Representing Agent Contracts with Exceptions using XML Rules, Ontologies, and Process Descriptions

Presentation of Paper at the International Workshop on Rule Markup Languages for Business Rules on the Semantic Web, held in conjunction with the 1st International Semantic Web Conference, June 14, 2002, Sardinia, Italy

Benjamin Grosof

MIT Sloan School of Management

bgrosof@mit.edu   http://www.mit.edu/~bgrosof/

Terrence Poon

MIT Computer Science

tpoon@alum.mit.edu

Examples of Contract Provisions Well-Represented by Rules in Automated Deal Making

- Product descriptions
  - Product catalogs: properties, conditional on other properties.
- Pricing dependent upon: delivery-date, quantity, group memberships, umbrella contract provisions
- Terms & conditions: refund/cancellation timelines/deposits, lateness/quality penalties, ordering lead time, shipping, creditworthiness, biz-partner qualification, service provisions
- Trust
  - Creditworthiness, authorization, required signatures
- *Buyer Requirements (RFQ, RFP) wrt the above*
- *Seller Capabilities (Sourcing, Qualification) wrt the above*

What Can Be Done with the Rules in contracting, & negotiation, based on our SweetDeal approach to rule representation

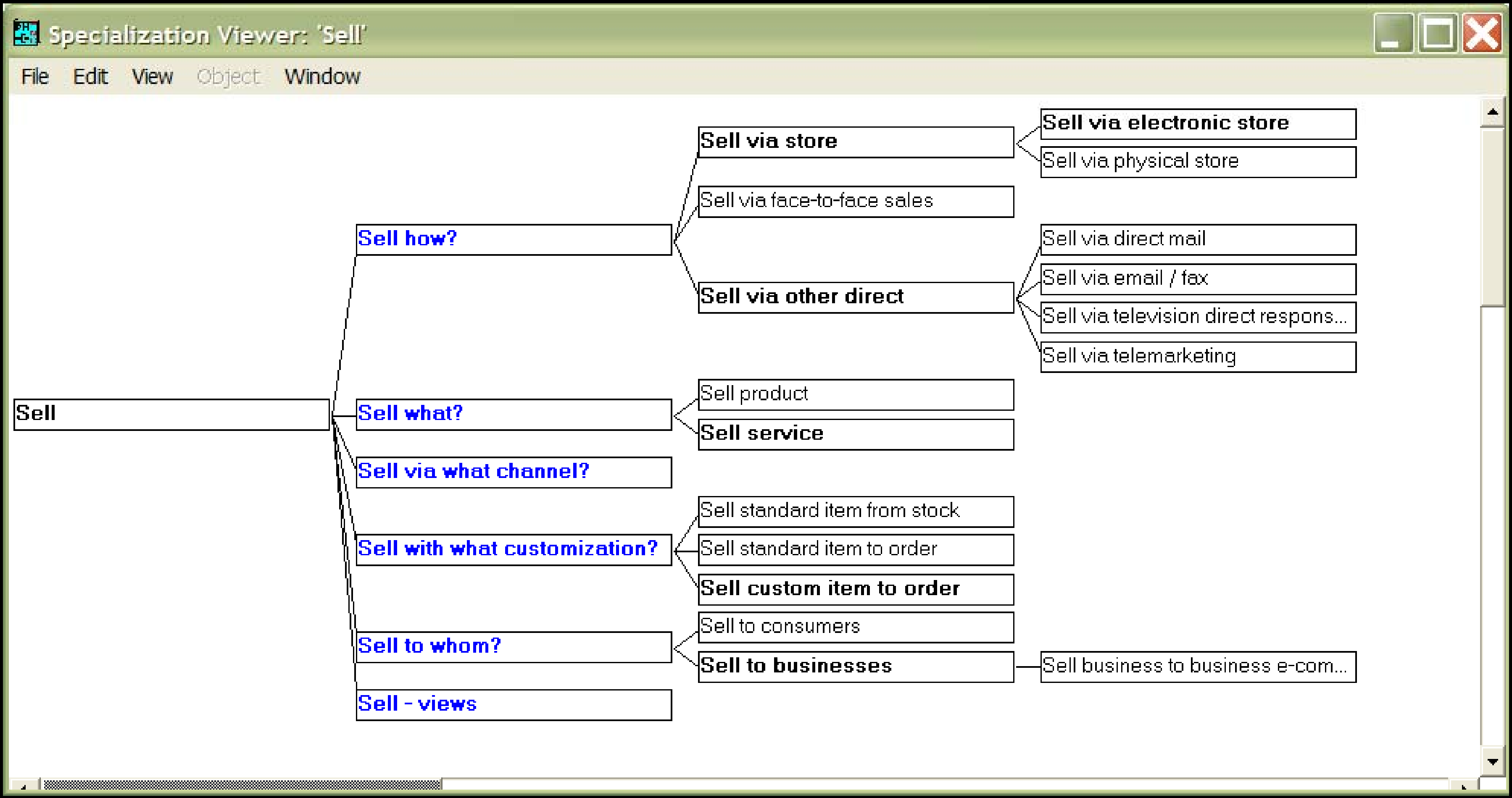
- Communicate: with deep shared semantics
  - via RuleML, inter-operable with same sanctioned inferences
  - ⇔ heterogeneous rule/DB systems / rule-based applications (“agents”)
- Execute contract provisions:
  - infer; ebiz actions; authorize; ...
- Modify easily: contingent provisions
  - default rules; modularity; exceptions, overriding
- Reason about the contract/proposal
  - hypotheticals, test, evaluate; tractably
  - *(also need “solo” decision making/support by each agent)*

