Supporting Contract Execution through Recommended Workflows

Roger Tagg[1], Zoran Milosevic[2], Sachin Kulkarni[2], Simon Gibson[2]

[1]University of South Australia, School of Computer and Information Science
Mawson Lakes, SA 5095, Australia
Roger.Tagg @unisa.edu.au

[2]CRC for Enterprise Distributed Systems Technology (DSTC)
Level 7, GP South, University of Queensland,
Brisbane, Q 4072, Australia
[zoran | sachink | sgibson] @dstc.edu.au

Abstract. This paper extends our previous research on e-contracts by investigating the problem of deriving business process specifications from business contracts. The aim here is to reduce the risk of behaviour leading to contract violations by encouraging the parties to a contract to follow execution paths that satisfy the policies in the contract. Our current contract monitoring prototype provides run-time checking of policies in contracts. If this system was linked to workflow systems that automate the associated business processes in the contract parties, a finer grain of control and early warning could be provided. We use an example contract to illustrate the different views and the problems of deriving business processes from contracts. We propose a set of heuristics that can be used to facilitate this derivation.

1 Introduction

Most business transactions are based on a contract of some form. However, in most of today's organizations, including their IT systems support, contracts are treated as isolated entities, far removed from their essential role as a governance mechanism for business transactions. This can lead to many problems, including the failures to detect in timely manner and react to business transaction events that could result in contract violations or regulatory non-compliance.

As a result, several vendors have begun offering self-standing enterprise contract management software [2][3][4][6]. These systems consist mostly of a number of pre-built software components and modules that can be deployed to specific contract requirements. However our earlier work [7][8] suggests that a more generic approach is needed that more closely reflects contract semantics, in particular in terms of the governance role. This means adopting higher level modelling concepts that directly reflect the business language of a contract and the policies that express constraints on the parties involved. Examples of these are obligations, permissions, prohibitions, authorisation etc. This implies a need for specialised languages to express these contract semantics.

In previous papers we presented our language-based solution for the expression of contract semantics in a way suitable for the automation of contract monitoring [7][8]. This language, Business Contract Language (BCL), is used to specify monitoring conditions that can be then interpreted by a contract engine. This paper investigates to what extent the semantics of contracts can be used to infer business processes which, if followed by the trading partners, would help reduce the risks associated with contract non-compliance. Such processes may be able to provide a finer grain of monitoring to complement that achievable through the BCL alone. We refer to these business processes as ‘recommended’ business processes - to reflect the fact that they can only be a guiding facility for managing activities related to contracts, and that the different parties’ organisational policies and cultures may impose limitations on how far business processes can be structured or how strictly they should be mandated.

The paper begins with a motivating example of a water supply maintenance situation that could benefit from the automation of contract related activities. In the subsequent section we describe how BCL can be used to express monitoring conditions for this system. We then present a model of the same contract seen as a business process, following which we discuss the problems associated with the translation of contract conditions into a business process, referring to the lessons we learned in trying this. Next we present a proposed approach for derivation of business process from a contract, based on heuristics related to different types of contract and clause. This is followed with an overview of related work. The paper concludes with a summary of areas for future research and a brief conclusion.

2 Motivating Example

In this fictitious example, Outback Water (OW) is a utility organisation that provides water to agriculture, industry (primarily mining and oil/gas extraction) and small towns in certain central parts of Australia. It operates some storage lakes and both open irrigation canals and pipelines.

OW makes contracts with maintenance subcontractors for servicing and maintaining its assets (e.g. pumps, valves, etc) located in its facilities in various areas. The contracts are of a repetitive and potentially continuing nature. Contracts are for a year and cover a list of assets that have to be maintained.

From the point of view of OW's service to its customers, its Quality of Service (QoS) objective is to ensure that the average and worst loss of service to any customer is within a stated maximum number of days. OW uses MTBF (Mean Time Between Failures) and MTTR (Mean Time To Repair) as its main measures of asset availability.

