Towards the Specification and Verification of Legal Contracts*

Alireza Parvizimosaed
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
aparv007@uottawa.ca, https://scholar.google.com/citations?user=m18pXpkAAAAJ

***Abstract*—A contract is a legally binding agreement that expresses high-level requirements of parties in terms of obligations, powers and constraints. Parties' actions influence the status of a contract and shall comply with its clauses. Manual contract monitoring is very laborious in real markets, such as transactive energy, where plenty of complex contracts are running concurrently. Furthermore, liability, right and performance transition through run-time operations such as subcontracting, assignment and substitution complicate contract interpretation. Automation is needed to ensure that contracts respect desirable properties and to support monitoring of compliance and handling of violations. In this thesis research, I propose an innovative ontology that defines fundamental contractual notions (such as the ones mentioned above) and their relationships, on which is built a specification language, called *Symboleo*, that provides syntax and axiomatic semantics of contracts via first-order logic. Symboleo enables the development of advanced automation tools such as a compliance checker that monitors contracts at run-time, and a model checking verification method that analyzes liveness and safety properties of contracts. This paper reports on the problem domain, research method, current status, expected contributions, and main foreseen challenges.**

***Index Terms*—Legal Contract, Specification Language, Model Checking, Smart Contract, Ontology.**

I. Introduction and Motivation

A legal contract is an enforceable bond that regulates acts of participants in a business such as trading. Contracts are expressed in natural language and determine legal requirements (i.e., *obligations* and *powers*) of parties. For example, a seller promises to deliver an ordered good to a buyer within three days. The lifecycle of a contract spans a wide spectrum of activities, from formation to drafting to performance. Offers, acceptance and consideration are some of the main elements of contract *formation*. An offer represents the intention of a party for being in an agreement for exchanging at least two assets as considerations (e.g., a product and money), and a counterparty's acceptance completes the negotiation and legally binds the parties to a *drafted* contract. Contract *performance* starts as soon as an offer is accepted or awaits for contract's start time arrival. A contract's state, obligations and powers all change during contract performance, which complicates manual compliance checking. For example, in Ontario's energy sector, Independent Electricity System Operator (IESO) forms more than 30,000 contracts per quarter, which must be monitored at five minutes interval [1].

In large scale projects (e.g., in construction), obligations are also *subcontracted* to third parties at multiple levels in order to speed up project progress and reduce costs. However, sequential subcontracting may sacrifice the quality of work and general regulations [2]. Contract assignment and substitution are two other legal operations whereby rights, liability or performance are transferred to a new party at run-time. These operations again make contract analysis more difficult owing to the mobility of liability, rights and performance.

Smart contracts, proposed by Szabo [3] over two decades ago, represent an approach aiming to overcome the aforementioned issues. Smart contracts are software systems that monitor the execution of a legal contract to ensure compliance. The Requirements Engineering community as contributed many guidelines into how to extract system requirements from general laws and regulations [4], but less attention was paid to contracts, especially in a context where smart contracts become software programs themselves. It is important to note that smart contracts are independent from blockchains (e.g., they can run on conventional platforms and databases), although many recent implementation approaches take advantage of blockchain technology [5].

In this thesis research, I propose *Symboleo* as a logical specification language for smart contracts that supports early verification (through model checking of properties, at design time) as well as run-time compliance checking. The research questions of interest are:

RQ1: What formal concepts should a specification language support for legal contracts?

RQ2: How can contracts specified with that language be verified against safety and liveness properties at design time?

RQ3: How can contracts specified with that language be used to check compliance at run-time, while supporting appropriate reactions to detected violations?

This research is conducted in four steps. First, for RQ1, I survey many textual contracts from different business areas (including supply chain, energy, construction, and software development) to discover fundamental concepts and their internal relations, and finally produce a *contract ontology*.

Partially funded by an NSERC Strategic Partnership Grant titled *Middleware Framework and Programming Infrastructure for IoT Services* and by SSHRC's Partnership Grant *Autonomy Through Cyberjustice Technologies*.

