Mining Business Contracts for Service Exceptions

Xibin Gao, Munindar P. Singh, *Fellow, IEEE*, and Pankaj Mehra

Abstract—A contract is a legally binding agreement between real-world business entities whom we treat as providing services to one another. We focus on business rather than technical services. We think of a business contract as specifying the functional and nonfunctional behaviors of and interactions among the services. In current practice, contracts are produced as text documents. Thus the relevant service capabilities, requirements, qualities, and risks are hidden and difficult to access and reason about.

We describe a simple but effective unsupervised information extraction approach and tool, Enlil, for discovering service exceptions at the phrase level from a large contract repository. Our approach involves preprocessing followed by an application of linguistic patterns and parsing to extract the service exception phrases. Identifying such noun phrases can help build service exception vocabularies that support the development of a taxonomy of business terms, and also facilitate modeling and analyzing service engagements.

A lightweight online tool that comes with Enlil highlights the relevant text in service contracts and thereby assists users in reviewing contracts. Enlil produces promising results in terms of precision and recall when evaluated over a corpus of manually annotated contracts.

Index Terms—Contract analysis, service exceptions, text mining

1 Introduction

We address the challenge of modeling and analyzing (business) service engagements. Service engagements inherently involve the interaction of autonomous parties and are naturally specified at a high level in terms of contracts. Contracts can help formalize business processes through which service engagements are realized. They describe the expectations that each participant may have of the others and offer the potential of legal recourse should those expectations not be met. Thus service engagements are almost always specified via a contract, although the contracts involved demonstrate a wide range of complexity.

Because of the importance of service engagements and contracts to the world economy, they are increasingly being studied in computer science [1], [2]. Existing approaches are top-down in that they each propose a model for services and contracts and establish its technical properties. They represent such properties as manually populated metadata and use them to determine how to manage the lifecycle of a contract. In contrast, we adopt a bottom-up approach wherein we examine existing real-life contracts to understand what knowledge and structure we can induce from them. In this sense, our approach is complementary to the above types of approaches. Analysis such as ours can automatically yield part of the knowledge needed by the more traditional approaches.

Modern enterprises manage a large number of active contracts for business operations. Such contracts are usually expressed in unstructured text, but contain rich knowledge about business processes, customer relations, legal risks, and financial implications. Mining contracts can yield actionable knowledge that can help decision makers better regulate business operations, adapt to ever-changing customer demands, maximize financial performance, and mitigate risks.

To this end, we study existing textual contracts with a view to extracting useful knowledge from them. Our approach and tool, dubbed *Enlil* after the Sumerian god of storms (a classical contractual exception), builds on existing tools for text processing and data mining. We find some important differences between the contracts domain and the traditional application domains of text analysis. In particular, contracts appear both to involve longer and more complex sentences and to follow a more routinized structure using a small set of templates than in normal language. The routinized structure facilitates analysis.

Our special focus in this paper is on *service exceptions* or contingency conditions that are described within a contract. For our purposes, contract text is a type of formal legal text [3]. Each contract consists of one or more clauses that specify what each of the participating parties may expect from the others. Contract clauses usually also specify exceptions. Importantly, in practice, the specification of such exceptions exhibits frequent use of a small number of linguistic patterns.

This paper approaches the field of *service computing* in the broad sense. In particular, we are concerned primarily with business services, as indicated by value transfer and coproduction [4], [5]. Business services contrast with technical services, such as Web or grid services, for which a suitable modeling involves the exchange of information such as by a client invoking an operation and the service providing a response. In the technical services literature, the term *contract* sometimes refers to software descriptions, roughly the functionality or type signature of a service, such as might be specified using the Web Services Description Language (WSDL). Other standards address describing the nonfunctional behaviors of a Web Service as well. However, our focus is on a contract as a legal binding between service provider and service consumer.

Enlil extracts domain-specific contracting-relevant knowledge from a large repository of service contracts. Such knowledge can help build service vocabularies that support the development of a taxonomy of business terms. Further, the extracted knowledge facilitates modeling and analyzing service engagements in different domains. Enlil includes a lightweight online tool for automatically annotating important aspects of a service contract, so it can be readily used as annotator for service contracts.

Organization

This paper is organized as follows. Section 2 introduces the technical problem of mining service contracts. Section 3 introduces Enlil's system architecture. Section 4 evaluates the Enlil prototype. Section 5 explains previous work on contracts and text analysis. Section 6 summarizes our conclusions and discusses future work.

2 Problem: Mining Service Contracts

Our approach is both important and viable because modern enterprises usually manage a large number of textual contracts: an average of 40,000 active contracts for a Global 1000 corporation.[1] It is important because improving the treatment of contracts would lead to gains in productivity in setting up and enacting service engagements. It is viable because the contracts corpora (such as are available within each enterprise) serve as crisp and comprehensive knowledge sources that can be automatically mined, as our work demonstrates.

Our particular interest is in mining contracts to discover actionable knowledge regarding the service engagements that the contracts specify. We focus on the exceptions that can be identified from service contracts, broadly because of their importance to the growing field of enterprise risk management, and specifically because of the technical challenge such exceptions pose to the development of robust business processes. The automatic discovery of exceptions by mining contracts can help a business better meet customer demand, conform to regulations, avoid unnecessary financial loss, and hedge against legal risks.

A contract may potentially list one or more exception conditions along with each of its clauses. A contract usually does not list the risks because risks are internal to each party. However, a contract would list the remedies, if any, offered in the case of an exception. Such remedies may represent risks to the remedying party and may indicate the magnitude of the risk perceived by the remedied party.

For example, an IT services contract may say that data access may be lost due to a network outage and may specify a refund of $100 in case of service outage. In this case, the exception is the data access loss due to network outage, the risk to the provider is the $100 it would have to pay, and the risk to the consumer is mitigated by the $100 it would receive. Each party would face additional risks not included within the contract.

The most insidious exceptions are those that a contract fails to anticipate. Enlil can help a designer readily determine what exceptions are incorporated in a contract and what exceptions that occur in other contracts in the same domain have been omitted from a specific contract. Knowing the missing exceptions would be a reason for a participant to reject a contract or to negotiate to modify the terms of a contract before accepting it.

Definition 1: Exception: A potential circumstance that poses an adverse condition for a business or that does not conform to a rule or generalization.

To better appreciate the importance of exceptions, consider the following manufacturing service agreement between FASL LLC and Fujitsu Limited:[2]

- In case of **any defect in Serviced Products**, Fujitsu shall, at Fujitsu's option, (a) rework the applicable Serviced Products, or (b) issue a credit to FASL.
- Fujitsu shall ship all Serviced Products in accordance with the delivery schedule contained in the applicable Purchase Order, and shall promptly notify and consult with FASL in case of **any expected delays in shipping Serviced Products**.
- If **FASL fails to make any payment on or before the required payment date**, FASL shall be liable for interest on such payment at a rate equal to ten percent (10%) per annum or the maximum amount allowed by Applicable Law, whichever is less.
- This Agreement shall be deemed to have been drafted by both Parties and, in the event of **a dispute**, no Party hereto shall be entitled to claim that any provision should be construed against any other Party by reason of the fact that it was drafted by one particular Party.
- In the event that **FASL intends to stop delivering Purchase Orders for Services with respect to any Products**, it shall deliver to Fujitsu four (4) months' prior written notice thereof, provided that (subject to the provisions of Section 5.2 below) no such notice shall be delivered prior to December 1, 2003.