Overview I: SweetDeal, Exception Handlers, Web Services

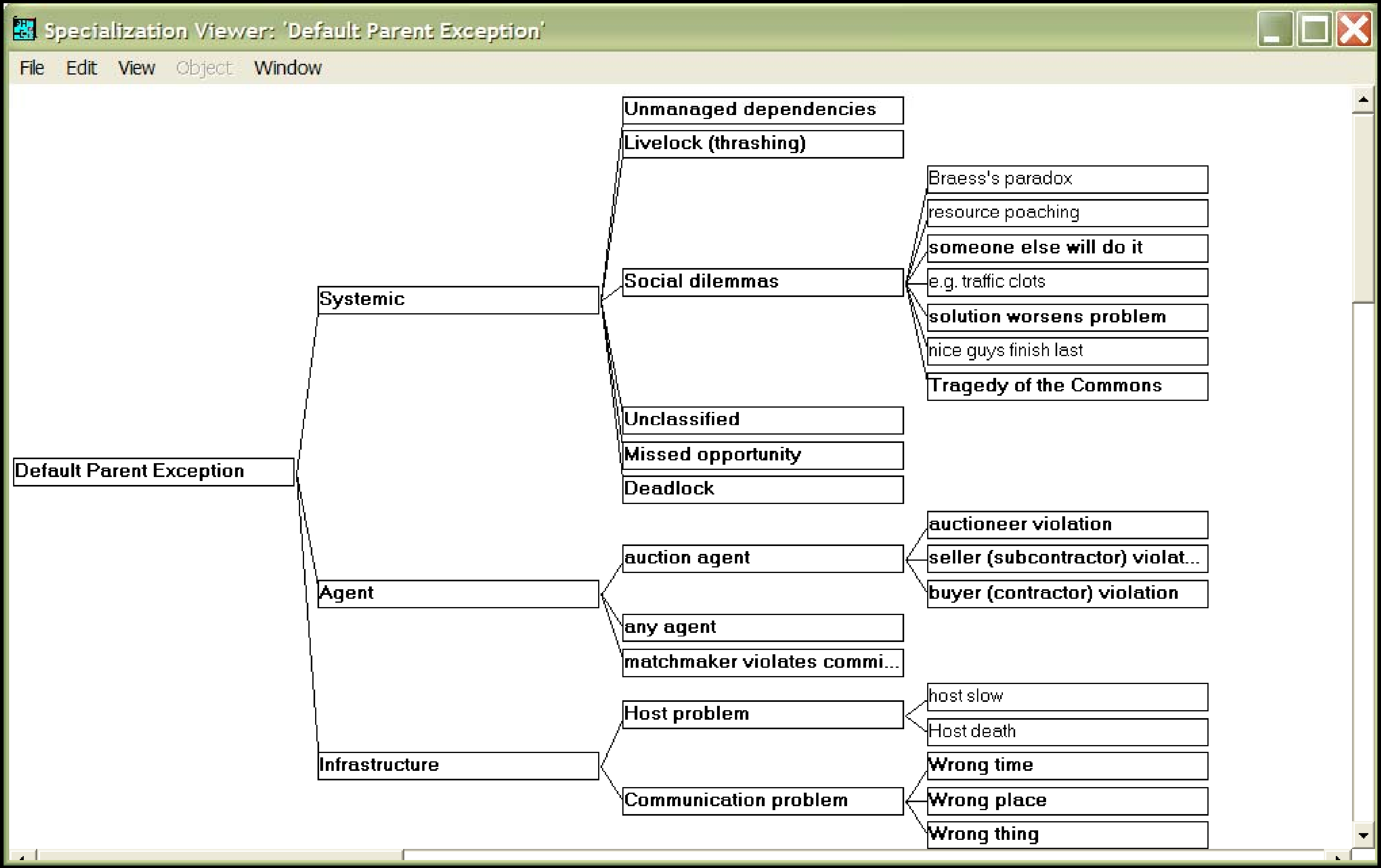
- This work is part of **SweetDeal**: rule-based approach for e-contracting
- Advantages of rule-based: (use Situated Courteous LP KR in RuleML)
  - high level of conceptual abstraction to specify; modularly modifiable; reusable; executable
  - esp. good for specifying *contingent* provisions

