

Translating business contract into compliant business processes

Z. Milosevic¹, S. Sadiq, M. Orłowska,

*School of Information Technology and Electrical Engineering,
The University of Queensland,
Brisbane, QLD 4072, Australia.
{zoran, maria, shazia}@itee.uq.edu.au*

Abstract

This paper presents an approach for translating legalese expression of business contracts into candidate business activities and processes while ensuring their compliance with contract. This is a progressive refinement using logic-based formalism to capture contract semantics and to serve as an intermediate step for transformation. Particular value of this approach is for those organisations that consider moving towards new approaches to enterprise contract management and applying them to their future contracts.

Keywords: *Business Contracts, Internal Business Processes, Collaborative Business Processes.*

1. Introduction

Business processes have been traditionally designed with the aim of achieving some internal goals such as an increasing efficiency or reduction of cost in performing internal business activities. In doing so, businesses were focusing on optimal design (or indeed redesign) of their internal processes. Increasingly however, business process designers are becoming cognisant of a need to consider external influences such as the impact of regulatory compliance requirements or the role of business contracts as governing mechanism for cross-organisational interactions.

In order to provide an explicit support for linking processes with external constraints, there is a need to start discovering and documenting rules and patterns that could be applied when trying to map such external constraints onto the design of business processes. This is of particular importance for processes crossing organisational boundaries.

The aim of this paper is to provide a first step in this direction, by proposing an approach which can be used to progressively transform legalese version of contract into business activities and processes, compliant with the contract and in a form which is executable by the underlying process engines. Our current focus is on considering the constraints from business contracts, but the analysis is quite generic so that it can be applied more broadly, in terms of any kind of external constraints.

The next section provides discussion about legalese form of contracts and different types of contract conditions in such forms. Section 3 introduces an example that will be used to illustrate various concepts and ideas in the rest of the paper. This is followed by the description of a contract formalism, which we call a Formal Contract Language (FCL), first introduced in [6], but taking into account significant prior research results of a series of authors [3], [5], [8]. The subsequent section presents our transformation approach and provides several discussion points. The paper concludes with a list of key findings and an outline of several future research directions.

2. Contracts – key legalese structures

From a system-theoretic point of view, a contract is an agreement that specifies part of the collective behaviour of two or more objects [19]. In the world of law, these objects can be trading partners (individuals or organisations) considered as legal entities and the agreement reflects their mutual promises, commitments and expectations while being enforceable by law. The behaviour of an object may involve many other aspects of behaviour, either internal behaviour or interactions with other objects which are subject to other agreements and contracts.

¹ Also with Deontik, Australia (www.deontik.com)

From a legal point of view, the contract-governed part of the collective behaviour can typically be expressed through one or more of the legally-centric types of contract conditions. We have identified several categories of such conditions or legalese structures:

1. The declaration of *pre-existing external constraints* from the environment which apply to the contract as a whole or to the variables in the contract. These for example may be rules and policies from various types of law, such as taxation law, employment law or business contracts law;
2. *Definitional expressions* explaining meaning of certain terms in contracts, e.g. that price is nominated in the Australian dollars;
3. The declaration of a *period of validity* when the contract is in effect or a duration for the contract;
4. The statement of *core normative policies* such as obligations, permissions and prohibitions [9] that apply to the parties involved, either directly from the contract in question or from the environment, as per category 1 above. Note that in the contracts, the core normative policies will typically apply to a subject of a policy but will often mention a target or beneficiary of the policy; in cases where a beneficiary is omitted, it may be inferred from the context. In the deontic logic literature, if a normative policy explicitly mentions one or both of the subject or target, they are referred to as directed modalities and if neither of these was mentioned the deontic statement is referred to as a general modality. Further, obligation policies can be of two types, one representing high-level policies, typically stating a goal to be achieved and lower level policies which explicitly state actions of a subject;
5. The statement of other enterprise policies that reflect typical terms used by business and which can mostly be reduced on the core policies above; we refer to these as *compound normative policies*; examples are the concepts of rights, liabilities, commitment and responsibility;
6. The statement of policy-related actions that cover transfer of normative modalities between principals and agents, such as various forms of delegation; in this paper we call them *policy-transfer* (delegation) actions;
7. The specification of events that signify occurrence of violations of the policies or the events that signify situations that could potentially lead to some violations in future; in this paper we call them *attention events*;
8. Second-effect policies to be invoked in cases of violations of any of the above policies; we call these *reparation policies*, as per [8];

9. The expression of *force-majeure* conditions, explaining circumstances which are beyond control of either parties; these need to be mentioned, although they might not be able to invoke of any subsequent measures;
10. A number of *structuring constructs* introduced for the purposes of grouping these expressions and supporting reuse of the existing fragments, e.g. a contract clause can consist of a number of other legal statements.

The legalese structures above can be combined in various ways to reflect the specific circumstances that apply to the contract in question.

3. Example

This section introduces an example of a contract which will be used to highlight key concepts mentioned above and to illustrate other ideas as they are introduced in subsequent sections. The example is mostly based on our earlier scenario presented in [18].