This ontology is an extension of the Legal Unified Foundational Ontology (UFO-L) [6]. Second, again for RQ1, I propose a formal specification language, namely Symboleo, that adopts propositional and first-order logic to formally describe the syntax and axiomatic semantics of smart contracts and their requirements. Events, time points, and time intervals are fundamental elements of Symboleo used to streamline reasoning about events and time. In the third step for RQ3, a compliance checker tool applies acceptance tests to domain-independent axioms (i.e., Symboleo's semantics) to verify their correctness property. This tool will be extended to support run-time compliance checking based on streams of events. Finally, for RQ2, a mapping to an SMT solver is being developed in order to verify safety and liveness properties of contracts. I iterate through all steps based on new types of contracts being considered, specified, and checked, as suggested in the *Design Science Research* (DSR) methodology, and in turn refine the ontology, Symboleo, and its verification/compliance methods.

The rest of this paper is organized as follows. Section II discusses related work, including legal ontologies, specification languages and verification methods. The research approach is summarized in section III. Section IV presents a contract ontology together with the syntax and semantics of Symboleo. Section V presents our evaluation strategies, whereas Section VI briefly explains foreseen technical challenges. Expected contributions are presented in section VII, and then section VIII concludes and highlights future work.

II. Related work

In the past decade, the *International Workshop on Requirements Engineering and Law* (RELAW) has explored several relationships between requirements and contracts [7]–[10], including compliance and subcontracting aspects. However, none has considered smart contracts nor the level of automation they enable.

UFO-L is a comprehensive legal ontology, grounded in UFO, that models legal relations (i.e., correlation and opposite) among legal positions (i.e., Hohfeldian legal concepts). However, contractual notions such as *asset* and even *contract* have not been addressed in UFO-L [11]. Kaliban et al. [12] introduced a multi-tier ontology of contracts with three levels of abstractions where the highest determines common entities of prevalent types of contracts, the middle specifically investigates a business contract, and the lowest provides detailed templates of a particular contract type. The ontology is designed to monitor execution of contracts through business workflows, but ignores time and detailed obligations that are important prerequisites for legal reasoning. Similarly, Karlapalem et al. [13] conceptually model contracts through an entity-relation data model, and include business process aspects of a contract rather than legal concepts. Goodchild et al. [14] also outline a preliminary specification of business-to-business contracts in which no legal notion is derived from clauses, and hence legal reasoning over their XML-based language is limited to clauses, not legal concepts and relations.

From a contract-as-process perspective, a contract encompasses interrelated obligations and rights whose performance shifts the contract to a new situation through time [15], [16]. Theoretically, such approaches enable contract monitoring, but practically many unforeseen states are generated during a contract's lifecycle, which can freeze monitoring and reasoning processes. On the other hand, contractual terms are not adequately detailed to explore sequences of dependent terms.

Business processes capture social interactions among participants who are responsible toward each other. Social commitments, as ethical obligations tied to society, are conceptually close to legal obligations. However, the legal norms, flexibility that *power* provides, and contract dependency on legal norms differentiate legal contracts. Chesani et al. [17] represent the lifecycle of a commitment as a state machine, and formally specify semantics of the machine transitions through the Event Calculus. In addition to commitment, Dalpiaz et al. [18] take into account business processes, and so integrate social commitments, participants and social constraints into a social interaction protocol. Apart from the specification, an algorithm checks compliance of events that may violate the protocol. Other approaches [19]–[22] either create a new commitment for a third party in place of the older one, or keep the older one to specify commitment delegation.

In addition, there are several efforts targeting the formalization of legal concepts. Prisacariu and Schneider [23] propose an extension of the μ-calculus to specify intuitive properties of a contract based on deontic notions of obligation, permission and prohibition. He et al. [24] suggest a metamodel as the basis for its specification language (SPESC), which has a natural-language-like syntax and an informal description of its semantics. FCL [25] is a propositional and deontic-based logic developed to reason about contrary-to-duty obligations, which are less generic than the concept of power, the latter enabling the creation, suspension, or elimination of obligations upon violations or other situations.

III. Research Method

The research method is inspired from Hevner's Design Science Research (DSR) [26]. DSR is an artifact-oriented methodology that iteratively employs methods and techniques to develop target artifacts and eventually improves human and organizational capabilities.

