Practical Contract Storage, Checking, and Enforcement for Business Process Automation

Alan Abrahams[1], David Eyers[2], and Jean Bacon[3]

[1] University of Cambridge, Computer Laboratory, William Gates Building, 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK,
asa28@wharton.upenn.edu
[2] University of Cambridge, Computer Laboratory, William Gates Building, 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK,
David.Eyers@cl.cam.ac.uk
[3] University of Cambridge, Computer Laboratory, William Gates Building, 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK,
Jean.Bacon@cl.cam.ac.uk

Abstract. We show how Kimbrough's Disquotation Theory, a formal theory about sentences that embed propositional content, can be profitably applied to the creation of computational environments for monitoring and enforcing electronic commerce contracts using pervasive, mainstream industrial technologies such as Java and relational databases. We examine the notion of an occurrence and provide a structural representation of this abstraction. We show how contractual provisions - obligations, permissions, prohibitions, and powers - can be stored, monitored, and enforced. Detailed examples illustrate how a query coverage-determination mechanism can be used to check inter-organizational contractual provisions against internal policies and external legislation for dynamic conflicts. The work presented here demonstrates that an extended version of Kimbrough's theory presents a novel and promising means of storing interrogable and executable specifications for e-commerce workflow applications.

1 Introduction

Implicit and explicit contracts are an important mechanism for coordinating the activities of an organization. In their contracts, organizations express which states of affairs are desirable and undesirable. They specify which legal states of affairs are achievable, and how, and which are not. In each case, a subjective attitude towards propositional content is stated: the state of affairs is obliged, forbidden, permitted, obtains in law, or does not, according to some clause. We demonstrate here that representing and reasoning about subjective, propositional content has applications in the automation of commerce.

We favor a broadly declarative approach. A procedural execution style hardwires process sequences and requires a complex, manual search-and-fix

strategy each time a business's contracts change. The obligations, permissions, and powers expressed in contracts are not explicit in procedural code, being lost in line-by-line invocations. Traditional imperative languages, which provide no direct representation of rights and duties of parties, therefore introduce an impedance mismatch between the contracts that specify the desired system behavior, and the software that implements these requirements. In conventional imperative language approaches, rules and procedures are obscured in computer code and not directly accessible or modifiable; instead of reproducing rules in procedure manuals, memoranda, and dispersed across computer code, a centralized rule base should be used to ensure locality of update and avoid update inconsistencies [Lee88a]. Our goal is to store explicit representations of the obligations, permissions, and powers inherent in contracts in order that we may use these stored contractual provisions to executably specify workflow applications. Our approach to the representation of rights and duties has been informed by the vast literature on deontic logic, and in particular by Kimbrough's Disquotation Theory. In our software realization, we adopt a declarative, event-driven approach, where each contract provision is explicitly stored and monitored, and software and human components may consult the provisions to determine what currently holds, and what to do next.

Kimbrough's Disquotation Theory [Kim01] is a nascent formal theory that can be used for representing and reasoning about sentences that embed propositional content, such as those commonly found in electronic commerce contracts. We find ourselves in broad agreement with Kimbrough's theory and wish to offer many detailed amendments and extensions. We seek to describe a software implementation of Kimbrough's theory that provides an environment in which we can examine practical application development cases. Our contribution is to move beyond a purely logical view and towards a system that can be implemented and used. Applications and practice very properly have roles in informing the development of the theory. Many have preceded us in the theory department. A vast philosophical literature exists on deontic logic,[1] speech acts [Aus62,SV85], and event semantics.[2] Our goal here is to present a practical approach aimed at mimicking the devices of a particular theory and establishing its plausibility through testing with real applications. We hope to contribute to the theory through an understanding of how to implement the promising ideas. We explain the data structure and software features implemented in our prototype development and execution environment, EDEE, which is capable of representing, storing, and enforcing electronic commerce contracts over diverse platforms using occurrence stores. Our software includes a novel continuous query mechanism for determining which stored provisions apply to a new occurrence.

[1] See [MW93], [And58], [And62], [Das99], [Mak86], [Lin77], [Lee88a], and [PS97] for starters.

[2] See, e.g., [Dav80], [Ben88], [Par90], [JM00], and [PV00].