Ignoring the underlined phrases for the time being, let us examine the bold highlighted text in each of the

1. http://en.wikipedia.org/wiki/Contract_management

2. http://contracts.onecle.com/spansion/fujitsu-mfg-2003-06-30.shtml

above clauses. Each such snippet describes an event that indicates an exception faced by the parties to the contract.

The technical problem we address is how to mine contract text to identify the exceptions it refers to. But mining contracts offers challenges not present in some more commonly studied text forms. Contract text tends to involve long sentences with complex nested structure including legal jargon and complicated noun phrases.

Accordingly, we propose a simple but effective unsupervised pattern-based approach for identifying noun phrases indicating exceptions from contract text. For extracting specific semantic relationships from text, pattern-based approaches, such as due to Hearst [6], not only outperform the more general key phrase extraction methods, such as due to Frank et al. [7] and Zha [8], but are also simpler to implement. We evaluate Enlil using the well-known Onecle repository of *real* contracts.[3]

The benefits of Enlil include: (1) discovering service exception vocabularies for different contract domains; (2) highlighting the exceptions in a business contract; and (3) helping develop a taxonomy of exceptions that commonly arise in business operations in each domain.

3 Approach

Our approach consists of the following steps. We describe the three main steps in the remainder of this section.

Step 0: Preprocess contract text by stripping HTML tags and other noise, and segmenting the text into a collection of sentences. We use an off-the-shelf HTML-to-text converter [9] to strip off all the hypertext tags. Next we segment the clean text into a collection of sentences using a sentence delimiter [10].

Step 1: Extract sentences referring to exceptions by applying linguistic patterns.

Step 2: Construct noun phrases from the above sentences using an existing natural language parser.

Step 3: Identify noun phrases corresponding to exceptions.

Currently, Enlil takes contracts retrieved from the Onecle repository as input. The input format could be easily changed by suitably modifying the preprocessing step. Figure 3 shows

a simplified view of Enlil system's architecture.

3.1 Step 1: Extract Sentences that Refer to Exceptions

Revisiting the FASL-Fujitsu contract snippets shown in Section 2, we see that each snippet includes an underlined phrase. Each such phrase describes a syntactic pattern, that is, a pattern phrase, and is textually followed or preceded by a phrase that potentially identifies an exception. Exploiting the routinized nature of contract text, we introduce pattern phrases as a basis for extracting suitable sentences.

Definition 2: Pattern Phrase: A phrase serving as a textual pattern that identifies some important information.

We have crafted a small set of pattern phrases in the style of Hearst [6]. Our patterns are geared toward extracting exceptions from English contracts. A typical pattern is of the form ⌜in (the) case of NP⌝, wherein NP is a noun phrase. Here NP indicates the exception we are trying to extract. A clause specifying the corresponding remedy may follow the pattern, but we focus on exceptions in this paper. Other patterns are formed in the same vein.

Our remaining patterns include *in (the) event of*, which selects noun phrases, and *if*, *in (the) event that*, and *in (the) case that*, which select (sentential) clauses. Although noun phrases are simpler in structure than clauses, they often express exceptions independent of a specific contract context. Further, lThe noun phrases can form the foundation for a taxonomy of exceptions that arise in different business domains. Thus we focus exclusively on noun phrases in this paper. Our technical problem is to identify noun phrases that describe exceptions.

Definition 3: Noun Phrase: A phrase whose head is a noun or a pronoun, optionally accompanied by a list of modifiers.

Definition 4: Exception Phrase: A noun phrase that describes an exception.

We extract the sentences in a contract that match the specified pattern phrases. In particular, in this step, we use the above pattern phrases merely as lexical filters. This is a fast and easy step that substantially reduces the number of sentences that we have to deal with in the subsequent, far more complex, steps.

Definition 5: Pattern Sentence: A sentence that contains a pattern phrase.

Extracting pattern sentences is straightforward. Given a collection of sentences, we check each sentence to determine if it contains any of the identified pattern phrases. If yes, we extract the sentence; otherwise, not. When this process finishes, we have obtained a new collection of sentences, each of which contains a candidate exception phrase.

3.2 Step 2: Construct Noun Phrases

We parse each sentence that matches our lexical patterns using Lingpipe [11], a natural language processing toolkit. Lingpipe provides high-performance part-of-speech tagging (POS), which involves assigning syntactic tags such as Noun (nn), Adjective (jj), Adverb (rb), and so on, to each lexeme. For example, the following input sentence

> In the event of any such delay or failure, the party affected shall promptly notify the other

3. http://contracts.onecle.com/

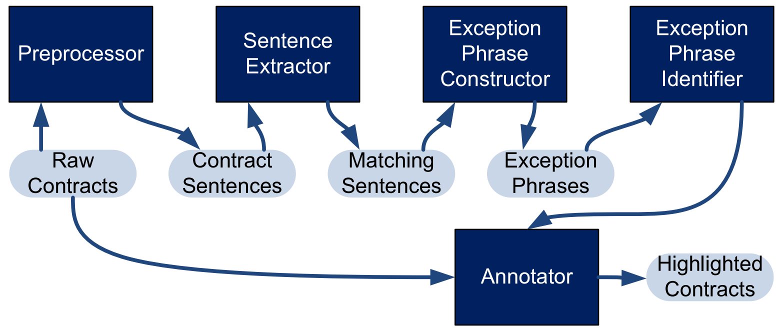


Fig. 1. System architecture and data flow in Enlil.

> party in writing and use all commercially reasonable efforts to overcome the event or circumstance causing the delay or failure as soon as practicable.

yields

> In/in the/at event/nn of/in any/dti such/jj delay/nn or/cc failure/nn ,/, the/at party/nn affected/vbn shall/md promptly/rb notify/vb the/at other/ap party/nn in/in writing/vbg and/cc use/vb all/abn commercially/rb reasonable/jj efforts/nns to/to overcome/vb the/at event/nn or/cc circumstance/nn causing/vbg the/at delay/nn or/cc failure/nn as/cs soon/rb as/ql practicable/jj ./.

We use the Lingpipe noun phrase chunker to aggregate relevant words to form noun phrases based on a grammar of English. A noun phrase can begin with a part of speech such as determinant, pronoun, and adjective, and can have other modifiers such as present particle and past particles. For example, the phrase *any expected delay* is a noun phrase containing a determiner or quantifier, a past particle, and a noun. Lingpipe includes a set of rules for chunking noun phrases. We introduced some noun phrase rules to handle the longer meaningful phrases that arise in contracts. For example, our rules treat the verb *causing* as helping continue a noun phrase instead of terminating it. Thus, the text *an accident causing a delay* is parsed as one noun phrase even though a prefix of it, namely, *an accident* is also a noun phrase.