This thesis research will result in three main artifacts, one for each research question: the Symboleo language (including its underlying ontology and axioms, and an editor), a design-time verification tool for model-checking liveness/safety properties, and a run-time compliance checking tool. To develop and validate these artifacts, I iterate through three steps: 1) interpret sample contracts to explore new concepts, 2) refine the ontology, Symboleo, and verification methods and tools, and 3) evaluate the artifacts through case studies, developed in collaboration with industrial partners from different areas.

So far, I have studied multiple types of contracts, and developed several iterations of the ontology and the Symboleo language. Moreover, I implemented an event-oriented reasoner

tool that enables the testing of Symboleo specifications (e.g., for supply chain contracts) as well as for checking the correctness of the language itself. For the communication of the results, two conference papers on Symboleo are currently under review.

IV. Proposed Solutions

A. Ontology

The contract ontology is drawn from three sources:

1) Legal theories claim that the basis of a contract contains distinct legal concepts. For instance, Hohfeld's theory introduces eight correlative or opposite concepts [27]. Furthermore, Alexy classified these concepts as rules and principles and used Hohfeldian relations to expand his theory's legal relation aspect [28]. In this research, I summarize Hohfeldian norms to *obligations* and *powers* where an obligation is that which a party ought to do, and a power entitles a party to alter the state of obligations, powers or contracts. For instance, *"a supplier is obligated to deliver two tons of beef to a buyer in return of $1000"* is a conditional obligation that enforces the supplier to deliver in case it gets paid. *"The supplier is entitled to terminate a contract if the payment is more than 5 days late"* is a conditional power of the supplier power whose exertion terminates the sales contract.
2) A Legal Core Ontology (LCO) is a specialization of an upper-level ontology that represents imperative legal concepts. UFO-L [11] is a LCO based on Alexy's theory and is grounded in the Unified Foundational Ontology (UFO). This domain-independent ontological model reuses concepts of law, but is not specific to contracts. In turn, a contract is a particular aspect of law categorized in contract law. A *contract* is a composition of obligations and powers whose relations differ from UFO-L's. For instance, UFO-L solely converts a power to correlative or opposite concepts by means of "relators" whereas a power might influence other obligations, powers or contracts. At least two parties are bound to an agreement, and may be assigned to roles. Parties exchange at least two assets through a contract. Although UFO-L covers concepts such as party, asset, role, power and obligation norms, their relations are altered in a contract context. The proposed ontology inherits these concepts from UFO-L, but adds new concepts and relations.
3) The analysis of sample contracts is another ontology elicitation technique. My research team and I manually interpreted 50 contracts and online terms and conditions, and then annotated and classified legal and logical concepts and relations. This led us to enrich the ontology by adding important concepts such as event, time point and time interval, as well as relations such as subcontracting.

Fig. 1 depicts the proposed ontology. All white concepts, except contract, are specializations of UFO-L. Briefly, a *situation* is a state of affairs that is true or false at a given moment. *Events* may bring about a situation. A *legal position*, either power or obligation, is defined between a creditor and a debtor to bring about a situation (i.e., the consequent) if a precondition (i.e., the antecedent) is true. Once a creditor exerts a power, the state of a legal position or contract is modified, e.g., an obligation is created or discharged, or a contract is terminated.

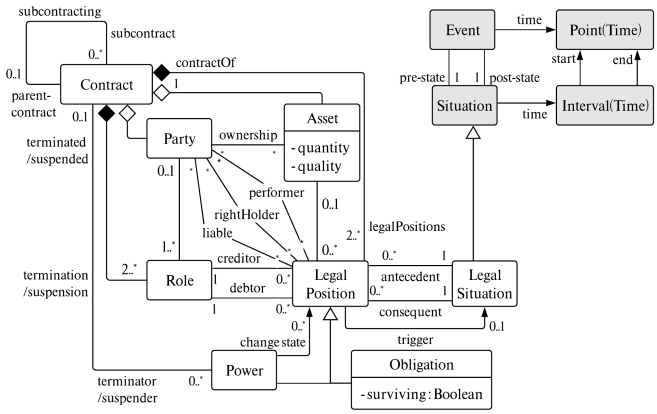

Fig. 1. Proposed contract ontology

At least one (*liable*) party is responsible to fulfill an obligation towards other parties (*rightHolder*). An obligation *performer* is a party who is permitted to fulfill the obligation. Typically, debtors are liable and performer, while creditors are right holders of obligations. However, run-time operations (e.g., subcontracting, assignment and substitution) transfer or share these positions to/with parties. Nobody is liable to a power, while a creditor has the right to exert a power.