Consider a contract between an Outback Water (OW) company that provides irrigation water to a number of different customers encompassing agriculture, industry (primarily mining and oil/gas extraction) and small towns in a central part of Australia. The OW operates storage lakes feeding into both open irrigation canals and pipelines, some of which are hundreds of kilometres in length. The OW irrigation system can be monitored remotely and the system includes many pumps and other assets that need constant monitoring and maintenance.

The OW has chosen a subcontractor to provide maintenance services for various assets in the water system. The terms of engagement are formalised by a legal contract that will govern interactions between the OW and the selected subcontractor. The contract provides legal binding between parties and each party has an obligation to satisfy the contract conditions by appropriately implementing their business processes during the term of the contract.

The agreed contract will thus specify conditions for maintenance subcontractors to service and maintain pumps and related equipment in various facilities operated by OW. It is expected that the relationship between OW and the subcontractor will have a long-term nature, but the contract will be initially for a year and will apply to a specific list of assets that are to be maintained.

This maintenance service contract is given next.

MAINTENANCE SERVICE CONTRACT

This Deed of Agreement is entered into as of the Effective Date identified below.

BETWEEN Outback Water (To be known as the OW) AND OZ Pumps (To be known as the Subcontractor)

WHEREAS (OW) enters into an agreement with (Subcontractor) for Maintenance Services, to be known as (Service) subject to the following terms and conditions:

1 Definitions and Interpretations

- 1.1 Price is a reference to the currency of the Australia unless otherwise stated.
- 1.2 This agreement is governed by Australia law and the parties hereby agree to submit to the jurisdiction of the Courts of the Queensland with respect to this agreement.
- 1.3 MTBF is Mean Time Between Failures and MTTR is Mean Time To Repair

2 Commencement and Completion

- 2.1 The commencement date is scheduled as January 30, 2006.
- 2.2 The completion date is scheduled as January 30, 2007.
- 2.3 The (OW) shall notify the (Subcontractor) of possibility of extension for 1 year by 3rd quarter of the contract

3 Service and QoS Delivery

- 3.1 The (Subcontractor) shall make its best efforts to ensure that the following QoS conditions are met:
 - not exceed the maximum asset down time on any one asset
 - not exceed the call-out time limit on more than 5% of emergencies in a month
 - average above the specified MTBF and below the MTTR over a monthThe maximum or minimum values are provided in schedule A of the contract.
- 3.2 (Subcontractor must inform (OW) within 24 hours of any event that might affect the ability to achieve the quality of service, e.g. resignation of subcontractor engineers, recurring problem with certain asset types
- 3.3. The (Subcontractor) shall not re-assign maintenance to another party, i.e. Sub-Subcontractor
- 3.4. The (OW) will provide access to all asset sites based on service requirements

4 Reports and notifications

- 4.1 The (Subcontractor) will submit monthly reports on all preventative maintenance activities and emergency events, including full timing details and description of problems and action taken, broken down into labour, parts and materials.
- 4.2 The (OW) will provide list of assets to be maintained, with clear instructions of the maintenance cycles required (asset lists are in a schedule to the contract, maintenance manuals are in associated paper or on-line documents)
- 4.3 The (OW) will provide clear MTBF and MTTR targets
- 4.4 The (OW) will provide feed back to the subcontractor any information received about problems with the water

supply, including emergencies reported by its customers within 24 hours

4.5 After each of the 1st and 2nd quarters, the (OW) will give guidance to the subcontractor on how any shortcomings in the service might be improved.

5 Payment

- 5.1 The (Subcontractor) shall submit monthly invoices to (OW) for services performed during that period
- 5.2. The (OW) shall make full payment of (Subcontractor) invoices within 30 days of receipt

6 Termination

- 6.1 The (OW) can terminate the contract after three QoS violations

The contract above is structured in terms of the clause groups according to the legalese structures above, namely in terms of:

- Definitional clauses (Clause Group 1), stating the external laws and policies that need to be obeyed, and explaining the meaning of some variables applicable to many assets, such as Mean Time Between Failures (MTBF) and Mean Time To Repair (MTTR).
- Commencement and completion clauses (Clause Group 2), defining the validity of the contract and conditions for its renewal
- Service delivery clauses, including Quality of Service (QoS) clauses (Clause Group 3), stating QoS and other service conditions that need to be provided by the subcontractor, in consideration for payment for their services; these clauses may also include conditions that apply to the OW company in ensuring that certain preconditions needed for the subcontractor are also met. This section of contract contains most of the core normative (or deontic) expressions that apply to the main service delivery by the Subcontractor.
- Reporting and notification clauses (Clause Group 4), referring to the obligations of each party in either making available some documents (by the OW) or producing reports during the maintenance process (by the subcontractor) as part of regular maintenance processes; these clauses would also describe the notifications that need to be sent by either party to indicate some potential problem or emergency situation.
- Payment clauses (Clause Group 5), stating different conditions needing to be fulfilled for payment processes
- Termination clauses (Clause Group 6) stating permissions under which parties can terminate the contract.

