# Standardizing XML Rules: *Rules for E-Business on the Semantic Web* (Invited Talk)

Benjamin Grosof

MIT Sloan School of Management, Information Technology group

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

*Slides presented at IJCAI-01 Workshop on E-business and the Intelligent Web, Aug. 5, 2001*

*http://www.ijcai-01.org ; http://www.csd.abdn.ac.uk/ebiweb/*

# *Outline of Talk*

- Introduction:  Background, Motivation
- Fundamental Technical Issues and Approaches
  - heterogeneous commercial rule systems/rep'ns
  - evolutionary strategy for standards
  - logic programs and extensions
- Latest iteration:  RuleML
  - Webizing
- Next Steps

# *Important KR's today in E-Business*

- Rules, relational databases
  - emerging standard: RuleML
- Description Logic, frames, taxonomies
  - emerging standard: DAML+OIL
- (other) Classical Logic
  - emerging standard: Knowledge Interchange Format (KIF)
- Bayes Nets & Decision Theory: probabilities, dependencies, utilities
  - early, primarily for researchers: Bayes Net Interchange Format (BNIF)
- (other) Data Mining inductive predictive models: neural nets, associations, fuzzy, regressions, … -- early: Predictive Model Markup Lang.
- *Arguably:* Semi-Structured Data: XML Query, RDF
- *Arguably:* UML

# *Applications of Agent Communication in Knowledge-Based E-Markets (KBEM)*

- Bids  in auctions and reverse auctions
- Orders  in supply chain or B2C
- Contracts/Deals/Proposals/RequestsForProposals
  - prices;  product/service descriptions;  refunds, contingencies
- Buyer/Seller interests, preferences, capabilities, profiles
  - recommender systems; yellow pages; catalogs
- Ratings, reputations; customer feedback or problems
- Demand forecasts  in manufacturing supply chain
- Constraints  in travel planning
- Creditworthiness, trustworthiness, 3rd-party recommendations
- *Industry-verticals:  computer parts, real estate, …*

# *Technology Research Directions: KR for Agent Communication*

- Aims:
  - deeper reasoning intra-agent
    - "understanding" what receive
  - more modularity in:
    - content
    - software engineering
  - KR of the kind needed for e-market applications
    - catalogs, contracts, negotiation/auctions, trust, profiles/preferences/targeting, …
  - play with XML standards, capabilities, mentality

# *Technology Research Direction: KR 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

# *Why Standardize Rules Now?*

- Rules as a form of KR (knowledge representation) are especially useful:
  - relatively mature from basic research viewpoint
  - good for prescriptive specifications (vs. descriptive)
    - a restricted programming mechanism
  - integrate well into commercially mainstream software engineering, e.g., OO and DB
    - easily embeddable; familiar
    - vendors interested already: Webizing, app. dev. tools
- *⇒⇒ Identified as part of mission of the W3C Semantic Web Activity*

# *Vision: Uses of Rules in E-Business*

- 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; → 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.

# *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.)*

# *Standardizing XML Rules: Overall Goals*

- Provide a basis for a standardized rule markup language, with declarative KR semantics
  - interoperability of heterogeneous rule systems and applications
  - information integration of heterogeneous rule KB's/services
- Start with commercially important flavors of rules
- Start simple with a kernel KR, then add extensions incrementally.

# *Standardizing XML Rules: More Goals*

- Add extensions incrementally to:
  - raise KR expressiveness and syntactic convenience
  - connect cleanly to procedural mechanisms
  - pass-thru/bundle-in system-specific (meta-)info
  - exploit Web-world functionality, standards
- Synergize with other KR aspects of Semantic Web:
  - RDF; Ontologies: DAML+OIL/Description-Logic
    - rules in/for ontologies, ontologies for/of rules
- Complement XML non-SW ontologies already evolving
- Synergize with other Web standards: P3P APPEL, XML Query, Web Services, ...