B. Finite State Machines

Fig. 2 represents the states of contracts and legal positions in three Finite State Machines (FSMs) that enable compliance and property checking.

An obligation is an entity expressing requirements of an obligee (i.e., creditor). An obligation is created in various ways. Generally, contract activation triggers some obligations, but suspensive obligations take place solely if their preconditions are satisfied. A created obligation stays inactive until the creditor takes proper actions. For example, the previous delivery obligation would be in effect if and only if the buyer pays $1000 to the supplier. In some cases, the creditor's action is not required, e.g., *"the supplier shall always keep beefs frozen"*. An obligation may expire if its creditor acts too late. A debtor may fulfill or violate its obligation. A power may also create, discharge, suspend, resume or terminate an obligation.

The creation, activation, suspension, resumption and expiration of powers are similar to obligations'. The exertion of a power may terminate, suspend or resume contracts and obligations. *Surviving obligations*, however, keep their position beyond the contract termination, e.g., *"participants shall not disclose sales information until six months after the agreement termination"*. A contract terminates successfully when powers and (non-surviving) obligations are no longer active.

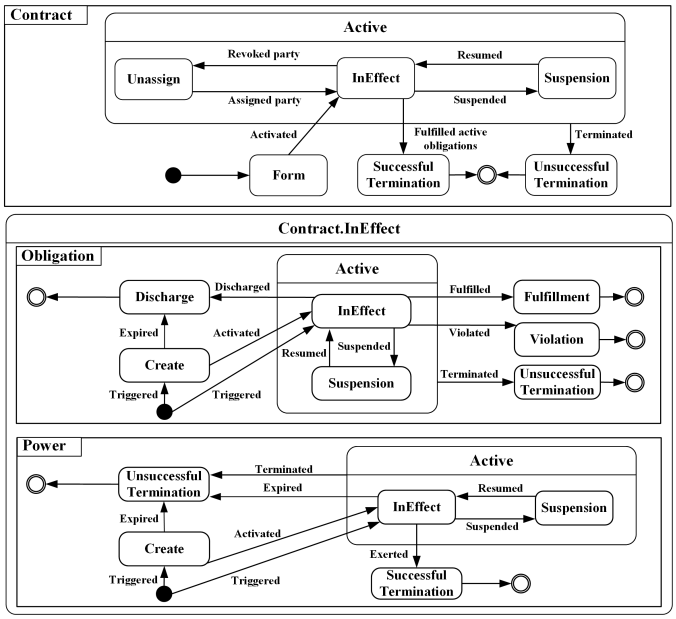

Fig. 2. FSM of the contract, obligation and power concepts

C. Symboleo: a specification language

Symboleo is a specification language of smart contracts that is developed based on the proposed ontology and state machines. The language adopts a first-order and time-based logic for reasoning about time points, time intervals and events. This language is also close to event calculus thanks to common time point, event and proposition concepts.

— **Syntax**. Symboleo's syntax is briefly described in Table I, which shows a contract's *Domain* concepts, including but not limited to events and assets, with their attributes. These are defined outside of a contract, but are instantiated in a contract's *Declarations*. The signature of a contract include a name and multiple input parameters that are valued individually at instantiation time. Pre- and post-conditions might restrict the beginning and successful ending of a contract. For example, a sales contract is valid if the good's owner is the seller. In the table, a *proposition* is a logical composition of events, states of obligations/powers/contract and shorthand predicates. For instance, *"⟨event⟩* ***happensBefore*** *⟨time point⟩"* denotes that the event happens before a certain time point. An extensible library of shorthands is being developed for the language.