4. Contract formalism

The identification of several categories of legalese contract expression above is a step towards a more structured way of representing contracts. This structuring provides a better starting point for considering the use of several existing logics and/or normative systems theories to arrive at a formal (i.e. logic) representation of contracts. Example of such logics are modal, deontic and temporal logic, along with a recently proposed logic of violations [5]. In many respects the scope of normative systems overlaps with the scope of these logics. This is because they analyse those systems whose behaviour is covered by a set of norms, being statements of constraints on expected behaviour such as what behaviour is allowed, what behaviour must not occur, what behaviour is required and so on.

4.1 From legalese structures to deontic constraints

A detailed examination of the semantics of the contract conditions above reveals that their legal intent and form are closely related to various types of behavioural expressions that apply to the parties in contract. Most of these behavioural expressions are in the form of core normative policies (or deontic modalities), the core of which are obligations, permissions and prohibitions.

Deontic constraints express what parties to the contract are required to perform (obligations), what they are allowed to do (permissions), or what they are not allowed to do (prohibitions).

In general, deontic constraints apply to the behaviour of actors playing roles in some policy context (contract being a special case of a set of policies). This context is typically determined by the way the roles are configured into collaborative structures and by the pre-existing constraints from the environment of this context space. In the specific case of legal contracts, this context is the contract itself, consisting of various types of contract conditions as discussed above, all of which will apply to the two or more roles specified in the contract. These roles in turn could be played by different legal entities, and policies apply to these legal entities when they accept the constraints of the respective roles.

Deontic constraints are structured in terms of:

- *constraints* on behaviour typically expressed in terms of events and relationship between events, closely reflecting the specification of various policies as part of contract conditions. The event (relationship) constraint describes the expected behaviour of the party in question. Events describe the actions of the parties that are subject to the

constraint, but also other occurrences such as expiration of deadlines, or actions of other parties.

- the specification of the *modality* that applies to the party, e.g. an obligation, permission or prohibition.
- a *subject* to which modality and behavioural constraints apply. A deontic modality may explicitly identify a *beneficiary* (or target) from the subject side in a modality expression, although the beneficiary may be implied from the contract.
- *Triggering conditions* which signify that normative policies are in force; these can be temporal events, but also other events, such as violation of other policies.

The semantics of modality can be expressed in terms of a set of observations chosen to determine whether the corresponding policy is satisfied or not. For example, the prohibition modality can be checked through an observation of occurrence(s) which are contrary to the prohibition statement.

Note that the style of constraints in deontic expressions is different from the style of expressions in traditional business process specification languages. The focus of the process languages is on the description of control and data flows to support repeatable and automatable activities. A lesser emphasis is given to the detailed description of various types of associations between roles and process activities, and no emphasis is normally given to the organizational consequences of the violations of agreed behaviour.

Recent business process specifications, however, are increasingly adopting an event-based style of behaviour which better suits needs for more flexibility such as real-time process adaptations. Events can be used to determine process flows in real-time as a result of the outcome of a specific task or to generate notifications of specific events (e.g. natural events or temporal events) or the recognition of an emerging economic or market trend.

We believe that the event-oriented style of expression is suitable for the expression of deontic constraints and the emerging business processes specifications centred on the flow of events can be exploited as a mechanism to facilitate the derivation of business processes compliant with contract conditions.

4.2 From deontic constraints to contract formalism

This section presents a formalism for describing deontic constraints, based on commonly accepted principles of deontic logic, but extended with the formalism for treating violations. The formalism, called the Formal Contract

Language (FCL) was first presented in [6], and is based on the logic of violations developed by Governatori and Rotolo [5], used to represent violation in contracts.

4.2.1. Formalising deontic constraints

Deontic logic extends first order logic with the modal operators O , P and F denoting obligations, permissions and prohibitions. The modal operators satisfy the following deontic relationships:

$$OA \equiv \neg P\neg A \quad \neg O\neg A \equiv PA \quad O\neg A \equiv FA \quad \neg PA \equiv FA.$$

The modal operators also satisfy the following relationship $OA \rightarrow PA$, meaning that if A is obligatory, then A is permitted. This relationship can be used to ensure checking of the internal consistency of the obligations in contracts: it is possible to execute obligations without doing something that is forbidden.

As stated before, the deontic constraints in contracts apply to the roles involved in the contract, specifically to the subject to which constraints apply, and possibly including the target or beneficiary. In case of obligation this can be denoted using the expression $O_s A$ to be read as ‘ s has the obligation to do A ’, or ‘ A is obligatory for s ’. If a beneficiary is mentioned, the expression is extended, i.e. $O_{s,b}$ to be read as ‘ s has the obligation to do A with respect to b ’.

