Automating Law in the Small: Contracts, Regulations, and Prioritized Argumentation

Invited Talk for the

International Conference on Artificial Intelligence and Law (ICAIL-2001)
held at Washington University, St. Louis, MO, USA, May 21-25, 2001

Prof. Benjamin Grosof

Information Technology group,

MIT Sloan School of Management

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

Overview

- *Aiming to be provocative*
- What is Law in the Small
  - example: e-signatures: issues & opportunities
- 1st Steps: Automating Agent Contracting
  - Approach: Inter-operable XML Rules represent parts of Contract Content
  - knowledge representation: declarative rules in XML
    - specify, infer/act, assemble, evaluate, modify
  - value of prioritized default reasoning/argumentation
    - pragmatics, modularity ; via Courteous Logic Programs
- Discussion: Directions for the Glorious Future
  - regulations, bureaucratic policies & processes

What is "Law *in the Small*"

- "Hum-drum": agreements, "rules & regulations"
- contracts; e-signatures; authorizations
- regulations; bureaucratic forms, processes
- routine, but lots of details to be worked out & dealt with
  - what we deal with every day
  - not intrinsically controversial, usually doesn't → court
  - no TV channels or shows, lacks glamour
- goal: minimize run-time human lawyer labor
- represent business policies and processes, many of which have legal aspects or legal weight

Law in the Small (continued)

- *Dream: Automate it*
  - specify
    - modify
  - infer
    - act, decide
  - communicate
    - find relevant

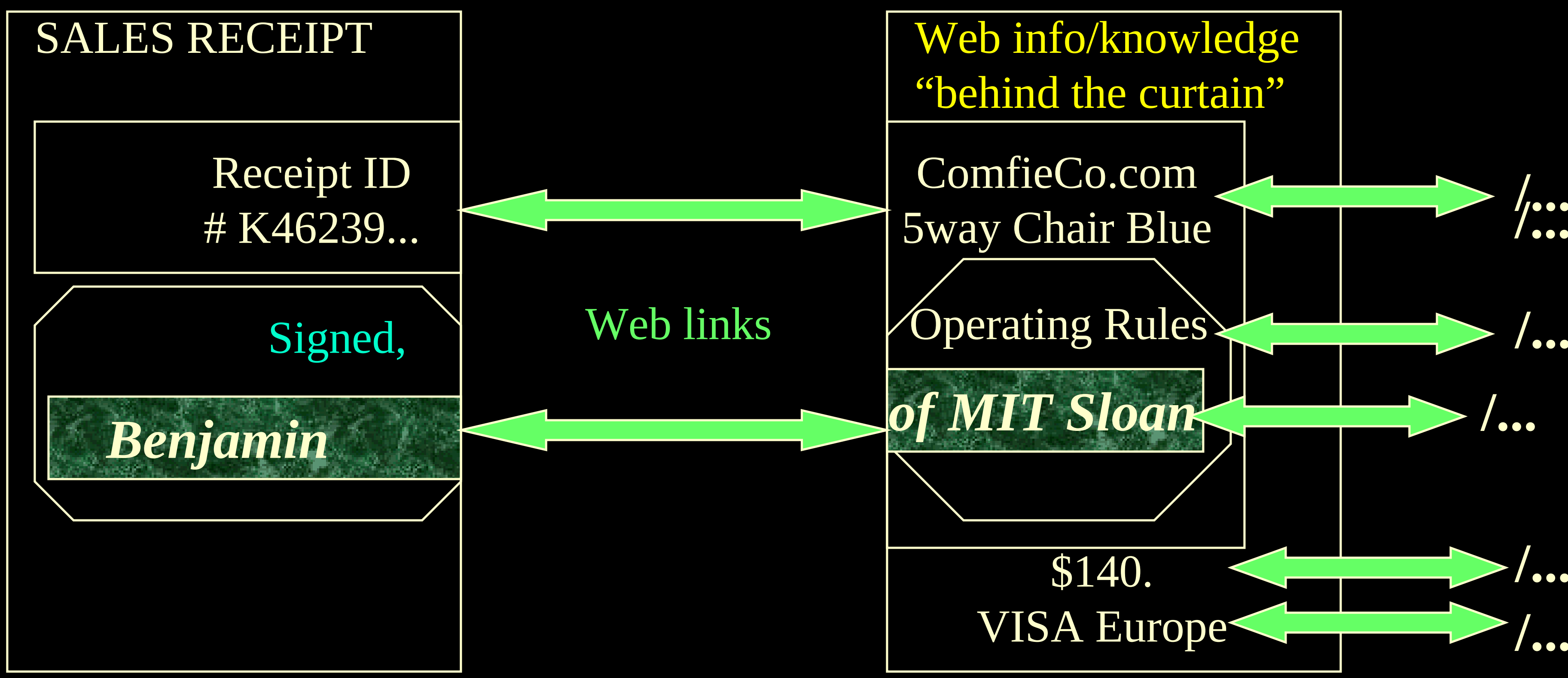
Deeper Issues of E-Signatures

- WHAT'S THE DEAL ? ... !!
- SIGN AS WHAT ?? ... !!

- *Vision/Approach:* A net of documents combined by links, on the Web

Looks Simple To Start... then Gets Interestingly Precise

A Vision/Approach of what Web & Agents enable



Intelligent Agents in Web E-Commerce

- *Today:* especially in the discovery phase of shopping
  - sales agents: recommend products, target ads
  - buyer agents: find vendors; compare offers on price, delivery, and availability
- *Coming soon to a world near you:…*
  - billions/trillions of agents
  - ...with smarts: knowledge gathering, reasoning, economic optimization
  - ...doing our bidding
    - but with some autonomy

Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

Automating Contracting

- "Contract" in broad sense: = offering or agreement.
- "Automate" in deep sense: =
  - 1. <u>Communicatable</u> automatically.
  - 2. <u>Executable</u> within appropriate context of contracting parties' business processes.
  - 3. <u>Evaluable</u> automatically by contracting parties.
    - "reason about it".
  - 4. <u>Modifiable</u> automatically by contracting parties.
    - negotiation, auctions.

