issue of allocating resources for running workflows under the role-based authorization control, which is one of the most popular authorization mechanisms. By taking into account the authorization constraints, the resource allocation strategies are developed in this paper for both human resources and computing resources. In the allocation strategy for human resources, the optimization equation is constructed subject to the constraint of the budget available to hire human resources. Then the optimization equation is solved to obtain the number of human resources allocated to each authorization role. The allocation strategy for computing resources calculates not only the number of computing resources, but also the proportion of processing capacity in each resource allocated to serve the tasks assuming each role. The simulation experiments have been conducted to verify the effectiveness of the developed allocation strategies. The experimental results show that the allocation strategy developed in this paper outperforms the traditional allocation strategies, which do not consider authorization constraints, in terms of both average response time and resource utilization.

1. Introduction

This work considers a type of workflow in pervasive computing. These workflows are abstracted from the enterprise-level application in the business or commerce area. In a workflow of this type, some activities (or tasks) are run on computing resources, which are called Computing Tasks (CT) in this paper, and some need to be processed by human resources (e.g., the employees in a bank), which are called Human Tasks (HT). Nowadays, an employee in an enterprise that is involved in processing the human tasks in certain business processes is often equipped with a wireless device (e.g., iPad or smart phone), so that the activities can be processed wherever the relevant employees are. When a human task needs the employee's attention, an alert or the task itself is sent to his or her wireless device and then the employee spends a certain amount of time to process it, for example, to read the relevant information related to the task and then make corresponding decisions. Human involvement introduces additional authorization concerns. Research has been conducted on the topic of security and authorization constraints in the workflow context [1–4]. Role-Based Authorization Control (RBAC), under which the users are assigned to certain roles while the roles are

* Corresponding author. Tel.: +44 2476573802.
E-mail addresses: liganghe@dcs.warwick.ac.uk (L. He), nadeem@dcs.warwick.ac.uk (N. Chaudhary), saj@dcs.warwick.ac.uk (S.A. Jarvis).

0167-739X/$ – see front matter © 2013 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.future.2013.09.030

associated with prescribed permissions, is a popular authorization control scheme applied to workflow executions [5].

Many workflow management strategies have been developed to enhance the performance of workflow executions [6]. When we design workflow management/scheduling strategies or plan resource capacities, it is often assumed that when a task is allocated to a resource, the resource will accept the task and start the execution once the processor becomes available [7–9]. However, the authorization policies impose further constraints on task executions and therefore, may incur performance penalty and affect both application- and system-oriented performance. The following example illustrates such a situation.

A bank will need both human tasks and computing tasks to support its business. A human task may involve a person (i.e., a user in the RBAC terminology) with an official position (i.e., a role in RBAC, e.g., a branch manager) signing a document; a computing task may involve running an application (often deployed as a service) on a computing resource to assess risk for an investment. Further, the computing applications may be hosted in a central resource pool (e.g. a cluster or a cloud) [10], and the invocation of an application may be automated without human intervention, which we term an Automated Computing Task (ACT), or for security reasons, can only be initiated by a user with a certain role and be executed under that role/user, which we term a Human-aided Computing Task (HCT). The following authorization constraints are often encountered in such scenarios [11]: (1) Role constraints: A human task may only be performed by a particular role; a computing application may only be invoked by assuming a particular role; (2) Temporal constraints: A role or a user is only activated during certain time intervals (e.g., a staff member only works morning hours); (3) Cardinality constraints: The maximum number of tasks (computing or other) running simultaneously under a role is N. It is common to find such authorization constraints and interaction between human and automated activities in other application domains such as healthcare systems [12], the manufacturing community [13,14], and so on. Human intervention and associated authorization clearly affects the processing of tasks and impacts on both application-oriented performance (e.g. mean response time of workflows) and system-oriented performance (e.g. utilization of the computing resource pool). For example, a task may have to wait in the waiting queue due to the temporal constraint and/or cardinality constraint of the role that the task is assuming, even if there are free resources in the system. Therefore, when planning resource capacities for workloads in such situations, we need to take into account the impact of authorization constraints.

