# DAML Rules Update and Issues

Expressive Features and Abstract Syntax;
Use Cases & Scenarios, Requirements, and Tools;
RuleML & relationships to RDF, OWL, Query, and Services;
Description Logic Programs, Procedural Attachments, and Negation

*Presentation for Rules Breakout sessions of DAML PI Meeting,*
*Apr. 8-10, 2003, Miami, FL, USA. [http://www.daml.org](http://www.daml.org)*

Benjamin Grosof*

MIT Sloan School of Management

bgrosof@mit.edu [http://www.mit.edu/~bgrosof/](http://www.mit.edu/~bgrosof/)

Thanks to Mike Dean* and Stefan Decker for agenda suggestions.

* co-leads of DAML Rules effort

## *OUTLINE OF SLIDES*

- Primer  Presentation (15min), from Apr. 8, 2003
  - Introduction
  - Background on Description Logic Programs
- Main  Breakout's  Presentations (totaling 1 hour), from April 9, 2003
  - except for part by Stefan Decker on Use Cases, and some other skimmed documents – RuleML Working Note outline and RuleML abstract syntax excerpts by B. Grosof
- Outbrief Presentation (20min), from April 10, 2003

- Optional Slides
  - SweetDeal
  - Semantic Web Services
  - DLP Background

## *Primer: Intro*

## *What is "DAML Rules"?*

- Generally: new rules stuff specifically related to DAML program
  - e.g., OWL, DAML-Services, and their application scenarios
- Focus: RuleML (esp. since Oct '02 PI Meeting)
  - Horn Logic Programs + extensions/restrictions = sub-languages
  - Webizing: URI's for predicates etc., facilitate modules
  - Negation as failure, prioritized conflict handling, strong negation
  - "Reactivity": Procedural attachments for actions, queries; events
- Language Expressive Features, Syntax; Tools; Use Cases, Scenarios
- Relationships to OWL and RDF and Query:
  - OWL/RDFS ontologies used or defined by Rules
  - Description Logic Programs semantics for ↔ OWL
  - RDF, OWL syntaxes for RuleML; unordered abstract syntax to bridge
  - Relationships to DQL, RDF Query approaches; expressiveness needed
- Use in Services, security
- Coordination with:
  - Joint Committee, RuleML Initiative, W3C, SWS Coalition, Oasis
  - *(These are locus of most technical discussions on Rules, to date.)*

## *Top-Level Goals -- for overall Breakout*

- Update all on latest relevant progress and news
  - e.g., there's lots on relationships to RDF, OWL, Query, W3C
- Share news generally from folks -- e.g., what tools using / making
- Discuss technical issues, e.g., relationships to RDF, OWL, Query, Services
- Set some near-term focus and plans for DAML Rules effort

## *Focus Areas -- for overall Breakout*

- requirements, use cases, and language features
  - negation & defaults? procedural attachments? Major commercial systems all have them!
  - more use cases needed – where?

- relationship to RDF, OWL, Query
  - Syntax directions?: abstract syntax approach; “object-oriented” argument collections; RDF, OWL encodings; queries incl. path / graph expressions
  - Expressive focus?: Description LP for OWL; ~ Horn for RDF Query
  - Concepts of combinations?: E.g., also: pile of DL $\cup$ LP axioms.

- relationship to Services and security
  - procedural attachments/“reactivity” – how critical?