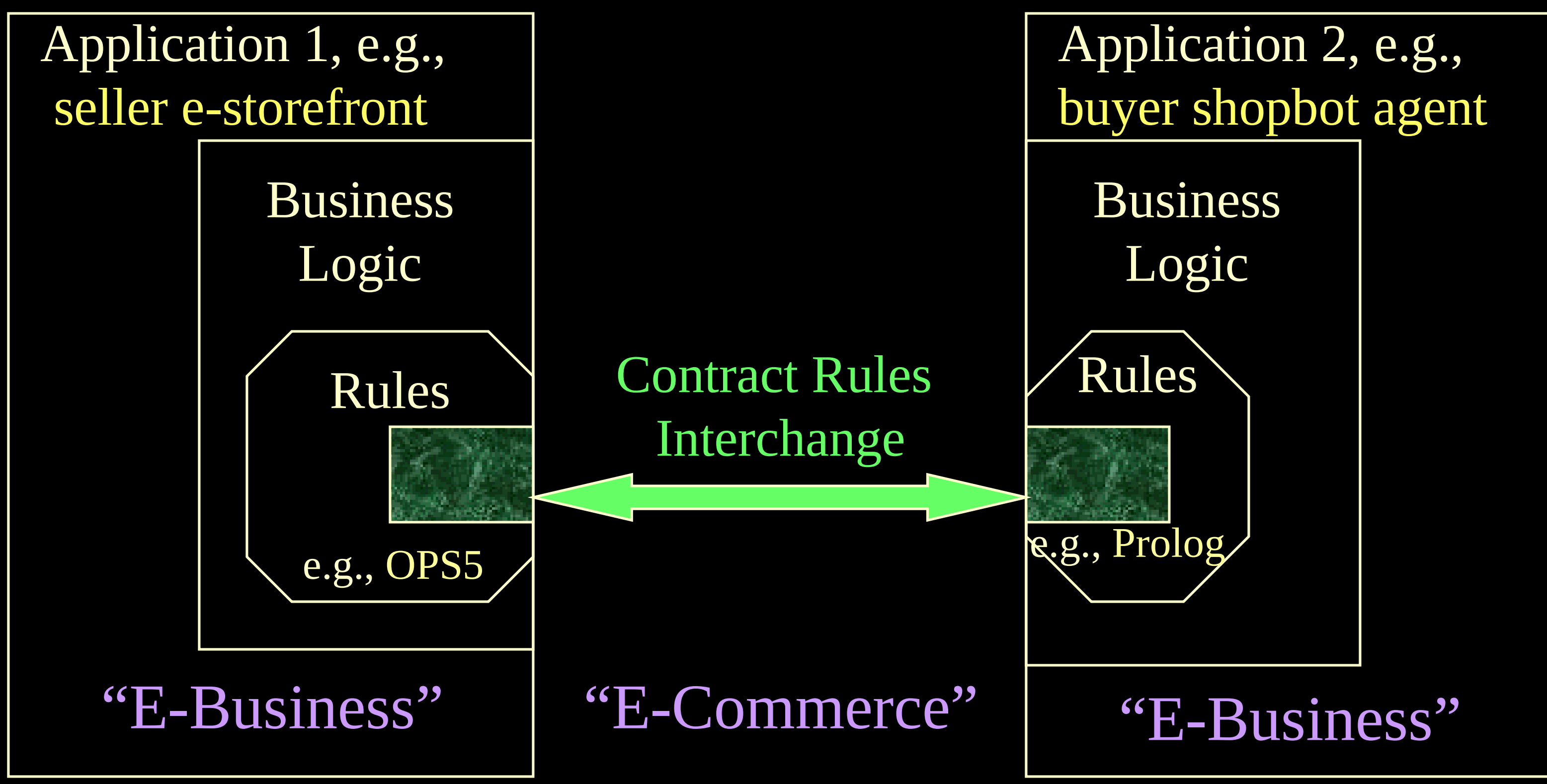
Approach: *Rule-based Contracts for E-commerce*

- Rules as way to specify (part of) business processes, policies, products: as (part of) contract terms.
- Complete or partial contract.
  - As default rules. Update, e.g., in negotiation.
- Rules provide high level of conceptual abstraction.
  - easier for non-programmers to understand, specify, dynamically modify & merge. E.g.,
  - by multiple authors, cross-enterprise, cross-application.
- Executable. Integrate with other rule-based business processes.

Examples of Rules in Agent Contracts & Deal Making

- Product descriptions
  - Product catalogs: properties, conditional on other properties.
- Price vs. quantity vs. delivery date.
  - Discounting, incl. for groups.
- Terms & conditions
  - Service provisions
  - Refunds, cancellations.
  - Surrounding business processes, e.g., lead time to order.
- Trust
  - Creditworthiness, authorization, required signatures
- *Buyer Requirements (RFQ, RFP) wrt the above*
- *Seller Capabilities (Sourcing, Qualification) wrt the above*

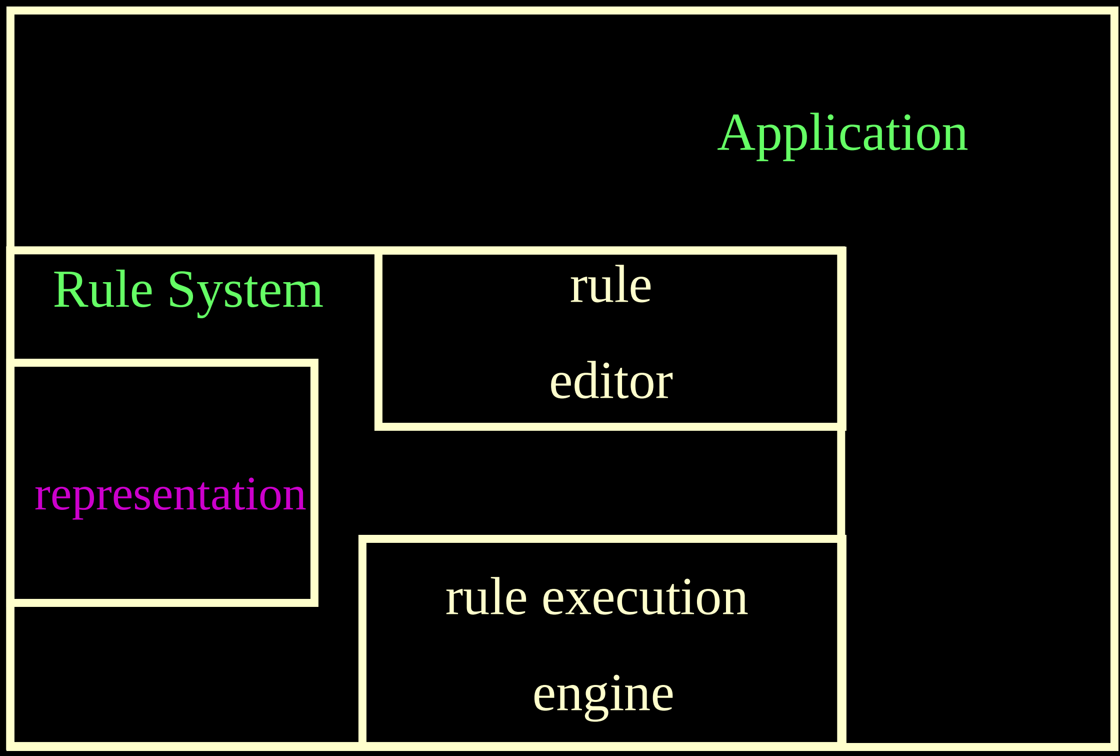
Contract Rules across Applications / Enterprises



Contracting parties integrate e-businesses via shared rules.

Application Using Rules

•



 by Benjamin Grosof

ZOOM-OUT: *larger Vision: rules in e-business overall*

- Rules as an important aspect of coming world of Internet e-business: rule-based business policies & business processes, for B2B & B2C.
  - represent seller's offerings of products & services, capabilities, bids; map offerings from multiple suppliers to common catalog.
  - represent buyer's requests, interests, bids; $\rightarrow$ matchmaking.
  - represent sales help, customer help, procurement, authorization/trust, brokering, workflow.
  - high level of conceptual abstraction; easier for non-programmers to understand, specify, dynamically modify & merge.
  - executable but can treat as data, separate from code
    - potentially ubiquitous; already wide: e.g., SQL views, queries.
- Rules in communicating applications, e.g., embedded intelligent agents.