# *Incremental Strategy of Standards Development*

- *Initial Step*: *Keep It Simple*, focus primarily on:
  - Currently Commercially Important (CCI) kinds of rules
  - with XML syntax
  - with shared semantics and interoperability
  - *BUT: foresee to max. smooth evolution, back-compatibility*
- *Later*: get *fancier* in regard to:
  - Web-izing: features, synergy with other standards
  - KR expressiveness
  - incorporate new fundamental research results & consensus
- *Rationale: speed acceptance & deployment; avoid "bleeding edge"*

# *Technical Challenge #1: which initial core KR semantics?*

- *Analytic Insight [many]:*
  - Horn FOL is a shared KRsem. E.g., KIF conformance level
- *Analytic Insight [Grosof 99]:*
  - *!!Can do better -- closer, more expressive!!*
  - Start with Horn Logic Program (LP), esp. Datalog
    - closer correspondence to what CCI rule systems actually do
    - generate ground-literal conclusions only, no other "tautologies" (e.g., OR's)
    - Unique Names Assumption (UNA) is typical; opt.: explicitly add equalities
    - {Datalog + {bounded # logical variables per rule} } is frequent, tractable
  - Extend LP to negation, priorities, procedures
    - needed in CCI rule systems, fairly well-understood fundamentally

# *Technical Challenge #2: how to handle CCI non-monotonicity?*

- CCI non-monotonicity is heavily used, includes:
  - negation
  - priorities (Prolog, OPS5, DB updates, inheritance exceptions)
    - Common CCI Theme: enable modularity in specification

- *Analytic Insight [many]:*
  - negation-as-failure (NAF), not classical negation, is the form of negation typically used in CCI
    - more natural/easy to implement, more flexible

# *Semantics of Negation As Failure in CCI*

- canonical semantics of NAF in LP is well-understood theoretically since 1990's:
  - Well-Founded Semantics (WFS); nuanced for unrestrictedly recursive rules
  - consensus has formed in fundamental research community
  - only modestly increases computational complexity compared to Horn (frequently linear, at worst quadratic)

- ...but practice in Prolog and other CCI is often "sloppy" (incomplete / cut-corners) relative to canonical semantics
  - in cases of recursive rules, WFS algorithms required are more complex
  - ongoing diffusion of WFS theory & algorithms, beginning in Prolog's

# *Ordinary Logic Programs as Shared KR*

- {Horn LP} + NAF = “Ordinary” LP (OLP)
  - a.k.a. “general”, “normal”, …
  - e.g., “pure” Prolog is backward-direction OLP

# *how to handle CCI non-monotonicity? continued*

- *Synthetic Insight* [Grosof 97..99]:
  - "Courteous" LP (CLP) [Grosof 97..99] is able to represent the basic kinds of priorities used in CCI
    - static rule sequence, e.g., in Prolog
    - dynamically-computed rule sequence, e.g., in OPS5
    - inheritance with exceptions
    - DB updates
  - CLP only moderately increases computational complexity compared to OLP (frequently linear, worst-case cubic)
  - CLP modular for software engineering
    - compileable into OLP (preserving ontology)

# *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)
- ← 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.

# *Technical Challenge #3: how to handle CCI procedural aspects?*

- *Ignoring procedural control (cf. inferencing control strategies)…*
- CCI procedural aspects are heavily used, including:
  - Prolog: built-ins
  - OPS5/ECA: actions, some conditions
    - key to embeddability in mainstream software dev.
  - "triggers" and "active rules" in relational DB's
- *Analytic Insight* [Grosof 99]:
  - view as procedural attachments (cf. KR theory)

# *how to handle CCI procedural aspects? continued*

- *Synthetic Insight* [Grosof 95..00]:
  - "Situated" LP (SLP) [Grosof 97..00] appears able to represent the basic kinds of procedural attachments used in CCI, though with more discipline(/restrictions)
    - "aproc" = external attached procedure
    - "effecting": drawing pure-belief conclusion triggers invocation of action aproc for sake of its side-effects
    - "sensing": test pure-belief antecedent condition by invoking purely-informational query to aproc
    - discipline: restrict state changes from external procedures
      - querying (sensor) attached procedures does not change state
      - performing effector associate predicates with external procedures