## *Breakout Agenda -- Schedule*

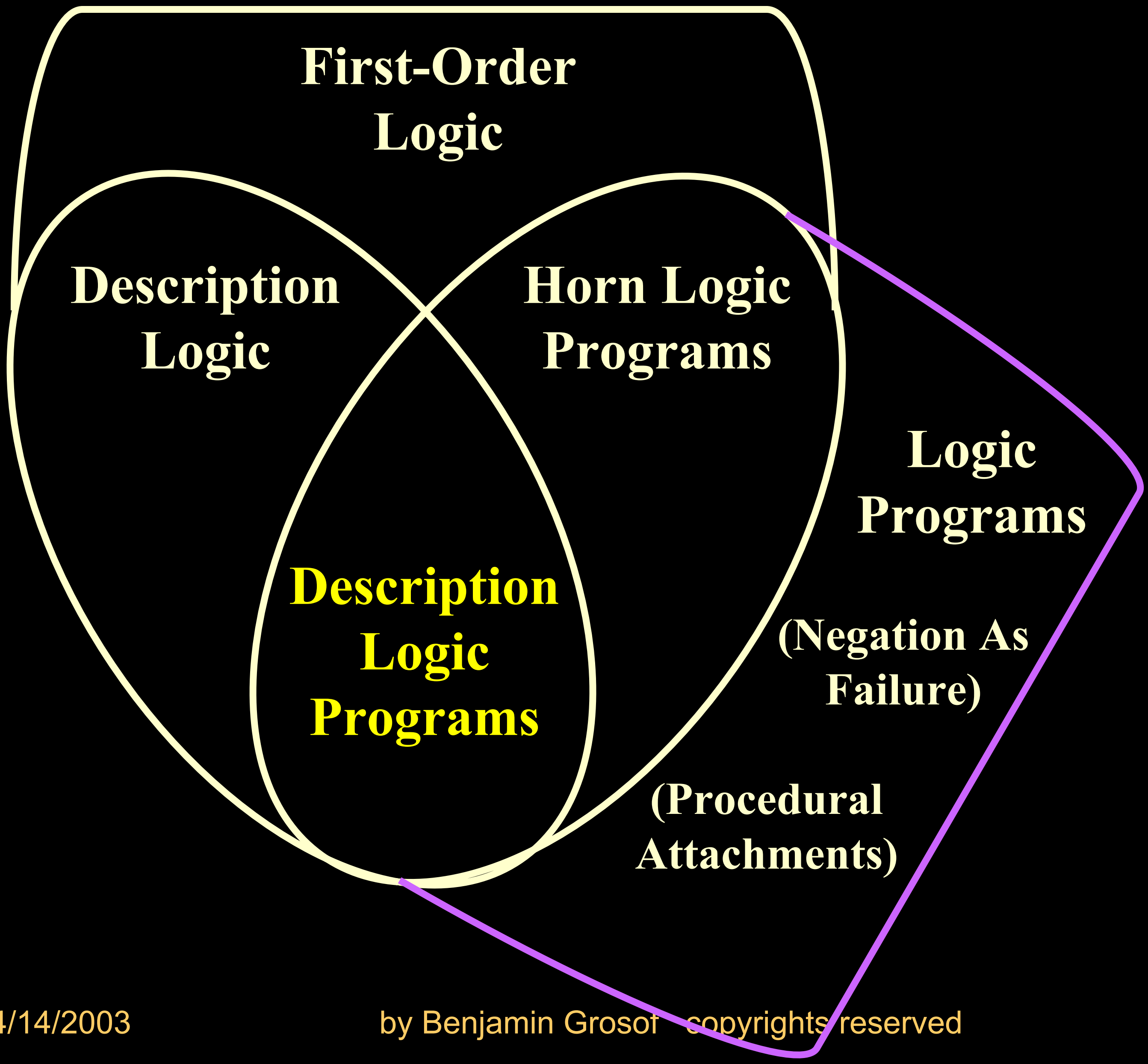
- *1. 13:00-13:25* Overall Update on DAML Rules and RuleML
- *2. 13:25-13:50* Rules Use Cases and Requirements effort
- *3. 13:50-14:50* RuleML in relation to RDF, OWL, and Query
- 10-min BREAK
- *4. 15:00-15:30* Rules and Services
- *5. 15:30-16:00* Setting Focus and Plans

## *Coordination with other breakouts*

- In Services breakout:
  - Rules in use cases & scenarios (9:00-10:00)
- In Query breakout:
  - Rules relationship to RDF Query approaches incl. DQL     (sometime during 10:00 - 12:00)

## *Primer: DLP Background*

## *Venn Diagram: Expressive Overlaps among KR's*



# *updated Overview of DLP Features*

- DLP captures a complete subset of DL, containing RDFS plus more
- RDFS subset of DL permits the following statements:
  - Subclass, Domain, Range, Subproperty (also SameClass, SameProperty)
  - instance of class, instance of property
- DLP also completely captures following DL statements beyond RDFS:
  - Using the Intersection connective (conjunction) in class descriptions
  - Stating that a property (or inverse) P is Transitive or Symmetric.
  - (Some other stuff)
  - "OWL Feather"
- DLP can *largely but partially* capture: most other DL features.
  - Use skolemization, explicit equality, integrity constraints.
- Translation simpler to define from DL $\Rightarrow$ LP than DL $\Leftarrow$ LP.
- Bridge easily to Relational DBMS (SQL) – which is LP-based.
  - *Scaleability of LP/DB engines >> DL engines , as |instances|* $\uparrow$.

## *LP as a superset of DLP*

- "Full" LP, including with non-monotonicity and procedural attachments, can thus be viewed as including an "ontology sub-language", namely the DLP subset of DL.

## *LP Task Scenarios / Use Cases*

- Key aim:  import DL ontologies into LP rulebase.
- 
- ⇒   Consistency of the result/merge is an issue.