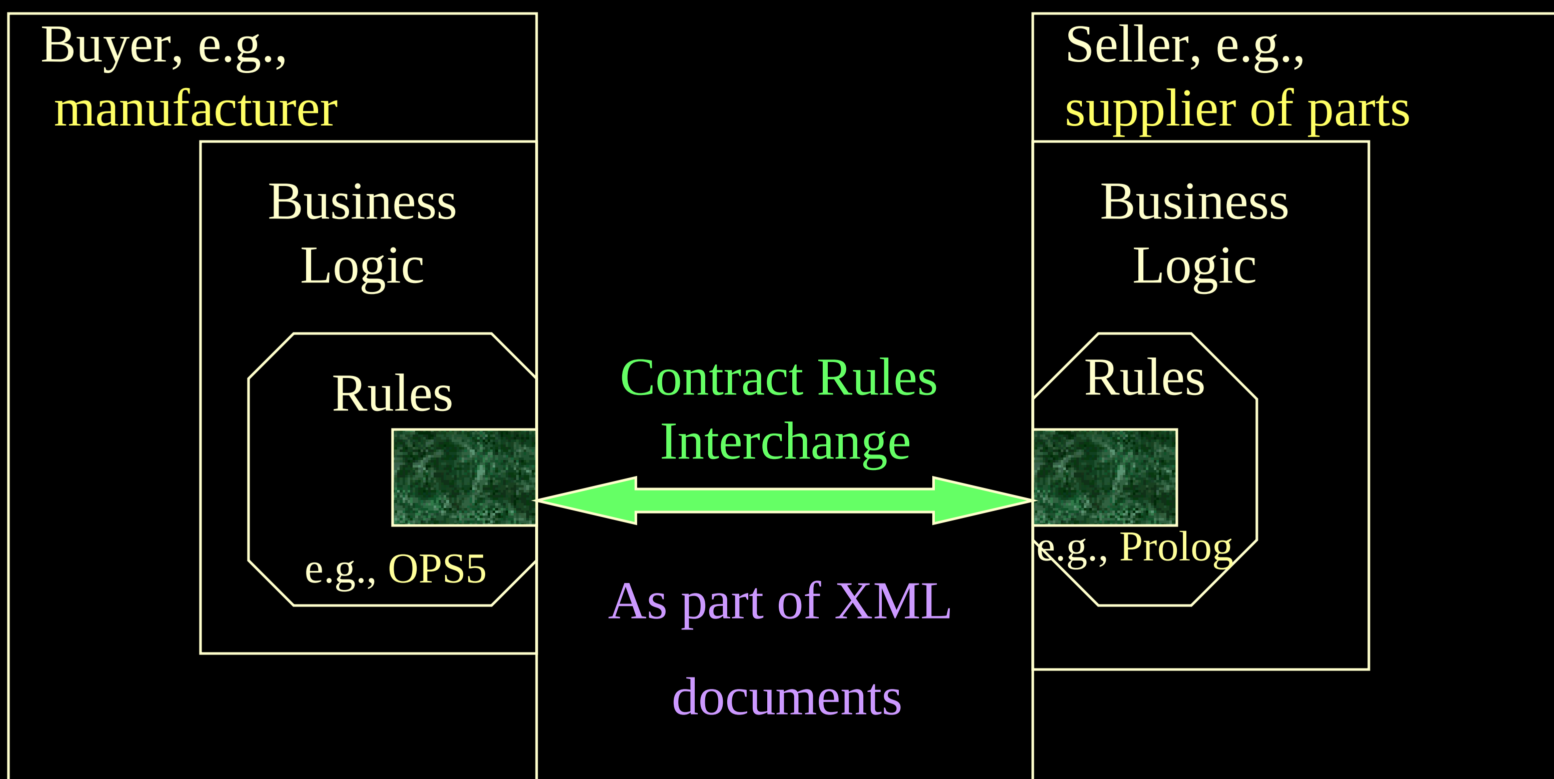
Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

Bidding in Negotiation
(e.g., in manufacturing supply chain)

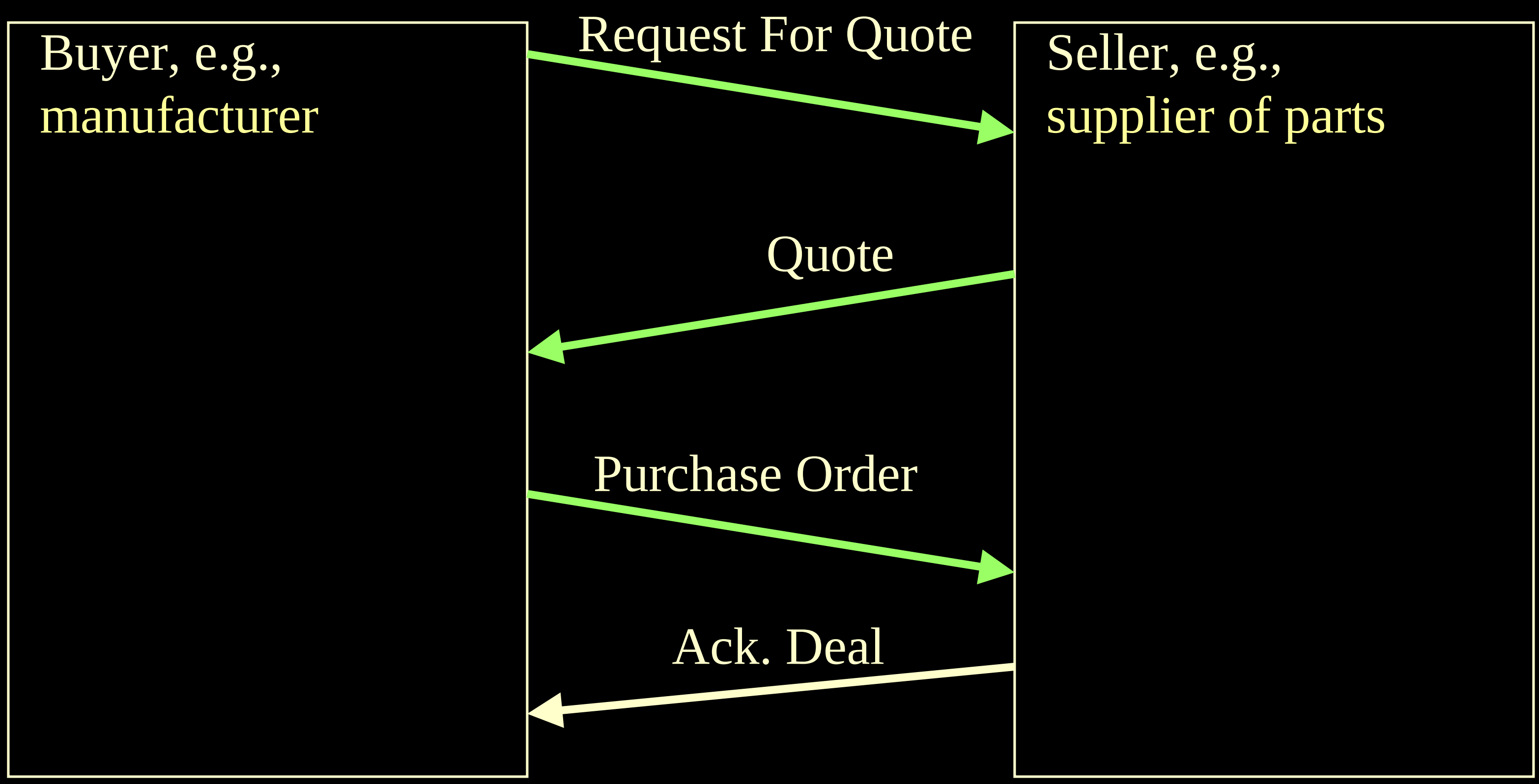
- Use Interlingua to represent contents of:
  - Requests For Quotation or Proposal, i.e., statements of buyer interests, that initiate negotiation, esp. inter-enterprise in B2B.
  - responses to such RFQ's / RFP's by seller: bids, proposals, quotes, ….
  - proposals and counter-proposals and "side information" exchanged during back-and-forth negotiation / bargaining between buyer and seller.
  - *In short: content of bids and requests for bids:*
    - partial then complete.
  - statements of seller/supplier capabilities/interests, e.g., important for source selection as well as bargaining.

