Rules and DAML:
Description Logic Programs, Rule-based Semantic Web Services, their Application Scenarios; and RuleML Update

Presentation for DAML PI Meeting,
Oct. 17, 2002, Portland, OR, USA.
This version slightly further edited, dated Oct. 21, 2002.

Benjamin Grosof

MIT Sloan School of Management

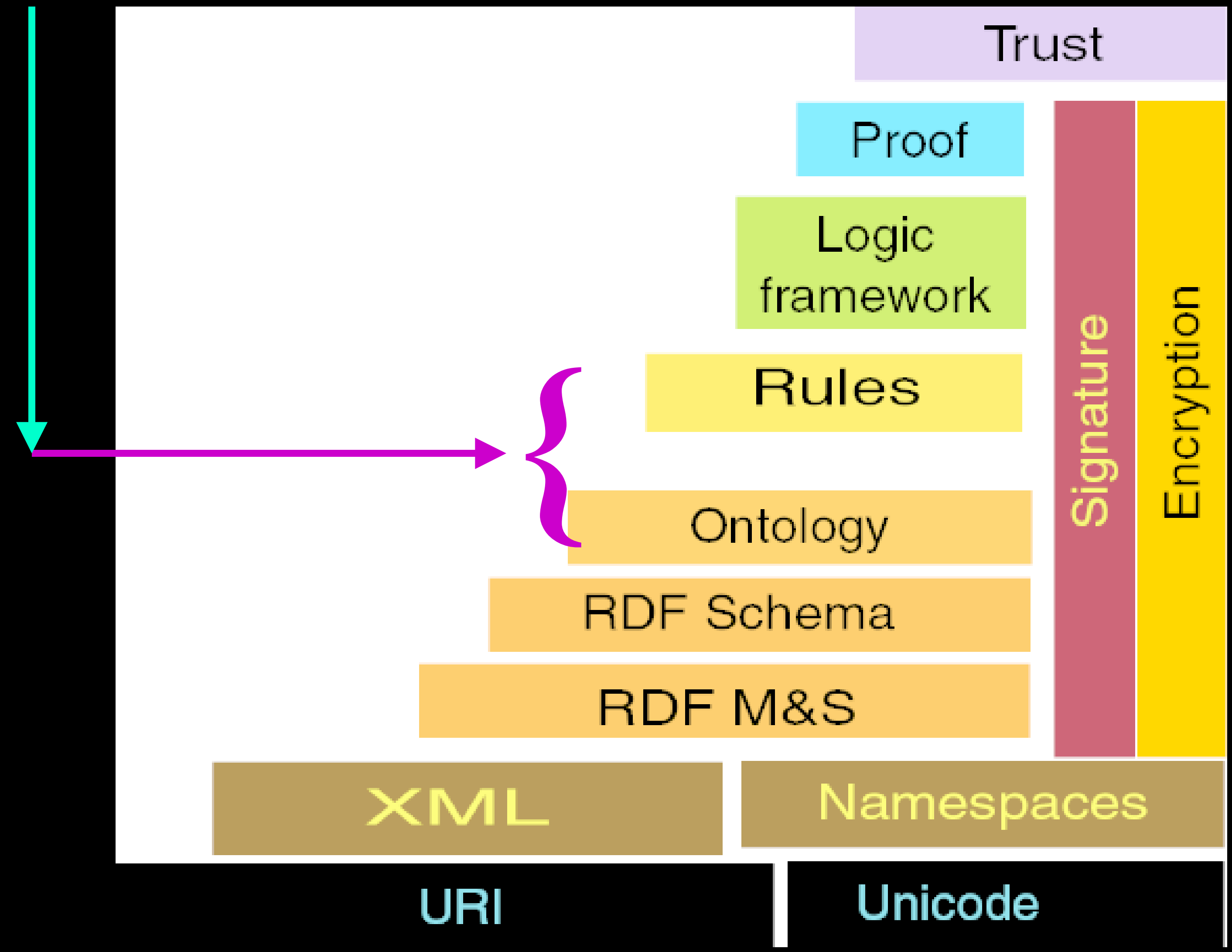
bgrosof@mit.edu   http://www.mit.edu/~bgrosof/

What is "DAML Rules"?