- Ways to achieve robustness:
  - 1. Use DLP for ontologies, rather than full DL.
  - 2. Exploit LP's nonmonotonic expressiveness:
    - Negation as failure; or more generally:
    - Courteous LP's  prioritized conflict handling

## *Hybrid DL+LP Task Scenarios/Use-Cases*

- 1. Service descriptions combining LP rules and DL ontologies
- 2. Rules for knowledge translation: e.g.,
  - translating/merging ontologies (or rules)

## *MAIN SLIDES FOLLOW*

## *PART I. SLIDES FOLLOW*

# *Part I. Overall Update -- Outline*

- *Intro: Goals, Focus, Agenda*
- Description Logic Programs: expressiveness ↑ ; papers ; tool
- RuleML language features; Working Note outline (Boley, Grosof, & Tabet)
- Rules Uses Cases & Requirements draft (Decker, Dean, & McGuinness)
- relationship to Query in RDF, incl. DQL
  - survey draft (Prud'hommeaux & Grosof)
  - use cases drafts (Miller, Reggiori & Seaborne)
- RDF/OWL syntax for RuleML:
  - abstract syntax, object-oriented argument collections, minimizing order
- W3C News: on Query & Rules, e.g. Plenary Mar '03, www-rdf-rules
- News: RuleML tools, scenarios
- Upcoming: ISWC Rules Workshop (deadline 6/15)

## *More: updated Overview of DLP Features*

- DLP captures a complete subset of DL, containing RDFS plus more
- RDFS subset of DL permits the following statements:
  - Subclass, Domain, Range, Subproperty (also SameClass, SameProperty)
  - instance of class, instance of property
- DLP also completely captures following DL statements beyond RDFS:
  - Using the Intersection connective (conjunction) in class descriptions
  - Stating that a property (or inverse) P is Transitive or Symmetric.
  - (Some other stuff:) disjunction or existential in subclass expression, universal in superclass expression.
  - "OWL Feather" – subset of OWL Lite
- DLP can *largely but partially* capture: most other DL features:
  - Cardinality, existential in superclass, universal in subclass, functionality of property (or inverse).
  - But NOT: (general) negation, disjunction in superclass
  - Use skolemization, explicit equality, integrity constraints.
- Translation simpler to define from DL $\Rightarrow$ LP than DL $\Leftarrow$ LP.
- Bridge easily to Relational DBMS (SQL) – which is LP-based.
  - *Scaleability of LP/DB engines >> DL engines, as |instances| $\uparrow$.*

## *more details on Overall Update*

- Description Logic Programs:
  - WWW-2003 paper [Grosof, Horrocks, Volz, & Decker]
  - Follow-on working paper [Volz, Motik, Horrocks, & Grosof] on more expressiveness, SweetOnto translator tool for OWL to RuleML and DB
  - SweetOnto tool to be available publicly in ?May

- relationship to Query in RDF, incl. DQL
  - survey draft (Prud'hommeaux & Grosof)
    - Horn fundamental expressiveness seems to suffice ?
    - Path/graph expressions required in syntax?
  - use cases drafts (Miller, Reggiori & Seaborne)
    - Lessons?

## *more details on Overall Update, continued*

- RuleML language features; Working Note outline (Boley, Grosof, & Tabet)
- (see file ruleml-working-note-summary-040803.txt)

## *more details on Overall Update, continued*

- W3C News: lot of interest in Query & Rules, e.g.
  - W3C Plenary Mar '03 discussions at Semantic Web Architecture sessions
  - Many different systems already
  - www-rdf-rules as interest group that combines
  - Joint Committee archives public
  - RuleML / DAML Rules technical discussion mainly on Joint Committee and/or www-rdf-rules mailing lists
  - Issue: focus of potential new Working Group
  - Plan: RuleML Working Note, Rules Use Cases, WG Charter

## *more details on Overall Update, continued*

- News: RuleML tools, implemented scenarios
  - Several new tools available now or soon
    - Editors, translators, inference engines
    - XSB, Jess, OWL, SQL, KIF
  - New implemented application scenarios:
    - financial knowledge integration (ECOIN)
  - See www.ruleml.org and www.daml.org/rules and ebusiness.mit.edu/bgrosof

## *Breakout Agenda -- Schedule*

- *1. 13:00-13:25* Overall Update on DAML Rules and RuleML
- *2. 13:25-13:50* Rules Use Cases and Requirements effort
- *3. 13:50-14:50* RuleML in relation to RDF, OWL, and Query
- 10-min BREAK
- *4. 15:00-15:30* Rules and Services
- *5. 15:30-16:00* Setting Focus and Plans

## *PART I. DISCUSSION*

- all share their news
  - how DAML'ers are using rules now
- agenda refinement

## *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.
    - SQL99 even has recursive rules.
- Production rules (OPS5 heritage): e.g.,
    – Jess, 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, e.g., XSB: *"logic programs" as a full programming language.*
- *(Lesser: other knowledge-based systems.)*

## *PART II.  SLIDES*

- Presentation by Stefan Decker on Use Cases effort by him and collaborators
- See separate file(s)

## *Breakout Agenda -- Schedule*

- *1. 13:00-13:25* Overall Update on DAML Rules and RuleML
- *2. 13:25-13:50* Rules Use Cases and Requirements effort
- *3. 13:50-14:50* RuleML in relation to RDF, OWL, and Query
- 10-min BREAK
- *4. 15:00-15:30* Rules and Services
- *5. 15:30-16:00* Setting Focus and Plans

## *PART III.*
## *Suggested Discussion Focus*

- Relationships to OWL and RDF and Query:
  - OWL/RDFS ontologies used or defined by Rules
  - Description Logic Programs semantics for ↔ OWL
  - RDF, OWL syntaxes for RuleML
    - unordered abstract syntax to bridge
  - Relationships to DQL, RDF Query approaches; expressiveness needed:
    - Horn enough for RDF Query?
    - Path/graph expression syntax needed for RDF Query?
    - Lessons from RDF Query use cases?