3.3 Step 3: Identify Exception Noun Phrases

We identify a noun phrase as relevant based on whether it relates to any of the patterns we used to extract the sentences above. Specifically, for the above patterns, determining relevance involves checking whether a noun phrase immediately follows a pattern phrase. If so, we include it in the results; otherwise, we ignore it.

An additional intuition that we capture involves the use of conjunction words (*and*, *but*, *or*, *either or*, *neither or*, and so on) usually indicate a "coordination" or semantic similarity of the phrases they connect. For example, in "The tax proposal was simplistic and well-received" we know that *well-received* is a positive word, and it is connected to *simplistic* by *and*, so we infer that *simplistic* in this context is also a positive word. The conjunction rule is widely used in predicting semantic orientation of adjectives [12] and building opinion lexicons [13].

Definition 6: Conjunction Rule: If an exception phrase is connected with a noun phrase by a conjunction, then the noun phrase is likely to be an exception phrase.

The following is a example of a conjunction occurring in a real contract.

- In the event of **litigation** or **arbitration** the prevailing party shall be entitled to interest, as specified by law, reasonable attorney fees, and court costs.

Notice that our previous step identifies *litigation* as an exception phrase in the contract because it follows the pattern *in the event of*, we can apply the *conjunction rule* to infer that *arbitration* is also an exception phrase. In our approach, we use only the conjunction word *or* for expansion, because it occurs frequently in the sentences that match our patterns and has limited ambiguity.

We apply the conjunction rule in the obvious manner: if two noun phrases are conjoined and one is included as an exception phrase, then so is the other. Algorithm 1 details this method.

4 Evaluation

We now systematically evaluate our approach by highlighting important properties of contracts, the prevalence of exceptions in them, and the quality of our results.

4.1 Statistics about the Corpus

We consider a corpus of 2,647 contracts from Onecle for some evaluations. As Table 1 shows, our pattern sentences are prevalent in contracts across seven major domains of interest to services.

Algorithm 1 Applying the conjunction rule.

Require: Noun phrase set $P = \{p_1, p_2, \cdots, p_k\}$ of sentence s, conjunction word list $W = \{w_1, w_2, \cdots, w_n\}$, and exception phrase list $L = \{l_1, l_2, \cdots, l_m\}$

for all p_i in P AND p_i not in list L **do**
  for all w_j in W, l_r in L **do**
    if p_i connect to l_r with w_j **then**
      Add p_i list L
    end if
  end for
end for
return List L

TABLE 1
Distribution of our patterns across contracts of different domains over our entire corpus.

Contract Type	Contracts	Matches	Average
Licensing	1,364	3,838	2.8
Consulting	501	509	1.0
Outsourcing	9	21	2.3
Supply	207	733	3.5
Manufacturing	206	577	2.8
Purchase	142	591	4.1
Stock Options	218	1,153	5.3
Overall	*2,647*	*7,422*	*2.8*

Figure 2 shows the distributions of different lengths of the sentences that match our patterns (indicating apparent exceptions) and in all sentences in corpus. The distributions are largely parallel, indicating that sentences of all lengths are equally likely to match our patterns. Figure 3 shows contracts distribution and the number of pattern matches in a contract with respect to its length. As we can see, the longer a contract is the more pattern sentences it usually contains. At the same time, we conjecture that, because longer contracts are more complex, the prevalence of pattern phrases in a long contract potentially helps tackle the contract's complexity.

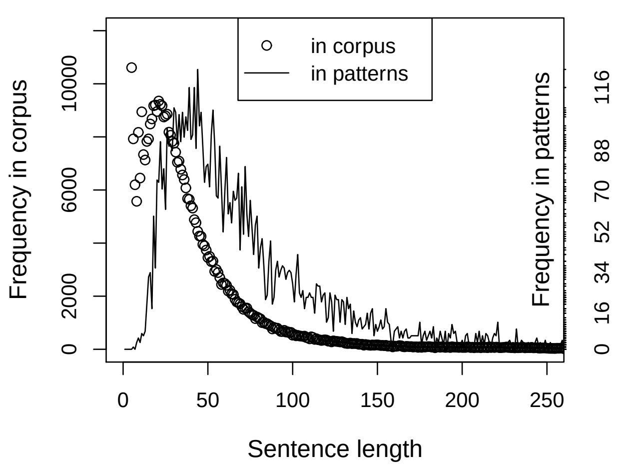


Fig. 2. Distribution of lengths (numbers of words) of the matched sentences and all sentences across all 2,647 contracts studied.

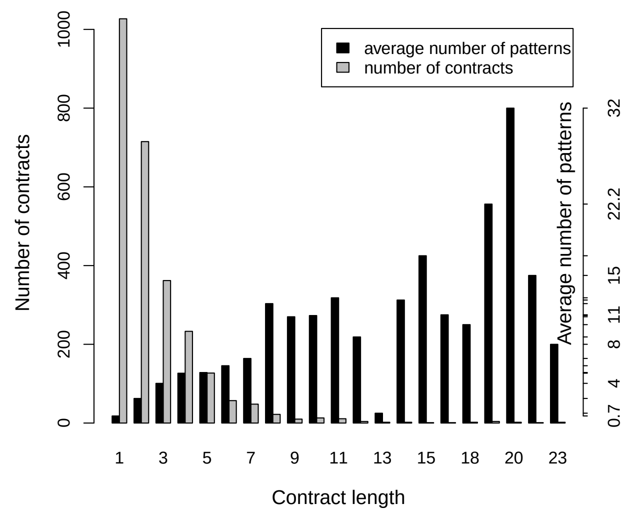


Fig. 3. Distribution of contracts and exception patterns for contracts of different lengths (based on all 2,647 contracts studied). The contract length is measured in hundreds of sentences.

To evaluate the effectiveness of Enlil in extracting exception phrases, we manually annotate the following five (arbitrarily selected) manufacturing contracts from the Onecle repository, namely, those between (1) Minnesota Mining and Manufacture Company (3M) and Sepracor Inc.,[4] (2) Novoste Corporation and BEBIG Isotopen,[5] (3) DrugAbuse Sciences, Inc. and Eon Labs manufacturing, Inc.,[6] (4) FASL LLC and Fujitsu Limited,[7] and (5) Lucent Technologies Inc. and CD Radio, Inc.[8]

In the above five manufacturing contract documents, our patterns yield a total of 24 matching sentences. Table 2 shows some statistics for the extracted sentences. As we can see, sentences that contain pattern phrases are usually quite long. Recall that sentences are delimited by periods and sometimes an entire paragraph that uses several semicolons (common in real-life contracts) can appear as one long sentence.

4. http://contracts.onecle.com/sepracor/3m.mfg.2001.12.20.shtml
5. http://contracts.onecle.com/novoste/bebig.mfg.2001.06.20.shtml
6. http://contracts.onecle.com/drugabuse/eon.mfg.2000.07.20.shtml
7. http://contracts.onecle.com/spansion/fujitsu-mfg-2003-06-30.shtml
8. http://contracts.onecle.com/sirius/lucent.ic.1998.04.24.shtml

TABLE 2
Statistics of sentence length (number of words) over our entire corpus.