- Answer 1 (general):
  - new stuff about rules that relates specifically to the DAML program, including to DAML+OIL, DAML-Services, and their application scenarios

- Answer 2 (narrower):
  - the hybridization of DAML+OIL with Logic Program rules
    - original aim: extend expressiveness of DAML KR beyond DAML+OIL
      - for defining ontologies, and for rules plus ontologies
    - current thrust focuses on *Description Logic Programs* as KR

Motivation from Semantic Web "Stack"



[Diagram http://www.w3.org/DesignIssues/diagrams/sw-stack-2002.png is courtesy Tim Berners-Lee]

Outline: Rules wrt DAML+OIL, DAML-Services

- Description Logic Programs (DLP)
- Rule-based Semantic Web Services (RSWS)
- Application Scenarios
- Other misc. on Rules and DAML
- RuleML update (brief)

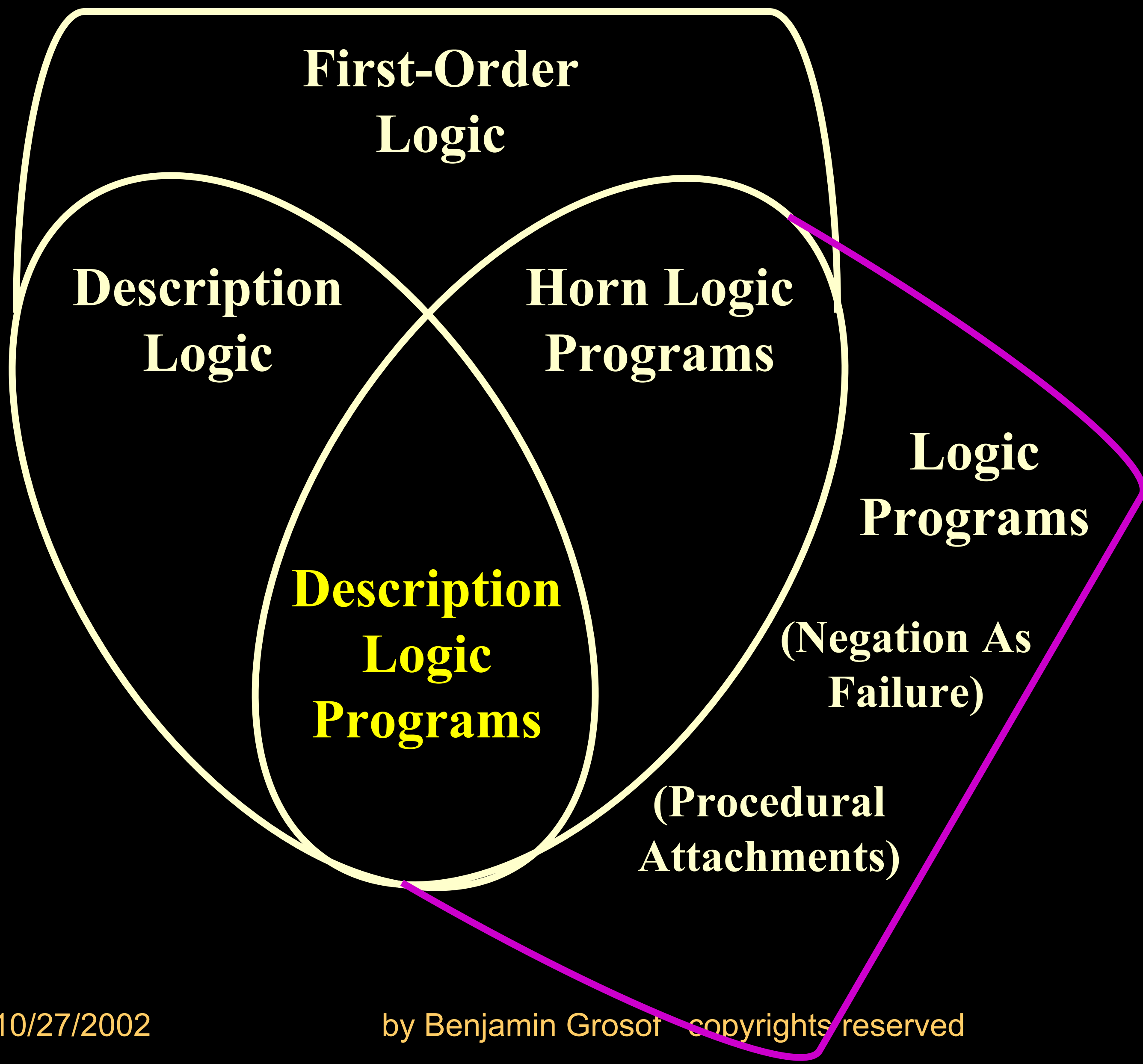
Description Logic Programs (DLP)