# *Situated LP's: Overview*

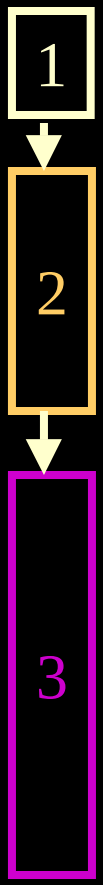
- phoneNumberOfPredicate ::s:: BoeingBluePagesClass.getPhoneMethod .
  *ex. Of sensor statement*
- shouldSendPagePredicate ::e:: ATTPagerClass.goPageMethod .
  *ex. effector statement*
- Sensor procedure may require some arguments to be ground, i.e., bound; in general it has a specified binding-signature.
- 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.

# *Going Beyond KIF*

- KIF is KR Ag. Comm. Lang.'s point of departure:
  - Intent:  general-knowledge interlingua.
  - Emerging standard, in ANSI commmittee.
  - Main focus:  classical logic, esp. first-order.
    - This is the declarative core, with deep semantics.
  - Has major limitations:
    - general-purpose-ness
    - logically monotonic
    - pure-belief
      - no invoking of procedures external to the inference engine.

# *Criteria for Agent-Communication Rule Representation*




- *High-level:* Agents reach common understanding; ruleset 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

# *IBM's Business Rules Markup Language (BRML) and CommonRules*

- The above approach with SCLP as core KR has been...
- embodied in IBM BRML 1.0 .. 2.1 [mid-99 to mid-00]
- implemented in IBM CommonRules 1.0 .. 2.1

- Limitations:
  - 1-vendor
  - shallow: XML/Web mechanisms/ conceptualization
  - shallow: ontologies