## *PART III. Agenda*

- 1350-1415 background presentation
  - *proposed ABSTRACT SYNTAX for RuleML: approach, examples
    - encoding RuleML syntax in RDF or OWL
    - unorderedness in RDF/OWL vs. orderedness in XML-S, commercial systems
    - object-oriented argument collections in RuleML
  - List of other topics, in prep for discussion
    - rules on top of ontologies, e.g., in SweetDeal
    - Description Logic Programs
    - RDF triples as facts in rules
    - relationship to RDF Query Systems and to DQL
    - querying remote systems via procedural attachments
    - mixing of RuleML encoded in RDF/OWL with use by rules of OWL ontologies
    - Rules expressive features: which and where are useful
    - scenarios of usage of rules together with RDF Query, DQL

- 1415-1450 discussion

## *PART III. Intro to Abstract Syntax for RuleML, continued*

- Address need for syntax specification to interoperate between current XML-Schema/DTD spec and:
  - RDF encoding
  - OWL encoding
  - Human-oriented concise string syntax, e.g., Prolog-y or Lisp-y style
  - Alternatives within XML-S, DTD, OWLwrt “Abstract Syntax for RuleML”

## *PART III. Intro to GBNF*

- Challenge: unordered (OWL, RDF) vs. ordered (XML-S)
- Challenge: represent contents vs. macro expansion

- New meta-syntax: GBNF “Generalized BNF for XML” or “Grosof BNF”
  – Unordered concatenation AND ordered concat.
  – Containment statements AND macro statements
  – Spirit of semi-structured databases, plus schema info
    - Treat attributes as elements; treat their defaults as pre-processing macro

## *PART III. Intro to Abstract Syntax for RuleML, continued*

- Various Expressive Features
- Object-oriented style
  - Unordered yet unambiguous children as contents
  - “roled lists”: Argument collections for a predicate/atom or function/term
    - with named user-defined “roles”, similar to columns of a DB relation
  - AND tuples
  - Nestably
- Quite concise.

## *PART III. Presentation on Abstract Syntax for RuleML*

- wrt "Abstract Syntax for RuleML":
  - – see file of working draft by B. Grosof:
    - ruleml-abstract-syntax-032803-excerpts.txt

## *PART III. Presentation on OWL Syntax for RuleML*

- DAML+OIL syntax for RuleML (“DamlRuleML”) since Apr ‘02exists already
- DamlRuleML draft was specified and translator was implemented to (XML-DTD) RuleML and to Jess, as part of SweetJess work
- See paper “SweetJess: Translating DamlRuleML to Jess”
  - by [Grosof, Gandhe, & Finin], Proc. Rules Workshop at ISWC 2002. Also available at http://ebusiness.mit.edu/bgrosof

# *Translating a Rule from (Daml)RuleML to Jess*

```
<damlRuleML:imp>
   <damlRuleML:_rlab>
      <damlRuleML:ind>steadySpender</damlRuleML:ind>
   </damlRuleML:_rlab>
   <damlRuleML:_body>
     <damlRuleML:andb>
        <damlRuleML:atom>
           <damlRuleML:_opr>
              <damlRuleML:rel>shopper<damlRuleML:rel>
           </damlRuleML:_opr>
           <damlRuleML:var>Cust</damlRuleML:var>
        </damlRuleML:atom>
        <damlRuleML:atom>
          <damlRuleML:_opr>
             <damlRuleML:rel>spendingHistory<damlRuleML:rel>
          </damlRuleML:_opr>
          <damlRuleML:tup>
            <damlRuleML:var>Cust</damlRuleML:var>
            <damlRuleML:ind>loyal</damlRuleML:ind>
          </damlRuleML:tup>
        </damlRuleML:atom>
      </damlRuleML:andb>
    </damlRuleML:_body>
```

## *Continued: Translating a Rule from (Daml)RuleML to Jess*

```
<damlRuleML:_head>
    <damlRuleML:atom>
      <damlRuleML:_opr>
        <damlRuleML:rel>giveDiscount<damlRuleML:rel>
      </damlRuleML:_opr>
      <damlRuleML:tup>
       <damlRuleML:ind>percent5</damlRuleML:ind>
       <damlRuleML:var>Cust</damlRuleML:var>
      </damlRuleML:tup>
    </damlRuleML:atom>
   </damlRuleML:_head>
  </damlRuleML:imp>
```

