Automated Reasoning with Legal XML Documents

Laurence L. Leff
Western Illinois University
Department of Computer Science
Macomb IL 61455
309 298 1315

mflll@wiu.edu

ABSTRACT

We have integrated the Jess Expert System tool from Sandia Labs [4] with the Xerces XML parser [7]. We submit to this software contracts and court filings for litigation involving those contracts. These are written as per a contract standard submitted to the Legal XML standards group [14] and the court filing proposed standard[15]. The system determines if a summary judgment request can be granted based on the submitted affidavits, contracts, and other documents.

Categories and Subject Descriptors

J.7 [Computers in Other Applications]: I.2.1 [Applications and Expert Systems] *Law*

General Terms

Standardization, Legal Aspects

Keywords

Contracts, Civil Procedure, Court Automation, Jess, Rule-Based Expert Systems, XML

1. INTRODUCTION

Those familiar with electronic commerce trends know that XML is being used and proposed for many applications. XML is a way that industry groups can specify the format for structured information. XML standards are available for insurance information[8], purchase orders [1]and financial exchanges [16] and many other industries. The Gartner group estimates that 70 per cent of business-to-business transactions will involve XML [17]. The Legal XML Work Group is developing standards for Legal documents; they have work groups developing products for court filing documents, contracts, transcripts, and legislation. The software described herein manipulates the documents found on the first two workgroup's web pages.

Contract cases constitute half of all civil litigation. Half of these were plaintiffs trying to recover payment for goods or service rendered. In one year, these represented at least 90,000 suits in state court as well as 50,000 federal cases. This survey [1] only included the 75 largest counties and does not include rural areas or small claims court suits. I believe that much of this contract litigation is routine.

All or much of the data for a contract case are often in the records of the participants, particularly when the contract was formed electronically. This is in contrast to tort cases such as car accidents, where much of the data on which the reasoning is to occur must be obtained by interviewing witnesses.

Gardner[5], pointed out that the Restatement of Contract Law is a series of rules, perfect for conversion to an expert system. Furthermore, the Rules of Civil Procedure that are promulgated by most court systems map nicely into the rules of an expert system.

This paper integrates the two showing how a combination of rules reasoning on the contract itself, in XML form, and on the court filing documents, also in XML form, provide a way of routinizing contract-related litigation.

1.1 The Jess XML integration

I observed that much XML software is written using Java with several freeware XML parsers available for downloading[2],[7], and [9]. And I knew of the Java Rule-Based Expert System tool called Jess [4]. Thus, I developed an integration of these two products. These include Jess User Functions that load XML documents and convert the DOM tree into a series of Jess facts. A fact is created for each XML tag and each attribute found in the document loaded. Then, ordinary Jess rules can be used to reason about the XML loaded. As a single XML document would create many facts, I created routines that will selectively expand parts of an XML document. After all the reasoning is completed for a document or part thereof, the corresponding facts can be removed from the Fact List. In the expert system domain, and in Jess, this is known as "retraction." All these can be done from inside Jess using the packages I wrote (see [13]).

Lastly, I allowed the rules themselves to be expressed as XML. The rule-writer can simply cut-and-paste the XML from a sample document into the rule set, replacing attribute values or included text with markup corresponding to Jess variables. (See examples below.)

2. EXAMPLE

To demonstrate the power of this mechanism, our sample Jess rule-set reads in a contract, a lawsuit initiation (a complaint), affidavit, and a request for summary judgment. Our XML format for the latter follows the proposed Court Filing standard. However, some parts of that standard not relevant to this

Conference '00, Month 1-2, 2000, City, State.

reasoning or discussion are omitted below for ease of explication. The rule-based reasoning determines whether the plaintiff is entitled to summary judgment based upon whether they met the contract terms and if so, issues a fact that the plaintiff is entitled to summary judgment.

2.1 The Input

Our example reads in the following four XML files.

1. An XML contract specifying that Acme pizza will deliver ten boxes of pizza to Joan Smith and Joan Smith, in turn, will pay $80.00. This obeys the first instance of Contract DTD proposed to the Legal XML Contract Work Group[10].