# *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)

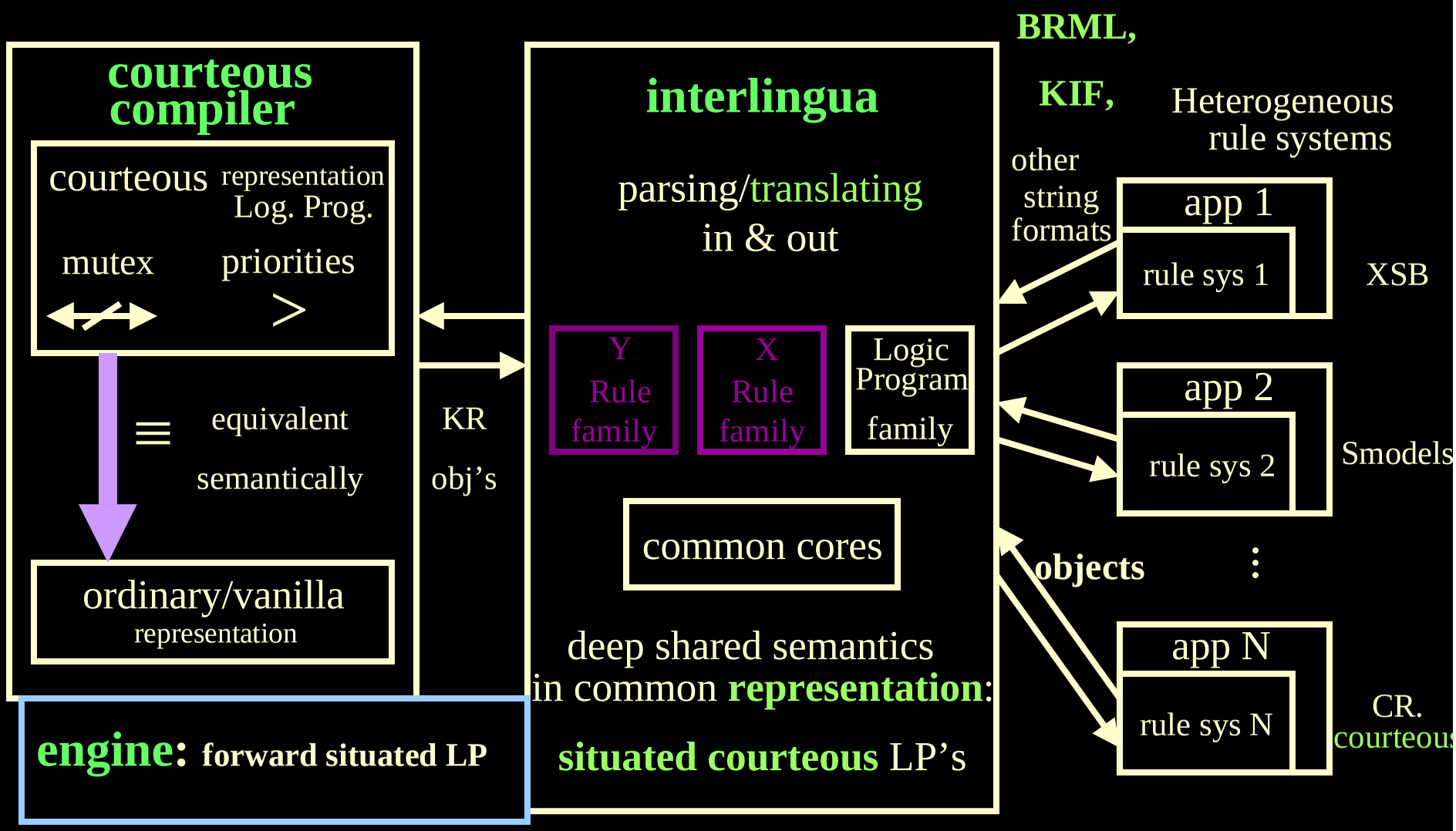
& KIF.

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

BRML is content language for XML-ified FIPA

Agent Communication Language.

# *Current-version IBM CommonRules*



# *RuleML Initiative [8/00..present] takes BRML as point of departure*

- Limitations of BRML:
  - 1-vendor
  - shallow: XML/Web mechanisms/ conceptualization
  - shallow: ontologies

- RuleML: [DTD V0.7 1/01 , V0.8 7/01]
  - independent of any one vendor
  - deepen wrt XML/Web mechanisms/conceptualization

# *RuleML Initiative Organization*

- Organization currently informal
- Aiming to morph into W3C activity, if possible
- A dozen or so “participant” institutions from each of academe and industry
- A lot of mindshare already:  W3C, DAML; BOF’s

- http://www.mit.edu/~bgrosof/#XMLRules, points at http://www.dfki.de/ruleml

# *RuleML has some First Steps of Webizing Rule KR*

- URIs for logical vocabulary and knowledge subsets
  - RuleML V0.8: predicates, functions, rules, rulebases
  - RuleML V0.8: labels for rules/rulebases
- Support RDF:
  - RuleML V0.8:
    - syntax: mostly unorderedness of graph
    - … with explicit orderedness
    - partial first drafts of alternative RDF syntax
- Support evolution and tight description of KR expressive classes:
  - RuleML Syntax defined as generalization-specialization lattice of DTD's
    - uses XML entity mechanism

# *RuleML's First Steps of Webizing Rule KR (continued)*

- Exploratory features in RuleML 0.8 [FEEDBACK PLEASE!]:
  - meta "role" convention in DTD: to aid RDF-friendliness
  - argument "roles" for atom/term argument lists
    - step toward OO support and RDF support