Contract Rules during Negotiation

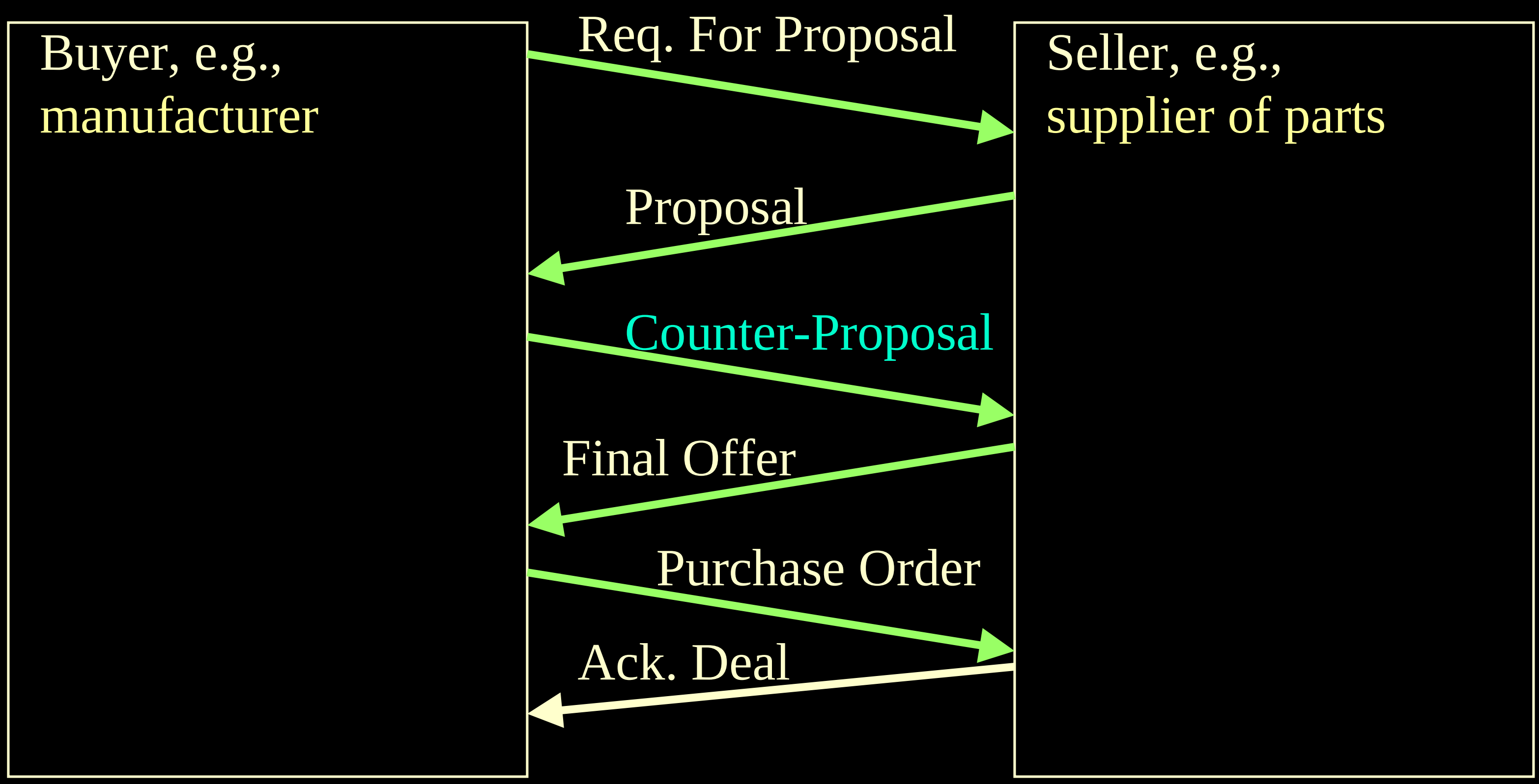


Contracting parties NEGOTIATE via shared rules.

Exchange of Rules Content during Negotiation: example



Exchange of Rules Content during Negotiation: example



 by Benjamin Grosof

Negotiation Example XML Document: Proposal from supplierCo to manufCo

- <negotiation_message>
-   <message_header>
-       <proposal/>
-       <from> supplierCo </from>
-       <to> ManufCo </to>
-    </message_header>
-    <rules_content>
-          …[see next slide]
-    </rules_content>
-    …
- </negotiation_message>
-
- Example of similar message document format:
- FIPA Agent Communication Markup Language (draft industry standard).

Negotiation Ex. Doc. Rules: Proposal from supplierCo to manufCo

- …
  <usualPrice> price(per_unit, ?PO, \$60) $\leftarrow$
- purchaseOrder(?PO, supplierCo, ?AnyBuyer) $\wedge$
- quantity_ordered(?PO, ?Q) $\wedge$ (?Q $\geq$ 5) $\wedge$ (?Q $\leq$ 1000) $\wedge$
- shipping_date(?PO, ?D) $\wedge$ (?D $\geq$ 24Apr00) $\wedge$ (?D $\leq$ 12May00).
- <volumeDiscount> price(per_unit, ?PO, \$51) $\leftarrow$
- purchaseOrder(?PO, supplierCo, ?AnyBuyer) $\wedge$
- quantity_ordered(?PO, ?Q) $\wedge$ (?Q $\geq$ 100) $\wedge$ (?Q $\leq$ 1000) $\wedge$
- shipping_date(?PO, ?D) $\wedge$ (?D $\geq$ 28Apr00) $\wedge$ (?D $\leq$ 12May00) .
  overrides(volumeDiscount , usualPrice) .
- $\perp \leftarrow$ price(per_unit, ?PO, ?X) $\wedge$ price(per_unit, ?PO, ?Y) GIVEN (?X $\neq$?Y).
- ...

Negotiation Ex. Doc. Rules: Counter-Proposal *from manufCo to supplierCo*

- …
  <usualPrice> price(per_unit, ?PO, \$60) $\leftarrow$ ...
- <volumeDiscount> price(per_unit, ?PO, \$51) $\leftarrow$
- purchaseOrder(?PO, supplierCo, ?AnyBuyer) $\wedge$
- quantity_ordered(?PO, ?Q) $\wedge$ (?Q $\geq$ 5) $\wedge$ (?Q $\leq$ 1000) $\wedge$
- shipping_date(?PO, ?D) $\wedge$ (?D $\geq$ 28Apr00) $\wedge$ (?D $\leq$ 12May00) .
  overrides(volumeDiscount , usualPrice) .
- $\perp$ $\leftarrow$ price(per_unit, ?PO, ?X) $\wedge$ price(per_unit, ?PO, ?Y) GIVEN (?X $\neq$?Y).
- <aSpecialDeal> price(per_unit, ?PO, \$48) $\leftarrow$
- purchaseOrder(?PO, supplierCo, manufCo) $\wedge$
- quantity_ordered(?PO, ?Q) $\wedge$ (?Q $\geq$ 400) $\wedge$ (?Q $\leq$ 1000) $\wedge$
- shipping_date(?PO, ?D) $\wedge$ (?D $\geq$ 02May00) $\wedge$ (?D $\leq$ 12May00) .
- overrides(aSpecialDeal, volumeDiscount) .
- overrides(aSpecialDeal , usualPrice) .
- ...

Simply *added* rules!

In XML: Business Rules Markup Language

- <clp>
- <erule rulelabel=”usualPrice">
- <head>
- <cliteral>
- <predicate name="price" arity="3"/>
- <larglist>
- <lfunction name="per_unit"/>
- <variable name="PO"/>
- <function name="$60"/>
- </larglist>
- </cliteral>
- </head>
- <body> … *(see next page)* </body>
- </erule>
- …
- </clp>

Business Rules Markup Language for Negotiation Example (continued)

- <body>
- <andb>
- <fcliteral>
- <predicate name="purchaseOrder" arity="3"/>
- <larglist>
- <variable name="PO"/>
- <lfunction name="supplierCo"/>
- <variable name="AnyBuyer"/>
- </larglist>
- </fcliteral>
- <fcliteral>
- …
- </fcliteral>
- ...
- </andb>
- </body>

Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

EECOMS Example of Conflicting Rules: Ordering Lead Time

- Vendor's rules that prescribe how buyer must place or modify an order:
- A) 14 days ahead if the buyer is a qualified customer.
- B) 30 days ahead if the ordered item is a minor part.
- C) 2 days ahead if the ordered item's item-type is backlogged at the vendor, the order is a modification to reduce the quantity of the item, and the buyer is a qualified customer.

- Suppose more than one of the above applies to the current order? **Conflict!**

- Helpful Approach: **precedence** between the rules. Often only *partial* order of precedence is justified. E.g., C > A.