Equivalent in  JESS:

```
(defrule steadySpender
    (shopper ?Cust)
    (spendingHistory ?Cust loyal)
    =>
    (assert (giveDiscount percent5 ?Cust) ) )
```

## *PART III. More Topics*

- rules on top of ontologies, e.g., in SweetDeal
- Description Logic Programs
- RDF triples as facts in rules
- relationship to RDF Query Systems and to DQL
- querying remote systems via procedural attachments
- mixing of RuleML encoded in RDF/OWL with use by rules of OWL ontologies
- rules expressive features: which and where are useful
- scenarios of usage of rules together with RDF Query, DQL

## *Breakout Agenda -- Schedule*

- *1. 13:00-13:25* Overall Update on DAML Rules and RuleML
- *2. 13:25-13:50* Rules Use Cases and Requirements effort
- *3. 13:50-14:50* RuleML in relation to RDF, OWL, and Query
- 10-min BREAK
- *4. 15:00-15:30* Rules and Services
- *5. 15:30-16:00* Setting Focus and Plans

## *PART IV. Background – Outline*

- Rule-based Semantic Web Services
  - Motivate procedural attachments, e.g., for actions in business processes

- Situated Logic Programs, as declarative abstraction of usual kinds of procedural attachments

## *Rule-based Semantic Web Services*

- Rules/LP in appropriate combination with DL as KR, for RSWS
  - DL good for categorizing: a service overall, its inputs, its outputs
- Rules to describe service process models
  - rules good for representing:
    - preconditions and postconditions, their contingent relationships
    - contingent behavior/features of the service more generally,
      - e.g., exceptions/problems
  - familiarity and naturalness of rules to software/knowledge engineers
- Rules to specify deals about services: cf. e-contracting.

## *Rule-based Semantic Web Services*

- Rules often good to executably specify service process models
  - e.g., business process automation using procedural attachments to perform side-effectful/state-changing actions ("effectors" triggered by drawing of conclusions)
  - e.g., rules obtain info via procedural attachments ("sensors" test rule conditions)
  - e.g., rules for knowledge translation or inferencing
  - e.g., info services exposing relational DBs
- Infrastructural: rule system functionality as services:
  - e.g., inferencing, translation

## *Application Scenarios for Rule-based Semantic Web Services*

- SweetDeal [Grosof & Poon 2002] configurable reusable <u>e-contracts</u>:
  - LP <u>rules</u> about agent contracts with exception handling
  - … <u>on top of</u> DL <u>ontologies</u> about business processes;
  - *a scenario motivating DLP*

- Other:
  - <u>Trust</u> management / <u>authorization</u> (Delegation Logic) [Li, Grosof, & Feigenbaum 2000]
  - <u>Financial</u> knowledge integration (ECOIN) [Firat, Madnick, & Grosof 2002]
  - <u>Privacy</u> policies (P3P APPEL)
  - Business <u>policies</u>, more generally

## *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.
    - SQL99 even has recursive rules.
- Production rules (OPS5 heritage): e.g.,
    - Jess, 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, e.g., XSB: *"logic programs" as a full programming language.*
- *(Lesser: other knowledge-based systems.)*

## *Heavy Reliance on Procedural Attachments in Currently Commercially Important Rule Families*

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

- Relational databases, SQL: Built-in sensors, e.g., for arithmetic, comparisons, aggregations. Sometimes effectors: active rules / triggers.

- Production rules (OPS5 heritage): e.g., Jess
  – Pluggable (and built-in) sensors and effectors.
- Event-Condition-Action rules:
  – Pluggable (and built-in) sensors and effectors.

- Prolog: e.g., XSB.
  – Built-in sensors and effectors. More recent systems: more pluggability of the built-in attached procedures.

# *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 or remote invocation/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.

## *SweetJess: Translating an Effector Statement*

```
<damlRuleML:effe>
  <damlRuleML:_opr>
    <damlRuleML:rel>giveDiscount</damlRuleML:rel>
  </damlRuleML:_opr>
  <damlRuleML:_aproc>
    <damlRuleML:jproc>
      <damlRuleML:meth>setCustomerDiscount</damlRuleML:meth>
      <damlRuleML:clas>orderMgmt.dynamicPricing</damlRuleML:clas>
      <damlRuleML:path>com.widgetsRUs.orderMgmt
        </damlRuleML:path>
    </damlRuleML:jproc>
  </damlRuleML:_aproc>

</damlRuleML:effe>
```

Associates with predicate P : an attached procedure A that is side-effectful.

- Drawing a conclusion about P triggers an action performed by A.

*jproc* = Java attached procedure.

*meth, clas, path* = its methodname, classname, pathname.

