Modeling and Reasoning about Business Processes under Authorization Constraints: A Planning-Based Approach

Alessandro Armando  Enrico Giunchiglia  and  Marco Maratea  Serena E. Ponta
Security & Trust Unit, and DIBRIS  DIBRIS - University of Genova  SAP Research
FBK, Trento, and UNIGE, Italy  Viale F. Causa 15, Genova, Italy  Sophia-Antipolis, Mougins, France
armando@fbk.eu  {enrico,marco}@dibris.unige.it  serena.ponta@sap.com

Abstract

Business processes under authorization control are sets of coordinated activities subject to a security policy stating which agent can access which resource. Their behavior is difficult to predict due to the complex and unexpected interleaving of different execution flows within the process. Therefore, serious flaws may go undetected and manifest themselves only after deployment. This problem may be tackled by applying formal methods to reason about business process models. In this paper we outline the main contributions in this application domain of (Armando et al. 2012), that uses the action-based planning language C and the Causal Calculator tool CCALC. C is used to specify a business process from the banking domain that is representative of an important class of business processes of practical relevance, and proved to be a rich and natural formal specification language in this domain. CCALC is then used to automatically solve three reasoning tasks that arise in this context. We also compare C with the SMV specification language used in model-checking: the comparison highlights some key advantages of C in the business process domain.

Modeling Business Processes under Authorization Constraints with C

C (Giunchiglia and Lifschitz 1998) is a propositional action language for expressing actions and how they affect the world described with a set of atomic formulas called fluents. C allows for two kinds of propositions: static laws of the form

\[ \text{caused } F \text{ if } G \]  

and dynamic laws of the form

\[ \text{caused } F \text{ if } G \text{ after } H, \]  

where \( F \) and \( G \) are fluent formulas (i.e. formulas composed by fluent symbols only) and \( H \) is an action formula, (i.e. a formula composed of action and fluent symbols). The informal meaning of (1) is “\( G \) is the cause for \( F \) to be true”, while for (2) its informal meaning is “given a state in which \( H \) is true, in the next state \( G \) is the cause for \( F \) to be true” (if \( H \)

Solving Reasoning Tasks with CCALC

An action description \( D \) is composed by both action and fluent symbols, the fluents are the preconditions, and the actions are to be executed.

Static and dynamic laws have been used to specify the Loan Origination Process (LOP), a business process from the banking domain that is representative of an important class of business processes of practical relevance as it features many aspects that frequently occur in practice: non trivial interplay between the control flow and the security policy, sophisticated access-control policies, events and tasks with nondeterministic, conditional and indirect effects. The process is represented in (Armando et al. 2012) by means of an extended elementary net (see, e.g. (Frau, Gorrieri, and Ferugato 2008)), i.e. a simple Petri net extended with conditional arcs between places and transitions. The specification of the security policy is given in terms of a basic access control model, in our case the Role-Based Access Control (RBAC) model (Sandhu et al. 1996), possibly enriched with features providing the flexibility required by the application domain (e.g. delegation) and mechanisms that are necessary to meet mandatory regulations (e.g. separation of duty constraints). According to the RBAC model, in order to perform a task an agent must be assigned a role enabled to execute the task and the agent must be also active in that role.

In this domain, C supports

- the separate specification of the workflow and of the associated security policy;
- the formal and declarative specification of a wide range of security policies;
- the specification of a variety of business process features, e.g. events, tasks with nondeterministic, indirect, and conditional effects; and, most importantly,

the integration of all the above aspects.

Copyright © 2013, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.
A Comparison between $\mathcal{C}$ and SMV

Languages $\mathcal{C}$ and SMV, which are supported by CACLC and NuSMV (Cimatti et al. 1999), respectively, are compared, focusing on the ability of the two languages to manage changes and updates of the specifications. The two languages differ in a fundamental way in the semantic: In $\mathcal{C}$, if there is no cause for a fact, the fact can be neither true nor false (and thus the formula corresponding to the specifications is unsatisfiable). In SMV declared facts are exogenous, i.e., they can arbitrarily change value in the transition from one state to the other (unless there is some other rule constraining their values). As a consequence, while in $\mathcal{C}$ modelers can elaborate the specification incrementally, this feature is seldom supported by SMV. In (Armando et al. 2012), this claim is substantiate by comparing $\mathcal{C}$ and SMV specifications on some typical scenarios of the business process domain, e.g., where an agent is granted the execution of a task iff she is a potential owner of the task or delegated to perform it, or a situation in which agents are not granted the execution of tasks by default unless they are assigned this duty by, e.g., an administrator, or they are delegated.

In (Armando et al. 2012) we have presented an planning-based approach to the formal specification and automatic analysis of business processes under authorization constraints. By using the $\mathcal{C}$ planning language, we have been able to both greatly simplify the specification activity, and allow for the separate specification of the workflow and of the associated security policy, while retaining the ability to perform a fully automatic analysis of the specifications by using the Causal Calculator CCALC. The experiments we have presented indicate that our approach can be profitably used to execute a number of reasoning tasks particularly important from the application viewpoint, and a comparison with the SMV specification language of model-checkers have highlighted some advantages of $\mathcal{C}$.

Acknowledgment. This work is partially supported by the FP7-ICT-2007-1 Project no. 216471, “AVANTSSAR: Automated Validation of Trust and Security of Service-oriented Architectures” and by the project “Automated Security Analysis of Identity and Access Management Systems (SIAM)” funded by Provincia Autonoma di Trento in the context of the “Team 2009 - Incoming” COFUND action of the European Commission (FP7). It is also partially supported by the Autonomous Region of Sardinia (Italy) and the Port Authority of Cagliari (Sardinia, Italy) under grant agreement L.R. 7/2007. Tender 16 2011, CRP-49656 “Metodi innovativi per il supporto alle decisioni riguardanti l’ottimizzazione delle attività in un terminal container”.

References


