Contracts, Policies, and Prioritized Rules in XML Agent Communication

Benjamin Grosof

bgrosof@mit.edu http://www.mit.edu/people/bgrosof/home.html

MIT Sloan School of Management

(Professor in Information Technology group)

Slides of Invited Talk presented at 18th meeting of FIPA, on 7/19/00, held at the UMBC Technology Center, Baltimore, MD, USA.

FIPA = Foundation for Intelligent Physical Agents, an industry standards organization. http://www.fipa.org

UMBC = University of Maryland at Baltimore County

Outline

- 1. Vision & Approach: rules in automated contracts.
- 2. A new knowledge representation for contract rules:
  - Courteous Logic Programs in XML: Business Rules Markup Language
  - Radical advance in modularity & conflict handling.
  - Radical advance in inter-operability.
- 3. Application piloting:
  - Negotiation, e.g., supply chain collaboration (EECOMS $29M project).
    - *(Other: auctions, e.g., travel; storefronts, e.g., books; authorization)*
- 4. Conclusions; Implications for FIPA:
  - CLP as candidate content language for FIPA
  - Content in ContractNet FIPA protocol.
  - Importance of non-monotonicity for SL role in FIPA ACL
- 5. Current Work
  - *DAML. Procedural attachments. Trust and delegation. XML-ify KIF.*

Advantages of XML-encoding ACL msgs

- *Proposed to FIPA 8/98 by [Labrou, Grosof, Finin].*
  - *Paper at IJCAI-99 ACL Wksh. (→ book in late 2000).*

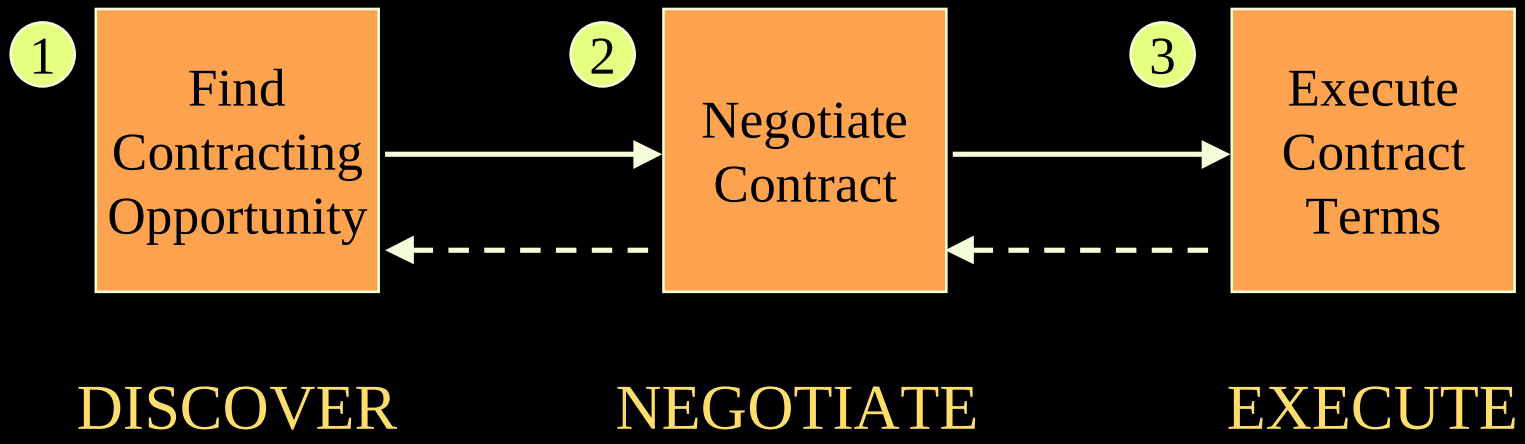
- Parsing and browsing: easier development.
  - Off-the-shelf XML parsers, easy to modify syntax.
  - Relatively human-understandable.
- More Web-friendly: easier integration; security, …
- Exploit links for ACL-msg pragmatics & detailed semantics:
  - Point to ontologies, agent capability and identity info, protocols, validators, inference engines, documents.

- Ditto for (multiple) content languages within ACL msgs.

Goal: Automate Contracting

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

Contracting 1-2-3



- Applies to any contracting, electronic or not.
- May iterate or interleave these steps.
- Boundaries not necessarily sharp.

Angle of Attack

- Rules aspect as an early step.

Idea/Vision: 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.

"Business Rules": Two Senses

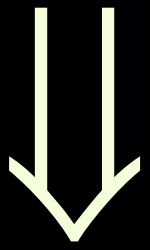
- 1. Broader/looser/shallower sense: "Business Rule" =
  - some piece of interesting/significant application logic, desirable to separate out somehow from rest of code.
  - e.g., how to calculate an insurance annuity.
  - *Holy Grails: reusability; appropriate OO scope, e.g., exceptions.*
- 2. Narrower/deeper sense: "(Business) Rule" =
  - "if-then" cf. logical knowledge representation (KR).
  - *Potential virtues: deeper connection to SQL and rule-based systems; easier merging & updating; more ways to use & analyze; simplicity for non-programmers.*
  - *Holy Grail: expressiveness PLUS practical tractability & engineering.*

- **We focus on (2.), and hope to often map (1.) → (2.).**

Larger Vision: rules in e-business overall

- Rules as an important aspect of coming world of Internet e-business: rule-based business processes for both B2B and 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 business processes, e.g., 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.

Angle of Attack: Rules aspect as an early step.



Need to improve the...

- Fundamental knowledge representation for rules:
  - semantics
  - syntax
- … for:
  - inter-operability + executability
  - specification dynamically & by non-specialists

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, ...

Examples of Rules in Contracts

- Terms & conditions, e.g., price discounting.
- Service provisions, e.g., rules for refunds.
- Surrounding business processes, e.g., lead time to order.
- Price vs. quantity vs. delivery date.
- Cancellations.
- Discounting for groups.
- Product catalogs: properties, conditional on other properties.
- Creditworthiness, trustworthiness, authorization.

