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
FBK, Trento, and UNIGE, Italy
armando@fbk.eu

DIBRIS - University of Genova
Viale F. Causa 15, Genova, Italy
{enrico,marco}@dibris.unige.it

SAP Research
Sophia-Antipolis, Mougins, France
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 $\mathcal{C}$ and the Causal Calculator tool CCALC. $\mathcal{C}$ 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 $\mathcal{C}$ with the SMV specification language used in model-checking: the comparison highlights some key advantages of $\mathcal{C}$ in the business process domain.

Modeling Business Processes under Authorization Constraints with $\mathcal{C}$

$\mathcal{C}$ (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. $\mathcal{C}$ allows for two kinds of propositions: *static laws* of the form

$$\textbf{caused } F \textbf{ if } G \qquad (1)$$

and *dynamic laws* of the form

$$\textbf{caused } F \textbf{ if } G \textbf{ after } H, \qquad (2)$$

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 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 Ferigato 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, $\mathcal{C}$ 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.

Solving Reasoning Tasks with CCALC

An *action description* D is composed by a set of casual laws and can be conveniently represented by a *transition diagram*, i.e. a directed graph that describes the transition "causally explained" by the laws in D. Given an action description D and a query (i.e. a grounded fluent formula), CCALC automatically checks whether there exist paths of a given length n in the transition diagram, i.e. if there is plan of length

n leading to a state is which the query is satisfied: for this bound n it

1. produces, through the process of literal completion (Giunchiglia et al. 2004), a description of the transition diagram in the form of a set of clauses (where a clause is a disjunction of literals) ϕ, such that there is a one-to-one correspondence between the paths of length n[1] of the transition diagram satisfying the query and the assignments satisfying ϕ; then
2. the formula ϕ is fed to a SAT solver together with the set of clauses corresponding to the query; and
3. if the SAT solver returns a satisfying assignment, then the corresponding path is returned to the user.

The reasoning tasks we have dealt with in (Armando et al. 2012) are:

- *Verification of Security Properties*: to establish whether the control flow together with the security policy meets the expected security properties. The security policy of a business process manages the access of agents to tasks, and should ensure that undesirable behaviors, e.g, frauds, do not occur.
- *Synthesis of the Permission Assignment*: for a given number of agents, synthesize a security policy for the business process under given security requirements. In particular, we synthesize the permission assignment for an RBAC model for the LOP given some requirements for the user and the permission assignment.
- *Resource Allocation Plan*: for a given number of agents and for all execution flows, find (if any) an assignment of activities to agents ensuring the process executability according to the given security policy.

A Comparison between $\mathcal{C}$ and SMV

Languages $\mathcal{C}$ and SMV, which are supported by CCALC 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 specification 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.

Conclusions

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

Armando, A.; Giunchiglia, E.; Maratea, M.; and Ponta, S. E. 2012. An action-based approach to the formal specification and automatic analysis of business processes under authorization constraints. *Journal of Compututer and System Sciences* 78(1):119–141.

Cimatti, A.; Clarke, E. M.; Giunchiglia, F.; and Roveri, M. 1999. NuSMV: A new symbolic model verifier. In Halbwachs, N., and Peled, D., eds., *11th International Conference on Computer Aided Verification (CAV 1999)*, volume 1633 of *Lecture Notes in Computer Science*, 495–499. Springer.

Frau, S.; Gorrieri, R.; and Ferigato, C. 2008. Petri net security checker: Structural non-interference at work. In Degano, P.; Guttman, J. D.; and Martinelli, F., eds., *Formal Aspects in Security and Trust*, volume 5491 of *Lecture Notes in Computer Science*, 210–225. Springer.

Giunchiglia, E., and Lifschitz, V. 1998. An action language based on causal explanation: Preliminary report. In Mostow, J., and Rich, C., eds., *Proc. of 15th National Conference on Artificial Intelligence (AAAI 1998)*, 623–630. AAAI Press.

Giunchiglia, E.; Lee, J.; Lifschitz, V.; McCain, N.; and Turner, H. 2004. Nonmonotonic causal theories. *Artificial Intelligence* 153(1-2):49–104.

Sandhu, R. S.; Coyne, E. J.; Feinstein, H. L.; and Youman, C. E. 1996. Role-based access control models. *Computer* 29(2):38–47.

[1] n is set accordingly to the possible transitions in the extended elementary net.