The contract is summarised in the following table:

	Obligations: subcontractor
s1	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 month The maximum or minimum values are provided in a schedule to the contract.
s2	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, replacement parts and materials.
s3	Inform the asset operator 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
s4	Submit monthly invoices of money due to the subcontractor.
	Obligations: asset operator (OW)
ow1	Pay the subcontractor on monthly invoice within 30 days.
ow2	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)
ow3	Provide clear MTBF and MTTR targets
ow4	Feed back to the subcontractor any information received about problems with the water supply, including emergencies reported by its customers within 24 hours
ow5	Give the subcontractor access to all the asset sites.
ow6	After each of the 1st and 2nd quarters, give guidance to the subcontractor on how any shortcomings in the service might be improved.
	Permissions: asset operator
ow7	May take on an additional subcontractor in the event that the appointed subcontractor is having difficulty in meeting the QoS targets.
ow8	After the 3rd quarter of the contract, may give the subcontractor notice to quit or to be asked to continue for another year
	Prohibitions: subcontractor
s5	Not allowed to re-assign maintenance tasks to a sub-sub-contractor.

Table 1 Representation of the contract between Outback Water and a Maintenance Subcontractor

3. Expressing Contract Monitoring Conditions Using BCL

BCL is a language developed specifically for the purpose of monitoring behaviour of parties involved in business contracts. Key concepts of this language are [7]:

- Community – a container for roles and their relationships in a cross-enterprise arrangement. A Community may be instantiated from a Community Template.
- Policy – general constraints on behaviour for a community, expressed as permissions, prohibitions, obligations or authority. In combination these make up the terms of the contract.
- State – information containing the value of variables relevant to the community; may change in respect to events or time.

- Event – any significant occurrence generated by the parties to the contract, an external source, or a temporal condition.
- Event Pattern – an expression relating two or more events that can be detected and used in checking compliance with a policy (see [5] for similar concepts).

The BCL concepts introduced above can be used to express a model for a specific business contract, such as that between OW and a sub-contractor. These models are then interpreted by a contract engine, to enable evaluation of actual contract execution versus agreed contract terms. This evaluation requires access to the contract-related data, events and states as they change during the execution of business processes.

In the water supply example there are a number of clauses that are suitable for run time monitoring, but for brevity we choose only the clauses under s1 in Table 1. The contract should have a schedule describing each asset and the availability objectives associated with that asset. As part of the contract the sub-contractor must submit a monthly report outlining all tasks performed whether routine maintenance or emergency repairs. This report should contain basic details that will be used to calculate adherence to the QoS metrics. The report will need to identify the asset, and contain a description of the task, the start time and the finish time. In addition to this, for any emergency task the actual time of failure should be indicated.

To begin with, an overall community should be defined for the entire contract. In this example, each asset has some of its own monitoring behaviour and so each asset can be seen as being a sub-community of the overall community. Figure 1 outlines some of the required constructs. The full lines indicate event flow and the broken lines indicate data flow.