In case of certain breaches of policies in contract by actors playing the roles in a contract, special policies may be included to express the respective obligations for these actors. These policies can vary from pecuniary penalties to the termination of the contract itself. In deontic logic, this type of expression, namely the activation of certain obligations in case of other obligations being violated, is referred to as contrary-to-duty obligations (CTD) or reparation obligations. The reparation obligations are in force only when normative violations occur and are meant to ‘repair’ violations of primary obligations. Thus a reparation policy is a conditional obligation arising in response to a violation, where a violation is signalled by an unfulfilled obligation. The expression of violation conditions and the reparation obligations is an important requirement for formalising contracts, design subsequent business processes to minimise or deal with such violations and also to support the monitoring of business contracts

There are a number of different approaches in deontic logic to formalise CTD obligations, but in this paper we use a simple logic of violation, to avoid danger of logical paradoxes that some other approaches may involve [3]. This logic is also suitable to model chains of violations and is described next.

4.2.2. Formalising violations of deontic constraints

In addition to using the logic based approach to specifying core deontic constraints, we thus provide a simple logic of violation.

The violation expression consists of the primary obligation, its violation conditions, an obligation generated upon the violation condition occurs, and this can recursively be iterated, until the final condition is reached. This final condition is one which cannot be violated and this it is to be a permission. We introduce the non-boolean connective \otimes , whose interpretation is such that $OA \otimes OB$ is read as “ OB is the reparation of the violation of OA ”. In other words the interpretation of $OA \otimes OB$, is that A is obligatory, but if the obligation OA is not fulfilled (i.e., when $\neg A$ is the case, thus resulting in a violation of the obligation OA), then the obligation OB is activated and becomes in force until it is satisfied or violated. In the latter case a new obligation may be activated, followed by others in chain, as appropriate.

4.2.3. Formal Contract Language (FCL)

The FCL brings together two set of atomic symbols: a numerable set of propositional letters p, q, r, \dots , intended to represent the state variables of a contract and a numerable set of event symbols $\alpha, \beta, \gamma, \dots$ corresponding to the relevant events in a contract. Formulas of the logic are constructed using the deontic operators O , P , negation \neg and the non-boolean connective \otimes (for the reparation operator).

The formulas of FCL will be constructed in two steps according to the following formation rules:

- every propositional letter is a literal;
- every event symbol is a literal;
- the negation of a literal is a literal;
- if X is a deontic operator and l is a literal then Xl and $\neg Xl$ are modal literals.

We now use the following set of formation rules to introduce \otimes -expressions, i.e., the formulas used to encode chains of obligations and violations:

- every modal literal is an \otimes -expression;
- if Ol_1, \dots, Ol_n are modal literals and l_{n+1} is a literal, then $Ol_1 \otimes \dots \otimes Ol_n$ and $Ol_1 \otimes \dots \otimes Ol_n \otimes Pl_{n+1}$ are \otimes -expressions.

Each condition or policy of a contract is represented by a rule in FCL, where a rule is an expression

$$r:A_1, \dots, A_n \vdash C$$

r is the name/id of the policy, A_1, \dots, A_n , (the *antecedent* of

the rule), is the set of the premises of the rule (alternatively it can be understood as the conjunction of all the literals in it) and C is the conclusion of the rule.

Each A_i is either a literal or a modal literal and C is an \otimes -expression. The meaning of a rule is that the normative position (obligation, permission, prohibition) represented by the conclusion of the rule is in force when all the premises of the rule hold. For example, the clause 3.2 (“Subcontractor must inform OW within 24 hours of any event that might affect the ability to achieve the quality of service, e.g. resignation of subcontractor engineers, recurring problem with certain asset types”) can be represented as

$$r: \alpha \vdash O_{Sub,OW} \beta$$

where α is the event symbol corresponding to the event “an event that might affect the ability to achieve the quality of service”, and β is the event symbol corresponding to the action “inform OW within 24hrs after event α ”. The policy is activated, i.e., the subcontractor is obliged to inform the OW within 24 hrs, when the event α has occurred.

Another example shows how a violation condition and reparation policies can be represented (although this is not explicitly shown in the contract):

MissingClause: InvoiceReceipt \vdash O_{OW} *PayWithin30Days*

$$\otimes P_{SubContractor} \textit{RequestPayWithInterest}$$

So, upon receipt of an invoice, the OW is obligated to pay it within 30days, but if they fail to do so, the Subcontractor is permitted to request payment with interest.

The connective \otimes permits combining primary and CTD obligations into unique regulations. The operator \otimes is such that $\neg \neg A \equiv A$ for any formula A and enjoys the properties of associativity $A \otimes (B \otimes C) \equiv (A \otimes B) \otimes C$, duplication and contraction on the right, $A \otimes B \otimes A \equiv A \otimes B$. The right-hand side of the last equivalence states that B is the reparation of the violation of the obligation A . That is, B is in force when $\neg A$ is the case. For the left-hand side we have that, as before, a violation of A , i.e., $\neg A$, generates a reparation obligation B , and then the violation of B can be repaired by A . However, this is not possible since we already have $\neg A$.

The formation rules for \otimes -expressions allows a permission to occur only at the end of such expression. This is due to fact that a permission can be used a reparation of a violation, but it is not possible to have violation of a permission. Sometimes contracts contain other mutual normative positions such as delegation, empowerment, rights and so. Often these notions can be effectively

represented in terms of complex combinations of directed obligations and permissions. Hence violations to such complex notions result in violations to the obligations describing such notions.