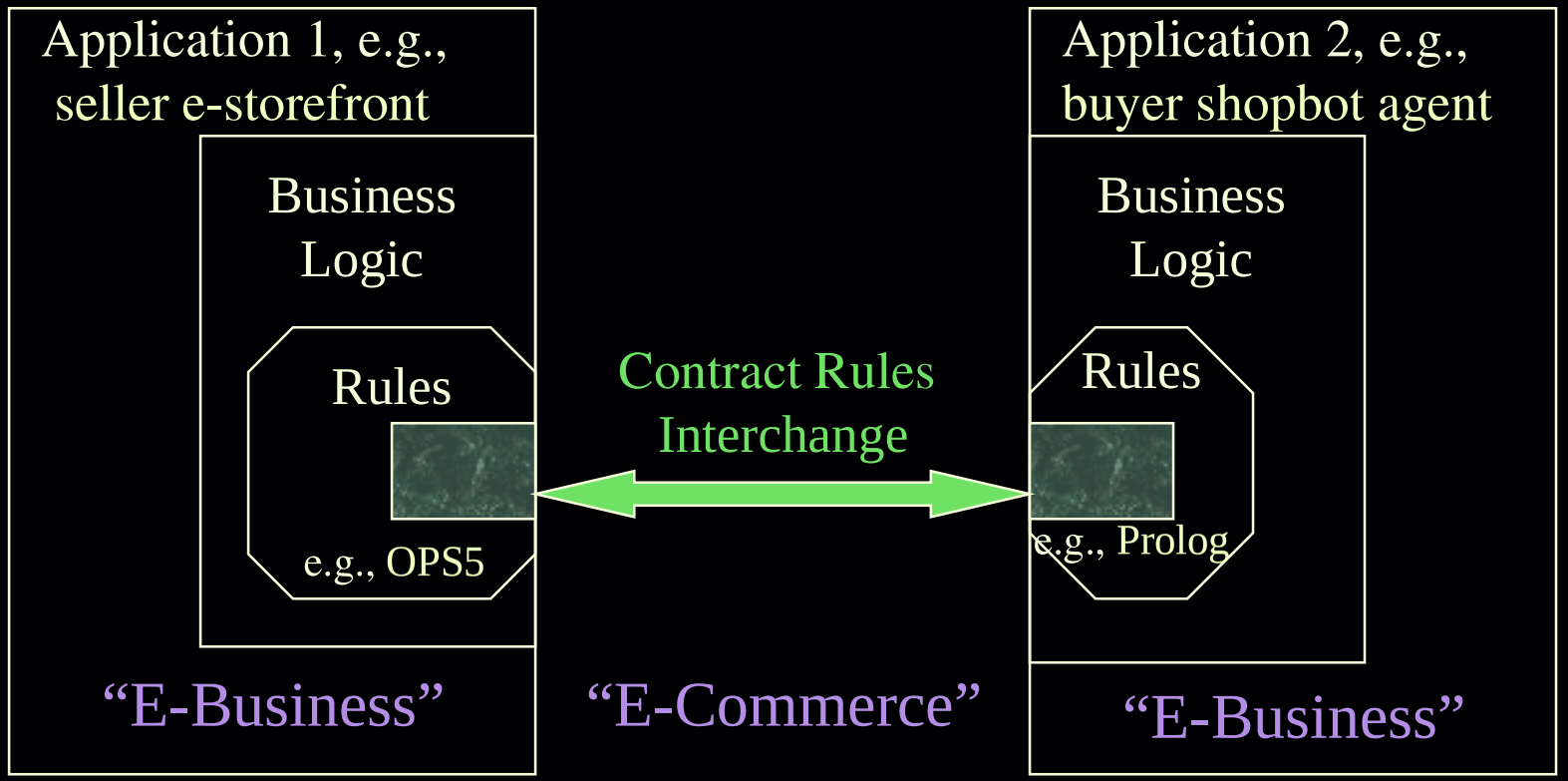
EECOMS Example of Conflicting Rules: Ordering Lead Time

- 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.

Contract Rules across Applications / Enterprises



Contracting parties integrate e-businesses via shared rules.

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

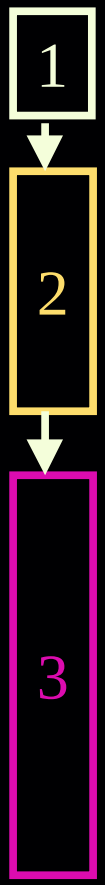
- Use Rule Interlingua to represent products (or services), related business policies and/or processes, e.g., in catalog or during negotiation.
  - E.g., conditions on how to return an item for repair, or to deliver an order.
  - Key: declarative knowledge representation:
    - begin with Ordinary Logic Programs; then extend; encode 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: Business Rules Markup Language. Standards related.
- 4. Generalize: Situated LP's. Procedural Attachments for tests, actions.
- Implementation: IBM CommonRules free on AlphaWorks since 7/99.
-
- 5. Generalize: Delegation LP's. Complex delegation in authorization & trust. Explicitly multi-agent representation. Compiler to OLP/CLP.

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

Criteria for Contract Rule Representation



- *High-level:* Agents reach common understanding; contract is easily modifiable, communicatable, executable.
- Inter-operate: heterogeneous commercially important rule systems.
- Expressive power, convenience, natural-ness.
- ... but: computational tractability.
- Modularity and locality in revision.
- Declarative semantics.
- Logical non-monotonicity: default rules, negation-as-failure.
  - essential feature in commercially important rule systems.
- Prioritized conflict handling.
- Ease of parsing.
- Integration into Web-world software engineering.
- Procedural attachments.

OLP

Courteous

XML

Situated

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.

Courteous LP's: Example

- <leadTimeRule1> orderModificationNotice(?Order,14days)
- ← preferredCustomerOf(?Buyer,?Seller) ∧
- purchaseOrder(?Order,?Buyer,?Seller) .
- <leadTimeRule2> orderModificationNotice(?Order,30days)
- ← minorPart(?Buyer,?Seller,?Order) ∧
- purchaseOrder(?Order,?Buyer,?Seller) .
- <leadTimeRule3> orderModificationNotice(?Order,2days)
- ← preferredCustomerOf(?Buyer,?Seller) ∧
- orderModificationType(?Order,reduce) ∧
- orderItemIsInBacklog(?Order) ∧
- purchaseOrder(?Order,?Buyer,?Seller) .
- overrides(leadTimeRule3 , leadTimeRule1) .
- ⊥ ← orderModificationNotice(?Order,?X) ∧
- orderModificationNotice(?Order,?Y) GIVEN ?X ≠ ?Y.