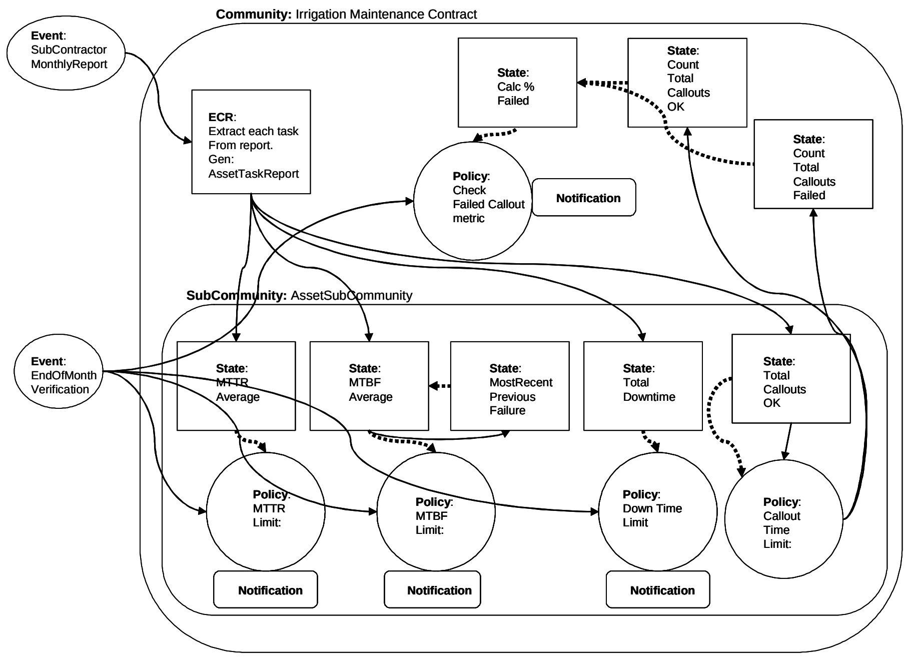
Community: Irrigation Maintenance Contract
Event: SubContractor MonthlyReport
ECR: Extract each task From report. Gen: AssetTaskReport
State: Calc % Failed
State: Count Total Callouts OK
State: Count Total Callouts Failed
Policy: Check Failed Callout metric
Notification
SubCommunity: AssetSubCommunity
Event: EndOfMonth Verification
State: MTTR Average
State: MTBF Average
State: MostRecent Previous Failure
State: Total Downtime
State: Total Callouts OK
Policy: MTTR Limit:
Policy: MTBF Limit:
Policy: Down Time Limit
Policy: Callout Time Limit:
Notification
Notification
Notification

Figure 1: BCL concepts for part of a water supply maintenance contract

The parent community template defines an event creation rule (ECR) that extracts each task from a SubContractorMonthlyReport event and passes the task as an event to the associated sub-community instance. There are a number of States that collect these events and perform an associated calculation. Policies can then use the value of these states as well as values defined in the contract schedule to determine whether constraints have been met or violated. The trigger for evaluating these Policies is when an event indicating the EndOfMonth is received. It should be noted that a Guard is placed on most of the Policies declaring that the SubContractorMonthlyReport must be received prior to the EndOfMonth event. Additional Policies could be used to enforce this behaviour but is not shown here for reasons of brevity. Notifications are used to notify human users that a violation has occurred. Table 2 provides BCL syntax for the specification of one fragment of the contract, namely asset downtime state, policies and notifications.

EventCreationRule: AssetTaskReport GenerateOn: SubContractorMonthlyReport ContentToGenerate: Loop through report and create an AssetTaskReport for each task	State: downtimeState On event: AssetTaskReport If its an emergency task calculate the total downtime and add it to total total = Total + (FinishDateTime - TimeOfFailure)
Policy: downtimeLimit Guard: SubContractorMonthlyReport On event: EndOfMonthVerification Checks if downtimeState value is greater than the defined value of MaxAssetDowntime metric	Notification: downtimeLimitNotification On event: downtimeLimitPolicyEvaluationEvent

Table 2: BCL syntax examples for asset downtime specifications

Note that although we use an event-driven approach for the monitoring, these are infrequent events and this contract can be characterised as a system-state invariant contract. For more information about various characteristics of contract clauses, see their classification in section 6.

4 Deriving Business Processes From Contracts

A contract exists for a limited purpose – to express constraints on the behaviour of signatories with the aim of achieving their individual objectives in the presence of uncertainty. It does not attempt to prescribe the “how” of a business process; rather, it is limited to what conditions that need to be satisfied for the parties to comply with the contract. In practice, in order to ensure that a contract is satisfied, the parties – separately and together – must have processes (which may be in part informal) for meeting their obligations under the contract.

A formal workflow might be able to add the following to contract management:

- Guidance to the human participants in each contract party, particularly where the staff involved are not experienced in the pattern of collaboration – answering “what do we do next?”
- Auditing: answering “who actually performed the constituent activities?” in case of a breach in the contract

- Early warning: if activities in either party are behind schedule at a detailed level, it may be possible to re-assign resources to remedy this.