4.2.4. Example

A complete representation of the maintenance contract in FCL is given below. Notice superscripts H, and D, next to several modalities. ‘H’ denotes a high-level policy that needs further refinement and ‘D’ denotes an action of delegation.

2.3: *3rdQuarterEnd,ExtensionYes* \vdash

$$O_{OW,Sub} \textit{ExtensionNotification}$$

3.1: *ContractStart* \vdash $O_{Sub}^H \textit{EnsureBestQoS}$

3.2: *OoSProblemEvent* \vdash $O_{Sub,OW} \textit{InformWithin24hrs}$

3.3: *ContractStart* \vdash $F_{Sub,Sub-Sub}^D \textit{AssignMaintenance}$

3.4: *AccessSiteRequest* \vdash $O_{OW,Sub}^H \textit{ProvideAccess}$

4.1: *ContractStart,BeginMonth* \vdash $O_{Sub,OW} \textit{SubmitMonthlyReport}$

4.2: *ContractStart* \vdash $O_{OW}^H \textit{ProvideListOfAssets}$

4.3: *ContractStart* \vdash $O_{OW}^H \textit{ProvideMTBFandMTTRTargets}$

4.4: *ProblemOrEmergency* \vdash $O_{OW,Sub} \textit{ProvideFeedback}$

4.5: *EndOfFirstQuarter* \vdash $O_{OW,Sub} \textit{GiveGuidance};$

$$\textit{EndOfSecondQuarter} \vdash O_{OW,Sub} \textit{GiveGuidance};$$

5.1: *BeginMonth* \vdash $O_{Sub,OW} \textit{SubmitMonthlyInvoice}$

5.2: *InvoiceReceipt* \vdash $O_{OW,Sub} \textit{FullPaymentWithin30days}$

6.1: *ThirdQoSViolation* \vdash $P_{OW} \textit{TerminateContract}$

Note that the FCL version above does not include predicates for the first and second sections of the contract. This form of contract provides a formalised version which can be used for various purposes, such as determining a consistency of an overall contract as also discussed in a related work in [15], but also as a starting point in assisting domain experts in constructing business processes that comply with this contract. To support this construction activity it is useful to have a methodology which would provide a systematic guide to the experts in deriving contract compliant processes, as presented next.

5. From contract to business processes

In this section we describe how a business contract stated in the FCL can be translated into business processes that are aligned with contract conditions. We will attempt to refine contract description into increasing level of detail, starting from mere cross-organisational interactions and

progressively refining it into internal processes. We will again use our contract example to illustrate the steps involved. Note that the example is not elaborated in full detail and many points were omitted for brevity.

We use BPMN notation [25] as our target process description language. We chose BPMN because this notation is suited to support business analysts and business process modellers and we believe that initial translation from business contracts into business processes needs to be undertaken by these specialists. Once this activity is performed and business process model is produced, it is up to the business process engine specialists to describe specific mapping from a BPMN model into an executable form of a process. Indeed, the BPMN specification comes with one such mapping, i.e. the mappings onto BPEL [23]. Many other mappings could be also possible.

In addition, BPMN provides an expressive, event-oriented approach to specifying contracts, which is a style of expression that is suitable for the description of business contracts and without imposing block-structure limitation adopted as part of BPEL specification.

Finally, BPMN provides quite suitable environment for supporting interactions defined by business contracts because it allows for support of process descriptions that range from internal processes to complex cross-organisational processes, involving several parties. Namely, using BPMN it is possible to describe *abstract (or public) processes* between parties, where the focus is on exchange of messages between them. In this case it is possible to either abstract away the *internal processes* of both parties or to abstract away the internal processes of the other partner only, depending on the circumstances. In the last case, the aim is to provide description of interactions from the point of view of one party. In the most detailed form, BPMN allows for the description of internal processes for all parties, along with the messages between them. In BPMN terms this is called a collaboration (global) process.

In what follows, we will use all these process variants as facilities to help the progressive refinement from the contract onto processes.

5.1 From FCL to abstract processes.

Consider the FCL representation of the contract given in the previous section. Our first step is to translate this contact representation into two abstract processes, corresponding to each of the parties and to identify the messages that can be consequences of the corresponding obligations. So, we first identify those obligations between the parties which explicitly state the subject and the target (or beneficiary) of the obligation. This is because many of such obligations will imply some form of message exchange between the parties. For example, the following

statement $QoSProblemEvent \vdash O_{Sub,OW} InformWithin24hrs$, implies that a Subcontractor will need to send a message to the OW to inform them (about some problem). This should also happen within 24hrs of the problem occurrence. This last statement is included as a BPMN Text Annotation. In fact, we will use BPMN text annotations (see Figure 1) as a way of stating deontic constraints on the process diagram. Note that we use the BMN annotations to show triggering conditions for obligations, such as beginning of month trigger for sending monthly report. Thus, they are used to denote the conclusion and antecedents of the FCL logical expression respectively.