The contract body contains a collection of obligations and powers classes. The signatures of an obligation is "*Trigger → Name :* ***O****(debtor, creditor, antecedent, consequent)*" and that of a power is "*Trigger → Name :* ***P****(creditor, debtor, antecedent, consequent)*". *Trigger* is a situation whose satisfaction instantiates the obligation (O) or power (P). Situations are expressed by propositions. The creditor and debtor are two roles. An antecedent determines the precondition of an obligation or power. An obligation's debtor is obliged to bring about the consequent situation while a power creditor has the right to exert that power. For example, in *O_1 : O(seller, buyer, true, happensBefore(delivered,delivered.delDueD))*, *o_1* is an instance of the delivery obligation whereby the buyer unconditionally shall deliver the ordered good before the due date. The constraint part contains safety and liveness properties that respectively ensure bad things never happen and contracts eventually terminate. For instance, the proposition *"NOT(isEqual(buyer, seller))"* prohibits that the buyer and seller roles are assigned to the same parties.

TABLE I
EBNF SYNTAX OF SYMBOLEO

⟨contractSpec⟩ ::= ⟨domainSpec⟩ ⟨contract⟩
⟨domainSpec⟩ ::= **Domain** ⟨name⟩ (⟨dConcept⟩ ';')$^+$ **endDomain**
⟨dConcept⟩ ::= ⟨name⟩ **isA** (⟨name⟩ ',')* ⟨name⟩ **with** (⟨att⟩ ',')* ⟨att⟩
⟨contract⟩ ::= **Contract** ⟨name⟩ '(' (⟨param⟩',')* (⟨param⟩',') ⟨param⟩ ')'
[**Declarations** (⟨declaration⟩';')*]
[**Preconditions** (⟨proposition⟩';')*]
[**Postconditions** (⟨proposition⟩';')*]
Obligations (⟨obligation⟩';')* (⟨obligation⟩';')
[**SurvivingObls** (⟨obligation⟩';')*]
[**Powers** (⟨power⟩';')*]
[**Constraints** (⟨proposition⟩';')*]
endContract
⟨att⟩ ::= ⟨pair⟩
⟨param⟩ ::= ⟨pair⟩
⟨declaration⟩ ::= ⟨pair⟩ **with** (⟨name⟩':='⟨name⟩',')* (⟨name⟩':='⟨name⟩)
⟨pair⟩ ::= ⟨name⟩ ':' ⟨name⟩
⟨obligation⟩ ::= ⟨name⟩':' [⟨proposition⟩ '→'] 'O' '('⟨name⟩','⟨name⟩ ',' ⟨proposition⟩ ',' ⟨proposition⟩ ')'
⟨power⟩ ::= ⟨name⟩ ':' [⟨proposition⟩ '→']
'P' '(' ⟨name⟩ ',' ⟨name⟩ ',' ⟨proposition⟩ ',' ⟨proposition⟩ ')'

— **Semantics**. Symboleo's semantics are specified using primitive predicates. Symboleo adopts five innovative primitive predicates given in Table II in addition to event calculus predicates [29]. This research mainly focuses on the monitoring aspects of contracts. At the moment, 27 axioms are proposed to specify explicit and implicit side effects of FSM transitions. For example, Axiom 1 formulates successful termination of a contract. *holdsAt(s,t)* checks if situation *s* is held at time point *t* while *fulfillment*, *InEffect*, *active* and *successfulTermination* indicate states of obligations and of the contract. The remaining axioms are available in [30].

TABLE II
PRIMITIVE PREDICATES OF SYMBOLAIO

e within s	situation *s* holds when event *e* happens
occurs(s, T)	situation *s* holds during the whole interval *T*, not just in any of its subintervals
initiates(e, s)	event *e* brings about situation *s*
terminates(e, s)	event *e* terminates situation *s*
happens(e, t)	event *e* happens at time instance *t*

Axiom 1. Contract *c* terminates when all obligations and powers of the contract are inactive except surviving obligations.

$$
\begin{aligned}
&happens(e,t) \wedge initiates(e, fulfillment(o)) \wedge \\
&\neg(holdsAt(fulfillment(o),t)) \wedge (e \textbf{ within } InEffect(c)) \wedge \\
&(\forall o'/o.contr.obl \mid surviving(o') \vee \neg(e \textbf{ within } active(o'))) \wedge \\
&(\forall p'/o.contr.power \mid \neg(e \textbf{ within } active(p'))) \\
&\quad \rightarrow initiates(e, successfulTermination(c)) \wedge \\
&\quad terminates(e, InEffect(c))
\end{aligned}
\tag{1}
$$

In the second step, we investigated informal semantics of substitution, subcontracting and assignment.