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., $\bot \leftarrow$ discount(?product,5%) $\wedge$ discount(?product,10%) .
  - E.g., $\bot \leftarrow$ loyalCustomer(?c,?s) $\wedge$ premiereCustomer(?c,?s) .
  - Permit classical-negation of atoms: $\neg$p means p has truth value *false*
    - implicitly, $\bot \leftarrow p \wedge \neg p$ 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.

Courteous Compiler

- Transformer compiles a courteous LP into an ordinary LP.
- A radically innovative approach in rules representation.
- "Compiles away" conflict, as modular add-on to rule system X's
  - inferencing
  - specification
- Enables courteous features to be added to, or implemented in, a variety of rule systems.
- Tractable: O(n^3). Incremental.

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: 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).

Business Rules Markup Language for Ordering Lead Time Example

```
<clp>
 <erule rulelabel="leadTimeRule1">
  <head>
   <cliteral predicate="orderModificationNotice">
    <variable name="?Order"/>
    <function name="days14"/>
   </cliteral>
  </head>
  <body>
  ...
  </body>
 </erule>
...
</clp>
```

[NB: here, OLD-version DTD!]

Business Rules Markup Language for Example (continued)

```
<body>
    <and>
      <fcliteral predicate="preferredCustomerOf">
        <variable name="?Buyer"/>
        <variable name="?Seller"/>
      </fcliteral>
      <fcliteral predicate="purchaseOrder">
        <variable name="?Order"/>
        <variable name="?Buyer"/>
        <variable name="?Seller"/>
      </fcliteral>
    </and>
  </body>
```

Business Rules Markup Language: Translators; Relation to Industry Standards Drafts.

```
<clp>
 <erule rulelabel="leadTimeRule1">
  <head>
   <cliteral predicate="orderModificationNotice">
    <variable name="?Order"/>
    <function name="days14"/>
   </cliteral>
  </head>
  <body>
   <and>
    <fcliteral predicate="preferredCustomerOf">
     <variable name="?Buyer"/>
     <variable name="?Seller"/>
    </fcliteral>
    <fcliteral predicate="purchaseOrder">
     <variable name="?Order"/>
     <variable name="?Buyer"/>
     <variable name="?Seller"/>
    </fcliteral>
   </and>
  </body>
 </erule>
 ...
</clp>
```

CommonRules includes

sample translators to

3 rule systems (incl. XSB, Smodels)

& KIF.

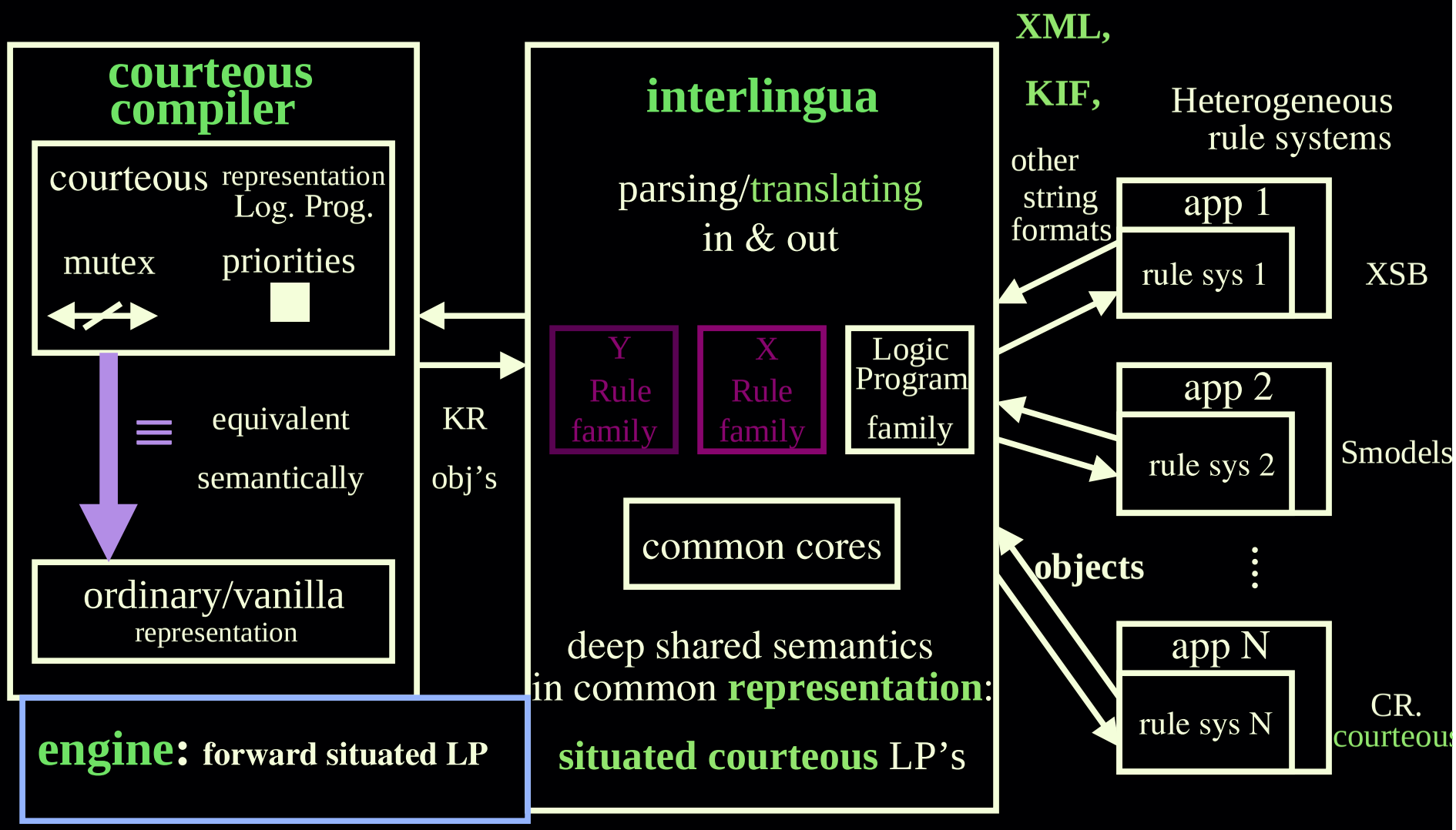
BRML $\supseteq$ { ANSI-draft KIF subset }.