- Here, newly extend to include exception handlers:
  - = violations of commitments $\rightarrow$ invoke business processes
  - more complex behavior
  - good for services, e.g., **deals about Web services**
  - process descriptions whose ontologies are in DAML+OIL
    - drawn from MIT Process Handbook, a previous repository
      - uniquely large & well-used (by industry biz process designers)
  - partially or fully specified by rules (executably)

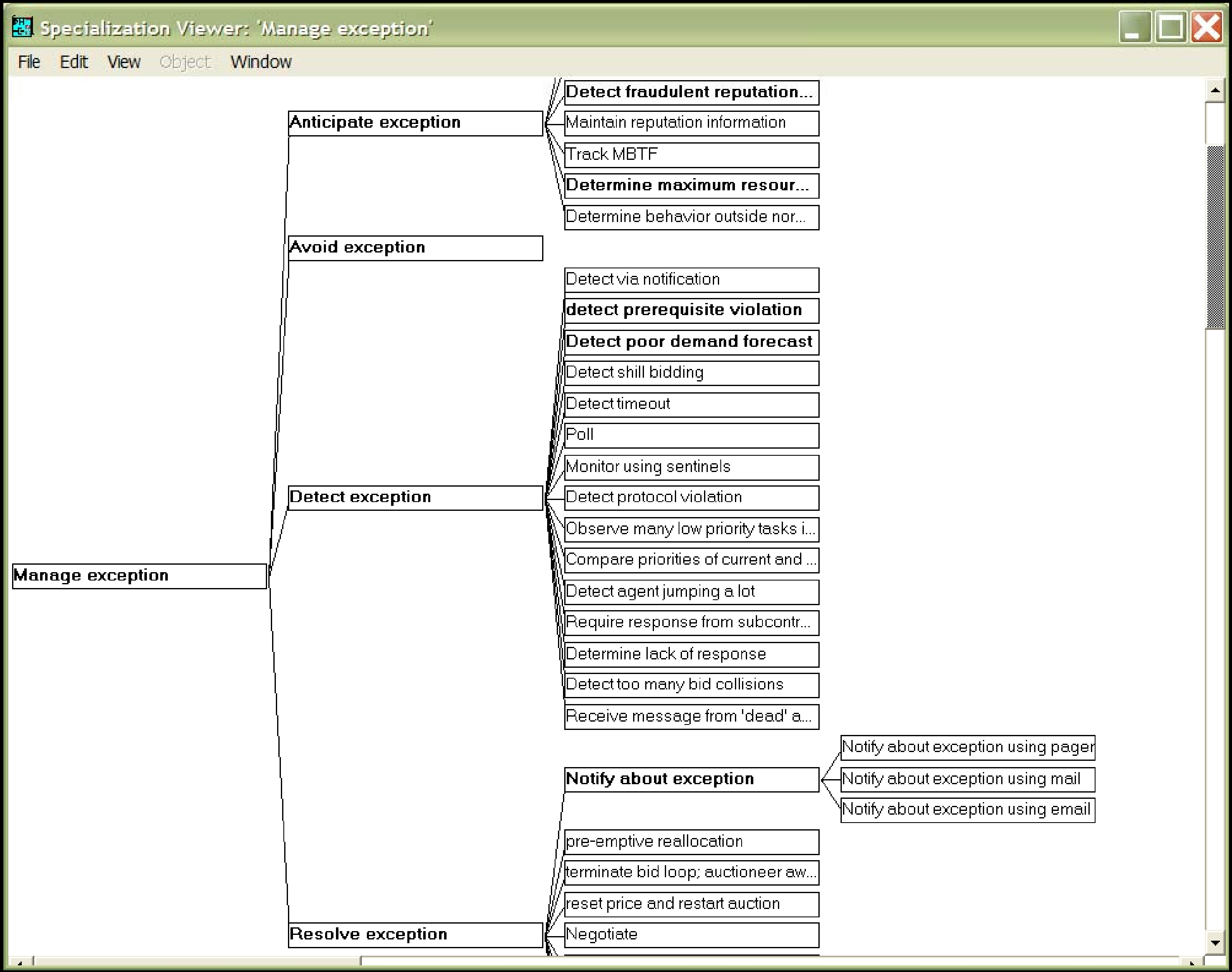
Some Specializations of "Sell" in the MIT Process Handbook (PH)