A business process will generally be at a finer level of detail than the contract clauses. When trying to derive business processes for the water supply example (see Figure 2), we needed to introduce a number of assumptions which were not explicitly stated in the contract. Examples of such "introduced" behaviour are activities such as Issue Work Order, Amend Work Order and Prepare Resolution plan. Another example is the activity: "the sub-contractor can be given notice" which implies that the asset owner must review performance against the contract. This finding is in line with our previous experience reported in [1] and is a result of the fact that the contract only states a broad framework of possible executions and that many behaviour trajectories can satisfy the policies stated in the contract.

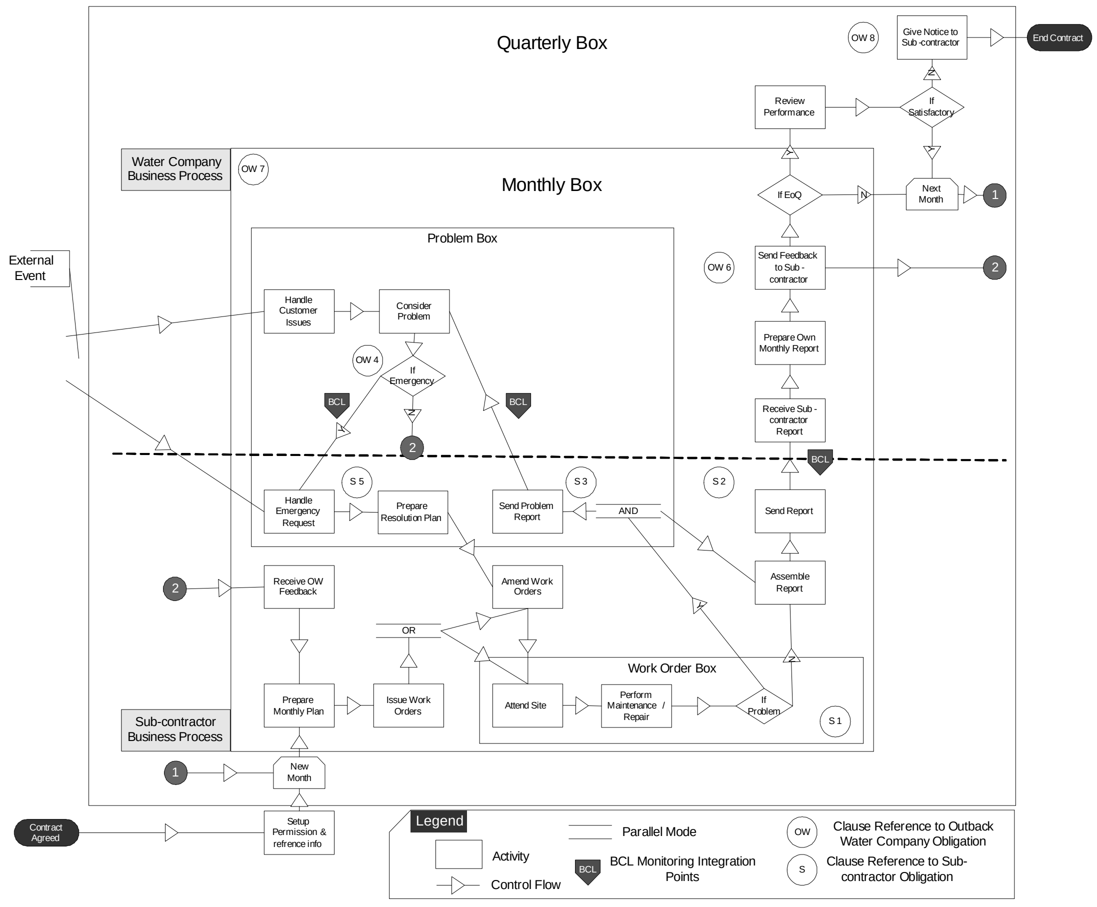

Figure 2: A business process for the water supply maintenance example

Therefore, the business process in Figure 2 is one possible way to satisfy the policies in this contract. The example also shows the separation of processes across OW and the sub-contractor; two levels of nesting for the month and quarter periods; and two repeating activities (problem and work order sub-processes). In addition it shows a need for supporting external events (not originating from within the workflow).