2. A Lawsuit on that Contract for $80.00

3. An affidavit stating that the Plaintiff delivered the Pizza.

4. A request for Summary Judgment by the plaintiff

These files follow:

The Contract:

```
<?xml version="1.0" encoding='UTF-8'?>
<!DOCTYPE Offer SYSTEM "xmlcontract.dtd">
<Offer ID="I003">
<PartyList>
<Party ID="P001">
<Name>Acme Pizza</Name>
</Party>
<Party ID="P002">
<Name>Joan Smith</Name>
</Party>
</PartyList>
<Clause ID="C002" By="P001" To="P002">
<Text>deliver ten boxes
Pizza</Text></Clause>
<Clause ID="C003" By="P002"
To="P001"><Pay>80.00</Pay></Clause>
<LawsuitTo>
<CourtInformation>
<Location>
<LocationId>Court 1</LocationId>
<LocationFunction>N.A.</LocationFunction>
</Location>
</CourtInformation>
</LawsuitTo>
<ApplyLawOf>NY</ApplyLawOf>
<EffectiveDate>
<Date>20000413</Date>
</EffectiveDate>
<ResponseDate>
<Date>20000415</Date>
</ResponseDate>
</Offer>
```

The Lawsuit Complaint

```
<?xml version="1.0" encoding='UTF-8'?>
<!DOCTYPE DocumentInformation SYSTEM
"lawsuit.dtd">
<DocumentInformation>
<Actors>
<Actor ID="PP">
<Name>Acme Pizza</Name>
<Role>Plaintiff</Role>
</Actor>
<Actor ID="DD">
<Name>Joan Smith</Name>
<Role>Defendant</Role>
</Actor>
</Actors>
<DocumentContent>
<Complaint>
<Prayer Amount="80.00"/>
</Complaint>
</DocumentContent>
</DocumentInformation>
```

The Affidavit

```
<?xml version="1.0" encoding='UTF-8'?>
<!DOCTYPE DocumentInformation SYSTEM
"lawsuit.dtd">
<DocumentInformation>
<Actors>
<Actor ID="PP">
<Name>Acme Pizza</Name>
<Role>Plaintiff</Role>
</Actor>
<Actor ID="DD">
<Name>Joan Smith</Name>
<Role>Defendant</Role>
</Actor>
</Actors>
<DocumentContent>
<Affidavit>
<Assert Clause="C002" ContractID="I003"/>
<Text>More Text about how he delivered the
Pizza
that the Court may not have to
Look at </Text>
</Affidavit>
</DocumentContent>
</DocumentInformation>
```

The Request for Summary Judgment

```
<?xml version="1.0" encoding='UTF-8'?>
<!DOCTYPE DocumentInformation SYSTEM
"lawsuit.dtd">
<DocumentInformation >
<Actors>
<Actor ID="PP">
<Name>Acme Pizza</Name>
<Role>Plaintiff</Role>
</Actor>
<Actor ID="DD">
<Name>Joan Smith</Name>
<Role>Defendant</Role>
</Actor>
</Actors>
<DocumentContent>
<RequestForSummaryJudgement
Case="http://web.mit.edu/leff/www/l2.xml"
FavorOf="plaintiff"/>
</DocumentContent>
</DocumentInformation>
```

2.2 The Rules

Our software reads rules in XML format. See [13]for information for a complete example showing how the same rules could have been written using Jess syntax.

```
<Rule Name="RuleTwo" Documentation="Match
Party Names to Real Names"
DocumentName="VariableName:dx">
<LHS>
 <Check>
```

```
  <Offer ID="VariableName:ContractID"
  DocumentName="VariableName:dx">
   <PartyList>
   <Party ID="VariableName:P">
   <Name><TextVariable
   VariableName="RealName"/></Name>
   </Party>
   </PartyList>
  </Offer>
 </Check>
</LHS>
<RHS>
 <Action>
  <assert><Template><TemplateName>RealName<
  /TemplateName><Slot><SlotName>PartyName</
  SlotName><Value><Variable>P</Variable></V
  alue></Slot>
  <Slot><SlotName>DocumentName</SlotName><V
  alue><Variable>dx</Variable></Value></Slo
  t><Slot><SlotName>HumanName</SlotName><Va
  lue><Variable>RealName</Variable></Value>
  </Slot><Slot><SlotName>ContractID</SlotNa
  me><Value><Variable>ContractID</Variable>
  </Value></Slot></Template>
  </assert>
 </Action>
</RHS>
</Rule>
<Rule  Name="RuleThree"
Documentation="give a name to each party in
the clause">
<LHS>
<Check DocumentName="VariableName:dThree">
  <assert>
   <Template><TemplateName>RealName</Templa
   teName><Slot><SlotName>PartyName</SlotNa
   me><Value><Variable>ByParty</Variable></
   Value></Slot><Slot><SlotName>HumanName</
   SlotName><Value><Variable>ByRealName</Va
   riable></Value></Slot></Template>
  </assert>
</Check>
<Check DocumentName="VariableName:dThree">
 <Offer ID="VariableName:ContractID">
  <Clause By="VariableName:ByParty" To=
  "VariableName:ToParty"
  ID="VariableName:ClauseID">
  <Text><TextVariable
  VariableName="ObligationText"/></Text>
  </Clause>
 </Offer>
</Check>
<Check DocumentName="VariableName:dThree">
 <assert>
  <Template><TemplateName>RealName</Templat
  eName><Slot><SlotName>PartyName</SlotName
  ><Value><Variable>ToParty</Variable></Val
  ue></Slot><Slot><SlotName>HumanName</Slot
  Name><Value><Variable>ToRealName</Variabl
  e></Value></Slot><Slot><SlotName>Document
  Name</SlotName></Slot><Slot><SlotName>Con
  tractID</SlotName><Value><Variable>Contra
  ctID</Variable></Value></Slot></Template>
 </assert>
</Check>
</LHS>
<RHS>
<Action>
 <assert>
  <Template><TemplateName>Obligation</Templ
  ateName><Slot><SlotName>DocumentName</Slo
  tName><Value><Variable>dThree</Variable><
  /Value></Slot><Slot><SlotName>By</SlotNam
  e><Value><Variable>ByRealName</Variable><
  /Value></Slot>
  <Slot><SlotName>To</SlotName><Value><Vari
  able>ToRealName</Variable></Value></Slot>
  <Slot><SlotName>ContractID</SlotName><Val
  ue><Variable>ContractID</Variable></Value
  ></Slot>
  <Slot><SlotName>ClauseID</SlotName><Value
  ><Variable>ClauseID</Variable></Value></S
  lot>
  <Slot><SlotName>ObligationText</SlotName>
  <Value><Variable>ObligationText</Variable
  ></Value></Slot></Template>
 </assert>
</Action>
</RHS>
</Rule>
```