Some Exceptions in the MIT Process Handbook



Some exception handlers in the MIT Process Handbook



Manage exception
- Anticipate exception
  - Detect fraudulent reputation...
  - Maintain reputation information
  - Track MBTF
  - Determine maximum resour...
  - Determine behavior outside nor...
- Avoid exception
- Detect exception
  - Detect via notification
  - detect prerequisite violation
  - Detect poor demand forecast
  - Detect shill bidding
  - Detect timeout
  - Poll
  - Monitor using sentinels
  - Detect protocol violation
  - Observe many low priority tasks i...
  - Compare priorities of current and ...
  - Detect agent jumping a lot
  - Require response from subcontr...
  - Determine lack of response
  - Detect too many bid collisions
  - Receive message from 'dead' a...
- Resolve exception
  - Notify about exception
    - Notify about exception using pager
    - Notify about exception using mail
    - Notify about exception using email
  - pre-emptive reallocation
  - terminate bid loop; auctioneer aw...
  - reset price and restart auction
  - Negotiate

Representing PH Process Ontology in DAML+OIL: Some Main Concepts

Define pr.daml

```
<daml:Class rdf:ID="Process">
  <rdfs:comment>A process</rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="CoordinationMechanism">
  <rdfs:comment>A process that manages activities between multiple
agents</rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="Exception">
  <rdfs:comment>A violation of an inter-agent commitment</rdfs:comment>
</daml:Class>

<daml:Class rdf:ID="ExceptionHandler">
  <rdfs:subClassOf rdf:resource="#Process"/>
  <rdfs:comment>A process that helps to manage a particular
exception</rdfs:comment>
</daml:Class>
```

Representing PH Process Ontology in DAML+OIL: More

```
<daml:ObjectProperty rdf:ID="hasException">
  <rdfs:comment>Has a potential exception</rdfs:comment>
  <rdfs:domain rdf:resource="#Process" />
  <rdfs:range rdf:resource="#Exception" />
</daml:ObjectProperty>

<daml:ObjectProperty rdf:ID="isHandledBy">
  <rdfs:comment>Can potentially be handled by, in some way </rdfs:comment>
  <rdfs:domain rdf:resource="#Exception" />
  <rdfs:range rdf:resource="#ExceptionHandler" />
</daml:ObjectProperty>
...
<daml:Class rdf:ID="ContractorDoesNotPay">
   <rdfs:subClassOf rdf:resource="#ContractorViolation"/>
   <rdfs:subClassOf>
    <daml:Restriction>
        <daml:onProperty rdf:resource="#isHandledBy"/>
        <daml:hasClass rdf:resource="#ProvideSafeExchangeProtocols"/>
     </daml:Restriction>
  </rdfs:subClassOf>
</daml:Class>
```