Once this process is in place, it would be then possible to use the events generated through the corresponding workflow system as input to a contract monitoring system, such as one that utilizes BCL and the underlying interpreter engine [7]. This figure highlights possible points where contract monitoring conditions can be applied (shown using the black BCL symbol).

5. Discussion Points

There are a number of considerations that we faced when working with this example. Some of the key questions and possible solutions are outlined in this section.

5.1 The Feasibility of Deducing Business Processes

How many activities are deducible by understanding the nature of the contract clauses? How much dependency between activities is explicitly stated in a contract?

In our analysis we found that, although we started from the same natural language specification as described above, the questions we had to ask for the two models, namely contract monitoring and workflow, were quite different. Several of the activities that the subcontractor should perform were not mentioned in the contract. We can deduce, for example, that the subcontractor must send a monthly report (and invoice) and inform OW immediately of any noteworthy problems. But it is not prescribed that the subcontractor must make a monthly plan and create work orders, or that they should revise the work orders following an emergency.

It is difficult to envisage any general rules that could be applied to all types of contracts and clauses.

5.2 Inter-organizational Workflow Versus Separate Workflows

Supposing we can deduce a recommended business process, how can it be usefully expressed?

One possibility is to propose a single inter-enterprise workflow. However this is not likely to be politically acceptable unless the parties to the contract have a very high level of mutual trust, and are not so concerned about their autonomy.

An alternative is to offer the workflow in two separate sections, one for each party, showing where they need to interact with the other, as in BPEL [10]. However highly autonomous parties may still object to too much detailed prescription, and may already have their own workflow patterns for performing services of this type. A water system maintenance subcontractor might have, for example, worked on maintenance for other clients. In this case, it would be better to leave the finer process detail as “black boxes” for each party’s managers to decide themselves.

It is really a part of contract negotiation to agree what level of integration of process and data the parties to a contract will subject themselves to. If cooperation needs to be close however, too little control might not be adequate. Someone in one organisation may need to ask the other “where exactly are you on this?” In such cases it could be desirable for each company to allow some inspection of their own local workflow status by their collaborators.

5.3 Dependence on Data Capture

How do we verify that data relating to the contract clauses is reliably captured? For example, are there remotely readable real time meters on the pumps, or does OW have to rely on the subcontractor? How do we know that the subcontractor’s engineer has properly serviced a pump, or that the required report contains at least the prescribed minimum details?

Verification of completion of activities is not necessarily assured by simply automating the workflow or the contract monitoring. Many workflow management systems (WfMS) allow a performer to simply click a link that says “Completed”.

In obligations and prohibitions, and in the effectiveness of the granting of permissions, how do we monitor non-compliance? In our example, how does anyone find out if a sub-sub-contractor has been called in discreetly, or that a key to an installation has not been provided? There are cases where it may be against the interests of one party to reveal the data.

In general, if contract clauses rely critically on the values of captured data, then a loop to verify those data may be needed. This can be added as an additional element in the recommended process.

5.4 Overriding the Contract Due to Force Majeure

A further question is, what happens if the contract itself has to be altered? An example might be a drought that caused a systematic failure in many pumps due to impurities. Such overrides would need to be reflected in any running workflows as well as the contract. If the contract is subject to such alteration, it would imply the need for any software supporting a workflow implementation of the business processes to allow easy adaptation of the workflow template at run time.

6 Proposals

This section provides a number of proposals to assist in deriving recommended business processes based on the contract expressions. They represent our early ideas to this mapping problem and will need to be further elaborated in our future research.