BRML is content language for XML-ified FIPA

Agent Communication Language.

IBM CommonRules Java alpha

free on Web since 7/99; 2000+ downloads



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 → modularity in specification.
  - courteous compiler → modularity in software engineering.
  - mutex's & conflict locales → keep tractability. (Compiler is O(n^3).)
- Novelty: do it in XML → ease of parsing, integration in Web engineering.

EECOMS Supply Chain Project: Overview

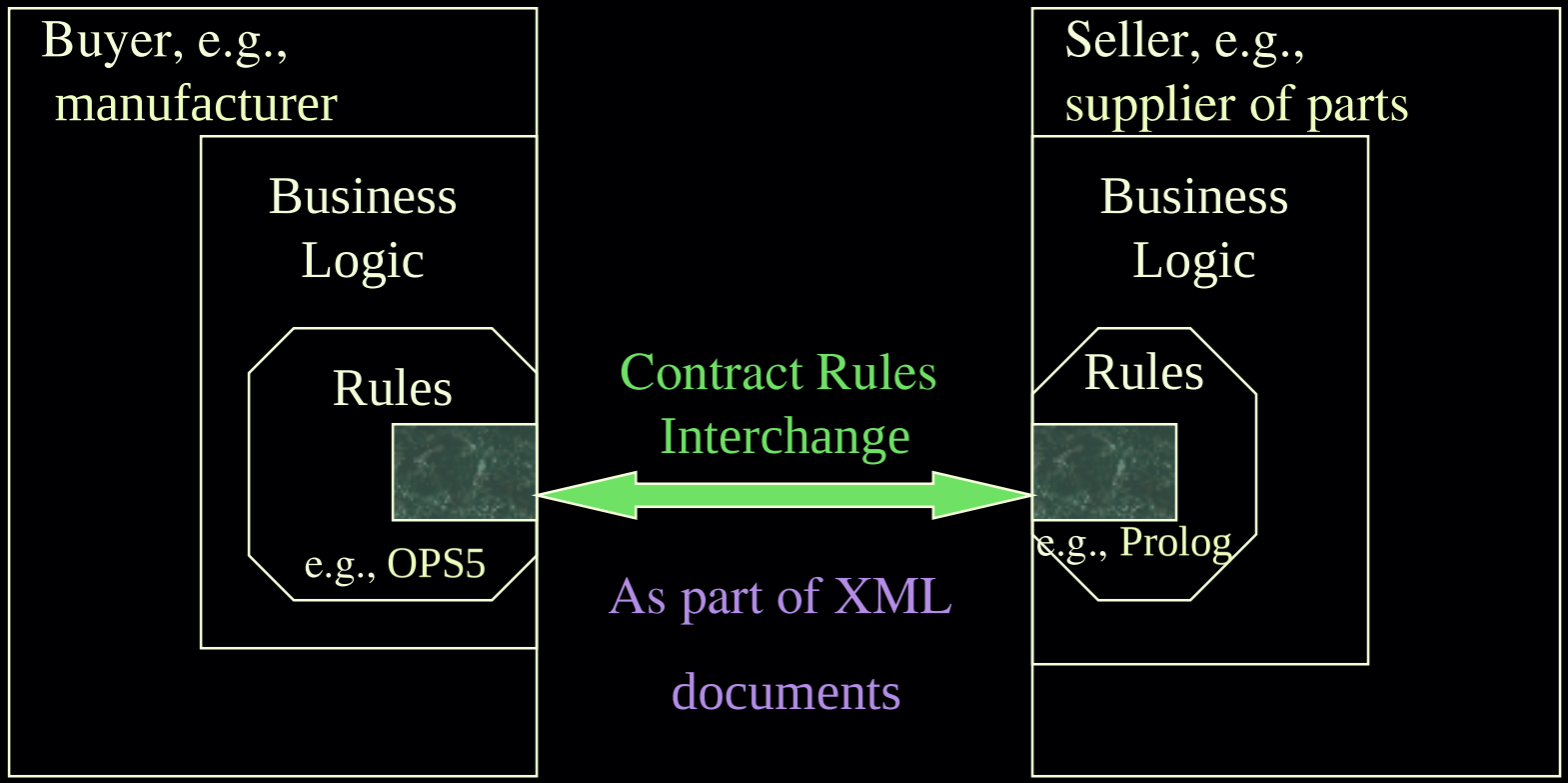
- EECOMS = Extended Enterprise Consortium for Integrated Collaborative Manufacturing Systems.
- Inter-enterprise supply chain integration/collaboration, in manufacturing.
- IBM-led consortium includes Baan, Boeing, TRW Consulting, smaller rules & tools co.'s, 3 universities.
- 50%-funded by US government's NIST Advanced Technology Program. $29Million over 3 years, ends 2001.
- 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.

Bidding in Negotiation

(e.g., in EECOMS manufacturing supply chain)