Equivalent in JESS: key portion is:

```
(defrule effect_giveDiscount_1
  (giveDiscount ?percentage ?customer)
  =>
  (effector setCustomerDiscount orderMgmt.dynamicPricing
                (create$ ?percentage  ?customer) ) )
```

## *Overview: Semantics of Situated Logic Programs*

- Definitional: complete inferencing+action occurs during an “episode” – intuitively, run all the rules (including invoking effectors and sensors as go), then done.
- Effectors can be viewed as all operating/invoked after complete inferencing has been performed.
  - Independent of inferencing control.
    - But often intuitively less appropriate if only doing backward inferencing.
  - Separates pure-belief conclusion from action.

## *Overview: Semantics of Situated LP, continued*

- Sensors can be viewed as accessing a virtual knowledge base (of facts). Their results simply augment the local set of facts. These can be saved (i.e., cached) during the episode.
  - Independent of inferencing control.
- The sensor attached procedure could be a remote powerful DB or KB system, a web service, or simply some humble procedure.
- Likewise, an effector attached procedure could be a remote web service, or some humble procedure. An interesting case for SW is when it performs updating of a DB or KB, e.g., "delivers an event".

## *Overview of Semantics of Situated LP, continued*

- Conditions:
  - Effectors have only *side* effects: they do not affect operation of the (episode's) inferencing+action engine itself, nor change the (episode's) knowledge base.
  - Sensors are purely informational: they do not have side effects (i.e., any such can be ignored).
  - Timelessness of sensor and effector calls: their results are not dependent on when they are invoked, during a given inferencing episode.
  - "Sensor-safeness": Each rule ensures sufficient (variable) bindings are available to satisfy the binding signature of each sensor associated with any of its body literals – such bindings come from the other, non-sensor literals in the rule body. During overall "testing" of a rule body, sensors needing such bindings can be viewed as invoked after the other literals have been "tested".

## *Overview: Semantics of Situated LP, Continued*

- Generalizations possible:
  - permit multiple sensors or effectors per predicate.
  - sense functions (or terms) not just predicates.
  - permit sensor priority – i.e, specify the prioritization of the facts that result from a particular sensor .
  - associate sensing with atoms/literals (or terms), but this is reducible to sensing predicates (or functions) – by rewriting of the rules.
- Challenge: error handling info returned from attached procedures

## *Example: Notifying a Customer when their Order is Modified*

- See extended version of B. Grosof WITS-2001 conference paper
  - "Representing E-Business Rules on the Semantic Web: Situated Courteous Logic Programs in RuleML"
  - Available at http://ebusiness.mit.edu/bgrosof

## *PART V. Agenda Topics for Discussion*

- Is LP Rules + Common Logic the right focus for “Rules” for
  - DAML?
  - Semantic Web?
  - Semantic Web Services?
- Layering:
  - What focus nearer-term
  - Can view Common Logic / FOL as point in RuleML’s expressiveness lattice (hierarchy) of sub-languages?
- Combining rules with OWL:
  - RuleML (or CommonLogic) on top of OWL ontologies
  - Description LP
  - Object-oriented syntax
  - Abstract syntax
- Use Cases and Application Scenarios

## *PART V. Agenda Topics for Discussion*

- Situated LP notion – useful?
- "Anarchic" scaleaability – challenge for non-monotonicity? For monotonicity?
  - Examples: view definitions in SQL, travel agent rulebase that you hand a set of sources
- Pairwise agent exchange vs. publishing
  - Message passing vs. Webpage-posting
- Implicit, vs. explicit persistently named, specification of rest of KB; explicit assumptions about use of nonmon rulebases
- Overall monotonicity of {KB entails p} relation.

## *Breakout Agenda -- Schedule*

- *1. 13:00-13:25* Overall Update on DAML Rules and RuleML
- *2. 13:25-13:50* Rules Use Cases and Requirements effort
- *3. 13:50-14:50* RuleML in relation to RDF, OWL, and Query
- 10-min BREAK
- *4. 15:00-15:30* Rules and Services
- *5. 15:30-16:00* Setting Focus and Plans

## *OUTBRIEF SLIDES FOLLOW*

## *What is "DAML Rules"?*

- Generally: new rules stuff specifically related to DAML program
  - e.g., OWL, DAML-Services, and their application scenarios
- Focus: RuleML (esp. since Oct '02 PI Meeting)
  - Horn Logic Programs + extensions/restrictions = sub-languages
  - Webizing: URI's for predicates etc., facilitate modules
  - Negation as failure, prioritized conflict handling, strong negation
  - "Reactivity": Procedural attachments for actions, queries; events
- Language Expressive Features, Syntax; Tools; Use Cases, Scenarios
- Relationships to OWL and RDF and Query:
  - OWL/RDFS ontologies used or defined by Rules
  - Description Logic Programs semantics for $\leftrightarrow$ OWL
  - RDF, OWL syntaxes for RuleML; unordered abstract syntax to bridge
  - Relationships to DQL, RDF Query approaches; expressiveness needed
- Use in Services, security
- Coordination with:
  - Joint Committee, RuleML Initiative, W3C, SWS Coalition, Oasis
  - *(These are locus of most technical discussions on Rules, to date.)*