Corpus	Size	Minimum	1st Quartile	Median	Mean	3rd Quartile	Maximum
Selected contracts	5	21	27.5	43	51.7	68.5	142
All manufacturing	206	9	31.0	50	67.0	79.0	474
Entire corpus	2,647	6	37.5	58	77.8	93.0	3,328

4.2 Quality of Identifying Exceptions

Unfortunately, no gold standard for exception extraction exists currently. Thus for our evaluation we need to annotate contracts manually. Accordingly, we manually annotated each service contract, marking the exception phrases as benchmark data. Manual annotation proved to be a challenging task for two reasons. First, because there is no existing standard, there is no ready reference for annotation. Second, the concept of exception itself is inherently ambiguous and comes in different expression forms.

As stated before, we restrict exceptions expressed in noun phrases to the scope of our system and annotation. We compare the exception phrases identified by Enlil with manually extracted phrases to compute the true and false positives and negatives (abbreviated TP, FP, TN, and FN, below). Using these, we can calculate the precision, recall, and F-measure—the most widely used metrics of the quality of a retrieval method. These metrics are defined below.

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{F-measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

When configured with the conjunction rule, Enlil extracted 29 phrases with three false positives and three false negatives; without the conjunction rule, it extracted 24 phrases with two false positives and seven false negatives. As Table 3 shows, applying the conjunction rule reduces the precision ever so slightly but increases recall as well as the F-measure.

TABLE 3
Precision, recall, and F-measure for selected manufacturing contracts.

Conjunction?	Precision	Recall	F-Measure
No Expansion	0.92	0.76	0.83
With Expansion	0.90	0.90	0.90

Each pattern leads to the extraction of some phrases from the contract text. Accordingly, we can compute the precision for each pattern that captures exceptions. Recall is not relevant in this scenario because we are interested in identifying all exceptions. Table 4 shows these results as well. As we can see, there is no substantial quality difference for the patterns we used.

TABLE 4
Precision achieved by different patterns for selected manufacturing contracts.

Pattern	Conjunction?	Precision
in (the) event of...	No Expansion	0.92
in (the) event of...	With Expansion	0.86
in (the) case of...	No Expansion	0.91
in (the) case of...	With Expansion	0.92

To convey a feel for the kinds of exceptions Enlil identifies, Table 5 and Table 6 shows the exception phrases

Now we analyze falsely extracted phrases. In the following sentence, “an inspection by the FDA” as well as “any other Regulatory Authority” are noun phrases that occur surrounding the pattern and a conjunction word. The conjunction *or* connects “the FDA” with “any other Regulatory Authority”, but the algorithm wrongly regards the conjunction as connecting “an inspection by the FDA” with “any other Regulatory Authority”. In other words, it falsely identifies “any other Regulatory Authority” as an exception phrase.

- In the event of **an inspection by the FDA** or **any other Regulatory Authority** that relates to the Product or the manufacture thereof, Eon shall notify DAS within twenty (20) business days in writing of the details and results of any such investigation.

Other phrases such as “any such circumstance” are not exceptions themselves, but refer to other clauses in the contract. To identify such conditions, techniques such as coreference resolution are needed, which we defer to future enhancements of Enlil.

When Enlil fails to identify an exception phrase, it mostly does so because (1) the HTML-to-text converter misses some paragraphs because of excessive noise in the input, (2) complicated sentence structure hides some of the target phrases, and (3) some unexpected patterns arise in the input.

In the following sentences, “an accusation of infringement pertaining to Licensed Product” and “undue delay” are exception noun phrases. However, Enlil misses these, as it does not include the corresponding patterns, which are shown underlined

TABLE 5
Sample exception phrases extracted from selected manufacturing contracts using pattern "in (the) event of."

In (the) event of ...	False Positive?	Expanded?
conflict		
inconsistency between any of the terms and conditions of this Agreement		Yes
an extraordinary increase in price due to such factors		
its merger		
an increase in material costs		
refunds		
an inspection by the FDA		
any other Regulatory Authority	Yes	Yes
any inconsistency between the terms and conditions of this Agreement and the terms and conditions of a Purchase Order		
any loss		
irreparable damage to Unfinished Products		Yes
a dispute		
any conflict		
a conflict between the applicable Business Terms and these terms and conditions		
a replacement		
any such circumstance	Yes	

TABLE 6
Sample exception phrases extracted from selected manufacturing contracts using pattern "in (the) case of."

In (the) case of ...	False Positive?	Expanded?
default in payment		
any filing with a Governmental Authority		
other transfer of substantially its entire business in aerosol		Yes
3M on the sale		
Force Majeure		
Product having a latent defect		
any defect in Serviced Products		
any expected delays in shipping Serviced Products		
material breach of this agreement caused by BEBIG		
settlement		
termination under Section 6.2 hereof		
litigation decisions affecting CD Radio		Yes
(ii)	Yes	

below. First, the pattern "in the event (that)" is often followed by an exception clause, not by an exception noun phrase, which is the kind of grammatical construct we seek here. Second, "without" generally introduces noise because of its ambiguity.

- Each Party will notify the other Party promptly in the event a Party receives **an accusation of infringement pertaining to Licensed Product**.
- 3M shall notify SEPRACOR without **undue delay** if it becomes aware that the time estimated for a task or tasks set out in the Scale-up Program will be insufficient to perform such task or tasks.

The current implementation of Enlil favors precision over recall. For one thing, we apply a handful of high-quality patterns. As Enlil can scale well to large datasets, we can obtain pretty good coverage of many aspects of the exceptions when the dataset is large enough.

4.3 Frequent Exception Phrases

We extract the commonly occurring exception phrases for a domain of interest to build a vocabulary of exceptions that arise in each domain. Such a vocabulary could be used to guide a contract reviewer in determining what a specific contract may be missing. And, it would form the basis for building a taxonomy of service exceptions for that domain. Table 7 reports some of the top phrases from our manufacturing corpus.

4.4 Performance

Although Enlil uses complex linguistic processing, it does so selectively on a small part of each contract. As a result, it offers fairly high performance. We evaluated the throughput on our corpus of 2,647 contracts in HTML of total size 182MB using a Toshiba Satellite L45-S7409 laptop with a 1.50GHz CPU, 1.5GB memory, and running Windows 7. A Perl module implements Steps 0 and 1 (preprocessing and extraction); it processes the corpus in 1,650 seconds. A Java module implements Steps 2 and 3 (construction and identification of exception phrases); it processes the sentences output by Step 1 in 250 seconds. So the overall throughput is 1.4 contracts per second. The average response time per contract is under 1 second,

TABLE 7
Frequent exception phrases extracted from all 206 manufacturing contracts in our corpus.

Head Noun	Frequency	Example Phrase
force majeure	51	force majeure
default	28	a default in the payment
merger	11	a merger
delivery	6	late delivery of consigned inventory
delay	5	an inexcusable delay of the delivery of such spare engine
cancellation	4	cancellation of a purchase order
defect	4	any defect in serviced products

demonstrating the viability of Enlil as an annotator tool.

4.5 Additional Validation: Cloud Services Contracts

To further demonstrate the efficacy of Enlil on cloud service contracts, just as we evaluated precision and recall in the manufacturing domain in the previous section, we study Enlil's capability of capturing cloud service specific exceptions in this section.