Rule Two will search for the Actors field of the Contract and create a Jess Fact of the form (RealName (PartyName=) (HumanName=)) Observe that the **Offer** and **PartyList** in the **LHS** match against the XML shown in the contract. (**LHS** stands for Left Hand Side and **RHS** stands for Right Hand Side.) However, the statement "**<PARTY ID="VaribleName:P>"** will cause the Jess variable, **P,** to be matched and set to the values in the contract read in. In our contract example, these would be **P001** and **P002**.

Rule Three matches the clauses with the physical name of the party. This enables them to be matched with the names in the lawsuit document. The **assert** and **Template** tags create intermediate facts when they appear in the **RHS**. They match against intermediate facts created by other rules when they appear in the **LHS**. Other tags are matched against the XML In the case of an attribute, the rule-writer simply uses the same attribute name and puts the VariableName:*Jess-Variable-Name*. To match against text within an XML tag, the rule-writer uses: <TextVariable VariableName="*Jess-Variable-Name*">

Other rules, described in[13], match each obligation of the plaintiff against affidavits (submitted in support of the motion for summary judgment). These verify that there is an **Assert Clause** in an affidavit for each and every **Text** clause in the contract. Then, the Prayer in the original complaint is compared with the amount that the contract stated should be paid. From these reasonings, the expert system determines if the summary judgment motion can be granted. To illustrate these, I include the XML version of the final rule below. This ensures that there is a complaint, the plaintiff wants summary judgment, and the plaintiff is legally entitled to summary judgment based on the affidavits and contracts submitted. These each involve checking for Jess facts generated earlier. In addition, we **Test** that the amount requested in the lawsuit's **Prayer** is less than the amount specified that the plaintiff should pay.

```
   <Rule Name="RuleNine" Salience="-99"
Documentation="Check To See if We have a
final Judgment">
   <LHS>
   <Check>
   <assert>
   <Template><TemplateName>Complaint</Templa
teName><Slot><SlotName>Plaintiff</SlotName><
```

```
Value><Variable>PN</Variable></Value></Slot>
<Slot><SlotName>Defendant</SlotName><Value><
Variable>DN</Variable></Value></Slot><Slot><
SlotName>LawsuitNumber</SlotName><Value><Var
iable>d</Variable></Value></Slot><Slot><Slot
Name>Amount</SlotName><Value><Variable>Dolla
rAmount</Variable></Value></Slot></Template>
    </assert>
    </Check>
    <Check>
    <assert>
    <Template><TemplateName>PlaintiffWantsSum
maryJudgment</TemplateName>
    <Slot><SlotName>CaseNumber</SlotName><Val
ue><Variable>d</Variable></Value></Slot>
    </Template>
    </assert>
    </Check>
    <Check>
    <assert>
    <Template><TemplateName>PlaintiffEntitled
toSummaryJudgment</TemplateName><Slot><SlotN
ame>LawsuitNumber</SlotName><Value><Variable
>d</Variable></Value></Slot>
    </Template>
    </assert>
    </Check>
    <Check>
    <assert>
    <Template><TemplateName>MustPay</Template
Name><Slot><SlotName>By</SlotName><Value><Va
riable>DN</Variable></Value></Slot><Slot><Sl
otName>To</SlotName><Value><Variable>PN</Var
iable></Value></Slot><Slot><SlotName>Payment
</SlotName><Value><Variable>Dollar1</Variabl
e></Value></Slot>
    </Template>
    </assert>
    </Check>
    <Check>
    <Test><RelationalOperator>ge</RelationalO
perator>
    <Variable1>Dollar1</Variable1>
    <Variable2>DollarAmount</Variable2>
    </Test>
    </Check>
    </LHS>
    <RHS>
    <Action>
    <assert>
    <Template><TemplateName>Judgment</Templat
eName><Slot><SlotName>To</SlotName><Value><V
ariable>PN</Variable></Value></Slot><Slot><S
lotName>From</SlotName><Value><Variable>DN</
Variable></Value></Slot><Slot><SlotName>Amou
nt</SlotName><Value><Variable>DollarAmount</
Variable></Value></Slot>
    </Template>
    </assert>
    </Action>
    </RHS>
    </Rule>
</RuleBase>
```