Courteous LP's: Ordering Lead Time Example

- <leadTimeRule1> orderModificationNotice(?Order,14days)
- $\leftarrow$ preferredCustomerOf(?Buyer,?Seller) $\wedge$
- purchaseOrder(?Order,?Buyer,?Seller) .
- <leadTimeRule2> orderModificationNotice(?Order,30days)
- $\leftarrow$ minorPart(?Buyer,?Seller,?Order) $\wedge$
- purchaseOrder(?Order,?Buyer,?Seller) .
- <leadTimeRule3> orderModificationNotice(?Order,2days)
- $\leftarrow$ preferredCustomerOf(?Buyer,?Seller) $\wedge$
- orderModificationType(?Order,reduce) $\wedge$
- orderItemIsInBacklog(?Order) $\wedge$
- purchaseOrder(?Order,?Buyer,?Seller) .
- overrides(leadTimeRule3 , leadTimeRule1) .
- $\perp \leftarrow$ orderModificationNotice(?Order,?X) $\wedge$
- orderModificationNotice(?Order,?Y); GIVEN ?X $\neq$?Y.

EECOMS Supply Chain Project: Overview

- EECOMS = Extended Enterprise Consortium for Integrated Collaborative Manufacturing Systems.   Completed Project: 3/98 - 2/01
- Inter-enterprise supply chain integration/collaboration, in manufacturing.
- IBM-led consortium includes Baan, Boeing, TRW Consulting, smaller rules & tools co.'s, 3 universities.
- 50%-funded by US government's NIST Advanced Technology Program. $29Million over 3 years (3/98 - 2/01).
- Business Focus: improve **"agility"**: late delivery, plant line breakdown, larger than expected order. React quickly, including modify plans, schedules.
- Technical Focus: **rules and conflict handling** for automated collaboration: contracts, negotiation, authorization, workflow; virtual situation room for human collaborative workflow.
- Is follow-on to CIIMPLEX (IBM-led NIST ATP $22M) & challenges it identified.   Shares: consortium, scenarios, agent-based approach.

Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

Flavors of Rules Commercially Most Important today in E-Business

- E.g., in OO app's, DB's, workflows.

- Relational databases, SQL: Views, queries, facts are all rules.
- Production rules (OPS5 heritage): e.g.,
  - Blaze, ILOG, Haley: rule-based Java/C++ objects.
- Event-Condition-Action rules (loose family), cf.:
  - business process automation / workflow tools.
  - active databases; publish-subscribe.
- Prolog. *"logic programs" as a full programming language.*
- *(Lesser: other knowledge-based systems.)*

Contract Rules: Overall Approach

- Use Rule Interlingua to represent products (or services), related business policies and/or processes, e.g., in catalog or during negotiation.
  - E.g., conditions on how to return an item for repair, or to deliver an order.
  - Key: declarative knowledge representation:
    - begin with Ordinary Logic Programs; then extend; encode/Webize in XML.
- Executable specification; "situated" LP / procedural attachments is esp. useful.
- Partially-specified / template, esp. during process of negotiation.
- Complement XML ontologies already evolving for various domains.
  - Ontology = formally-represented vocabulary / definitions.
- Specify negotiations including to configure auction mechanisms.
  - content of bids and requests for bids: partial then complete.
  - which goods, which attributes (e.g., price, delivery-date) are at issue.
- *Later: Specify trust/authorization, including via delegation.*

Overview of Approach to Contract Rule Representation

- *Wanted:* Interlingua between heterogeneous: SQL, Prolog, OPS5, ECA.
- 1. Choose: Ordinary Logic Programs. Forward or backward chaining.
- 2. Generalize: Courteous Logic Programs. Prioritized Conflict handling; Compiler to OLP. Modularity in specification and software engineering.
- 3. XML-ify: cf. RuleML emerging industry standard (updates BRML).
- 4. Generalize: Situated LP's. Procedural Attachments for tests, actions.
- Implementation: IBM CommonRules free on AlphaWorks: V1.0 7/99, V2.1 currently.

Detailed in this talk: (1.)--(3.).

Criteria for Contract Rule Representation

1

- *High-level:* Agents reach common understanding; contract is easily modifiable, communicatable, executable.

2

- Inter-operate: heterogeneous commercially important rule systems.
- Expressive power, convenience, natural-ness.
- ... but: computational tractability.
- Modularity and locality in revision.

3

- Declarative semantics.
- Logical non-monotonicity: default rules, negation-as-failure.
  - essential feature in commercially important rule systems.

OLP

- Prioritized conflict handling.

Courteous

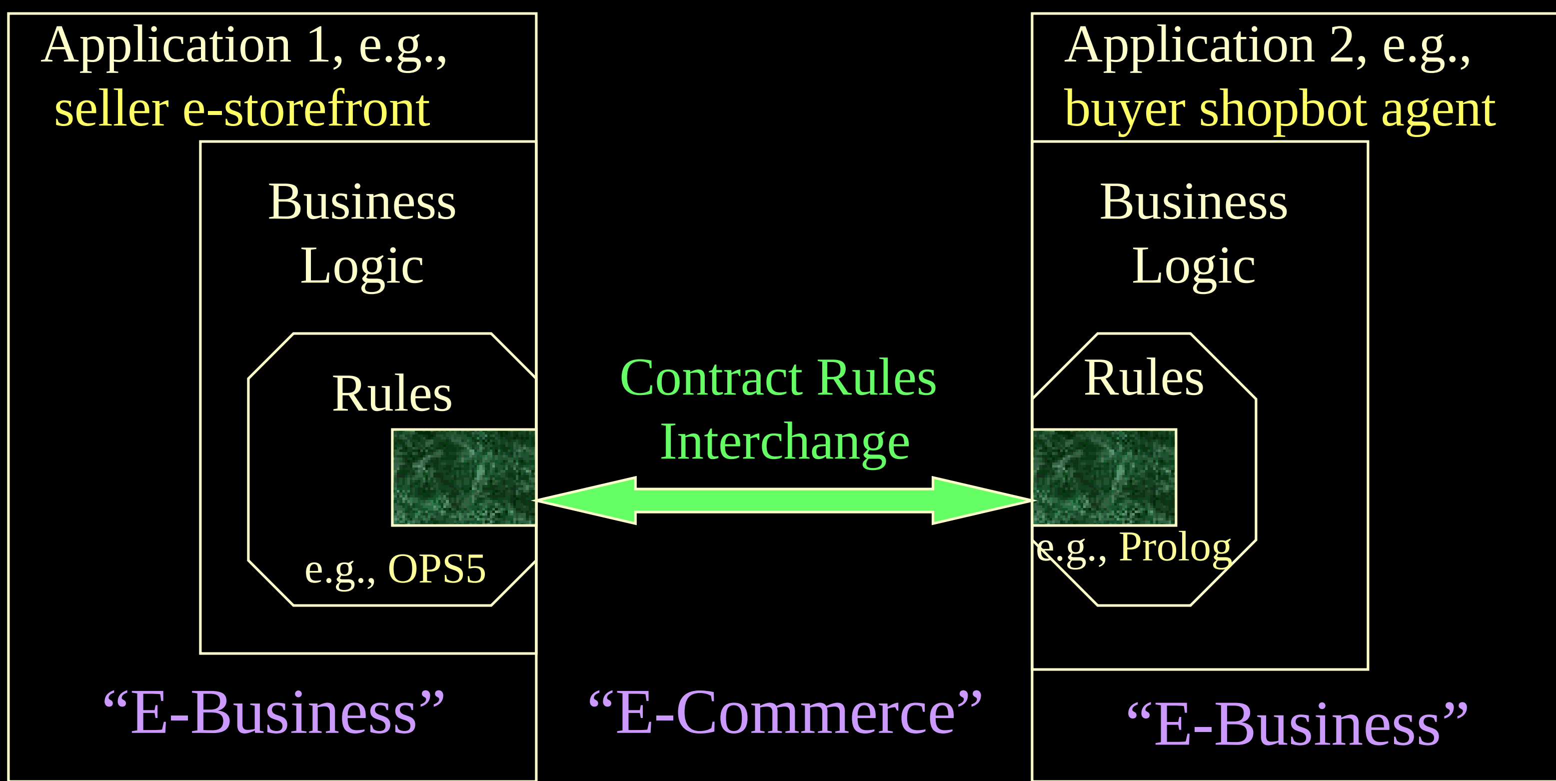
- Ease of parsing.
- Integration into Web-world software engineering.