A crucial challenge in applying and evaluating Enlil in the cloud services contract domain is the limited availability of cloud service contracts. Enlil is best used for discovering exceptions from a large contract repository, which it can do with high efficiency. However, cloud services is still an emerging area and sufficiently many related contracts are difficult to find. For this reason, we arbitrarily selected five terms of use for cloud services[9] from the Internet.

TABLE 8
Precision, recall, and F-measure for selected cloud services contracts.

Conjunction Expansion?	Precision	Recall	F-Measure
No Expansion	0.83	0.60	0.70
With Expansion	0.82	0.72	0.77

Enlil, when configured with conjunction rule, extracted 22 phrases and four of them are false positives, and thus it has a precision of 82% with a recall of 72%. The results are shown in Table 8.

A selection of the exceptions discovered by Enlil is shown in Table 9. These noun phrases mostly express business exceptions, but Enlil did find quality of service related exceptions such as "(hardware) failure," which can naturally have business consequences in a contract. We rate each extracted exception as either related to cloud services or general business. Among all the extracted true positives in the five cloud service contracts, 4/18=22% are related to cloud service exceptions.

9. https://aws.amazon.com/serviceterms/
http://www.cloud.bg/en/sla
http://www.opsource.net/OpSource-Cloud-Terms
http://www.rackspacecloud.com/legal
http://status.net/cloud-tos

5 Related Work

Our work in contract mining intersects with two research areas: service science and text mining. On the one hand, a contract is a service binding artifact, and thus service interactions are regulated by the contract. On the other hand, a contract is expressed in natural language, so text processing techniques apply naturally.

5.1 Contracts

Krishna and Karlapalem [1] formulate the entire contract lifecycle with special reference to service-oriented computing and illustrating the importance of moving from traditional to electronic contracts. They propose a methodology for contracts that gives special importance to exceptions. Indeed, Krishna and Karlapalem list (1) mining contracts and (2) developing general templates for contracts as two of four grand challenges. Our approach shows how to (1) mine contracts for exceptions at the phrase level and (2) by building a list of common exceptions, shows how to address the design of contracts as well.

Meneguzzi et al.'s effort [2] is part of the European Union's CONTRACT project framework, a comprehensive approach to model, reason about, and enact electronic contracts. Enlil complements the above work in two respects. One, Enlil can help acquire the knowledge of a particular setting that the CONTRACT framework can codify and operationalize. Two, Enlil brings up the typical business exceptions that arise in a domain and in this manner provides a basis for verifying whether a specific contract is sufficiently robust and that its enactments would accommodate the discovered exceptions.

Arenas and Wilson [14] distinguish between the operational and business levels of a contract. At the operational level, a contract can be expressed as policies, licenses, and service level agreements. Currently popular approaches for service agreements—such as WS-Policy [15], Web Service Level Agreement (WSLA) [16], Web Service Offerings Language (WSOL) [17], and Open Digital Rights Language (ODRL) Service profile (ODRL-S) [18]—largely emphasize operational details. At the business level, a contract is drafted by contract lawyers and executed by the participating

TABLE 9
Sample extracted exception phrases from selected cloud services contracts.

In (the) case/event of ...	Cloud service specific?
a conflict between the terms of these Service Terms and the terms of your agreement with us governing your use of our Services	
any inconsistency	
conflict with the Agreement	
a payment failure	Yes
a merger	
acquisition	
delay in processing of the order and the payment datas [sic] correctness	Yes
legal situations Host Color	
conflict between the terms contained in the Service Order and the terms in this Agreement	
a suspension by OpSource of Customer's access to any Service pursuant to Section 13.3	Yes
any termination by OpSource of any Service	
any dispute between the parties concerning interpretations	
a dispute between us regarding the interpretation of applicable law	
a (system) failure	Yes

organizations. There is thus a huge gap between the business and the operational levels.

Many researchers have recently begun to bridge the gap between the two levels of contracts [19]. Milosevic et al. [20] link contracts, processes, and services by mapping business contract conditions onto messages and business rules. Singh et al. [21] develop a high-level representation of business services that maps contracts to computations, and Telang and Singh [22] develop a formal business metamodel in terms of commitments and use models of cross-organizational businesses to verify concrete enactments realized via messaging. Desai et al. [23] study the challenges of validating business contracts based on their participants' valuations with respect to various contract events.

Exceptions have been studied in the context of representation, identification, and resolution. Molina-Jimenez et al. [24] introduce an architecture for exception resolution. Grosof and Poon [25] represent business contracts in RuleML, thereby enabling agents to automatically create, evaluate, negotiate, and execute contracts and to handle exceptions. Klein et al. [26] describe a methodology for identifying exceptions and finding suitable responses for these exceptions. Further, Klein et al. propose a taxonomy of exceptions.

However, existing approaches on contracts and exceptions do not interface well with the "legacy" of text-based contracts, which is how all serious business is still being conducted today. Research on the automatic extraction of exceptions from contracts has been rare, if not nonexistent—despite the exhortations of researchers such as Krishna and Karlapalem [1]. The present approach can feed the above approaches with concrete representations that they can formally reason with.

Khandekar et al. [27] proposed a system called MTDC (Methodology and Toolkit for Deploying Contracts) to map a business contract to deployable e-contracts based on the ER^{EC} data model of Karlapalem et al. [28]. The MTDC system takes advantage of knowledge of the domain (in which the contract applies) such as the contract type, and a list of keywords specific to the domain of the contract, and can extract sentences representing exceptions. Each sentence in a contract is classified as a clause, an activity, or an exception based on rules and supplemented with assistance from a human designer.

MTDC exemplifies a classification-based approach. In contrast, Enlil uses a pattern-based approach to extract sentences as an intermediary step to discover exceptions at the phrase level. MTDC classifies sentences in a contract as a *clause*, an *activity*, or an *exception*. That is, exceptions are labeled at the sentential level. The sentence classification approach in MTDC has its advantages when sufficient knowledge and human assistance is available. Our approach differs in some important ways. First, the patterns in Enlil are independent of the domain under consideration: we do not use domain-specific or contract-specific keywords to assist the sentence extraction (classification) process. We use the same patterns across different contract domains without any human intervention, and the result still proves to be effective. By contrast, MTDC requires a "designer" role to assist the system. Second, our main goal with Enlil is to extract exceptions from a large contract repository in different contract domains, that is, we seek to capture one important aspect of contract text. Efficiency, minimal supervision, and portability are significant motivations for us. By contrast, MTDC aims to produce a deployable e-contract, and its main motivations are accuracy in capturing several aspects of a single contract. Third, Enlil applies exception noun phrases, which are finer grained and more specific than sentences that involve exceptions. The extracted noun phrases are the semantic units that can lay a foundation for building ontologies for different contract domains.

In broad terms, because contracts are a type of legal document, work on knowledge extraction from regulatory text is indirectly related. Breaux et al. [29] and Kiyavitskaya et al. [30] extract rights and obligations from regulatory text to aid regulatory compliance. Koliadis et al. [31] extract key phrases and generate possible interpretations from predefined templates to contextualize regulatory policies. However, these approaches are mostly reliant on complex hand-crafted rules or heuristics, which are almost always manually applied. As a result, they are not easy to migrate to new settings and place onerous demands on the analyst.