- RuleML Tools beginning to appear
  - several links on website

# *Next Steps*

- RDF version of syntax -- planned for RuleML soon
- XML Schema version of Syntax specification -- planned for RuleML soon
- Situated Courteous LP DTD for RuleML -- my draft, soon public
- Implementation of translation and inferencing
  - MIT Sloan has work in progress
  - IBM has announced it will support in CommonRules V3
- “Header” meta-data
  - specify KR incl. expressive/syntactic restrictions

# *News from W3C*

- Informally Announced by Tim Berners-Lee at DAML mtg 7/20/01
- Internest Sub-Group with discussion list forming within the W3C Semantic Web Activity
- Mission statement being drafted as we speak
- Goal:  create charter and consensus for potential W3C Working Group on Rules, within Semantic Web Activity
  - maybe form W3C SW Rules WG in 6months
  - sibling of soon-to-be-formed W3C SW WG on Ontologies

- Contact me by email if interested.

# *My Current Related Research*

- Combine Ontologies (cf. DAML+OIL) with Rules
  - [joint with DAML'ers and others]
- "Distributed Belief Transfer"
- Use for Web Services
- Applications, incl. contracts as rulesets

- Thanks!
- Questions?
- For More Info:
  - http://www.mit.edu/~bgrosof → #XMLRules

# *OPTIONAL SLIDES FOLLOW*

# *Current Uses of Rules in E-Business*

- Inferencing in
  - business rules
  - workflow
  - database queries and triggers
  - intelligent agents, KB systems
- Transformation in (XML) document translation
- *Identified as a Design Issue of the W3C Semantic Web*

# *Automating 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.

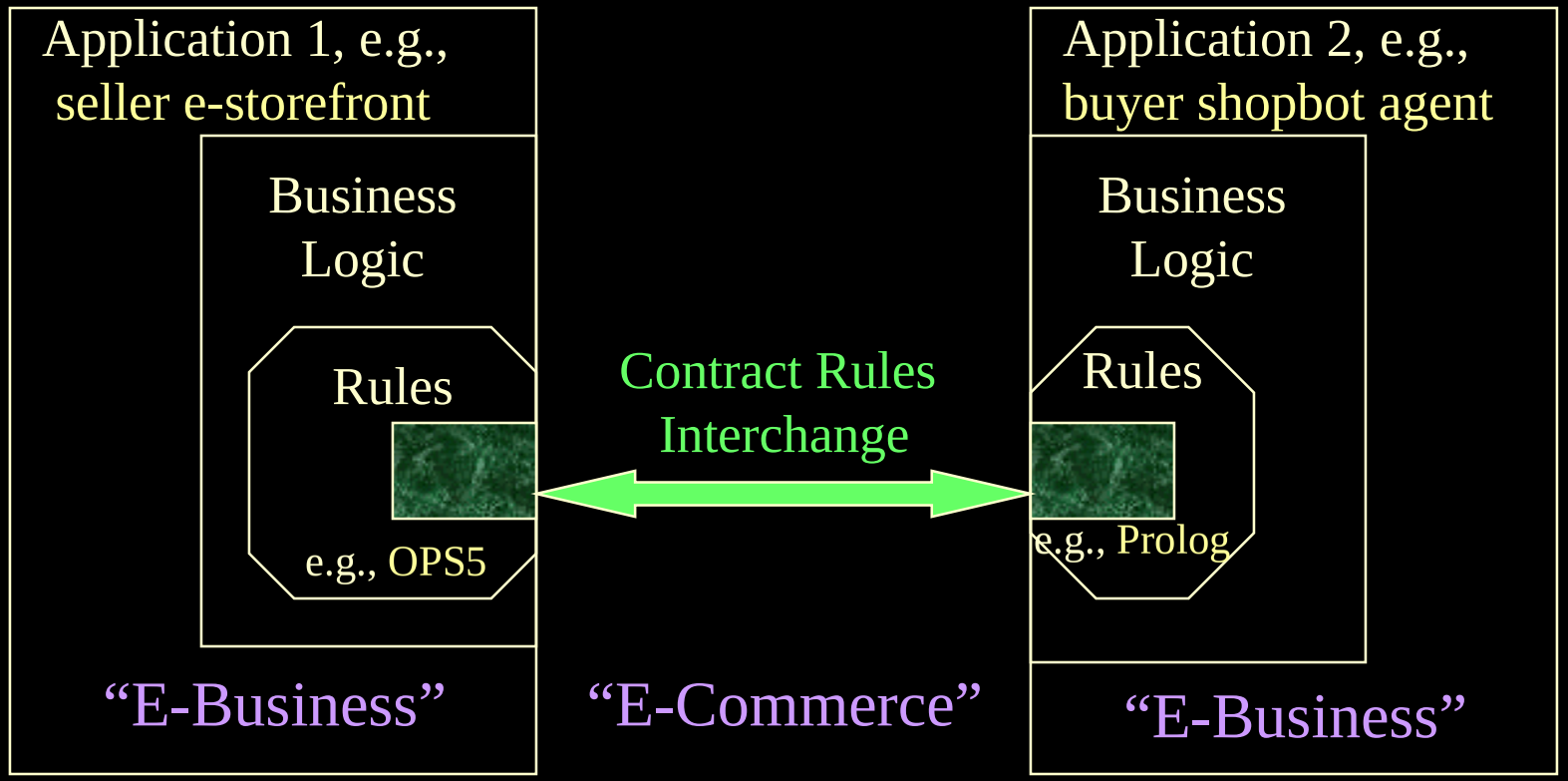
# *Idea/Vision #1: Rule-based Contracts for E-commerce*