- Status: [Grosof & Horrocks 10/02] working paper, Joint Committee discussions, including early use cases.

- Goal: understand relationship between DL and LP/HornFOL as KR's
  - Insight: expressive intersection is also
    a key to expressive combination/union

- 1st step: expressive intersection of DL and Logic Programs
  = "Description Logic Programs"
  (or "Description Rules")

Venn Diagram: Expressive Overlaps among KR's



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.

Candidate: First Order Logic

- FOL has practical and expressive drawbacks for <u>union</u> of DL and Rules:
  - Intractable
  - Lacks non-monotonicity and procedural attachments
  - Unfamiliar to mainstream software engineers

- Variant of DLP: “Horn Description Logic (HDL)”
  - Intersection of Horn Logic and Description Logic
  - Subset of FOL

- (general concept of “Description Rules”:   covers DLP or HDL)

Overview of DLP Features

- Essentially, DLP captures RDFS subset of DL -- plus a bit more.
- RDFS subset of DL permits the following statements:
  - Class C is Subclass of class D.
  - Domain of property P is class C.
  - Range restriction on property P is class D.
  - Property P is Subproperty of property Q.
  - a is an instance of class C.
  - (a,b) is an instance of property P.
- DLP also captures:
  - Using the Intersection connective (conjunction) in class descriptions
  - Stating that a property P is Transitive.
  - Stating that a property P is Symmetric.
- DLP can *partially* capture: most other DL features.
- Relevant technical issue in LP:
  - treatment of equality, e.g., uniqueness of names.

Examples of DL beyond DLP

- DLP is a *strict* subset of DL.
- Examples of DL that is not (completely) representable in DLP:
  - State a subclass of a complex class expression which is a disjunction. E.g.,
    - (Human $\cap$ Adult) $\subseteq$ (Man $\cup$ Woman)
  - State a subclass of a complex class expression which is an existential. E.g.,
    - Radio $\subseteq \exists$ hasSpeaker.Tuner
- Why not? Because: LP/Horn, and thus DLP, cannot represent a “disjunction in the head”.

Examples of LP beyond DLP

- DLP is a *strict* subset of Horn LP.
- Examples of Horn LP that are not (completely) representable in DLP:
  - A rule involving multiple variables. E.g.,
    - PotentialLoveInterestBetween(?X,?Y)

      ← Man(?X) /\ Woman(?Y).
  - Chaining (besides simple transitivity) to derive values of Properties. E.g.,
    - InvolvedIn(?Company, ?Industry)

      ← Subsidiary(?Company, ?Unit)

      /\ AreaOf(?Unit, ?Industry).
- Why not? Essentially because: Decidability of DLs crucially dependent on tree model property.
  - Intuition: DL's not used to represent "more than one free variable at a time".

Benefits: What DLP Enables, in Principle

- LP rules "on top of" DL ontologies.
- Translation of LP rules to/from DL ontologies.
- Use of efficient LP rule/DBMS engines for DL fragment.
- Development of ontologies in LP.
- Development of rules in DL.
- Translation of LP conclusions to DL.
- Translation of DL conclusions to LP.

DL Task Scenarios / Use Cases -- how well do they map to Rules?

- 1. Infer Categorization
  - Rules appear to often handle this well.

- 2. Infer Subsumptions
  - Rules appear to often be more awkward.

- 3. Configuration: seems to involve both categorization and subsumption.