XML

- Procedural attachments.

Situated

Contract Rules across Applications / Enterprises



Contracting parties integrate e-businesses via shared rules.

Ordinary Logic Programs as basic representation: Advantages

- Declarative: semantics is independent of inferencing procedure implementation, e.g., forward vs. backward chaining, sequencing of executing rules or conditions within rules.
- Expressive: relational expressions cf. SQL, large fragment of first-order logic, chaining, basic logical non-monotonicity (unlike first-order logic / ANSI-draft Knowledge Interchange Format).
- Efficient: computationally tractable given two reasonable restrictions:
  - 1. Datalog = no logical functions of non-zero arity.
  - 2. Bounded number v of logical variables per rule.
  - m = O(n^(v+1)), where n = ||LP||, m = ||ground-instantiated LP||.
  - Inferencing time is O(m) for broad case (stratified), O(m^2) generally (for well-founded semantics).
  - By contrast, first-order-logic inferencing is NP-hard.

Ordinary Logic Programs: Advantages (continued)

- Widely deployed and familiar:
  - relational DB's, SQL
  - Prolog
  - knowledge-based systems and intelligent agents
    - (e.g., IBM's Agent Building Environment)
- Common core shared semantically by many rule systems: e.g.,
  - relational DB's, SQL
  - Prolog
  - production rules (OPS5 heritage)
  - Event-Condition-Action rules
  - first-order-logic

Courteous LP's: the What

- Updating/merging of rule sets: is crucial, often generates conflict.
- Courteous LP's feature prioritized handling of conflicts.
- Specify scope of conflict via a set of *pairwise* mutual exclusion constraints.
  - E.g., $\perp \leftarrow$ discount(?product,5%) $\wedge$ discount(?product,10%) .
  - E.g., $\perp \leftarrow$ loyalCustomer(?c,?s) $\wedge$ premiereCustomer(?c,?s) .
  - Permit classical-negation of atoms: $\neg$p means p has truth value *false*
    - implicitly, $\perp \leftarrow p \wedge \neg p$ for every atom p.
- **Priorities** between rules: partially-ordered.
  - Represent priorities via reserved predicate that compares rule labels:
    - overrides(rule1,rule2) means rule1 is higher-priority than rule2.
    - Each rule optionally has a rule label whose form is a functional term.
    - overrides can be reasoned about, just like any other predicate.

Priorities are available and useful

- Priority information is naturally available and useful. E.g.,
  - recency: higher priority for more recent updates.
  - specificity: higher priority for more specific cases (e.g., exceptional cases, sub-cases, inheritance).
  - authority: higher priority for more authoritative sources (e.g., legal regulations, organizational imperatives).
  - reliability: higher priority for more reliable sources (e.g., security certificates, via-delegation, assumptions, observational data).
  - closed world: lowest priority for catch-cases.
- Many practical rule systems employ priorities of some kind, often implicit, e.g.,
  - rule sequencing in Prolog and production rules.
    - courteous subsumes this as special case (totally-ordered priorities), plus enables: merging, more flexible & principled treatment.

Prioritized argumentation in an opposition-locale.

Conclusions from opposition-locales <u>previous</u> to this opposition-locale {p1,...,pk}

↓

(Each pi is a ground classical literal. $k \geq 2$.)

Run Rules for p1,...,pk

↓

Set of <u>Candidates</u> for p1,...,pk:
Team for p1, ..., Team for pk

↓

Prioritized Refutation

↓

Set of <u>Unrefuted</u> Candidates for p1,...,pk:
Team for p1, ..., Team for pk

↓

Skepticism

↓

<u>Conclude</u> Winning Side if any: at most one of {p1,...,pk}

Courteous LP's: Advantages

- Facilitate updating and merging, modularity and locality in specification.
- Expressive: classical negation, mutual exclusions, partially-ordered prioritization, reasoning to infer prioritization.
- Guarantee consistent, unique set of conclusions.
  - **Mutual exclusion is enforced**. E.g., never conclude both p & ¬p.
- Efficient: low computational overhead beyond ordinary LP's.
  - Tractable given reasonable restrictions (Datalog, bound v on #var's/rule):
    - extra cost is equivalent to increasing v to (v+2) in ordinary LP's.
  - By contrast, more expressive prioritized rule representations (e.g., Prioritized Default Logic) add NP-hard overhead.
- Modular software engineering: via courteous compiler: CLP → OLP.
  - A radical innovation. Add-on to variety of OLP rule systems. O(n^3).

Courteous LP's: Keys to Tractability

- Overall: mutex's & conflict locales $\rightarrow$ keep tractability.
- LP's: disallow disjunctive conclusions, essentially. Classical allows $\Rightarrow$ NP-hard.
- LP's: disallow contraposition (= $\{\neg a \leftarrow . \, , \; a \leftarrow b \wedge c.\} \Rightarrow (\neg b \vee \neg c)\}$) which requires disjunctive conclusions. "Directional". Classical allows $\Rightarrow$ NP-hard.
- Highly expressive prioritized rule representations (e.g., Prioritized Default Logic, Prioritized Circumscription) allow minimal conflict sets of arbitrary size $\Rightarrow$ NP-hard overhead for conflict handling.
- Courteous conflict handling involves essentially only pairwise conflicts, i.e., minimal conflict sets of size 2. (Current work: possibly generalize to size k.)
  - Novelty: generalize to pairwise mutex's beyond $\perp \leftarrow p \wedge \neg p$, e.g., partial-functional, thus avoid need for contraposition and larger conflict sets.
- Courteous conflict handling is local within an opposition locale: a set of rules whose heads oppose each other through mutex's. Refutation and Skepticism are applied within each locale.

Summary: Courteous LP's in XML as Core KR

- Key Observations about Declarative OLP:
  - captures common core among commercially important rule systems.
  - is expressive, tractable, familiar.
  - advantages compared to classical logic / ANSI-draft KIF:
    - + + logical non-monotonicity, negation-as-failure.
    - – – disjunctive conclusions.
    - + + tractable.
    - + + procedural attachments: situated LP's.
- Cleverness of Courteous extension to the OLP representation:
  - prioritized conflict handling $\rightarrow$ modularity in specification.
  - courteous compiler $\rightarrow$ modularity in software engineering.
  - mutex's & conflict locales $\rightarrow$ keep tractability. (Compiler is O(n^3).)
- Novelty: do it in XML $\rightarrow$ ease of parsing, integration in Web engineering.

Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

Situated LP's: Motivation from Contracts

- For <u>executable</u> contract specification:
  - procedural attachments is esp. useful
  - ... thus <u>situated</u> logic programs is esp. useful
    - a new abstraction, highly declarative
    - introduced in: IBM Agent Building Environment '96.

Situated LP's: Overview

- Point of departure: LP's are pure-belief representation, but most practical rule systems want to invoke external procedures.
- Situated LP 's feature a semantically-**clean** kind of **procedural attachments**. I.e., they hook beliefs to drive procedural API's outside the rule engine.
- Procedural attachments for **sensing** (queries) when testing an antecedent condition or for **effecting** (actions) upon concluding a consequent condition. Attached procedure is invoked when testing or concluding in inferencing.
- Sensor or effector **link** statement specifies an association from a predicate to a procedural call pattern, e.g., a method.   A link is specified as part of the representation. I.e., a SLP is a conduct set that includes links as well as rules.

Situated LP's: Overview (cont.'d)

- phoneNumberOfPredicate ::s:: BoeingBluePagesClass.getPhoneMethod .
  ex. sensor link
- shouldSendPagePredicate ::e:: ATTPagerClass.goPageMethod .
  ex. effector link
- Sensor procedure may require some arguments to be ground, i.e., bound; in general it has a specified binding-signature.
- Enable dynamic loading and remote loading of the attached procedures (exploit Java goodness).

- Overall: cleanly separate out the procedural semantics as a declarative extension of the pure-belief declarative semantics. Easily separate chaining from action.