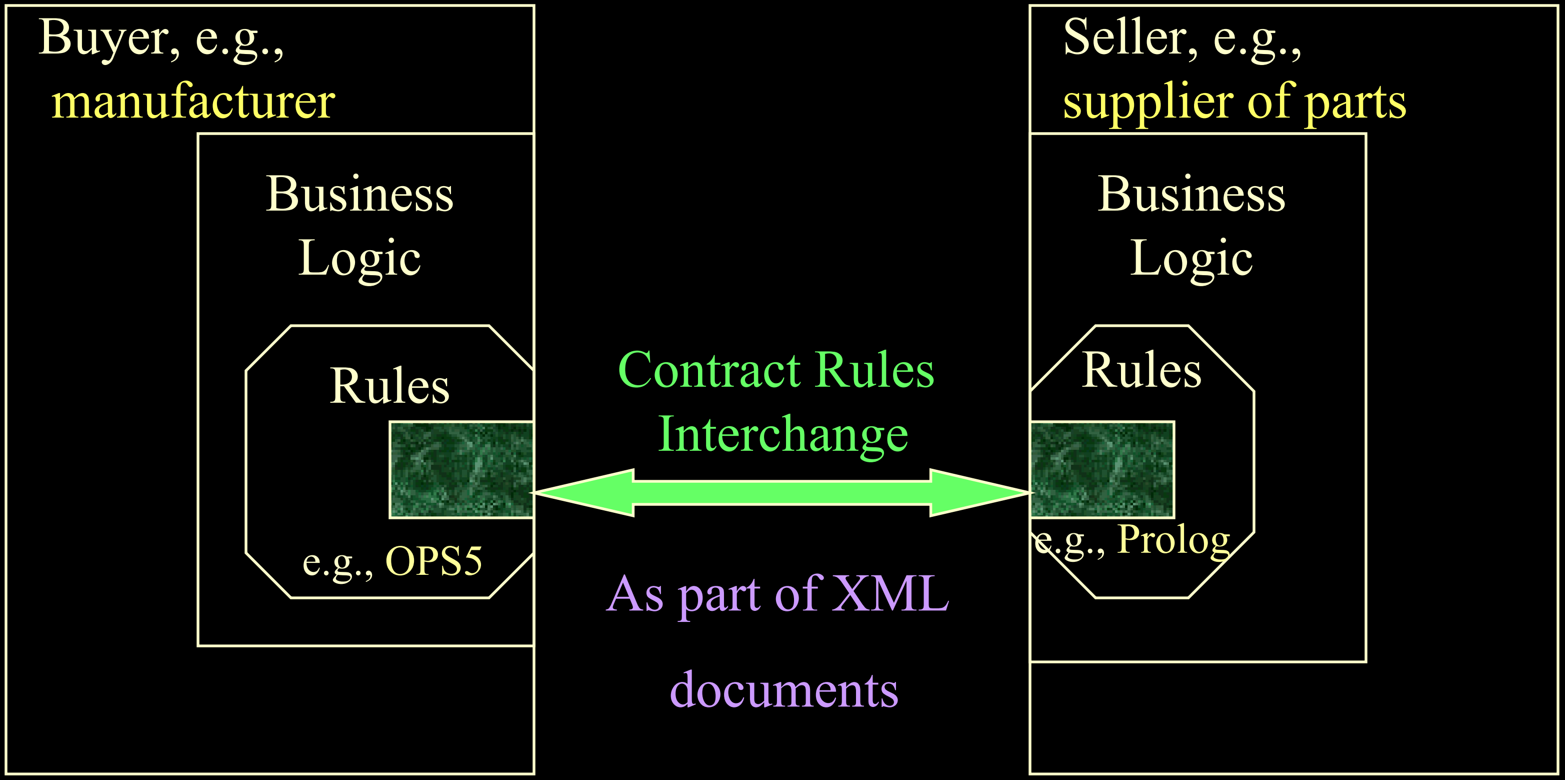
Representing New Contract Ontology in DAML+OIL

Define sd.daml

(imports pr.daml)