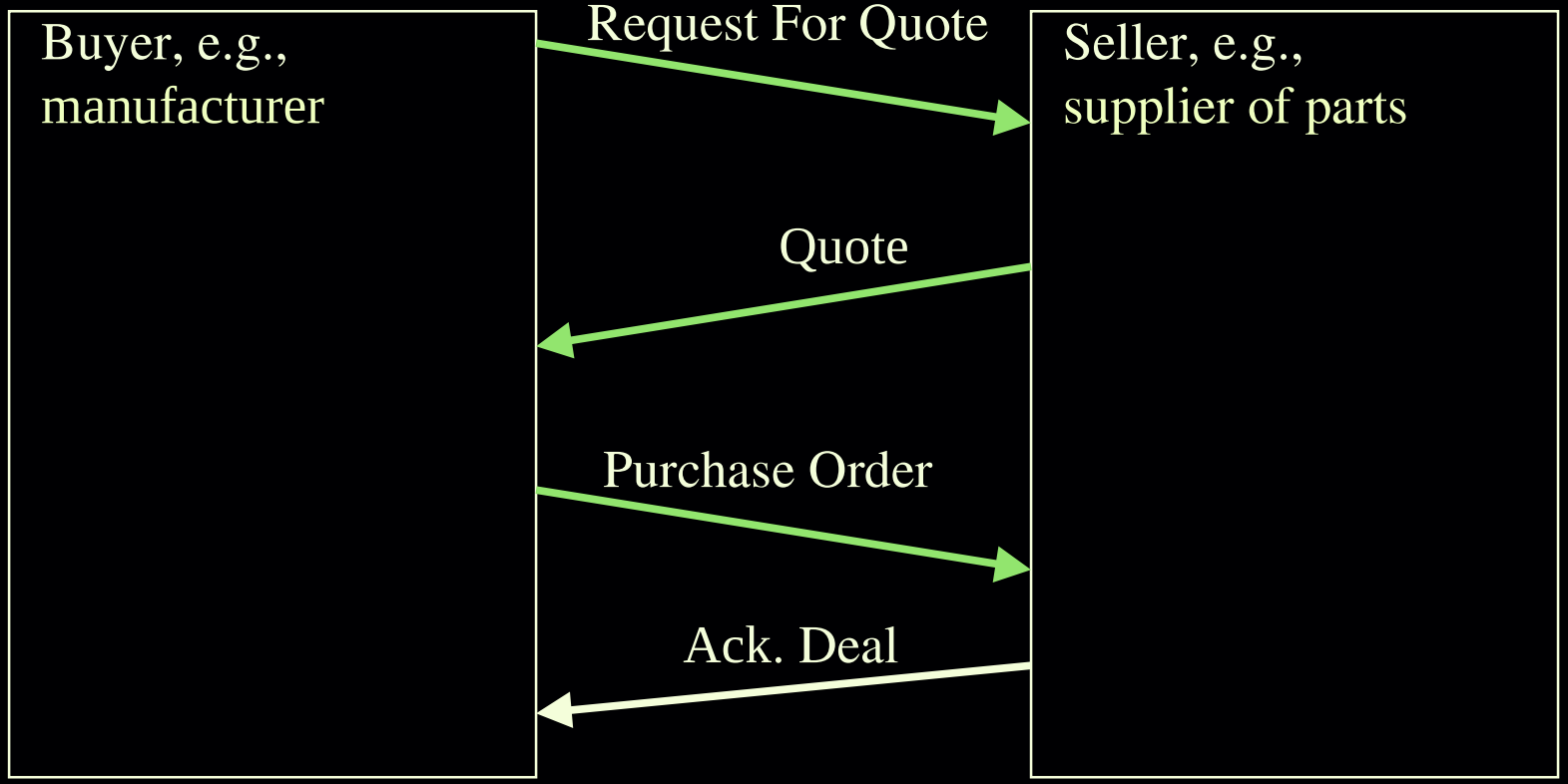
- Use Interlingua to represent contents of:
  - Requests For Quotation or Proposal, i.e., statements of buyer interests, that initiate negotiation, esp. inter-enterprise in B2B.
  - responses to such RFQ's / RFP's by seller: bids, proposals, quotes, ….
  - proposals and counter-proposals and "side information" 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