Some research applies text mining to analyze text artifacts in web services and software requirement for service matching, discovery, and key element identification. Yale-Loehr et al. [32] mine software requirement specifications (SRS) to discover shared services and make corresponding recommendations. They use a similarity based approach to compare keywords in SRSs and take advantage of sets of synonyms (termed *synsets*) identified in WordNet [33]. Guo et al. [34] propose an approach for improving the quality of semantic web service matching. They generate ontologies from web service descriptions and map between web services with the guidance of the ontologies. Spanoudakis et al. [35] discuss and lay out the foundations of principles for inconsistency and overlaps between SRSs. They address the problem of overlap identification and take steps towards providing a formal semantics for overlap relations. Hussain et al. [36] analyze SRSs to classify sentences as functional (for example, input, output, events) or nonfunctional (for example, performance, reliability, security) requirements. Hussain et al. use a set of keywords and part-of-speech tags and employ a text classifier based on Quinlan's C4.5 decision tree algorithm [37].

5.2 Text Mining

We apply a pattern-based natural language processing approach for finding exceptions in contract text. Pattern-based information extraction has been an active discipline in the past two decades. Despite their simplicity, linguistic pattern-based approaches yield surprisingly good results. We survey some important work in this area.

Hearst [6] pioneered the pattern-based approach by using it for automatic acquisition of hypernyms from Grolier's American Academic Encyclopedia. The hyponymy relation such as of *apple* to *fruit* indicates the *is a* relation. To extract such information, Hearst defines patterns of the type $\ulcorner NP_0$ such as $NP_1 \urcorner$. For example, the phrase *fruit such as apple* (if sufficiently frequent) conveys information that *apple* is a hyponym of *fruit*.

Berland and Charniak [38] apply a similar pattern-based approach to find nouns that satisfy *part-of* relations in the LDC North American News Corpus (NANC). The *part-of* relation indicates *part* and *whole* of the entities such as wheel to car. Berland and Charniak's patterns are of the type $\ulcorner NP_0$ of $NP_1 \urcorner$, which indicate a part-of relationship, as in *basement of building* that *basement* is a part of *building*.

Girju and Moldova [39] extract causal relations from text using an approach similar to the above on the TREC-9 data set, which is a collection of news articles. To extract causal relations from corpora, Girju and Moldova use the most explicit intra-sentential pattern $\ulcorner NP_0 \; V NP_1 \urcorner$, where V is a simple causative verb.

Hearst evaluates her approach against WordNet and obtains a precision of 57.55%. Berland and Charniak's approach yields 55% accuracy for the top 50 words, when evaluated against human annotated data. And, Girju and Moldova achieve 65.6% accuracy against the average performance on two human annotators on 300 relation pairs. In this context, our results of nearly 90% precision indicate that contracts are a promising domain and perhaps that additional information can be mined from them.

Leidner and Schilder [40] use Hearst patterns [6] to mine business risk vocabularies and build a taxonomy. They identify potential risks in financial reports. Leidner and Schilder use the Web as their corpus for vocabulary discovery and validation. In contrast, our system uses a set of contracts as its corpus, and its vocabulary discovery process is not based on the Hearst patterns.

Indukuri and Krishna [41] use an approach based on machine learning to study contract documents. They employ a binary support vector machine (SVM) to decide if a sentence in a contract is a clause. Indukuri and Krishna further classify the clauses into two categories: payment related or otherwise, on somewhat ad hoc grounds. In their experiment, they use n-gram models (with n ranging from one to four) to convert from text into feature vectors. They report the best result when n equals four. In contrast, we identify exception clauses and develop a domain-specific vocabulary of exceptions. Payment is inherently domain-independent so in that sense our problem is complementary to that of Indukuri and Krishna.

On the basis of linguistic processing, our method uses patterns as a clue to discover service exceptions at the phrase level. A basic pattern recognizer and a learning-based approach can also extract sentences or other linguistic context such as a text window consisting of the few immediately preceding and following sentences. Let us compare Enlil with these approaches. The difference between our approach and a traditional pattern recognizer system lies in their capabilities, motivations, and scopes. First, a naïve pattern recognizer simply returns a chunk of text, whereas Enlil extracts meaningful text syntactic units that associate with the specified patterns. Second, the pattern recognizer approaches are interested in the pattern matched text itself. However, Enlil is not interested in the patterns themselves, but uses patterns as clues for discovering exceptions. Third, the pattern recognizer generally runs on the surface, and does not involve chunking of meaningful text units, thereby requiring substantial human effort in understanding the extracted text. Enlil automatically extracts meaningful service exceptions at the phrase level.

6 SUMMARY

A contract is a legal agreement between real-world business entities whom we treat as providing services to one another. We focus on business rather than technical services. Service exceptions, as the focus of our study in this paper, reveal critical aspects of business service operations. As we live in an imperfect world, a timely capture of business exceptions and a proper handling of unexpected incidences can offer competitive advantage to an organization. Though rarely studied before in the service community, exceptions extraction at the phrase level can potentially help build a rich knowledge base for ontologies.

The novelty of our work lies in formulating and solving the problem by bridging text-based service contracts with natural language processing analytics. Empirical studies show that our approach is not only viable but also effective. As opposed to rule-based or machine learning approaches that address related tasks, such as the approach of Khandekar et al. [27], our approach requires minimal human intervention, yields better portability across different contract domains, and enjoys high efficiency on large text repositories. Capturing service exceptions at a semantic level is challenging because of their potential ambiguity and wide range of references. Harvesting exceptions from a vast amount of contract text is a daunting task. Enlil avoids the semantic challenges and takes advantage of a handful of text patterns to harvest the semantic units of exceptions at the phrase level.

Enlil demonstrates an *unsupervised* pattern-based approach for automatically extracting exceptions from contract text that is not only flexible, but also effective. We apply manual annotations solely for the purposes of evaluation and not to train the Enlil tool. Our approach is independent of the domain of the given contracts and requires minimal human effort. Figure 4 shows a screenshot the online tool that comes with Enlil when used on a real contract from the IT services domain. This illustrates a simple but valuable use of Enlil, wherein it highlights the relevant text in service contracts and thereby assists users in reviewing contracts.

Enlil can discover domain-specific exception vocabularies from contracts. For example, we may find phrases such as *late delivery* and *defect in products* as indicated in Table 7 more commonly in manufacturing contracts than in loan agreements, where terms such as “bankruptcy” and “insolvency” would appear more frequently.

It is worth observing that text analysis approaches, such as those summarized in Section 5, generally achieve precisions in the 55% to 70% range. Our results approaching 90% are perhaps an outcome of contract text being apparently (and not surprisingly) more routinized than normal English text. The quality of these results also indicates that the prospects for additional text analysis tools for contracts are promising.