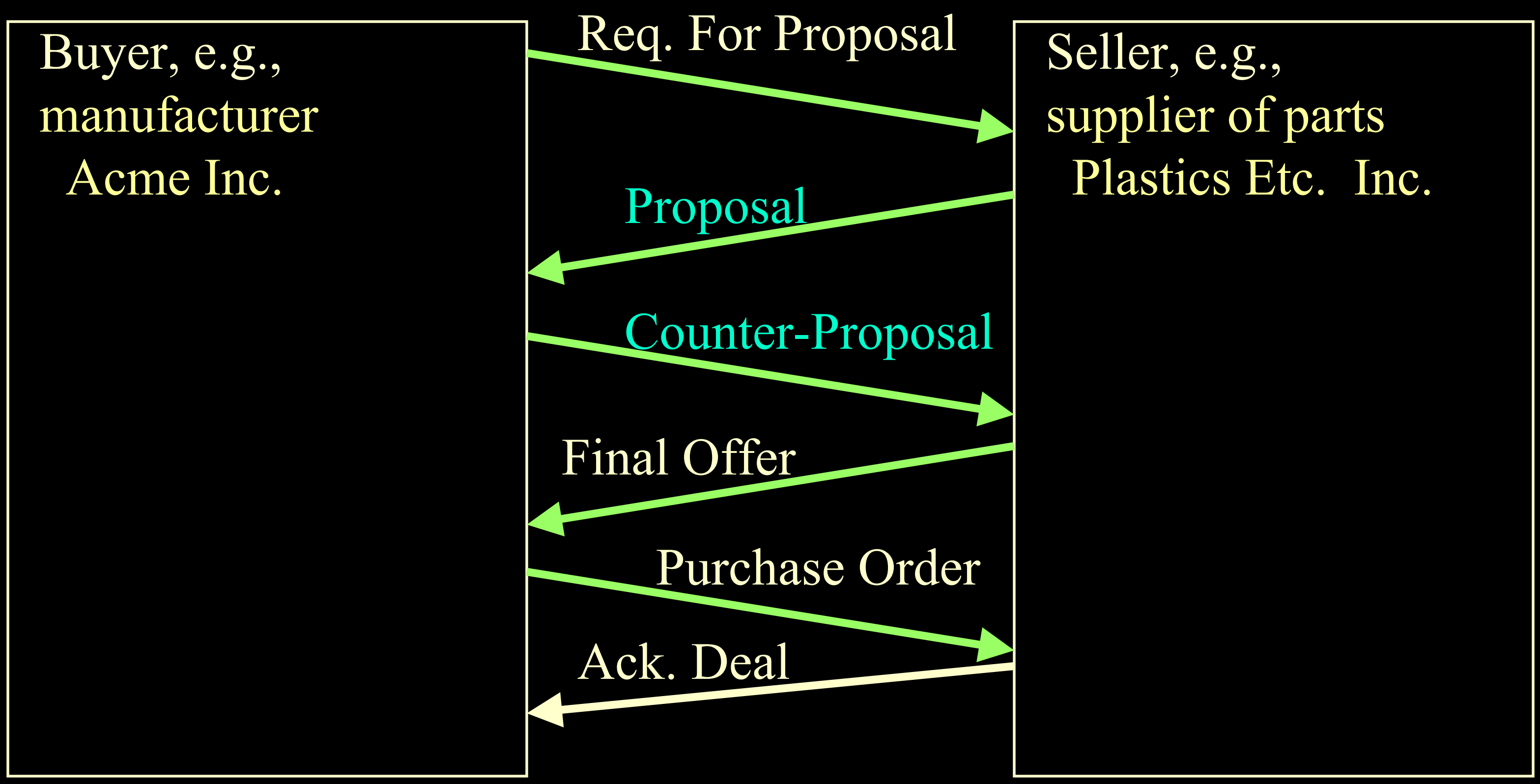
```
<daml:Class rdf:ID="Contract">
  <rdfs:subClassOf>
    <daml:Restriction daml:minCardinality="1">
      <daml:onProperty rdf:resource="#specFor"/>
    </daml:Restriction>
  </rdfs:subClassOf>
</daml:Class>

<daml:ObjectProperty rdf:ID="specFor">
  <rdfs:domain rdf:resource="#Contract" />
  <rdfs:range rdf:resource="http://xmlcontracting.org/pr.daml#Process"/>
</daml:ObjectProperty>

<daml:Class rdf:ID="ContractResult"/>

<daml:ObjectProperty rdf:ID="result">
  <rdfs:domain rdf:resource="#Contract" />
  <rdfs:range rdf:resource="#ContractResult" />
</daml:ObjectProperty>
```

Contract Rules during Negotiation



Contracting parties NEGOTIATE via shared rules.

Exchange of Rules Content during Negotiation: example


Buyer, e.g., manufacturer Acme Inc.
Seller, e.g., supplier of parts Plastics Etc. Inc.
Req. For Proposal
Proposal
Counter-Proposal
Final Offer
Purchase Order
Ack. Deal

Example Contract Proposal with Exception Handling
Represented using RuleML & DAML+OIL, Process Descriptions

Using concise text syntax

(SCLP textfile format)

for concise human reading

```
buyer(co123,acme);
seller(co123,plastics_etc);
product(co123,plastic425);
price(co123,50);
quantity(co123,100);
http://xmlcontracting.org/sd.daml#Contract(co123);
http://xmlcontracting.org/sd.daml#specFor(co123,co123_process);
http://xmlcontracting.org/sd.daml#BuyWithBilateralNegotiation(co123_process);
http://xmlcontracting.org/sd.daml#result(co123,co123_res);
shippingDate(co123,3); // i.e. 3 days after order placed
// base payment = price * quantity
payment(?R,base,?Payment) <-
  http://xmlcontracting.org/sd.daml#result(co123,?R) AND
  price(co123,?P) AND quantity(co123,?Q) AND
  multiply(?P,?Q,?Payment) ;
```