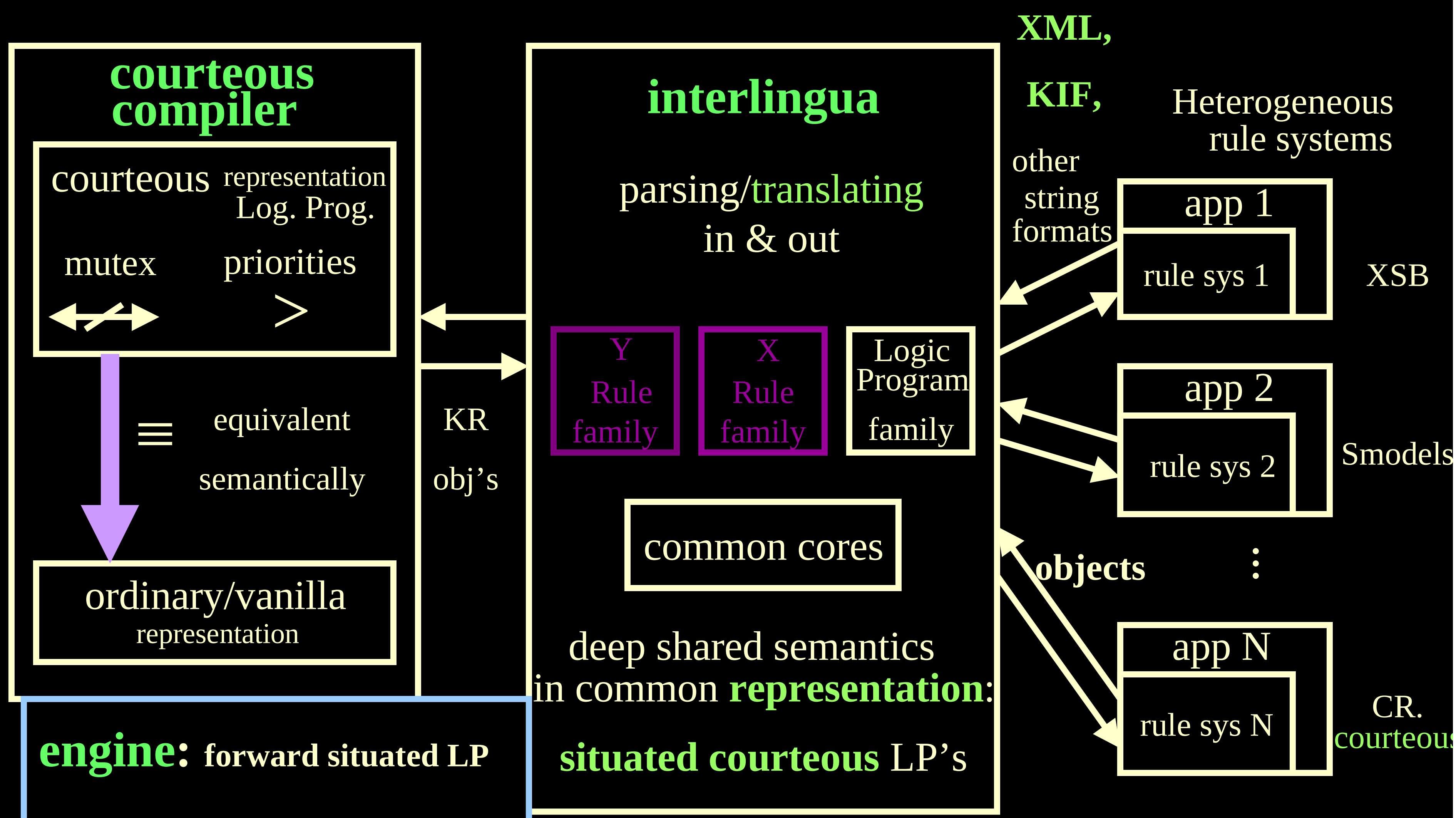
Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

Commercial Implementation & Piloting

- IBM CommonRules: AlphaWorks Java library
  - <u>implements</u> rule-based capabilities:
    - XML inter-operability; prioritized conflict handling
- Rule Markup Language: nascent <u>industry standards</u> effort
  - XML Knowledge Representation (KR) $\rightarrow$ make the Web be "Semantic"
  - KR: **Situated Courteous Logic Programs in XML**
- EECOMS industry consortium including Boeing, Baan, TRW, Vitria, IBM, universities, small companies
  - $29Million 1998-2000; 50% funded by NIST ATP
  - <u>application piloted</u>
    - contracting & negotiation; authorization & trust

Current-version IBM CommonRules



What's Doable Today *in rule-based agent contracting, based on our approach*

- Communicate:
  - XML, interoperable
  - ⇔ <u>heterogeneous</u> rule systems / rule-based agents
- Execute contract provisions:
  - infer; ebiz actions; authorize; ...
- Modify easily: contingent provisions
  - default rules; modularity
- Reason about the contract/proposal
  - hypotheticals, test, evaluate

Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

Related Work on Prioritized Rule KR

- Other approaches to prioritized logic programs
  - close: Defeasible Logic: Nute, Maher, Antoniou, *et al*
  - Prakken, Sartor
- Less close, less tractable: very-expressive prioritized logics
  - Prioritized Default Logic (Brewka),
  - Prioritized Circumscription (McCarthy, Lifschitz, Grosof)

Examples of Rules in Regulations & Bureaucratic Policies/Processes

- Taxes and Tariffs
  - rules from a variety of sources, e.g., jurisdictions
- Social Services: e.g., qualifying for benefits
- Notifications
- Penalties, Liabilities
- Authorizations, Permissions
  - e.g., access to medical records
- ...

Overview of Approach to: Policies for Trust and Security Authorization

- Use rule-based executable specification of security authorization policies, a.k.a. trust management: including delegation, certificates.
  - Straightforwardly generalizes Role-Based Access Control (RBAC).
  - We have the first step of an expressive extension of courteous LP's to handle delegation and certificates: Delegation Logic.
- Often, authorization/trust policy is really a part of overall contract or business policy, at application-level. This contrasts with authentication.
- Advantages of rule-based approach, esp. from declarative semantics:
  - easier integration with general business policy.
  - easier to understand and modify by humans.
  - provable guarantees of behavior of implementation.
  - principled handling of negation and conflict.

Delegation Logic (D1LP) Example: accessing medical records

- Problem: Hospital HM to decide: requester Alice authorized for patient Peter?
- Policies: HM will authorize only the patient's physician. HM trusts any hospital it knows to certify the physician relationship. Two hospitals together can vouch for a 3rd hospital.
  - HM says authorized(?X, read(medRec(?Y))) if HM says inRole(?X, physic(?Y)).
  - HM delegates inRole(?X, physic(?Y))^1 to threshold(1,?Z, HM says inRole(?Z,hosp)).
  - HM delegates inRole(?H,hosp)^1 to threshold(2 , ?Z, HM says inRole(?Z,hosp)).
- Facts: HC certifies Alice is Peter's physician. HM knows two hospitals HA and HB. HA and HB each certify HC as a hospital.
  - HC says inRole(Alice, physic(Peter)). HA says inRole(Joe, physic(Sue).
  - HM says inRole(HA,hosp). HM says inRole(HB, hosp).
  - HA says inRole(HC,hosp). HB says inRole(HC, hosp).
- Conclusion: HM says authorized(Alice, read(medRec(Peter))). *Joe NOT authorized.*

More Legal Applications: Visions

- regulations
- Alternative Dispute Resolution
- adjudication, legal decision-making
- …   ***?pointers?***

Also Currently Being Developed in the world today

- Delegations between agents
- XML Ontologies (Vocabularies)
  - knowledge representation: infer with definitional knowledge
  - specific domain/industry vocabularies
- DARPA Agent Markup Language: ontologies, rules
- Industry Standards:
  - Web
  - Agents, Business Processes, Workflow
  - E-Commerce
  - Industry-Specific
  - *Legal XML*
- *Law: Electronic Signatures, ...*