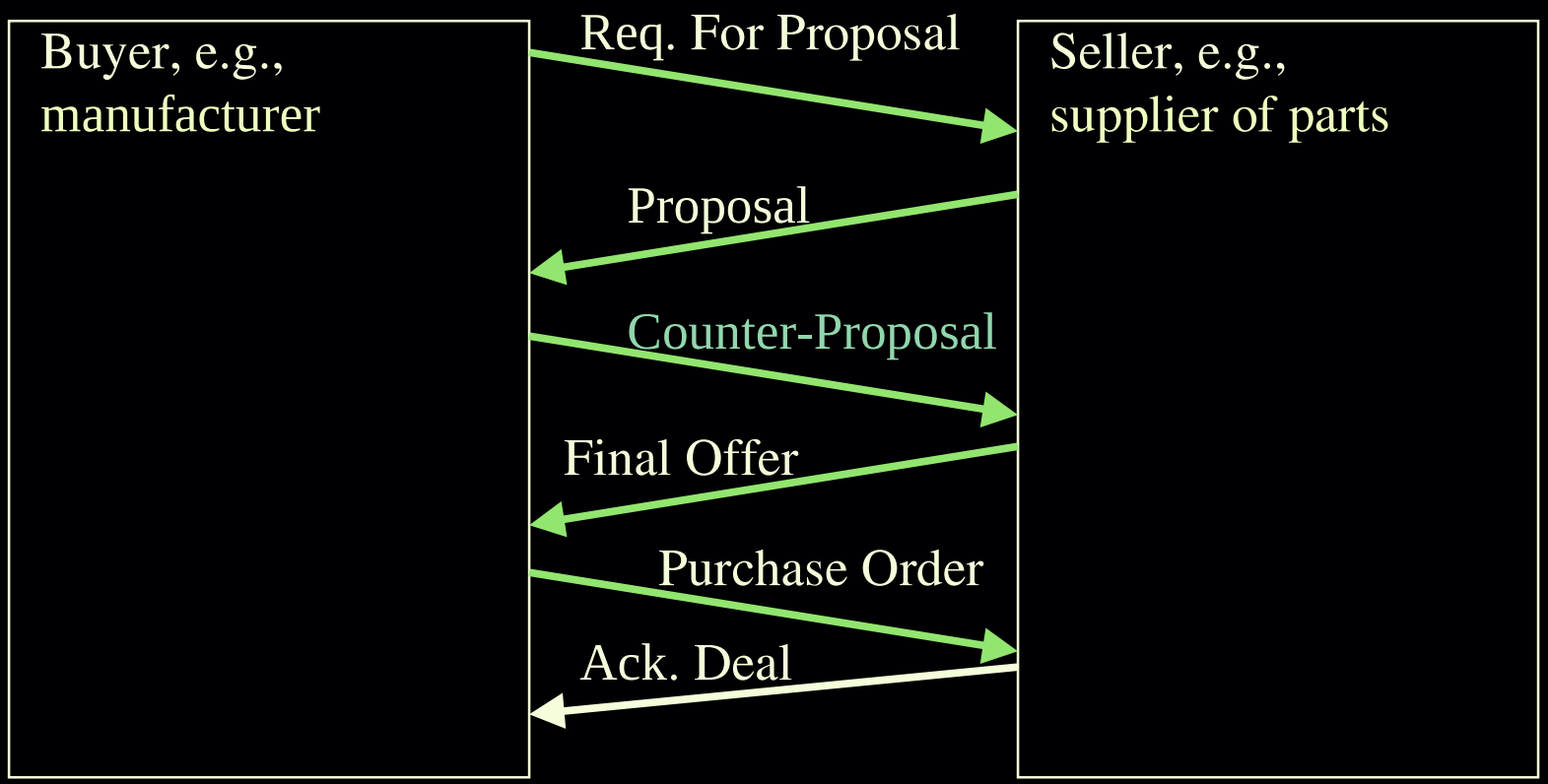


Contracting parties NEGOTIATE via shared rules.

Exchange of Rules Content during Negotiation: example



Exchange of Rules Content during Negotiation: example



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) ←
- purchaseOrder(?PO, supplierCo, ?AnyBuyer) ∧
- quantity_ordered(?PO, ?Q) ∧ (?Q ≥ 5) ∧ (?Q ≤ 1000) ∧
- shipping_date(?PO, ?D) ∧ (?D ≥ 24Apr00) ∧ (?D ≤ 12May00).
- <volumeDiscount> price(per_unit, ?PO, $51) ←
- purchaseOrder(?PO, supplierCo, ?AnyBuyer) ∧
- quantity_ordered(?PO, ?Q) ∧ (?Q ≥ 100) ∧ (?Q ≤ 1000) ∧
- shipping_date(?PO, ?D) ∧ (?D ≥ 28Apr00) ∧ (?D ≤ 12May00) .
  overrides(volumeDiscount , usualPrice) .
- ⊥ ← price(per_unit, ?PO, ?X) ∧ price(per_unit, ?PO, ?Y) GIVEN (?X ≠ ?Y).
- ...

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

- …
  <usualPrice> price(per_unit, ?PO, $60) ← ...
- <volumeDiscount> price(per_unit, ?PO, $51) ←
- purchaseOrder(?PO, supplierCo, ?AnyBuyer) ∧
- quantity_ordered(?PO, ?Q) ∧ (?Q ≥ 5) ∧ (?Q ≤ 1000) ∧
- shipping_date(?PO, ?D) ∧ (?D ≥ 28Apr00) ∧ (?D ≤ 12May00) .
  overrides(volumeDiscount , usualPrice) .
- ⊥ ← price(per_unit, ?PO, ?X) ∧ price(per_unit, ?PO, ?Y) GIVEN (?X ≠ ?Y).
- <aSpecialDeal> price(per_unit, ?PO, $48) ←
- purchaseOrder(?PO, supplierCo, manufCo) ∧
- quantity_ordered(?PO, ?Q) ∧ (?Q ≥ 400) ∧ (?Q ≤ 1000) ∧
- shipping_date(?PO, ?D) ∧ (?D ≥ 02May00) ∧ (?D ≤ 12May00) .
- overrides(aSpecialDeal, volumeDiscount) .
- overrides(aSpecialDeal , usualPrice) .
- ...

Simply *added* rules!

Configuring Auctions and Negotiations

- Specify negotiations including to configure auction mechanisms:
  - what are the negotiable parameters of a contract/deal
  - which goods, which attributes (e.g., price, delivery-date) are at issue.
  - ontology for such specification: e.g., separable vs. inseparable param.'s.
  - after auction, the negotiated parameters are added as high-priority facts to complete the contract.
  - rules about how to choose auction parameters.
- Pilot: configuring AuctionBot for travel trading-agent competition.

  - Collaborators: Dan Reeves, Prof. Michael Wellman of U. Michigan

More Application Pilots

- negotiation, e.g., supply chain collaboration.
- auctions, e.g., travel.
- storefronts, e.g., books.
- authorization, e.g., supply chain collaboration.

Conclusions

- 1. Contracts and business policies: much interesting content is rules.
- 2. How to represent rules is an issue.
  - Desiderata: declarative, inter-operable, non-monotonic, executable
  - KIF has major limitations: monotonic, pure-belief, no markup.
- 3. (Declarative) Ordinary Logic Programs: nicely inter-operable + executable.
- 4. Courteous Logic Programs:
  - Radical advance in modularity & conflict handling.
  - Tractably compilable to OLP.
- 5. Business Rules Markup Language; Advantages of XML-fying (incl. KIF).
- 6. Implementation available: IBM CommonRules (free Java alpha).
- 7. Application piloting:
  - negotiation, e.g., supply chain collaboration (EECOMS $29M project).
- 8. *Current Work including DARPA Agent Markup Language, Semantic Web:*
  - Situated CLP's for procedural attachments.
  - Trust, delegation, representing multi-agent transfers of beliefs.

Implications for FIPA

- Rules content deserves focus for e-business.
- CLP/BRML as candidate content language for FIPA.
- Content modularity in ContractNet FIPA protocol.

- Representational issues for logical formalism used in specifying FIPA ACL:
  - ?Want non-monotonicity?
  - ?Want reducibility to logic programs?

- Thanks!
- Questions?
- For More Info:
  - http://www.mit.edu/people/bgrosof/home.html
    - links to http://www.research.ibm.com/rules/

OPTIONAL SLIDES I FOLLOW: about CURRENT WORK

DARPA Agent Markup Language (DAML)

- DAML program, funded by DARPA:
  - raise the semantic level of agent communication over Web.
  - in XML.
- Doing grant project, collaborating with Prof. Tim Finin (UMBC).
- Another MIT DAML grant project is led by Tim Berners-Lee:
  - fleshing out the "semantic Web" concept.
  - has large commonality of goals/issues with ours:
    - knowledge representation, trust/authorization, reasoning about sources.