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)

Related Work to DLP

- CARIN [Halevy et al, late 90's] on extending DL with some aspects of LP. For DL-ish tasks.
- [Antoniou 2002] on Defeasible Logic rules + Description Logic (variant) ontologies

Outline:
Rules wrt DAML+OIL, DAML-Services

- Description Logic Programs (DLP)
- Rule-based Semantic Web Services (RSWS)
- Application Scenarios
- Other misc. on Rules and DAML
- RuleML update (brief)

Rule-based Semantic Web Services

- Rules/LP in appropriate combination with DL as KR, for RSWS
  - DL good for <u>categorizing</u>: a service overall, its inputs, its outputs
- Rules to describe <u>service process models</u>
  - rules good for representing:
    - <u>preconditions</u> and <u>postconditions</u>, their contingent relationships
    - <u>contingent</u> behavior/features of the service more generally,
      - e.g., exceptions/problems
  - familiarity and naturalness of rules to software/knowledge engineers
- Rules to specify <u>deals about services</u>: 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 e-contracts:
  - LP rules about agent contracts with exception handling
  - … on top of DL ontologies about business processes;
  - *a scenario motivating DLP*
- Other:
  - Trust management / authorization (Delegation Logic) [Li, Grosof, & Feigenbaum 2000]
  - Financial knowledge integration (ECOIN) [Firat, Madnick, & Grosof 2002]
  - Privacy policies (P3P APPEL)
  - Business policies, more generally

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

Using concise text syntax

(SCLP textfile format)

for concise human reading

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

SCLP TextFile Format for (Daml)RuleML

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

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

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

drm = namespace for damlRuleML

Example Contract Proposal, Continued: lateDeliveryPenalty exception handler module

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

Outline:
Rules wrt DAML+OIL, DAML-Services

- Description Logic Programs (DLP)
- Rule-based Semantic Web Services (RSWS)
- Application Scenarios
- Other misc. on Rules and DAML
- RuleML update (brief)

 by Benjamin Grosof

Other Misc. on Rules and DAML

- DAML+OIL syntax for RuleML: DamlRuleML; implemented in SweetJess
- Inclusion: DAML Includes, XIncludes
- Queries: DAML Query Language (DQL), ...
- Explanations and justifications

RuleML Update

- Overall: more tools, more participants.
- Situated courteous LP (SCLP) as extension of spec.
  - Implemented in SweetRules [Grosof 2001] inferencing and translation.
- DAMLRuleML draft spec.: DAML+OIL spec. for RuleML's syntax.
  - Implemented in SweetJess [Grosof, Gandhe, and Finin 2002].
- SweetJess translator of SCLP RuleML to/from Jess, inferencing via Jess.
  - 1st bridge between Prolog/RDBMS and OPS5/ECA.
- Reactive rules subgroup effort launching.
- Applications:
  - Configurable reusable e-contracts (SweetDeal).
  - Ontology-based financial knowledge integration (ECOIN).
- Oasis interest in "Policy RuleML" (tentative name) as possible TC.
  - RuleML for interchange between policy languages.
- Plan to engage on W3C front, as well.
- Events aimed for in 2003: W3C Plenary, WWW Conf., ISWC.

Other Issues in Rules

- Relationship to XQuery, RDF Query
- *(Open discussion….)*

Bibliography

- Antoniou, Grigoris. "A Nonmonotonic Rule System using Ontologies". Proc. Intl. Wksh. On Rule Markup Languages for Business Rules on the Semantic Web, held 6/02 at the 1st Intl. Semantic Web Conf. (ISWC-2002).
- Firat, Aykut and Madnick, Stuart, and Grosof, Benjamin. "Knowledge Integration to Overcome Ontological Heterogeneity: Challenges from Financial Information Systems". Proc. Intl. Conf. on Information Systems (ICIS), 12/02.
- Firat, Aykut and Madnick, Stuart, and Grosof, Benjamin. "Financial Information Integration in the Presence of Equational Ontological Conflicts". Proc. Wksh. on Information Technologies and Systems (WITS-02), held 12/02 at the Intl. Conf. on Information Systems (ICIS). *Describes ECOIN system.*
- Grosof, Benjamin. "Representing E-Business Rules for the Semantic Web: Situated Courteous Logic Programs in RuleML". Proc. Wksh on Information Technologies and Systems (WITS-01), held 2001 at the Intl. Conf. on Information Systems (ICIS). *Describes SweetRules tool as well as RuleML.*
- Grosof, Benjamin and Horrocks, Ian. "Description Logic Programs: Combining Logic Programs with Description Logic". Working Paper. Draft of 10/17/2002 is at: http://www.cs.mac.ac.uk/~horrocks/WebOnt/wp7-brief.pdf .