## *Breakout Agenda -- Schedule*

- *1. 13:00-13:25* Overall Update on DAML Rules and RuleML
- *2. 13:25-13:50* Rules Use Cases and Requirements effort
- *3. 13:50-14:50* RuleML in relation to RDF, OWL, and Query
- 10-min BREAK
- *4. 15:00-15:30* Rules and Services
- *5. 15:30-16:00* Setting Focus and Plans

## *Focus Areas -- for overall Breakout*

- requirements, use cases, and language features
  - negation & defaults? procedural attachments? Major commercial systems all have them!
  - more use cases needed – where?

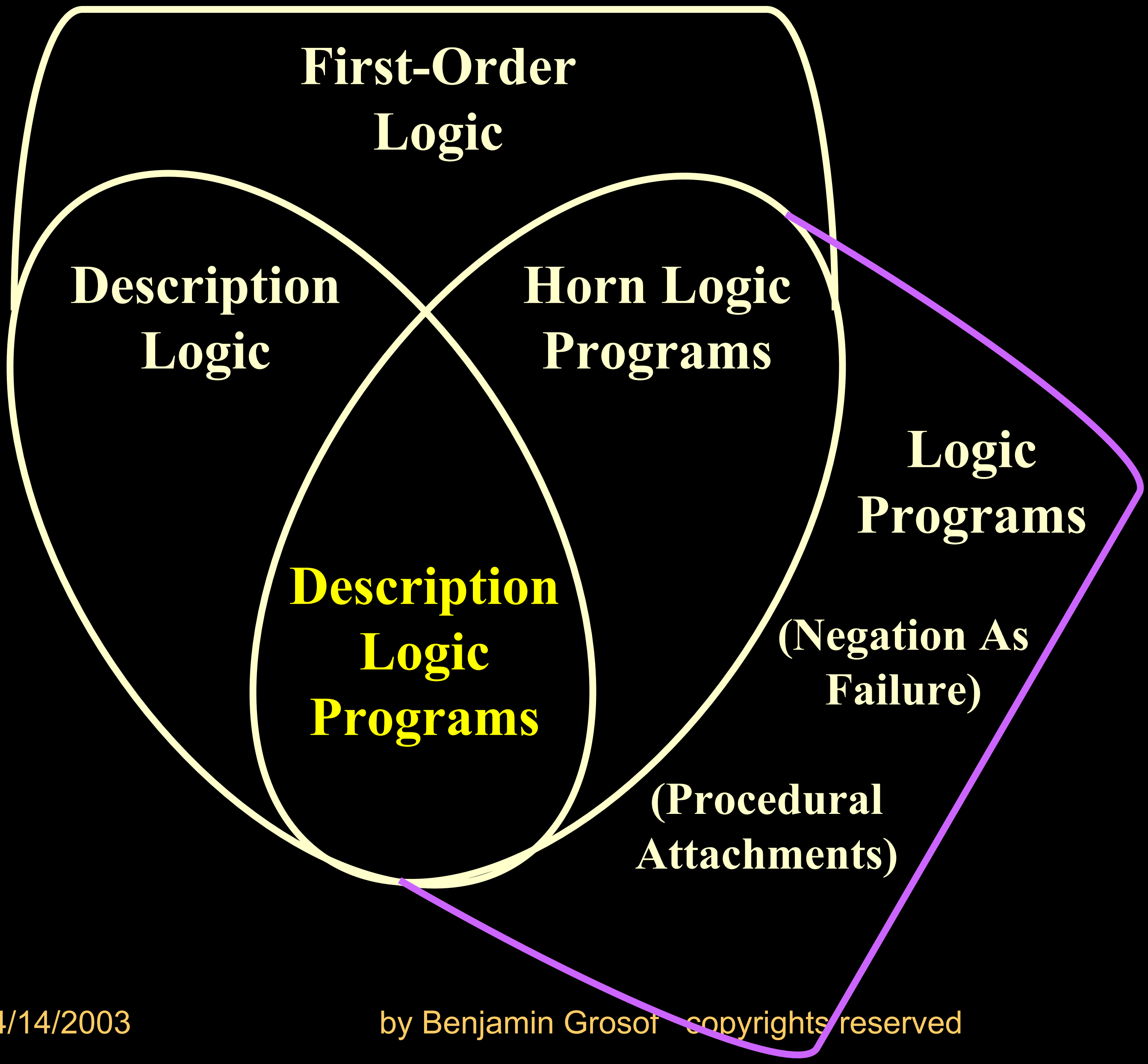
- relationship to RDF, OWL, Query
  - Syntax directions?: abstract syntax approach; “object-oriented” argument collections; RDF, OWL encodings; queries incl. path / graph expressions
  - Expressive focus?: Description LP for OWL; ~ Horn for RDF Query
  - Concepts of combinations?: E.g., also: pile of DL $\cup$ LP axioms.