— **Subcontracting**. A liable party partially or totally delegates performance of obligations to a third party through a subcontract. In specific situations, when stated in the contract, counterparties' consent is a prerequisite to any delegation. In all cases, the subcontractor takes over the performance of obligations while the original partner still preserves liability.

— **Substitution.** A substitution occurs if a party takes the place of another one in a contract and undertakes liability of obligations as well as their performance while the original party is no longer committed. Similar to subcontracting, consent of counterparties may be required; however, no new contract is created.

— **Right assignment.** This operation transfers the right of a party to someone else. For example, if a seller is entitled to get paid, the seller can transfer the right of payment to another party.

V. Evaluation Strategy

Symboleo's semantics and verification methods will be evaluated in three ways:

- Real contracts from existing industrial partners in Canada and France will be specified with Symboleo to assess the language's completeness and adequacy for the supply chain, energy trading and telecommunication areas. In this way, *RQ1* will be partially answered. So far, contracts for the sale of goods and for freight have been specified, while energy trading contracts and service level agreements are under development.
- A compliance checking tool is being developed. The short-term goal is to exercise and assess the correctness of the language's axioms, while enabling the execution of acceptance tests on Symboleo contracts, partially answering *RQ3*. A Prolog prototype was implemented (in Prolog), and the sales of good and freight contracts have been used as input. The tool executes sequences of events (tests) and checks whether expected situations are met. Tests will be developed in collaboration with our industrial partners. The long-term goal is to connect this prototype to an external source of streamed events (e.g., coming from IoT devices and other sources, via a Complex Event Processing engine), enabling the run-time monitoring of contracts.
- Another tool will be developed for the verification of safety and liveness properties at design time (RQ2). These properties which are encoded as temporal logic formulas represent the behaviour of contracts over time. For example, *powers* never activate contradictory obligations at a moment. This tool will transform a Symboleo contract specification into the input language of an existing SMT solver (Microsoft Z3 or others) to model-check properties. The same case studies will be used, with properties developed in collaboration with our partners.

VI. Technical Challenges

The diversity of legal contracts is a challenge. I will focus on business contracts, and will not cover contracts in domains such as marriages or employment. As mentioned, real (business) contracts are an undeniably useful source of concepts for the specification language. However, there is no guarantee that these concepts are sufficient to support other types of contracts.

Contracts are often subject to existing laws and regulations, which are specified outside individual contracts. This information (e.g., jurisdiction-related obligations and powers) likely needs to be encoded as well and imported by contracts.

Moreover, neither contract law nor contractual clauses demonstrate the side effects of a contract suspension or resumption. For example, a contract may entitle a party to suspend a contract in case of an obligation violation. In such situation, where the party exerts this suspension power, which obligations of which party get suspended? What happens to time-dependent terms after resumption? Who is liable for assets such as perishable goods during suspension? We suspect our formal contract specifications to be required to be more detailed than conventional contracts. Yet, flexibility in contractual obligations is at times convenient (especially in this COVID-19 era). These tensions between formality, completeness and flexibility will need to be studied.

The selection of an appropriate SMT environment for *RQ3* remains a challenge, assuming that one exists that will allow a mapping from Symboleo to its input language. One option might be to modify an open-source SMT solver to support missing concepts or algorithms, if needed.

VII. Expected Contributions

As explained before, the expected artefacts produced in the thesis will be the Symboleo language, with its ontology, FSMs and axioms (*RQ1*), with an SMT-based model-checking tool for design-time verification of liveness and safety properties (*RQ2*), and a tool for the testing of contracts and for their monitoring at run time (*RQ3*).

As the research is driven by DSR, I expect the current artefacts to evolve substantially as I progress through my research while considering an increasingly large number of real-life contracts and properties.