3. FUTURE WORK

We will expand our Jess/XML integration so that one can generate XML. The user will specify by putting the XML in the right hand side. Where replacement is to occur, they will use the **VariableName** and **TextVariable.** They will have roles in the right hand side analogous to those they have in the left hand side of rules.

With the capability of generating XML, we will use this expert system tool to read in XML contracts that confirm to industry standards for purchase orders and other business-to-business electronic commerce. The output will be contracts as per the proposals to the Legal XML Contract Work Group for Contract Standards [10] and [11].

The mapping will be described in the Operating Rules Markup Language [12], also proposed by these authors. By doing so, we hope to provide an integrated civil justice system going from contract evaluation to dispute resolution, whether by an arbitrator or a court, according to rules of procedure defined unambiguously by XML and Rule-Based reasoning.

4. ACKNOWLEDGMENTS

This work was done while I was a Visiting Scholar at the Massachusetts Institute of Technology Electronic Commerce Architecture Project during Summer 2000, in cooperation with Daniel Greenwood, Lecturer.

James Keane, in a conversation at the November 2000 Legal XML Face to Face, suggested that I use an XML input format for the rules.

5. REFERENCES

[1] Common Business Library, XML Common Business Library, http://www.xcbl.org

[2] Cover, R., Oasis, The XML Cover Pages, http://www.oasis-open.org/cover/xml.html#xmlSoftware

[3] DeFrances, Carol J., Smith, Steven K., Contract Cases in Large Counties, Bureau of Justice Statistics: Special report, February 1996 (NCJ- 156664)

[4] Friedman-Hill, Ernest, Jess, the Java Expert System, http://herzberg.ca.sandia.gov/jess/

[5] Gardner, A., An Artificial Intelligence Approach to Legal Reasoning, Bradford Books, Cambridge MA, 1987.

[6] Halvorson, M. and Himes, R., XML Standards Development Project Electronic Court Filing Draft Specification, http://www.legalxml.org/DocumentRepository/ProposedStandards/Clear/PS_10001/PS_10001_2000_07_24.htm

[7] IBM, XML Parser for Java, http://alphaworks.ibm.com/aw.nsf/frame?ReadForm&/aw.nsf/techmain/F62DB5F8684DCF6A8825671B00682F34

[8] ILingo.org, The Language of Insurance e-Business, http://www.ilingo.com

[9] Information Week, XML and Java for e-Business, http://www.informationweek.com/817/xml.htm

[10] Leff L., A Standard for Offer and Acceptance, http://www.legalxml.org/contracts/draft_documents/notes/UN_100XX_2000_04_22.html

[11] Leff, L., More On a Contract Standard, (submitted to the Legal XML Contracts Workgroup), http://www.wiu.edu/users/mflll/UN_100XX_2000_08_22.html

[12] Leff, L. and Greenwood, D., The Operating Rules Markup Language, http://ecitizen.mit.edu/ecap1.html

[13] Leff, L., XML/Expert Systems Integration With Applications to Contracts and Litigation, http://ecitizen.mit.edu/ecap3.html

[14] Legal XML, http://www.legalxml.org

[15] Legal XML, XML Standards Development Project Electronic Court Filing Draft Specification, http://www.legalxml.org/DocumentRepository/ProposedStandards/Clear/PS_10001/PS_10001_2000_07_24.htm

[16] Levitt, J. The Language of e-Business, Information Week, December 18/25, 2000. (also available as http://www.informationweek.com/817/xml.htm)

[17] Lemke, T., New Computer Language to Streamline Business-to-Business e-Commerce, Chicago Tribune, Section 4, Page 8, December 4 2000.