Figure 1 depicts two abstract processes between the parties (shown as BPMN pools) with several messages that were identified from the FCL statements. Note that the figure also shows the third party (sub-subcontractor) and the prohibition condition on the assignment of maintenance to this party by the subcontractor.

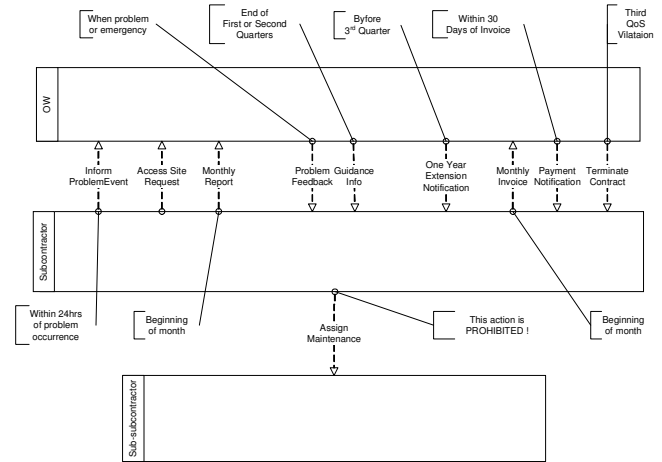


Figure 1: Abstract processes – message exchange

Notice that not all the mutual obligations from the FCL expressions could help in identifying such messages. This is in particular true for those obligations that are stated at a higher level of abstraction. For example, the expression $AccessSiteRequest \vdash O_{OW, Sub} ProvideAccess$, states an obligation on the OW to ensure the access to the sites (upon Subcontractor’s request) but this obligation does not necessary result in a message sent to the Subcontractor. Rather it should result in an internal task within the OW role which provides physical access to the sites. The fulfilment of this obligation can be only checked if some electronic sensors can be activated (or in much simpler cases, through the manual action of passing keys to the subcontractor).

Note that the messages identified do not necessarily need to represent a unique or even best solution for the interactions between the parties. However, they can be a good starting point for subsequent redesign as needed.

5.2 From an abstract process to a private process

Once public processes are identified, it is possible to provide a more detailed structure of internal processes of the parties, including the way messages (identified in the abstract processes) are generated or consumed by the tasks within them. This section will provide an example of how the internal processes of the OW can be designed, including the interactions with the subcontractor, so that contract conditions are satisfied. The example provides an OW centric view of the processes.

The FCL expressions of the contract can again be used in this exercise. To this end, we have identified several translation rules, as follows.

First, it is useful to look for the antecedents of each of the deontic expressions that indicate actions of the OW, i.e. which are of the form $O_{OW...}$ (i.e. they may or may not include target role). This is because antecedents represent certain occurrences, e.g. message arrivals, deadline expirations or state changes. In terms of BPMN constructs these would be start events or intermediate events, each of these of any of the BPMN available types. These triggers can then be used to start corresponding tasks of the internal processes. The statement $ContractStart \vdash O^H_{OW} ProvideListofAssets$, for example, means that when contract is activated, the OW is obliged to provide list of assets, which can be modelled as the respective task. In this case it is not possible to infer any interactions involving the Subcontractor role, so the ProvideListofAsset task (Figure 2) does not involve any messages to be sent to the Subcontractor. Once the contract is started, it is certainly possible for the Subcontractor to access list of assets but this detail was not shown in the example.

Second, some of the tasks activated by the events, can in turn generate messages to be sent to the other party. We observed that this would be most likely the case if the target role was specified as part of the deontic modality. An example of such a rule is in the statement $Problem/Emergency \vdash O_{OW,Sub} ProvideFeedback$.

Third, some obligations involve deadlines, and for this the triggering event should activate the tasks that need to be performed and also the deadline event, which if triggered before the completion of the activity, will signify violation of the obligation. An example of such a rule is $InvoiceReceipt \vdash O_{OW,Sub} FullPaymentWithin30days$. Figure 2 shows an example of the internal business process that is a

refinement from that shown in Figure 1. Again, many details are not further elaborated in the example, e.g. no compensation is provided for the payment violation.

It is important to note that the rules identified above can be considered only as a guiding mechanism for identifying some patterns for structuring internal activities. There are of course many possible designs for internal business processes and this is where it may be useful to consider further heuristics. Some initial considerations for such heuristics are presented in [18].

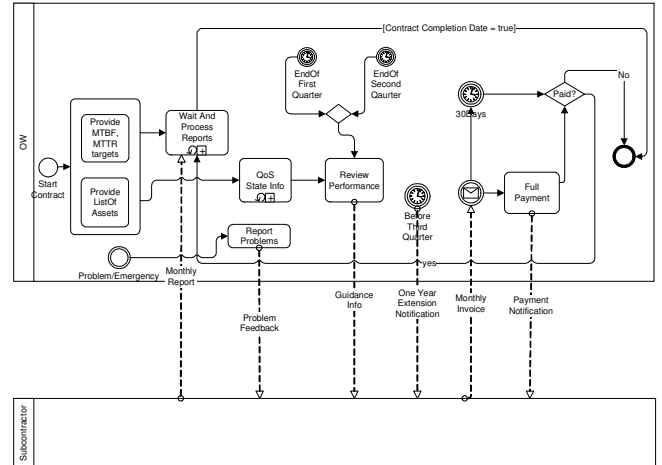


Figure 2: Outback Water's view – their internal process and Subcontractor's abstract process

5.3 From private processes to collaboration processes

The final step in the process derivation is the one in which both parties' business processes can be designed. This is similar to the step above, in that the same translation rules would apply to the Subcontractor role. The resultant BPMN collaboration business process is shown in Figure 3.