This research will contribute to the formal representation of requirements in (smart) contracts, with support for their analysis and monitoring, two activities in dire need of automation. This research goes beyond existing work by taking into consideration the concept of power, which enables (among other things) contracts to manipulate obligations dynamically and to react to detected violations, by considering time explicitly in contract analysis, and by supporting different levels of subcontracting. These language concepts are essential to make the specification and monitoring of (smart) contracts practical and beneficial.

VIII. Conclusion

A contract contains legal requirements of parties expressed in terms of obligations and powers. A power can manipulate a contract and its obligations at execution time. Due to the ambiguity of natural language (used in the vast majority of contracts) and the capabilities of powers and other advanced concepts, combined to complicated types of subcontracting relationships, a contract often encompasses inconsistent and even contradictory clauses, or important gaps. In addition, the real-time compliance checking of smart contracts is now enabled by the growth of IoT and blockchain-based technologies.

The Symboleo specification language is being proposed in this thesis to formally specify legal contracts and their requirements, find property violations, and automatically check compliance.

As a preliminary step, my colleagues and I reviewed 50 valid contracts and designed a contract ontology based on UFO-L that is compatible with Hohfeldian's notions. In addition, FSMs were developed to model the states of obligations, powers and contracts. I also defined the syntax and axiomatic semantics of Symboleo according to the ontology and state machines. I also implemented a compliance checking tool and conducted some acceptance tests on several realistic contracts to evaluate the correctness of Symboleo's axioms.

The next step will be for me to focus on *RQ2* and develop model-checking capabilities for Symboleo. I will then iterate over the three research questions by modeling and checking and increasing number of real contracts and properties from different domains.

Acknowledgment

I would like to express my deepest appreciation to Prof. John Mylopoulos and Prof. Daniel Amyot, my co-supervisors, for their endless support and help. I am also thankful to Prof. Luigi Logrippo and Sepehr Sharifi for collaboration on this research.

References

[1] IESO. (2020) A progress report on contracted electricity supply: Q4-2019. [Online]. Available: https://bit.ly/2RrEC8D

[2] V. W. Tam, L. Shen, and J. S. Kong, "Impacts of multi-layer chain subcontracting on project management performance," *International Journal of Project Management*, vol. 29, no. 1, pp. 108–116, 2011.

[3] N. Szabo, "Formalizing and securing relationships on public networks," *First Monday*, vol. 2, no. 9, 1997.

[4] P. N. Otto and A. I. Anton, "Addressing legal requirements in requirements engineering," in *15th IEEE International Requirements Engineering Conference (RE 2007)*. IEEE CS, 2007, pp. 5–14.

[5] D. Macrinici, C. Cartofeanu, and S. Gao, "Smart contract applications within blockchain technology: A systematic mapping study," *Telematics and Informatics*, vol. 35, no. 8, pp. 2337–2354, 2018.

[6] C. Griffo, J. P. A. Almeida, and G. Guizzardi, "Towards a legal core ontology based on Alexy's theory of fundamental rights," in *Multilingual Workshop on Artificial Intelligence and Law, ICAIL*, 2015.

[7] B. Berenbach, Ren-Yi Lo, and B. Sherman, "Contract-based requirements engineering," in *2010 Third International Workshop on Requirements Engineering and Law (RELAW)*. IEEE CS, 2010, pp. 27–33.

[8] R. Nekvi, R. Ferrari, B. Berenbach, and N. H. Madhavji, "Towards a compliance meta-model for system requirements in contractual projects," in *2011 Fourth International Workshop on Requirements Engineering and Law (RELAW)*. IEEE CS, 2011, pp. 74–77.

[9] B. Westphal, D. Dietsch, S. Feo-Arenis, A. Podelski, L. Pahlow, J. Morsbach, B. Sommer, A. Fuchs, and C. Meierhöfer, "Towards successful subcontracting for software in small to medium-sized enterprises," in *2012 Fifth IEEE International Workshop on Requirements Engineering and Law (RELAW)*. IEEE CS, 2012, pp. 42–51.