6.1 Analysis of the Types of Contracts and Contract Clauses Involved

Contract clauses vary a lot in style - and the contracts as a whole in their balance of these. The following classification is suggested, based on previous examples in the e-commerce area and the example we are currently using.

- System state invariant – this means that certain measurements on the real world situation must not be allowed to go outside certain bounds at any time. In our example this measurement could be the MTBF, MTTR and total down time. A procedure must exist within the party responsible for the system state for achieving this. In our case there has to be a maintenance plan, scheduling when each pump is going to be maintained. If the subcontractor falls behind on its work, then the impact on the contract may not be immediate. The MTBF and MTTR figures will only show up when they are next re-calculated and reported. Depending on the data, it may be possible to provide early warning of likely failure to meet the requirement
- Deadline – this means that some event must occur by a certain date (usually relative to a starting point or a previous event). In our case study, examples are submission of a monthly report, and of additional events and feedback in both directions. Early warning may be possible if the activities can be broken down into smaller measurable stages
- Event-dependent – this implies that some activity must occur following some specified event. In our example, the event could be an emergency in which an irrigation pump for a critical crop failed. In an e-business contract, the event could be the placing of an order.
- Artefact quality – for the contract to succeed, this implies an inspection stage, which may be followed by an iterative re-work loop. The artefact may be physical (e.g. delivered goods) or informational (e.g. a report or design).
- Nested – some contracts are at a single level, e.g. the once-off supply of a number of a particular product. More often the contract has multiple instances, in possibly more than one dimension. In our case study, we have multiple assets. Many other contracts cover multiple business cases, repeated orders etc. This implies processes at both the individual level and at the overall contract level.
- Periodic – some contracts are for a single instance of some activity, others are subject to regular calendar-based repetition, including our own example. Therefore there are processes that repeat within each calendar period.
- Exception specification – this explicitly states a process that is to be followed if things go wrong. In our example, OW can terminate the contract after the 3rd quarter. In other cases, there may be penalty clauses, procedures for agreeing extensions and so on. It often makes sense to provide prompting to parties to a contract that they should enforce their rights.

6.2 Heuristic Rules for Deriving Recommended Sub-workflows

While it is possible for contract architecture and business process model to be derived independently – as we have done – there does seem to be the opportunity for recommending a set of heuristic rules that may help to suggest the structure of the recommended workflow, based on clause characteristics.

The following table shows a summary of the heuristics that could be applied:

Heuristic	**Contract types**	**Deontic modality**	**Comments**
Introduce escalation branches (penalties, extensions etc)	Exception	Obligations, Permissions, Prohibitions	This is the easiest to derive, as the process is usually explicit in the contract
Introduce sub-processes for activities inside the nesting or periodicity	Nested, Periodic	All	Progress on the individual business cases, or periods, is the best early warning
Introduce loops for checking the deliverable and iterating to achieve quality	Quality	Obligations	The requestor may want to reserve the right not to accept the completion of the service.
Introduce planning activities corresponding to a required level of performance	Status	Obligations, Permissions	The party requesting the service may want to be confident that the subcontractor has adequate resources and procedures to meet the requirement
Introduce a re-negotiation phase in case the contract needs changing	Nested, Periodic	All	If things don't go right in one period, or on one business case, the parties may want to allow adjustment of the contract process itself
Introduce related reporting and other information flow phases	All except exceptions	Obligations	If required performance is specified, but no reporting activity, then this should be added

Table 3 Summary of Suggested Heuristics

6.3 Introduction of Additional "Accepted Practice" Sub-workflows

Some parts of widely-used business processes are available for re-use within some of the well-known workflow management systems, e.g. Action Works Metro [9]. Typical examples are getting feedback from a number of people on a draft document, or common business applications such as invoice/payment. Such business processes could be considered to be used as a potential solution for implementing certain processes that satisfy contract conditions.