SCLP TextFile Format for (Daml)RuleML

```
payment(?R,base,?Payment) <-
http://xmlcontracting.org/sd.daml#result(co123,?R) AND
price(co123,?P) AND quantity(co123,?Q) AND
multiply(?P,?Q,?Payment) ;
```

drm = namespace for damlRuleML

```
<drm:imp>
  <drm:_head> <drm:atom>
     <drm:_opr><drm:rel>payment</drm:_opr></drm:rel>    <drm:tup>
     <drm:var>R</drm:var> <drm:ind>base</drm:ind> <drm:var>Payment</drm:var>
  </drm:tup></drm:atom> </drm:_head>
  <drm:_body>
   <drm:andb>
     <drm:atom> <drm:_opr>
       <drm:rel href= "http://xmlcontracting.org/sd.daml#result"/>
         </drm:_opr> <drm:tup>
           <drm:ind>co123</drm:ind> <drm:var>Cust</drm:var>
          </drm:tup> </drm:atom>

... </drm:andb> </drm:_body>  </drm:imp>
```

Example Contract Proposal, Continued

- Buyer adds rule modules to the contract proposal to specify:
  - 1. detection of an exception
    - LateDelivery as a potential exception of the contract's process
    - detectLateDelivery as exception handler: recognize occurrence
  - 2. avoidance of an exception (and perhaps also resolution of the exception)
    - lateDeliveryPenalty as exception handler: penalize per day

- Rule module = a nameable ruleset → a subset of overall rulebase
  - can be included directly and/or imported via link; nestable
    - similar to legal contracts' "incorporation by reference"
  - an extension to RuleML; in spirit of "Webizing"

Example Contract Proposal, Continued: lateDeliveryPenalty exception handler module

```
lateDeliveryPenalty_module {
// lateDeliveryPenalty is an instance of PenalizeForContingency
//   (and thus of AvoidException, ExceptionHandler, and Process)
http://xmlcontracting.org/pr.daml#PenalizeForContingency( lateDeliveryPenalty) ;
// lateDeliveryPenalty is intended to avoid exceptions of class
// LateDelivery.
http://xmlcontracting.org/sd.daml#avoidsException(lateDeliveryPenalty,
  http://xmlcontracting.org/pr.daml#LateDelivery);
// penalty = - overdueDays * 200 ; (negative payment by buyer)
<lateDeliveryPenalty_def> payment(?R, contingentPenalty, ?Penalty) <-
  http://xmlcontracting.org/sd.daml#specFor(?CO,?PI) AND
  http://xmlcontracting.org/pr.daml#hasException(?PI,?EI) AND
  http://xmlcontracting.org/pr.daml#isHandledBy(?EI,lateDeliveryPenalty) AND
  http://xmlcontracting.org/sd.daml#result(?CO,?R) AND
  http://xmlcontracting.org/sd.daml#exceptionOccurred(?R,?EI) AND
  shippingDate(?CO,?CODate) AND shippingDate(?R,?RDate) AND
  subtract(?RDate,?CODate,?OverdueDays) AND
  multiply(?OverdueDays, 200, ?Res1) AND multiply(?Res1, -1, ?Penalty) ;
}
<lateDeliveryPenaltyHandlesIt(e1)> // specify lateDeliveryPenalty as a handler for e1
 http://xmlcontracting.org/pr.daml#isHandledBy(e1,lateDeliveryPenalty);
```

Example, Continued: Counter-Proposal

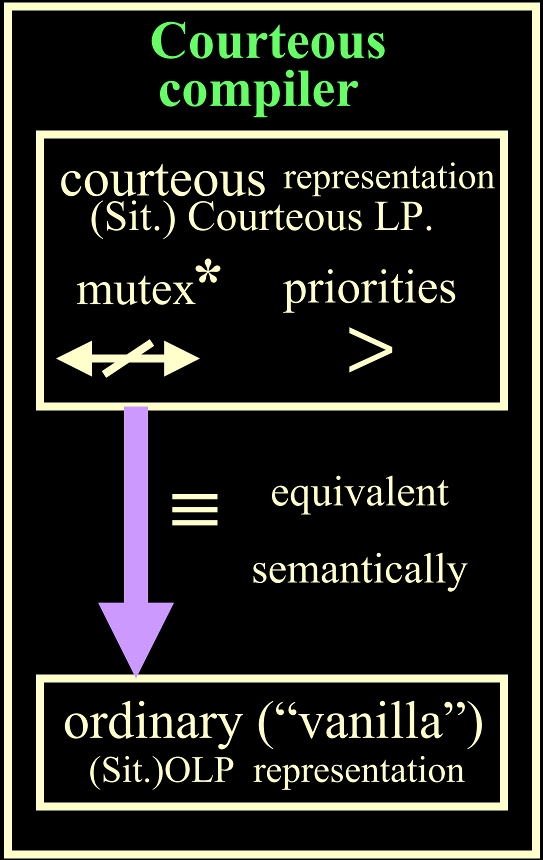
- Seller modifies the draft contract (it's a *negotiation*!)
- Simply adds* another rule module to specify:
  - lateDeliveryRiskPayment as exception handler
    - lump-sum in advance, based on average lateness
      - instead of proportional to actual lateness
  - higher-priority for that module than for the previous proposal, e.g., higher than lateDeliveryPenalty's rule module