- 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 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.

# *Contract Rules across Applications / Enterprises*




*Contracting parties integrate e-businesses via shared rules.*

# *RuleML: Overall Goals*

- Provide a basis for a standardized rule markup approach, with declarative knowledge representation (KR) semantics
  - Aid integration of heterogeneous rule systems and applications, via shared rule markup language
  - Start with commercially important flavors of rules
  - Complement XML ontologies already evolving for various domains
- Start simple with a kernel KR, then add extensions incrementally.
- Become an industry standard (e.g. via W3C)

# *Technical Approach of RuleML*

- Start with: Datalog Logic Programs with rules labeled *as kernel*
  - similar to Business Rules Markup Language *(IBM CommonRules)*
- Add: expressive extensions/restrictions, URI's
  - negation-as-failure (well-founded semantics); classical negation
  - prioritized conflict handling cf. Courteous Logic Programs *(stays tractable!)*
    - modular rulesets; modular compiler to Ordinary Logic Programs
  - procedural attachments: actions, queries ; cf. Situated Logic Programs
  - logical functions: standard built-ins, user-defined
  - 1st-order logic type expressiveness cf. Lloyd LP's, DAML+OIL, KIF
  - *more:* equivalence/rewriting rules; ... temporal, Bayesian, fuzzy, …
- Family of DTD's: a generalization-specialization hierarchy (lattice)
  - define DTD's modularly, using XML entities (~macros)
  - optional header to describe expressive-class using "meta-"ontology

# *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, ...

# *RuleML: Further Directions*

- move to XML Schema based rather than DTD based
- additional XML syntaxes: RDF; surface/"style-sheeted"
- more KR's: KIF/classical, Notation 3, Bayesian, fuzzy, rewriting, temporal, …

- provide Rule mechanism to emerging W3C standards:
  - Semantic Web / RDF, P3P, …

# *RuleML: Relevant Other Efforts in W3C and Markup*

- RDF, RDFS, DAML(+OIL), Semantic Web
- P3P privacy policies: APPEL rules
- XML Query

- Others:
  - XSLT
  - Predictive Model Markup Language (rules from data mining)