6.1 Limitations

Despite the high efficiency and demonstrated effectiveness of our approach, the inherent difficulty of the information extraction task exposes our approach to some limitations. Enlil relies on a handful of patterns. Because of the large variety of ways in which the concept of *exceptions* may be expressed in natural language, positive instances sometimes escape the routinized patterns. In the current implementation of Enlil, we manually select high-quality patterns to extract phrases expressing service exceptions, and it achieves high precision but may have sacrificed recall. We plan to dynamically expand patterns in an iterative fashion in the future. Further, although our approach only focuses on exceptions at the phrase level, some applications may make more sense with exceptions at sentential level. We plan to enhance Enlil to expand its extraction capability by including exceptions expressed in sentences.

6.2 Future Directions

Our effort on Enlil opens up several interesting directions for further study. Two, in particular, that are worth emphasizing are ontologies and temporal constraints. First, based on the service exceptions that Enlil extracts, phrase classification algorithms can further organize these vocabularies into categories. For example, some exceptions refer to financial conditions such as nonpayment, and some refer to natural disasters such as earthquakes. On top of that, a taxonomy of exceptions for a specific contract domain can potentially be generated automatically. It would be valuable to generate domain-specific ontologies of business service exceptions for potential use in evaluating contracts for completeness and authoring robust contracts.

Second, Enlil may be readily enhanced to extract other types of information from contract text. In particular, we observe that many business exceptions involve *temporal constraints* such as “late delivery of products” and “late payment.” A failure in the timely

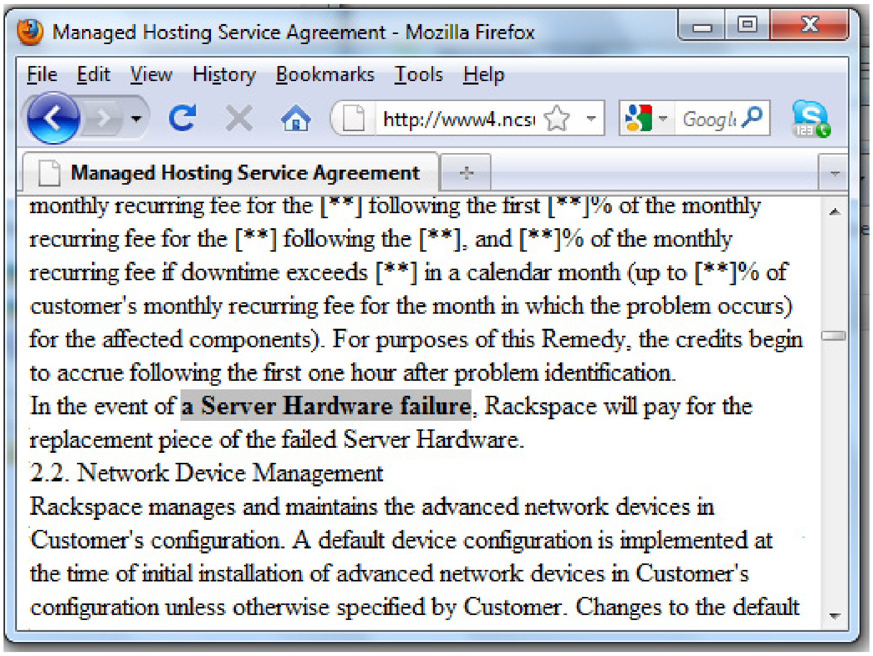


Fig. 4. A screenshot of Enlil as a browser add on.

delivery of a service can damage an organization reputation, disrupt enterprise activity, result in poor customer satisfaction, and ultimately in loss to bottom-line results. In addition, temporal relations, such as those indicated by *between*, *before*, and *after*, can provide critical information for regulating business activities. Mining business events and their temporal constraints from contracts can prepare a decision maker for possible violations and help an enterprise hedge against potential business risks. We plan to extend Enlil by extracting business events and their temporal constraints from contracts.

Acknowledgments

We are indebted to Dennis Bahler, Ting Yu, and Nirmit Desai for valuable comments on previous versions of this paper. We are also indebted to the anonymous reviewers for their helpful comments. Xibin Gao's effort on this paper was partially conducted while visiting the Digital Enterprise Research Institute (DERI) in Galway, Ireland.

References

[1] P. Radha Krishna and K. Karlapalem, "Electronic contracts," *IEEE Internet Computing*, vol. 12, no. 4, pp. 60–68, 2008.

[2] F. Meneguzzi, S. Miles, C. Holt, M. Luck, N. Oren, N. Faci, and M. Kollingbaum, "Electronic contracting in aircraft aftercare: A case study," in *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems*, 2008, pp. 63–70.

[3] Y. Yang and X. Liu, "A re-examination of text categorization methods," in *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 1999, pp. 42–49.

[4] R. J. Wieringa and J. Gordijn, "Value-oriented design of service coordination processes: correctness and trust," in *Proceedings of the ACM Symposium on Applied Computing (SAC)*. Santa Fe, New Mexico: ACM, 2005, pp. 1320–1327.

[5] P. P. Maglio and J. Spohrer, "Fundamentals of service science," *Journal of the Academy of Marketing Science*, vol. 36, no. 1, pp. 18–20, Mar. 2008.

[6] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," in *Proceedings of the 14th Conference on Computational Linguistics*. Association for Computational Linguistics, 1992, pp. 539–545.

[7] E. Frank, G. Paynter, I. H. Witten, C. Gutwin, and C. G. Nevill-Manning, "Domain-specific keyphrase extraction," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence (IJCAI)*. Morgan Kaufmann, 1999, pp. 668–673.

[8] H. Zha, "Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering," in *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2002, pp. 113–120.

[9] D. Muey, "http://search.cpan.org/∼dmuey/html-obliterate-0.3/."

[10] S. Yona, "http://search.cpan.org/∼shlomoy/lingua-en-sentence-0.25/."

[11] Alias-i, "http://alias-i.com/lingpipe."

[12] V. Hatzivassiloglou and K. R. McKeown, "Predicting the semantic orientation of adjectives," in *Proceedings of the 35th Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, 1997, pp. 174–181.

[13] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," in *Proceedings of the International Conference on Web Search and Web Data Mining*. ACM, 2008, pp. 231–240.

[14] A. Arenas and M. Wilson, "Contracts as trust substitutes in collaborative business," *IEEE Computer*, vol. 41, no. 7, pp. 80–83, 2008.

[15] S. Bajaj, D. Box, D. Chappell, F. Curbera, G. Daniels, P. Hallam-Baker, M. Hondo, C. Kaler, D. Langworthy, A. Malhotra *et al.*, "Web services policy framework (WS-Policy)," *Version*, vol. 1, no. 2, pp. 2003–2006, 2006.

[16] H. Ludwig, A. Keller, A. Dan, R. King, and R. Franck, "Web service level agreement (WSLA) language specification," 2003. [Online]. Available: http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf.

[17] V. Tosic, B. Pagurek, K. Patel, B. Esfandiari, and W. Ma, "Man-

agement applications of the web service offerings language (WSOL)," *Information Systems*, vol. 30, no. 7, pp. 564–586, 2005.

[18] G. Gangadharan, V. D'Andrea, R. Iannella, and M. Weiss, "ODRL service licensing profile (ODRL-S)," in *Proceedings of the 5th International Workshop for Technical, Economic, and Legal Aspects of Business Models for Virtual Goods*, 2007.