[10] L. d. S. Barboza, G. A. d. A. C. Filho, and R. A. C. d. Souza, "Towards legal compliance in it procurement planning in brazil's federal public administration," in *2016 IEEE 24th International Requirements Engineering Conference Workshops (REW)*. IEEE CS, 2016, pp. 229–238.

[11] C. Griffo, J. P. A. Almeida, and G. Guizzardi, "Conceptual modeling of legal relations," in *International Conference on Conceptual Modeling*. Springer, 2018, pp. 169–183.

[12] V. Kabilan, P. Johannesson, and D. M. Rugaimukamu, "Business contract obligation monitoring through use of multi tier contract ontology," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 2003, pp. 690–702.

[13] K. Karlapalem, A. R. Dani, and P. R. Krishna, "A frame work for modeling electronic contracts," in *International Conference on Conceptual Modeling*. Springer, 2001, pp. 193–207.

[14] A. Goodchild, C. Herring, and Z. Milosevic, "Business contracts for B2B," in *Workshop on Infrastructure for Dynamic Business-to-Business Service Outsourcing, ISDO*, ser. CEUR Workshop Proceedings, vol. 30. CEUR-WS.org, 2000. [Online]. Available: http://ceur-ws.org/Vol-30/paper8.pdf

[15] A. Daskalopulu, "Modelling legal contracts as processes," in *Database and Expert Systems Applications, 2000. 11th Int. Workshop*. IEEE, 2000, pp. 1074–1079.

[16] G. Governatori, F. Idelberger, Z. Milosevic, R. Riveret, G. Sartor, and X. Xu, "On legal contracts, imperative and declarative smart contracts, and blockchain systems," *Artificial Intelligence and Law*, vol. 26, no. 4, pp. 377–409, 2018.

[17] F. Chesani, P. Mello, M. Montali, and P. Torroni, "Representing and monitoring social commitments using the event calculus," *Autonomous Agents and Multi-Agent Systems*, vol. 27, no. 1, pp. 85–130, 2013.

[18] F. Dalpiaz, E. Cardoso, G. Canobbio, P. Giorgini, and J. Mylopoulos, "Social specifications of business processes with Azzurra," in *9th International Conference on Research Challenges in Information Science (RCIS)*. IEEE CS, 2015, pp. 7–18.

[19] Ö. Kafalı and P. Torroni, "Social commitment delegation and monitoring," in *Computational Logic in Multi-Agent Systems*. Springer, 2011, pp. 171–189.

[20] ——, "Comodo: collaborative monitoring of commitment delegations," *Expert Systems with Applications*, vol. 105, pp. 144–158, 2018.

[21] A. K. Chopra and M. P. Singh, "Multiagent commitment alignment," in *8th International Conference on Autonomous Agents and Multiagent Systems – Volume 2*. IFAAMAS, 2009, pp. 937–944.

[22] P. Yolum and M. P. Singh, "Reasoning about commitments in the event calculus: An approach for specifying and executing protocols," *Annals of Mathematics and Artificial Intelligence*, vol. 42, no. 1-3, pp. 227–253, 2004.

[23] C. Prisacariu and G. Schneider, "A formal language for electronic contracts," in *International Conference on Formal Methods for Open Object-Based Distributed Systems*. Springer, 2007, pp. 174–189.

[24] X. He, B. Qin, Y. Zhu, X. Chen, and Y. Liu, "SPESC: A specification language for smart contracts," in *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1. IEEE, 2018, pp. 132–137.

[25] G. Governatori and Z. Milosevic, "A formal analysis of a business contract language," *International Journal of Cooperative Information Systems*, vol. 15, no. 04, pp. 659–685, 2006.

[26] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design science in information systems research," *MIS quarterly*, pp. 75–105, 2004.

[27] W. N. Hohfeld, "Some fundamental legal conceptions as applied in judicial reasoning," *Yale Lj*, vol. 23, p. 16, 1913.

[28] R. Alexy, *A theory of constitutional rights*. Oxford University Press, USA, 2010.

[29] M. Shanahan, "The event calculus explained," in *Artificial intelligence today*. Springer, 1999, pp. 409–430.

[30] A. Parvizimosaed and S. Sharifi, "Symboleo Compliance Checker," May 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3840727