Again, it is important to state that this is one possible realisation solution. There can be many other realisations, which may reflect various design choices, various domain requirements as well as performance and quality measures.

5.4 Discussion points

This section summarises two categories of open issues identified from the above exercise and which would need to be addressed in future.

5.4.1. From process description to process implementation

In the discussions so far we have been concerned with business level specification of processes. It is precisely for

these reasons that we chose the BPMN notation – because it was developed for the high level descriptions of processes. However, in order to provide significant level of business process automation one relies on the availability of business process engines that provide an automation of business processes. In the case of BPMN the obvious engines of choice are those that implement BPEL semantics, because the BPMN specification provides mapping to the BPEL. However BPEL semantics provides a relatively restricted support for the execution of complex events along the lines of those that are for example described in [13]. It would be interesting to consider some other process engines that are more distributed in nature and that implement event-centric semantics such as those that follow the WS-CDL [26] approach or an engine proposed by Berry [1] [2].

addressed in this paper, one is concerned with how to design new processes based on the existing or new contract. In yet another cases one needs to determine whether there is sufficient trust between organisations to decide whether some third-party monitoring mechanisms need to be employed. All these issues need to be discussed and addressed when considering various deployment and operational issues associated with processes governed by business contracts.

6. Related Work

A number of authors have investigated the problem of formal representation of contracts or deontic foundational elements. This includes early work by Lee [11], a number of related efforts such as work by Farrell et al [4], based on earlier event calculus of Kowalski et al. [10], and recently Governatori, who proposed deontic and defeasible logic to represent contracts [8] and Linington and Milosevic [12]. In addition, there are many references that deal with the specification of business processes, and we only mention here standard developments such as BPEL [23], and WS-CDL [26].

To the best of our knowledge there are only few references that consider relationships between contracts and business processes, namely [22][2], while the area of deriving contract-compliant business processes has not been investigated in depth so far, apart from some initial ideas presented in [18]. We believe that this and an accompanying paper [7], embark on a new research theme that deserves significant future investigation.

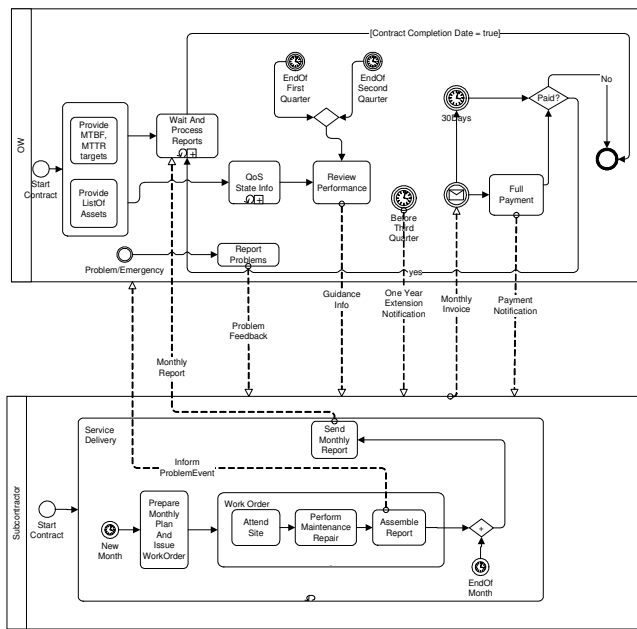


Figure 3: Collaboration processes

5.4.2. A methodology for deriving compliant processes

Our approach for the progressive development of collaborative business processes from the legalese form of contract is part of a more general methodology which we have recently proposed [16]. The aim of the methodology is to serve as a general guiding tool for business process modellers in their activities of developing processes that are compliant with contracts. In doing so, they need to consider various business setups of organisations and various pre-existing procedures, policies and cultures. In some cases for example, when there are existing processes in place, there needs to be checked whether these processes are compliant with business contracts. In other cases, such as those

7. Conclusions and Future Work

This paper provides an approach for progressively refining legalese form of contract into business activities and processes that are compliant with the contract. We have found that the representation of contract in a structured form such as the one based on a logic formalism, provides a more direct way in the identification of messages to be exchanged between parties as driven by their mutual obligations – as opposed by using straight legalese version. We have also found that FCL form can help in identifying some fragments of internal processes, typically the tasks that are activated by some triggering events and tasks which need to be involved in exchange of messages needed as part of fulfilment of mutual obligations. However, it is difficult to arrive at a detailed set of rules for the construction of internal processes – which in a way is to be expected as there may be many ways how processes can be realised to implement contract conditions. Some heuristics can be applied in a similar way as was proposed in [18] and

we expect that more of such heuristics will be developed over time as the patterns of use are identified and recorded. In fact, one of our future research directions is to study various such cases and to provide a richer set of heuristics, which are likely to be industry domain dependent.