[19] A. Arenas, M. Wilson, S. Crompton, D. Cojocarasu, T. Mahler, and L. Schubert, "Bridging the gap between legal and technical contracts," *IEEE Internet Computing*, vol. 12, no. 2, pp. 13–19, 2008.

[20] Z. Milosevic, M. Orlowska, and S. Sadiq, "Linking contracts, processes and services: an event-driven approach," in *Proceedings of the 3rd IEEE International Conference on Services Computing (SCC)*, 2006, pp. 390–397.

[21] M. P. Singh, A. K. Chopra, and N. Desai, "Commitment-based service-oriented architecture," *IEEE Computer*, vol. 42, no. 11, pp. 72–79, Nov. 2009.

[22] P. R. Telang and M. P. Singh, "Specifying and verifying cross-organizational business models: An agent-oriented approach," *IEEE Transactions on Services Computing*, vol. 4, 2011, in press.

[23] N. Desai, N. C. Narendra, and M. P. Singh, "Checking correctness of business contracts via commitments," in *Proceedings of the 7th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. Estoril, Portugal: IFAAMAS, May 2008, pp. 787–794.

[24] C. Molina-Jimenez, S. Shrivastava, and M. Strano, "Exception handling in electronic contracting," in *Proceedings of the 11th IEEE Conference on Commerce and Enterprise Computing (CEC)*, 2009, pp. 65–73.

[25] B. Grosof and T. Poon, "SweetDeal: Representing agent contracts with exceptions using XML rules, ontologies, and process descriptions," in *Proceedings of the 12th International Conference on World Wide Web*. ACM, 2003, pp. 340–349.

[26] M. Klein, C. Dellarocas, and J. Rodriguez-Aguilar, "A knowledge-based methodology for designing robust electronic markets," in *1st International Conference on Autonomous Agents and Multi-agent Systems (AAMAS)*, 2002.

[27] A. Khandekar, P. Krishna, and K. Karlapalem, "A methodology and toolkit for deploying contract documents as e-contracts," in *Tutorials, Posters, Panels and Industrial Contributions at the 26th International Conference on Conceptual Modeling*, vol. 83. Australian Computer Society, Inc., 2007, pp. 91–96.

[28] K. Karlapalem, A. Dani, and P. Krishna, "A frame work for modeling electronic contracts," in *Proceedings of 20th International Conference on Conceptual Modeling*, vol. 2224. Yokohama, Japan: Springer, Nov. 2001, pp. 193–207.

[29] T. D. Breaux, M. W. Vail, and A. I. Anton, "Towards regulatory compliance: extracting rights and obligations to align requirements with regulations," in *Proceedings of the 14th IEEE International Requirements Engineering Conference (RE)*. Washington, DC, USA: IEEE Society Press, September 2006, pp. 49–58.

[30] N. Kiyavitskaya, N. Zeni, T. D. Breaux, A. I. Anton, J. Cordy, L. Mich, and J. Mylopoulos, "Automating the extraction of rights and obligations for regulatory compliance," in *Proceedings of the 27th International Conference on Conceptual Modeling (ER)*, vol. 5231. Berlin, Germany: Springer-Verlag, October 2008, pp. 154–168.

[31] G. Koliadis, N. Desai, N. Narendra, and A. Ghose, "Analyst-mediated contextualization of regulatory policies," in *Proceedings of the 7th IEEE International Conference on Services Computing (SCC)*, 2010.

[32] A. Yale-Loehr, I. D. Schlesinger, A. J. Rembert, and M. B. Blake, "Discovering shared services from cross-organizational software specifications," in *Proceedings of the 7th IEEE International Conference on Services Computing*. IEEE Computer Society, 2010, pp. 186–193.

[33] G. A. Miller, "WordNet: A lexical database for English," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, Nov. 1995.

[34] H. Guo, A. Ivan, R. Akkiraju, and R. Goodwin, "Learning ontologies to improve the quality of automatic web service matching," in *Proceedings of the 16th International Conference on the World Wide Web*. IEEE, 2007, pp. 118–125.

[35] G. Spanoudakis, A. Finkelstein, and D. Till, "Overlaps in requirements engineering," *Automated Software Engineering*, vol. 6, no. 2, pp. 171–198, 1999.

[36] I. Hussain, L. Kosseim, and O. Ormandjieva, "Using linguistic knowledge to classify non-functional requirements in SRS documents," in *Proceedings of the 13th international conference on Natural Language and Information Systems: Applications of Natural Language to Information Systems*. Springer-Verlag New York Inc, Jun. 2008, pp. 287–298.

[37] J. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

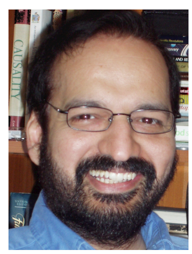
[38] M. Berland and E. Charniak, "Finding parts in very large corpora," in *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*. Association for Computational Linguistics, 1999, pp. 57–64.

[39] R. Girju and D. I. Moldovan, "Text mining for causal relations," in *Proceedings of the FLAIRS Conference*, 2002, pp. 360–364.

[40] J. Leidner and F. Schilder, "Hunting for the black swan: Risk mining from text," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics (ACL), System Demonstrations*, 2010, pp. 54–59.

[41] K. V. Indukuri and P. Radha Krishna, "Mining e-contract documents to classify clauses," in *Proceedings of the 3rd Annual ACM Bangalore Conference (Compute)*. ACM, 2010, pp. 1–5.

Xibin Gao Xibin Gao is a PhD student in the Department of Computer Science, North Carolina State University, Raleigh. He received a B.S. from Sichuan University and a M.S. from Chinese Academy of Sciences. His research interests include business intelligence, information extraction, natural language processing, and service-oriented computing.



Munindar P. Singh Munindar P. Singh is a Professor in the Department of Computer Science, North Carolina State University, Raleigh. His research interests include multiagent systems and service-oriented computing with a special interest in the challenges of trust, service selection, and business processes in large-scale open environments. His books include the coauthored *Service-Oriented Computing* (Wiley, 2005). Dr. Singh was the Editor-in-Chief of the *IEEE Internet Computing* from 1999 to 2002. He is a member of the editorial boards of *IEEE Internet Computing*, *Autonomous Agents and Multiagent Systems*, *Web Semantics*, and *Service-Oriented Computing and Applications*.

Pankaj Mehra Pankaj Mehra is the CTO and a cofounder of Inlogy, an enterprise software company that models and recommends people on the basis of the behavior they exhibit within their communication contexts. His research interests include semantics of enterprise information, cloud computing, and machine learning. His books include the co-authored *Storage, Data and Information Systems* (HP, 2007). Prior to Inlogy, Pankaj was Distinguished Technologist at HP, where he founded HP Labs Russia and led the incubation of Taxonom.com, an ontology-generation service. He also served on the faculty of Indian Institute of Technology, Delhi. Pankaj received his Ph.D. in Computer Science from the University of Illinois at Urbana-Champaign. He currently coordinates industry participation in Stanford University's Collaborative Data Management Initiative and serves on *IEEE Internet Computing* magazine's editorial board. Contact him at pankaj.mehra@IEEE.org.