- Courteous LP's prioritized conflict handling feature is used
- *NO *change* to previous proposal's rules needed!
  - similar to legal contracts' accumulation of provisions

Example Counter-Proposal's ruleset's prioritized conflict handling

```
// priority specified via syntactically reserved "overrides" predicate

 overrides(lateDeliveryRiskPaymentHandlesIt(e1),
         lateDeliveryPenaltyHandlesIt(e1) ) ;




// There is at most one avoid handler for a given exception instance.
// Consistency is enforced wrt this "mutex" integrity constraint.

  MUTEX
  http://xmlcontracting.org/pr.daml#isHandledBy(?EI, ?EHandler1) AND
  http://xmlcontracting.org/pr.daml#isHandledBy(?EI, ?Ehandler2)
GIVEN
  http://xmlcontracting.org/sd.daml#AvoidException(?Ehandler1) AND
  http://xmlcontracting.org/sd.daml#AvoidException(?Ehandler2) ;
```

Courteous feature: compileable, tractable



Tractable compilation:

O(n^3), often linear

Tractable inference: e.g., worst-case when no ctor's ("Datalog") & bounded v = |var's per rule| is equivalent to OLP with v $\rightarrow$ (v+2)

Preserves ontology.

Plus extra predicates for

- phases of prioritized argumentation (refutation, skepticism)
- classical negations

* classical negation too

Overview II: More New Contributions

- 1. Combine Situated Courteous Logic Programs (SCLP) case of RuleML with DAML+OIL; i.e., SCLP + Description Logic (DL)
  - rules "on top of" ontologies
  - show how and why to do as representational style (KR, syntax)
    - DAML+OIL class or property used as predicate in RuleML
      - heavily exploit feature of RuleML that predicate can be a URI
    - in progress: deeper semantics of the combination
  - more generally, 1st combo of nonmon RuleML / SCLP with DL
  - 1st combo of nonmon rules + DL (also Antoniou, independently)
- 2. Combine further with process descriptions
- 1st substantial practical e-business application domain scenario for 1., 2.
- Point of convergence between Semantic Web and Web Services
- 1st: approach to automate MIT Process Handbook using: a) XML ; b) powerful KR (but encoded only small fraction of its content so far!)
  - underline incapacity of DAML+OIL to represent default inheritance

Related Work: Ours & Theirs

- Previous Work on SweetDeal
  - Rule-based Approach; Requirements analysis for SW rule KR for e-contracting & e-business
  - ContractBot + AuctionBot: negotiation, auction configuration
  - EECOMS $29Million industry pilot on manufacturing supply chain: negotiation
- Recent Work on SweetDeal:
  - Contract fragments, with queryable repository
    - modules inclusion & naming: new technical aspects for RuleML
  - Contract-proposer "market" agent: GUI, with rule-based backend; semi-automated creation, modification, communication, inferencing
- Prototype running; publicly available soon
- Future Directions: Larger Projects:
  - Rule KR Technologies, esp. for Semantic Web Services
    - Current work: theory of {Description Logic $\cup$ LP}
  - Business Applications of Semantic Web Services
    - Deal Level of SW/Services; B2B, policies, supply chain, finance

DAML-S, WSMF

Antoniou '02

More Current & Future Work

- Representing Default Inheritance in Ontologies

- Relating to Semantic Web Services elements:
  - SOAP, UDDI, WSDL
  - DAML-S, WSMF;   WSFL/Xlang, …
  - E-Business/Agent Messaging, e.g., ebXML, UBL

- Relation to Legal aspects of Contracting   ; Legal XML