- relationship to Services and security
  - procedural attachments/“reactivity” – how critical?

## *Breakout Discussion I:  Expressiveness Requirements*

- Two kinds of rules are of interest:
  - 1. LP Rules / RuleML
  - 2. First-order logic / Common Logic
    - some like to  call an implication a “rule”.
  - These have substantial overlap.
  - Common Logic aims to support a RuleML subset
- Rules on top of ontologies – is a vital requirement / usage
  - Description LP a good tool for semantic aspect of this
  - Syntax:  URIref provides the basic capability
- Procedural attachments – are important
  - esp. for services, business processes, and “making the business case for rules”
  - e.g., query service calls upon another query service
  - Not well-understood how to do in First-order logic beyond LP

## *Venn Diagram: Expressive Overlaps among KR's*



## *Breakout Discussion II: Situated Logic Programs*

- Situated LP approach to procedural attachments in LP Rules:
  - Effectors for external side-effectful actions
  - Sensors for purely-informational external querying
  - Declarative semantics:
    - independence from inferencing control strategy
  - Much simpler than general planning or programming

  - Makes assumptions about attached procedures be more explicit
    - Interesting similarity to W3C's normative principles for GET and POST for general Web
  - Interesting approach overall
  - More feedback requested

## *Breakout Discussion III: Syntax*

- "Object-oriented" argument collections feature in RuleML:
  - Is useful (has a long history under various names)
  - … in Common Logic too
  - Interestingly:
    - can treat argument roles as part of ontology
  - Related also to enabling types for variables

- Abstract Syntax proposal for RuleML:
  - Terseness is appealing
    - (57 lines for nearly all current RuleML features.)
  - More feedback requested

## *Breakout Discussion IV: Use Cases*

- Use Cases & Requirements effort is ongoing
  - Stefan Decker presented
- Kinds of uses of rules include:
  - Derivation
  - Reactive, Transformation, Integrity Constraints:
    - Build upon Derivation, may not require (much) more in terms of fundamental expressiveness
- More use cases wanted!!!!!

## *Breakout Discussion V: Rules on the Web*

- *Lots* of discussion!!
- Clarified issue of fundamental goals/uses:
  - 1. "Messaging": Exchange of rules between a few parties or in limited/controlled context
    - Common in e-business, esp. B2B and early adopters
  - 2. *Vs.* "Posting": Fully public / very wide
    - Cf. vision of SW ontologies
  - These have different requirements emphases
    - Driven by different aims for reuse, composition, modification
    - Many felt: (2.) motivates desire for monotonicity
  - "Anarchic" scaleability as a goal
  - "This is Useful vs. "This is True" – clash of intutions?
  - Use cases helpful! E.g., descriptive *vs.* prescriptive; merging, travel agents, e-contracting, DB integration, …

## *Breakout Discussion VI: Nonmonotonicity*

- *Lots* of discussion!! … Actually got somewhere!!

- Meaning of asserting defaults: believed as premises
- Defaults' usefulness often includes:
  - being prescriptive, e.g., in open-source spirit
  - facilitating reuse: simplifies modification often to be just merging/updating
- Rulebase includes facts – crisply defines scope of "world" being closed. (Non-fact) rules and facts may originate from multiple Web sources. Once provided, then semantic closure occurs.

- Nonmon with disjunction/(FOL-LP) is not well understood enough for practicality, yet.

## *Breakout Discussion VII: Nonmon., cont.'d*

- Key requirement for reuse of defaults:
  - enough meta-knowledge about source and intended use context; e.g., reliability, reputation, etc.
- Prioritized default approach, cf. Courteous LP:
  - Many felt: is reasonable point of departure for rules on the Web, esp. when prioritized conflict handling is needed (e.g. Pat!!)
  - Can represent and infer meta-knowledge about sources, e.g.:
    - prioritization for merging/updating, based on authority, expertise, reliability, freshness, etc.
  - Paraconsistent: non-conflicting defaults go thru
  - Handles conflicts & keeps global consistency
  - Reduces tractably to normal LP (Horn + negation as failure)

## *Ongoing Discussion Venues*

- daml-rules@daml.org   DAML-Rules mailing list
- www.daml.org/rules    DAML-Rules  web page
- Joint Committee archives  -- see www.daml.org/committee
  - public to read, but not to post
- www-rdf-rules  W3C mailing list
- RuleML  www.ruleml.org; & ebusiness.mit.edu/bgrosof
  - You can join as a participant, then get on its mailing list
- BOF on Query & Rules at WWW-2003  (eric@w3.org contact)

## *For OPTIONAL SLIDES: see separate file*