6.4 Cross-checking of Business Process Models

As discussed earlier, the parties to a contract may wish to tailor any recommended workflows to meet their internal organisation culture, or they may already have their own workflows. Another approach is to analyse the difference between the process models of the individual parties and the "recommended" model. As we found from our own experience, even deriving a recommended model can introduce the possibility of inconsistency with the BCL model, so cross checking is also needed here. In our own example, we can highlight the fact that there is no explicit measuring of the call-out time in the process model. We may allow this to be included by the subcontractor in "Perform Maintenance/Repair", or we may feel that this does not encourage the call-out time to be reliably captured.

7 Related Work

Very few researchers have addressed the relationship between contracts and workflow. In the paper of Van den Heuvel and Weigand [11] and in the European Cross-Flow project [12], contracts are introduced as a means of coordinating, at a higher level, the workflows of individual organisations in a B2B environment. In the commercial field Dralasoft, a vendor of component workflow software, has recently (23/02/04) announced a link with diCarta [13], but further details are not yet known. We believe our approach is currently unique in trying to re-use contract information to infer the workflows that should exist within and between the parties.

8 Future Work and Conclusions

To further this work a natural follow on is to analyse a larger number of contracts to examine whether there are some other clause/contract characteristics and come up with a more comprehensive classification of contracts. This would also help identify possible further patterns which would suggest heuristics for deriving business processes from natural language expression of contracts. Until this is done, we believe that it is premature to develop software approaches for this derivation, such as intelligent agents which could be used for building knowledge bases containing suitable derivation heuristics. Further, the development of tools for cross checking between the BCL and process models is also dependent on a greater understanding of the variety of contract clause types. Another problem is to what extent derived workflows can be feedback into the contract negotiation.

It is worth noting that our original hypothesis was that it may be possible to translate a business contract expressed in a language such as BCL into a business process language that could be used in a workflow management system, but this did not prove to be realistic. This research found that the types of contract, and the nature of the politics between and within the parties to a contract, were too variable. We have proposed a set of heuristics that can help guide the design of recommended workflows that could guide parties to implement contract-compliant behaviour.

Acknowledgements

The work reported in this paper has been funded in part by the Co-operative Research Centre for Enterprise Distributed Systems Technology (DSTC) through the Australian Federal Government's CRC Programme (Department of Industry, Science & Resources).

The authors would also like to thank to Dr Michael Lawley and Tony O'Hagan for their comments to an earlier version of this paper.

References

[1] Z. Milosevic, G. Dromey, *On Expressing and Monitoring Behaviour in Contracts*, EDOC2002 Conference, Lausanne, Switzerland
[2] iMany, www.imany.com
[3] DiCarta, www.dicarta.com
[4] UpsideContracts, www.upsidecontract.com
[5] D. Luckham, *The Power of Events*, Addison-Wesley, 2002
[6] Oracle Contracts, http://www.oracle.com/appsnet/products/ contracts/content.html.
[7] P. Linington, Z. Milosevic, J. Cole, S. Gibson, S. Kulkarni, S. Neal, A unified behavioural model and a contract for extended enterprise, Data Knowledge and Engineering Journal, Elsevier Science, to appear.
[8] S. Neal, J. Cole, P. F. Linington, Z. Milosevic, S. Gibson, S. Kulkarni, *Identifying requirements for Business Contract Language: a Monitoring Perspective*, IEEE EDOC2003 Conference Proceedings, Sep 03.
[9] Action Technologies, Inc, www.actiontech.com
[10] www-106.ibm.com/developerworks/library/ws-bpel/
[11] van den Heuvel, W-J and Weigand, H "Cross-Organizational Workflow Integration using Contracts" jeffsutherland.org/oopsla2000/vandenheuvel/vandenheuvel.htm
[12] Damen, Z, Derks, W, Duitshof, M and Ensing, H "Business-to-Business E-commerce in a Logistics Domain" http://www.crossflow.org/ link to Publications
[13] Dralasoft Inc Press Release, http://www.dralasoft.com/news/dicarta.html