XML-ify-ing KIF

- … is work-in-progress (with Yannis Labrou)
- Full KIF (classical logic), not just overlap with CLP.

Situated LP's: Motivation from Contracts

- For executable contract specification:
  - procedural attachments is esp. useful
  - ... thus situated 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.

Trust in larger context of Business Policies and Contracts

- Trust/authorization is often closely tied to other business policies, e.g., pricing, bidding, customer selection, lead-time, service level. E.g.,
  - Risk of new business partner B, when supplier S makes bid.
    - ? Will B fulfill its commitments if B places an order?
    - ? Will S lose by reserving capacity while awaiting B's decision?
    - ? Will B leak information to competitors about B's pricing & capacity?

- *From another viewpoint:* Trust is what contracts are all about:
  - Contracts encode agreements that define conditions of trust.

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: Goal and Basic Approach

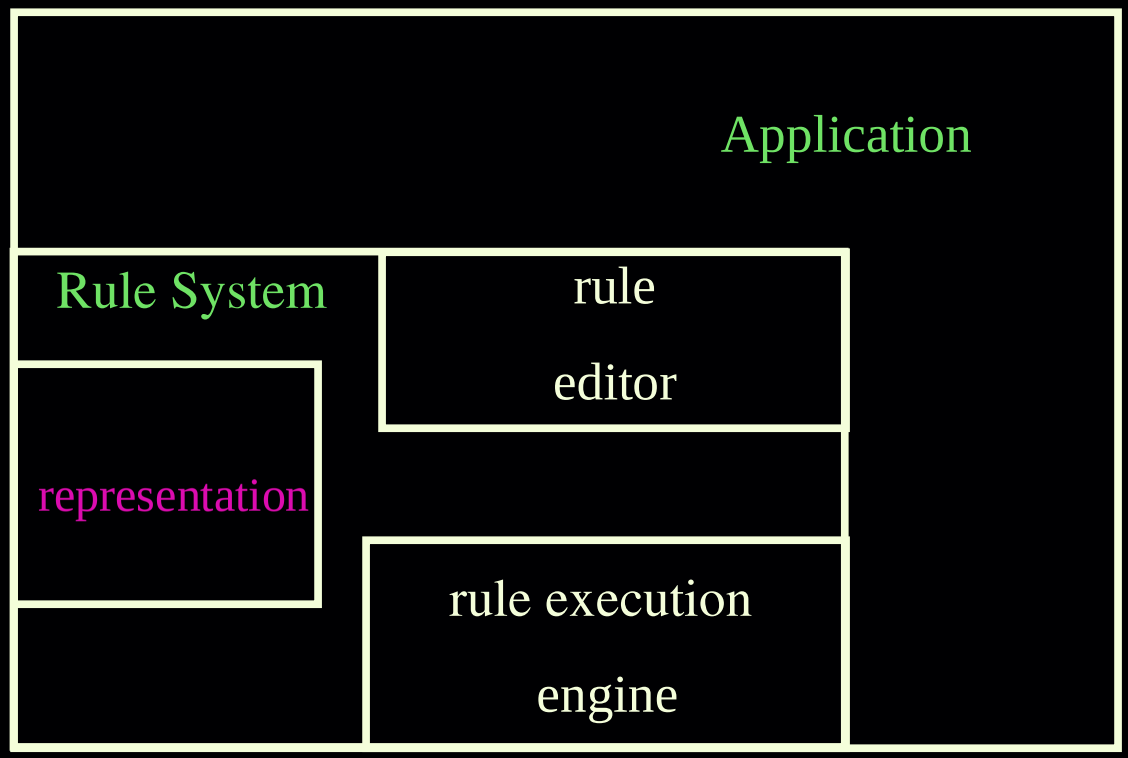
- Our goal: Develop a language that
  - can represent, with significant expressive power, policies and credentials for authorization in Internet scenarios
  - can provide mechanisms for delegation
  - has a clear declarative semantics

- 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)

OPTIONALS SLIDES II FOLLOW: MISC. ADD'NAL DETAILS

Application Using Rules



Content Languages

- Knowledge Sharing Effort (KSE) of early 1990's: Layered approach.
  - Propositional attitudes, e.g., query, inform, promise, fear.
  - **Content**: propositions, e.g., facts, rules.
  - Ontology: vocabulary, definitions.
- Multiple content languages:
  - Pre-existing: STEP, SQL, OQL, CCL, ...
  - KSE developed KIF for aim of being general-purpose.

Applications of Rules: earlier work on Agent Building Environment

- Can view generally in terms of business processes, including workflow.
  - rules are good to capture if-then conditionality esp. involving chaining.
- Embeddable technology for building rule-based intelligent agent capabilities into applications.
- Class of applications: filtering and routing of info items
  - mail, news, Lotus Notes documents
  - customer service / help desk
  - workflow in manufacturing: design changes, plant floor alerts
  - *also:* shopping agent

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)
- ← from(?msg,?s) ∧ customer(?s) ∧ ~urgency(?msg,low). *Example Rule*
-
- where the "?" prefix indicates a logical variable.
- Generally, a rule has the form of Head IF Body :
- H ← B_1 ∧ ... ∧ B_j ∧ ~B_j+1 ∧ ... ∧ ~B_m .
- where m ≥ 0 ; ∧ stands for logical "AND"; ← stands for logical "IF"; and H, B_1, ..., B_m are each an atom with form: Predicate(Term_1, ..., Term_k).
- A predicate = a relation. An atom semantically denotes a boolean.
- ~ stands for negation-as-failure (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).)