# *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.

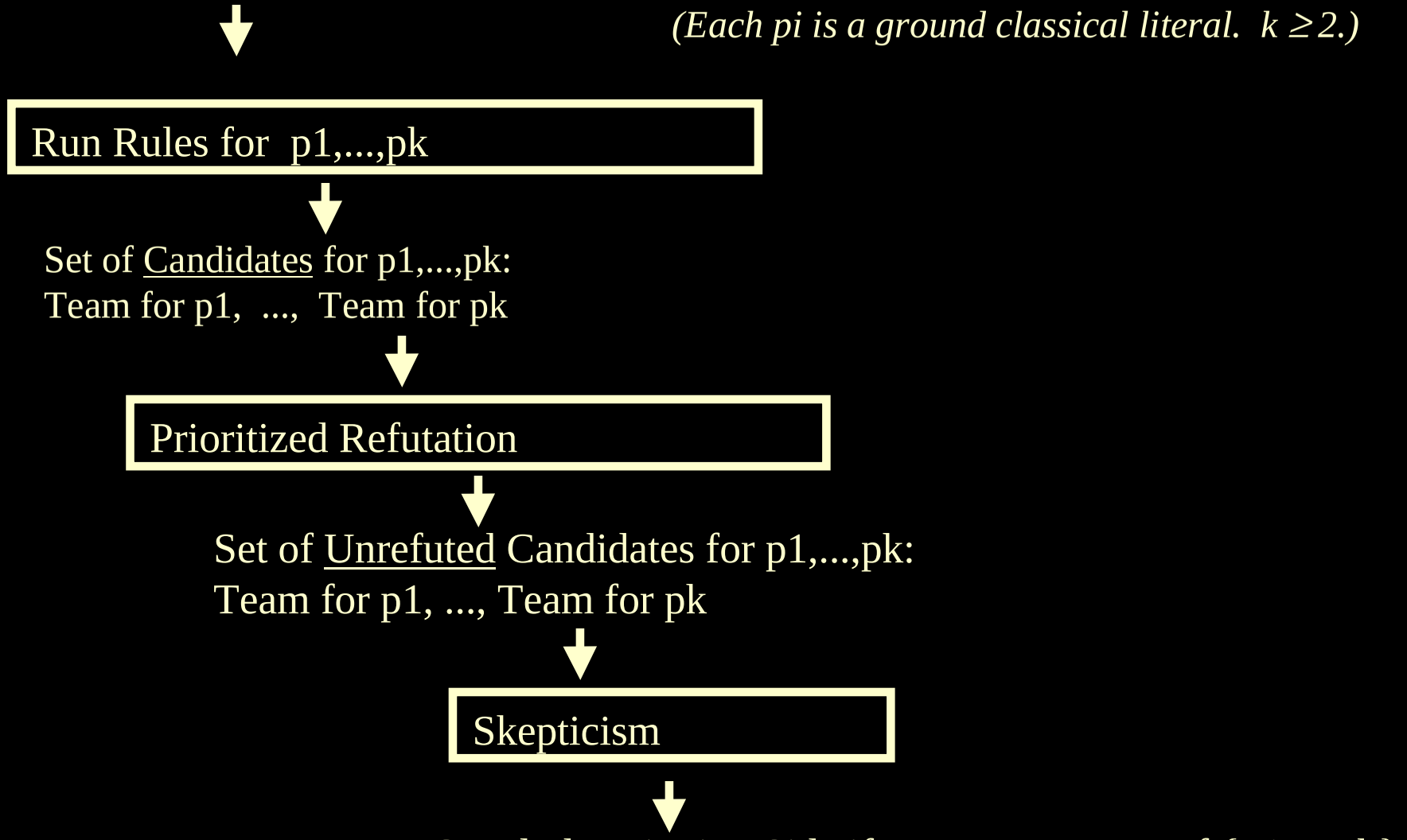
# *Priorities are available and useful*

- 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.

*Prioritized argumentation in an opposition-locale.*

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

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




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

# *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.

# *Summary: Courteous (Situated) LP's 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. And consistency.
  - courteous compiler → modularity in software engineering.
  - mutex's & conflict locales → keep tractability. (Compiler is O(n^3).)