Knowledge-Based Systems 62 (2014) 11–27

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Knowledge augmented policy conflict analysis for services collaboration

Zhengping Wu *, Yuanyao Liu

Department of Computer Science and Engineering, University of Bridgeport, 221 University Avenue, Bridgeport, CT 06604, USA

ARTICLE INFO

Article history:
Received 10 October 2012
Received in revised form 5 February 2014
Accepted 15 February 2014
Available online 12 March 2014

Keywords:
Policy conflict analysis
Knowledge augmentation
Temporal logic
Semantic extension
Services collaboration

ABSTRACT

In the services computing environment, collaborations are constrained by different requirements from different service providers and consumers. Administrators in different providers and users from different consumers use policies to define control rules and configurations of application environments. These control rules and application configurations reflect different performance requirements, management needs, and business contracts. When collaboration is necessary between services for a specific task, various performance and management requirements from individual services and their providers may have conflicts. The situation is even worse when the collaboration is a one-time event. In policy-based systems, these conflicts are reflected in policy conflicts. Thus, we propose a knowledge-augmented logical analysis framework for these policy conflicts in order to make services collaboration possible and smooth. In our policy conflict analysis engine, a knowledge base is used to supply critical information for analyzing dynamic relationships, hidden information, and constraints on attributes and relationships. More importantly, this information is embedded in logic expressions and reasoning processes so that explicit and implicit constraints between different elements can be integrated into one logical analysis framework. Two different case studies in web services and sensor network environments and their corresponding experiment results confirm the strength and applicability of our proposed policy conflict analysis framework.

1. Introduction

Collaboration is a clear trend for applications and services over the Internet. Guarantee for Quality of Service (QoS) is an important yet difficult task to accomplish among collaborating applications and services. However, different services from different providers have different requirements. For example, a real-time HD video service requires a large bandwidth for any synchronization or collaboration, and a messaging service requires low delay and secure transmission for collaboration or integration. These two services may need to be integrated and work concurrently in a multimedia application. In order to manage complex requirements between multiple service providers and service consumers, policy-based management can be applied and these requirements can be represented in policies. Policy-based management is an administrative approach to manage system usage and its governance rules within an information domain. More and more systems have adopted this policy-based management approach. In a collaborating services environment, a policy domain (domain hereafter) is a collection of elements and services administered in a coordinated fashion [1].

Collaborating services can support interactions and coordination between service providers and individual services, as well as service providers and service consumers. Different service providers can share their resources and build new services based on existing services. For example in Fig. 1a, Domain A contains two services: Financial Service *A* and Data Service *D*. Service *A* requires data service *D* to provide enough throughputs. Another Domain B contains two services: Message Processing Service *B* and Data Service *D*. Service *B* needs data Service *D* to respond to every request within a certain time limit.

When these two domains collaborate, they share the same data Service *D* (as illustrated Fig. 1b). At this point, data Service *D* has two policies from Service *A* and Service *B* respectively. However, different management requirements from these two services are reflected in different policies in a policy-based management environment. These requirements may conflict with each other. These conflicts of requirements are usually reflected in policy rule conflicts. For example, before Domain A and Domain B collaborate, they have their own policies to control services and have their own data services. We call the policy in Domain A “Policy 1”, and the policy in Domain B “Policy 2”. In “Policy 1”, Data Service *D*

* Corresponding author. Tel.: +1 203 576 4762.
E-mail addresses: zhengpiw@bridgeport.edu (Z. Wu), yuaoyaol@bridgeport.edu (Y. Liu).

http://dx.doi.org/10.1016/j.knosys.2014.02.019
0950-7051/

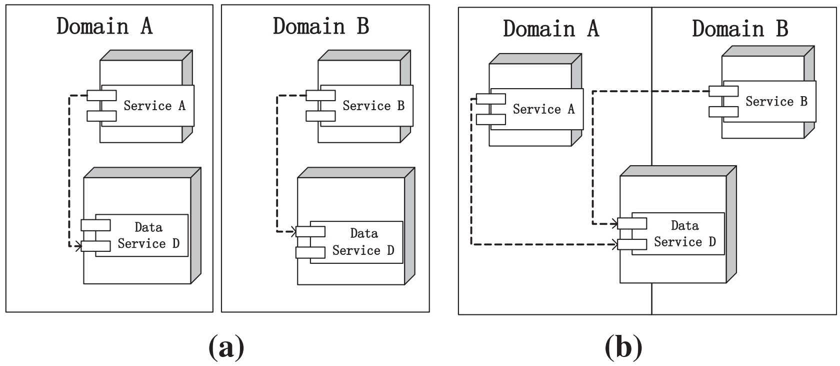

Fig. 1. Service in different domain.

has to provide enough throughputs for Service *A* and the maximum throughput is 2 MB/s. This maximum throughput value is adjustable according to the number of page views in Service *A*. In "Policy 2", Data Service *D* has to respond to any request from Service *B* within 100 time units (0.1 s). Page view on Service *A* is a dynamic and hidden factor, which affects requirements from Service *A*. Before collaboration, Data Service *D* provides certain throughput for Service *A* only. Therefore, Service *D* can use all of its capacity to serve *A*.