The security-aware scheduling issues are also investigated in the literature. The work in [15] developed a security overhead model for workflows and focused on three security aspects: (i) confidentiality, (ii) integrity and (iii) authentication. However, the work presented in this paper focuses on the security aspect of authorization. The work in [16] developed the security-aware resource allocation strategies for real-time DAG jobs in homogeneous clusters and heterogeneous cluster. The work in [17] proposed a Security-Aware Task (SEAT) graph model and based on this model, further developed the methods to achieve maximum security strength while satisfying the real-time constraints. Again, the security issues investigated in the above work do not include authorization. Moreover, those work do not consider human resources.

This paper investigates the issue of allocating both human resources and computing resources for running workflows under the role-based authorization control, so as to mitigate negative impact of authorization constraints on execution performance of workflows.

In the application domains of interest, the allocations of human resources and computing resources have different considerations. In the role-based authorization control, a human resource is affiliated with a role. The human resources with different roles will incur different salary costs (e.g., hiring a branch manager is more expensive than hiring a cashier). The budget is often a major factor of determining the allocation of human resources in enterprise applications. Therefore, this paper takes authorization constraints into account and develops an optimization method to allocate the proper amount of human resources for each role, so that the human tasks can achieve optimized performance subject to the budget limit for human resources.

Due to relatively low costs of computing resources, the cost is typically not a major concern for deploying low- or middle-end computing resources. When the workflows are running under authorization control, authorization constraints may incur performance penalty as discussed in the above workflow example in banks. Therefore, minimizing the overhead caused by the authorization constraints should be a main objective. In order to address this issue, this paper develops a strategy of allocating computing resources. The strategy is able to calculate (1) a proper number of computing resources allocated to host each service, and (2) the processor sharing proportion in each resource allocated to run the tasks assuming a certain role.

In this paper, a workflow consist of human tasks and computing tasks. A computing task involves invoking a computing service hosted in a central resource pool (e.g., a cluster or a cloud). It is assumed that the invocation of computing services can only be initiated by a user with a certain role. A human task is executed by a human resource with a certain role. A human task can also be regarded as invoking a human service provided by a user with a certain role. Therefore, we will discuss human tasks and computing tasks in a consistent manner in this paper.

It is assumed that a set of services (human service or computing service) is hosted by the resources (human resources or computing resources). A task (human task or computing task) in a workflow invokes one of the hosted services.

The rest of this paper is organized as follows. Section 2 presents the methods to calculate the arrival rate of the requests assigned to a role. Section 3 presents the method to allocate human resources, while Section 4 develops the method to allocate computing resources for hosting computing services. The experimental studies are presented in Section 5. Finally Section 6 concludes the paper.

2. Calculating the arrival rate under authorization

In the workflow context in this paper, a task in a workflow invokes one of the services running on the resources. In order to determine the amount of resources allocated to host services, this section first calculates the arrival rate of tasks for each service, which is the invocation rate of each service when there is no authorization control. However, under the authorization control, the tasks have to be assigned to a role before they can invoke the services, and the roles may have temporal and cardinality constraints. Consequently, the services' invocation rates may be different from those when there is no authorization. This section derives the arrival rate of tasks for each role, i.e., the rate at which the tasks are assigned to each role under the authorization constraints. Table 1 lists the notations used in the paper.

2.1. Calculating the arrival rates for services

$\mathbb{S} = \{s_1, \ldots, s_L\}$ denotes the set of services running on the resource pool.

$\mathbb{F} = \{f_1, \ldots, f_N\}$ denotes the set of workflows, which has N types of workflow. Different types of workflow may have different topologies of tasks. A task in a workflow invokes one of the services in S. A service invocation matrix, denoted as $C_{L\times N}$, can be used to represent which services are invoked by a workflow in F. The matrix has L rows and N columns. Row i represents service s_i,