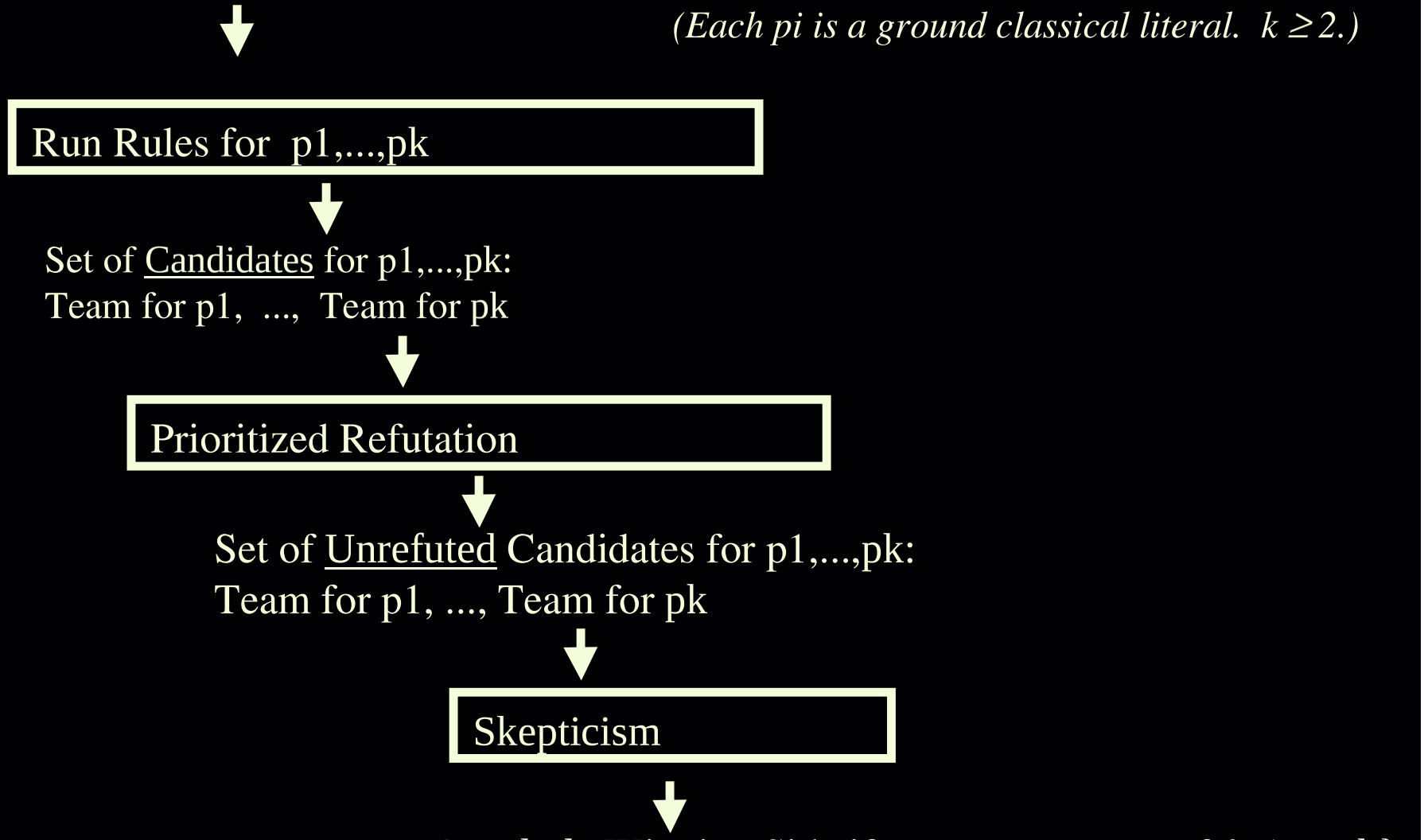
Courteous LP's: more details

- Optionally, insert here:
  - 3 phases of argumentation in
    - courteous semantics
    - post-compilation rules
  - sample post-compilation rule set

Prioritized argumentation in an opposition-locale.

Conclusions from opposition-locales previous to this opposition-locale {p1,...,pk}

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



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 $\bot \leftarrow p \wedge \neg p$, 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.

What Was Hard, What Is Clever (I)

- "Business rules are a nice notion, but they are inextricably tied to messy semantics of specific fragments of programming-language-level code and the particular programming languages and tools they are used with."
- "Rules cf. expert systems and Prolog are old hat and limited in usefulness to much smaller niches than say SQL for example." (Previous mainstream views.)
- → Analysis:
  - Declarative (O)LP is widely shared semantics: restricted but broadly expressive. It includes SQL as well as Prolog etc.
  - Arrival of XML era enables & encourages an interlingua.
    - Not a programming language. Not for experts only.

What Was Hard, What Is Clever (II)

- "Non-monotonic reasoning, in the sense of a principled declarative knowledge representation, is a very very hard area in which early high hopes have been followed by disappointing lack of progress towards intuitive semantics with computational tractability. It's impractical, theory-only, and irrelevant in any but the long-term." (Previous research-community mainstream view.)
  - related: "Rules don't scale well in specification: too hard to maintain."
- $\rightarrow$ Invent: Computationally practical, usefully expressive, intuitively natural, modular form of non-mon.: Courteous LP's.
  - Deep theoretical insight: locality notion; restrictions.
  - Mutual exclusions notion: motivated by applications.
  - Applications value: modular merge/update, dynamism.

What Was Hard, What Is Clever (III)

- "Adopting a new more expressive form of rules, requiring new engines and programmer training, faces a large hurdle of effort and incentives due to legacy inertia. It will only happen fairly slowly. Critical mass will be hard to achieve." (Conventional wisdom about software tools.)
- → Invent: Compiler approach for courteous etc..
  - Component bolts on to legacy technology.
  - Deep theoretical insight underlying it.

Web Storefronts: bookstore example

- B2C personalized promotions:
  - discounting *ex.: ← amounts purchased, group memberships, store card.*
  - showing targeted ads with incentives *ex.: ← genres purchased.*
- Rules & facts from:
  - marketing managers: with updates & merges
    - priorities from recency, authority, specificity
      - *ex.: premiere-customer care overrides accounting late-payment rules.*
      - *ex.: store manager's rule about humor; seasonal about calendars & gifts.*
  - data mining *ex.: ad targeting ← browsing behavior.*
  - DB *ex.: purchasing and browsing-history facts.*
  - dynamic Web session data *ex.: navigation & shopping-basket facts.*

INSERT Bookstore Web E-Storefront App Example SLIDES

- Running example in CommonRules: includes about 60 rules and facts.

- See IBM Research Report RC 21473 “DIPLOMAT...Demonstration”, which is an extended version of the paper appearing at AAAI-99 conference. This is available at http://www.research.ibm.com/rules/ .

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: Logic programs (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 (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.*