In Fig. 1a, when Service *A* and Service *B* collaborate together, they share Data Service *D*, which imposes certain overlap on "Policy 1" and "Policy 2" over Data Service *D*. If Service *A* does not request a very high throughput, Data Service *D* can serve two services very well. But the requirement of Service *A* may change implicitly due to the dynamic number of page views. In order to analyze this dynamic information, we can use temporal logic as the logical reasoning tool to help us detect conflicts before collaboration. However, if we use temporal logic only, we cannot find any conflict because no policy mentions the relationship between page view number and throughput (only throughput of Service *D* and response time of Service *D* is expressed in policies). A major drawback of extant logical representations is the lack of organizational principles for factors constituting a knowledge base [2]. However, relationships and information behind policies are important in policy conflict analysis, especially those implicit relationships. In order to make temporal logic suitable for analyzing policies in a similar situation compared to this example, knowledge support is necessary. In this paper, we introduce a knowledge-augmented approach (in the format of a semantic extension) for analyzing policy conflicts to overcome disadvantages mentioned above.

The rest of this paper is organized as follows. Section 2 covers related work in conflict analysis and temporal logic. In Section 3, we discuss policy model and corresponding conflict analysis approach. In Section 4, we introduce a knowledge-augmented temporal logic to help policy conflict analysis. In Section 5, we discuss our agent architecture for conflict analysis using this new logical tool. In Sections 6 and 7, we study two cases in web services and sensor network environments with their experiments respectively. In Section 8, we compare our approach with other extant approaches for policy conflict analysis in details. Section 9 concludes the paper with our major contributions and future directions.

2. Related work

Temporal logics have experienced rapid development in recent years. Various properties for temporal logics' complexity and axiomatizations are studied [3–5]. Logical expression capability makes temporal logic a good tool for system specification and verification. Recently, temporal logics are used more in reasoning and planning as well [6–8], especially in policy specification reasoning and analysis [9]. In distributed environments, one entity may carry multiple attributes and these attributes can have different definitions in different domains. The complexity of an information domain becomes a barrier for specification and verification of policies. There are two major categories of temporal logics that can be used to analyze temporal attributes. One is liner-time temporal logics; the other is branching-time temporal logics. In the first category [10,11], information is represented as constraints. In [10], authors implement a Dynamic Linear Temporal Logic (DLTL) to specify and verify systems with communicating agents and interaction protocols. Semantic facts of agent communication are specified by means of rules and constraints. In [11], authors describe a logical framework for Temporal Action Logic (TAL) that specifies and verifies interacting systems. This framework provides a simple formalization of communicative actions in terms of their effects and preconditions and the specification of an interaction protocol by means of temporal constraints. Another temporal logic [12] achieves effectiveness and simplicity through reduction of information from information domains. Authors present an A-LTL that inherits some properties from Liner-time Temporal Logic (LTL), including constraints. Interval Temporal Logic (ITL) [13,14] is another linear temporal logic working over finite time intervals. The Propositional Interval Temporal Logic (PITL) [15,16] is an extended Interval Temporal Logic, which considers semantic information through past operators. However, if some information elements cannot be expressed by logic operators, the accuracy of reasoning may be compromised. In [17], a Fuzzy Temporal Logic is proposed. The fuzzy temporal constraints are used for simple cases, where constraints are composed in a single interval. Constraints usually play as a supplement to logical reasoning, which contains limiting conditions from an information domain. Borrowing from this idea, we propose a semantic extension as an addition to temporal logic so that hidden and implicit relationships can be expressed and incorporated in temporal analysis. Meanwhile, a balanced point of time complexity and space complexity can be achieved through proper usage of this semantic extension.

Research in policy conflict analysis has attracted growing interest recently as autonomous and automatic system management has become popular. Dynamic policy analysis also has started to be studied recently. Logic languages [18,19] are widely used in this field. Temporal logic is widely used in different types of policy analysis frameworks. For example, First-order Temporal Policy-analysis Logic (FTPL) [18] is used to check whether a SPKI policy state satisfies a property specified in FTPL. This property check can be applied to static properties and static policies, which is insufficient for collaboration activities. In [19], Event Calculus is implemented in a logic-based policy analysis framework to