Current Work: Knowledge Representation on the Web

- Apply KR viewpoint and techniques to Web info
- "Web-ize" the KR's
  - exploit Web/XML hyper-links, interfaces, tools
  - think global, act global : as part of whole Web
- Radically raise the level of shared meaning
  - level = conceptual/abstraction level
  - meaning = sanctioned inferences / vocabularies
  - shared = tight correspondence
- "The Semantic Web", "The Web of Trust" [Tim B-L]
- Build: The Web Mark II

Current Work in KR on the Web: Challenges & Opportunities; Issues

- exploit emerging Web standards in XML suite
  - XML data, XML-ified APIs generally
  - beginning Rule Markup Language industry standards effort
    - related: Java Rule Engines standards effort
  - RDF, DAML+OIL Description Logic, Topic Maps, XML Query, P3P
  - XML-EDI, EDIFAC, EBXML, UDDI, ...
  - Industry verticals ontologies, IEEE Upper Ontologies, …
  - Legal XML, ...
- exploit other emerging agent-communication standards:
  - FIPA, OMG, ANSI Knowledge Interchange Format (KIF)
- inter-source context, conflicts, trust

- Thanks!
- Questions?
- Comments? Pointers?
- For More Info:
  - http://www.mit.edu/~bgrosof/
    - links to http://www.research.ibm.com/rules/

Outline

- Intro; Law in the Small
- Automating Agent Contracting
  - intro
  - examples, illustrating approach
  - approach details: KR, design rationale
    - Courteous Logic Programs in XML
      - value of prioritized default reasoning/argumentation
        - pragmatics, modularity
    - Commercial Implementation and Piloting
- Current Work; Related Work; the Glorious Future
  - regulations, bureaucratic policies & processes
  - XML standards, the Semantic Web

OPTIONAL SLIDES FOLLOW

Launch Vector: My Background
E-Commerce Agents, Rules: Techno + Biz

- Harvard BA math econ & mgm sci
- startups
- Stanford CS (Computer Science) PhD in AI
- IBM Watson Research: IA for EC
  - Led Intelligent Agents, Business Rules for E-Commerce
- MIT Sloan: Information Technology group
- Technology end of B-school IT world
- CS + Business Perspective (cf. Industry, cf. B-school):
  - how/where the technology is useful, important
  - business value; implications for processes & strategies
  - market evolutions; innovation paths; organizational changes

theory theory → ***practical theory + pilot app's***

Background in Law-related Research

- Overall: formally represent policies and info as rules
- Evidential Reasoning: probabilistic, fuzzy, ...
- Bureaucratic Processes as domain
  - pioneer within AI knowledge representation community
- Argumentation with rule-based beliefs:
  - efficient algorithms
  - theory
  - bridge to commercially practical rule-based/database systems
- Contracting & Negotiation, Authorization & Trust
- *Invited Speaker at 2001 ABA Spring Meeting > Business Law > Cyberlaw > Internet Law > E-Agents Task Force*
- *Invited Speaker at 2001 International Conference on AI & Law:*
  - "Automating Law in the Small: Contracts, Regulations, and Prioritized Argumentation"

The Web is becoming XML

- XML (vs. HTML) offers much greater capabilities for structured detailed descriptions that can be processed automatically.
  - Eases application development effort for assimilation of data in inter-enterprise interchange
  - A suite of open standards both current and emerging

- *Soon, Agents will Talk according to these standards…*
  - ∴ potential to revolutionize *interactivity* in Web marketplaces
    - B2B, ...

Declarative Semantics at Core

- Desire: deep semantics (model-theoretic) to
  - understand and execute imported rules.
- Possible only for shared expressive subsets: "cores".
  - Rest translated with superficial semantics.
- Approach: declarativeness of core / rep'n (in sense of knowledge representation theory).
  - A given set of premises entails a set of sanctioned conclusions. Independent of implementation & inferencing control (bkw vs. fwd).
  - Maximizes overall advantages of rules:
    - Non-programmers understand & modify.
    - Dynamically (run-time) modify.

Interlingua: Need Go Beyond KIF

- KIF has major limitations:
  - logically monotonic.
    - yet virtually all practical rule (and probability) systems are non-monotonic.
  - pure-belief, no procedural attachments.
    - yet most practical rule systems do invoke procedures external to the inference engine.
- Candidates to complement KIF exist:
  - logic programs, Bayes nets, ...

Ordinary Logic Programs as basic representation: Definition

- A LP is a set of (premise) rules; semantically, it specifies a set of conclusions.
- replyInterval(?msg,CustomerRep)
- $\leftarrow$ from(?msg,?s) $\wedge$ customer(?s) $\wedge$ ~urgency(?msg,low).

  Example Rule
-
- where the "?" prefix indicates a logical variable.
- Generally, a rule has the form of Head IF Body :
- $H \leftarrow B_1 \wedge \ldots \wedge B_j \wedge \sim B_{j+1} \wedge \ldots \wedge \sim B_m$.
- where $m \geq 0$; $\wedge$ stands for logical "AND"; $\leftarrow$ stands for logical "IF"; and $H, B_1, \ldots, B_m$ are each an <u>atom</u> with form: Predicate($Term_1, \ldots, Term_k$).
- A predicate = a relation. An atom semantically denotes a boolean.
- ~ stands for <u>negation-as-failure</u> (a.k.a. weak negation, default negation).
  - The negation-as-failure construct is logically non-monotonic.
  - Intuitively, ~p means p's truth value is either *false* OR *unknown*.

Ordinary Logic Programs: Definition (continued)

- Each argument Term_1, ..., Term_k is a term.
- A term is either a logical constant (e.g., “Joe”) OR a logical variable (e.g., “?msg”) OR a functional expression of the form:
- LogicalFunction(Term_1, ..., Term_k)
- A functional expression semantically essentially denotes a logical constant.
- A term, atom, or rule is called “ground” when it has no logical variables.

- A fact is a ground rule with empty body.
- A primitive conclusion has the form of a ground atom (compound conclusions are built up from these via logical operators such as AND etc.).
- Semantically, a rule or LP stands for the set of all its ground instances.
- (Observe that a rule body can represent an expression in relational algebra cf. relational DB’s (e.g., SQL).)

IBM CommonRules technology overview

- Java library: V1.0 released 7/30/99 on IBM AlphaWorks.
  - thousands of downloads via Web.
  - piloting in EECOMS $29Million NIST ATP project (IBM, Baan, Boeing, TRW, universities, other co.'s) on agile manufacturing.
    - negotiation & trust/security in supply chain collaboration.
- Basic rule representation: <u>Logic programs</u> (LP's).
  - LP's in declarative sense, not Prolog. E.g., forward or backward chaining.
  - representation = syntax + deep semantics.
    - semantics of rule set = its set of valid conclusions.

CommonRules technology overview (continued)

- Extends rule representation to:
  - Courteous LP's:
    - prioritized handling of **conflicts**, e.g., in updating/merging.
  - Situated (Courteous) LP's:
    - **procedural attachments** to invoke non-reasoning actions or queries, via methods external to inferencing engine.
- Courteous Compiler from courteous LP's to ordinary LP's.
- XML Interlingua and sample translators.
  - interlingua = common rule representation for translation between heterogeneous rule systems. Suitable to become industry standard.
- Sample Inferencing/Execution Engine:
  - forward-chaining situated courteous LP's.

Delegation Logic: Goal and Basic Approach

- Our goal: Develop a language that
  - can represent, with significant expressive power, policies and credentials for authorization in Internet scenarios
  - can provide mechanisms for delegation
  - has a clear declarative semantics
- Our approach: Delegation Logic (DL): multi-agent logic programs with delegation to complex delegatees
  - D1LP: extends negation-free OLP $\Rightarrow$ with delegation
  - D2LP: extends Courteous LP $\Rightarrow$ with delegation
  - Tractable "Delegation compiler" similar to courteous compiler.
- Collaborators: Ninghui Li (NYU→Stanford), Joan Feigenbaum (ATT→Yale)