We have also found that BPMN provides the style of process expression that fits well the business contracts domain. This is because BPMN provides facilities to represent both internal and cross-organisational interactions. In addition, BPMN comes with a number of event-centric modelling concepts which are a good fit for the event-oriented style of deontic constraints and all this makes it a good candidate for the progressive refinement approach as presented in the paper.

References

- [1] A. Berry. Describing and Supporting Complex Interactions in Distributed Systems. PhD thesis, University of Queensland, 2002.
- [2] A. Berry and Z. Milosevic. Extending choreography with contract constraints. *Int. J. of Cooperative Inf. Syst.*, 14, 2005.
- [3] J. Carmo and A.J.I. Jones. Deontic logic and contrary to duties. In D.M. Gabbay and F. Guenther, editors, *Handbook of Philosophical Logic.*, volume 8: 265–343. Kluwer, 2002.
- [4] A.D.H. Farrell, M.J. Sergot, M. Sallé, and C. Bartolini. Using the event calculus for tracking the normative state of contracts, *International Journal of Cooperative Information Systems*, Vol. 14, Nos. 2-3 (2005) 99-129.
- [5] G. Governatori and A. Rotolo. Logic of Violations: A GENTZEN System for Reasoning with Contrary-To-Duty Obligations. *Australasian Journal of Logic*, 4, 193-215, 2006.
- [6] G. Governatori, Z. Milosevic, *Dealing with contract violations: formalism and domain specific language* Proc. of the 9th International Conference on Enterprise Distributed Object Computing, Enschede, The Netherlands, Sep. 2005.
- [7] G. Governatori, Z. Milosevic, S. Sadiq, Compliance checking between business processes and business contracts, Proc. 10th IEEE EDOC Conference, 16-20 Oct, 2006.
- [8] G. Governatori. Representing business contracts in RuleML. *International Journal of Cooperative Information Systems*, 14, 2-3: 181-216, 2005.
- [9] A. J. I. Jones, *On the Characterisation of Law and Computer Systems: The Normative Systems Perspective* (1993)
- [10] R.Kowalski and M.Sergot. A Logic-Based Calculus of Events. *New Generation Computing*, 4:67--95, 1986.
- [11] R.M. Lee. A logic model for electronic contracting. *Decision Support Systems*, 4:27–44, 1988.
- [12] P. Linington, Z. Milosevic, J. Cole, S. Gibson, S. Kulkarni, S. Neal, *A unified behavioural model and a contract language for extended enterprise*, *Data Knowledge and Engineering Journal*, Elsevier Science, 51 (2004), p. 5-29.
- [13] D. Luckham, *The Power of Events*, Addison-Wesley, 2002
- [14] D. C. Ma, Maria E. Orlowska, Shazia W. Sadiq *Formal Considerations of Rule-Based Messaging for Business Process Integration*, Special Issue of Cybernetics and Systems: An International Journal, Vol 37/2 (2006).
- [15] Z. Milosevic, G. Dromey, *On Expressing and Monitoring Behaviour in Contracts*, EDOC2002 Conference, Lausanne, Switzerland.
- [16] Z. Milosevic, S. Sadiq, M. Orlowska, Towards a methodology for deriving contract-compliant business processes, Proc. 4th Int. Conference on Business Process Management, 5-7 Sept, 2006.
- [17] M. J. Sergot, A computational theory of normative positions, *ACM Trans. Comput. Log.*, p. Vol 2, No 4, 581-622, 2001.
- [18] R. Tag, Z. Milosevic, S. Gibson, S. Kulkarni, Supporting Contract Execution through Recommended Workflows, DEXA04 Conference., Zaragoza, Spain, Sept. 2004.
- [19] ITU-T Rec. X.902 | ISO/IEC 10746-2: Foundations, RM-ODP
- [20] ISO/IEC IS 15414, *Open Distributed Processing-Enterprise Language*, 2002.
- [21] Shazia Sadiq, Wasim Sadiq, Maria Orlowska (2005) *A Framework for Constraint Specification and Validation in Flexible Workflows*. Information Systems, Elsevier Science. Volume 30. Issue 5 pp. 349-378
- [22] W-Jan van den Heuvel, H. Weigand, *Cross-Organisational Workflow Integration using Contracts*, *Decision Support Systems*, 33(3): p. 247-265
- [23] www-106.ibm.com/developerworks/library/ws-bpel/
- [24] J. Cole, J. Derrick, Z. Milosevic, K. Raymond, *Author obliged to submit a paper before July 4: Policies in Enterprise Specification*, Proc. Policy01 workshop.
- [25] Business Process Management Notation (BPMN) www.bpmn.org
- [26] Web Services Choreography Description Language, 1.0, www.w3.org/TR/2004/WD-ws-cdl-10-20041217/