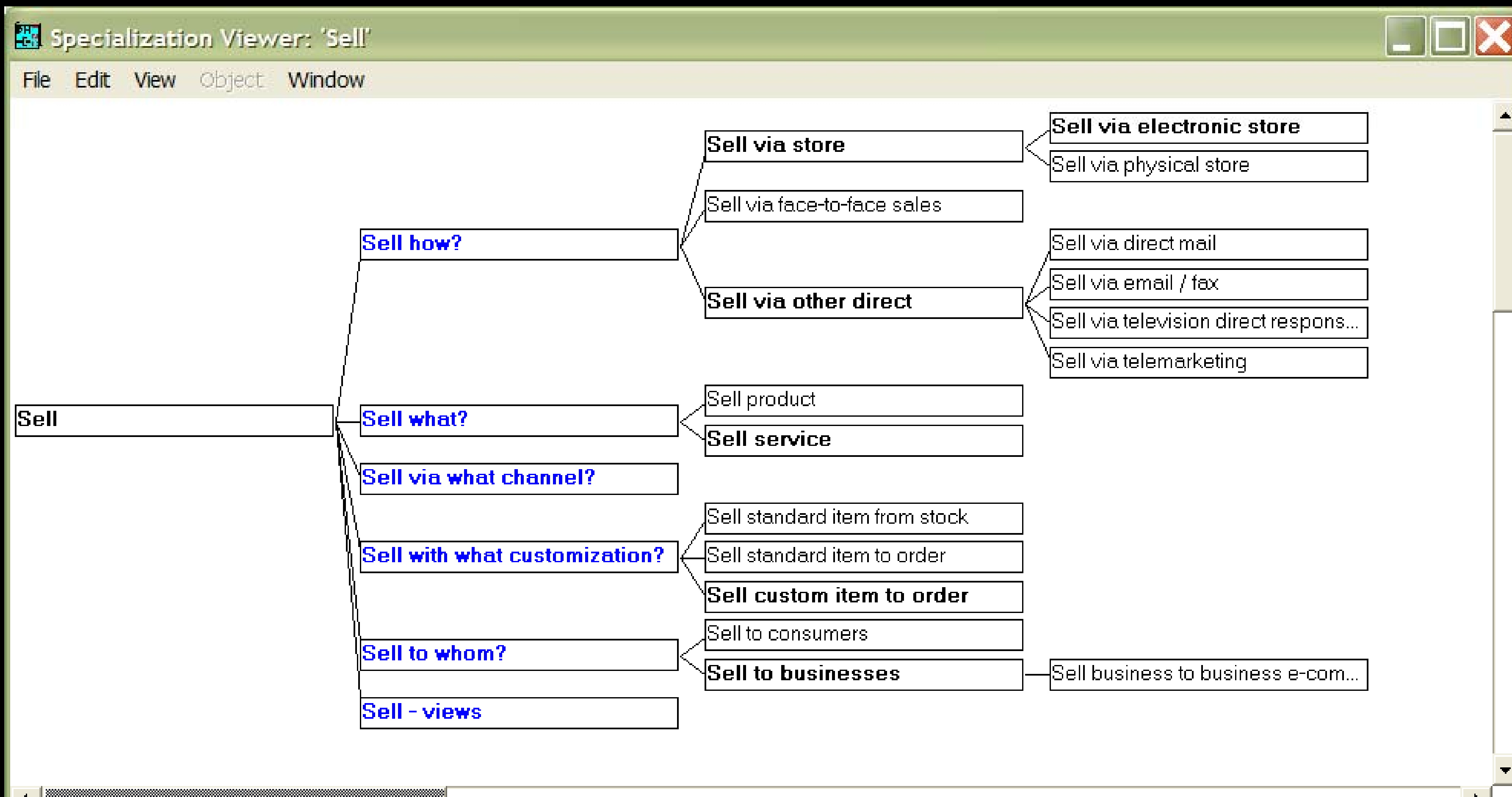
Bibliography, continued

- Grosof, Benjamin and Poon, Terrence. "Representing Agent Contracts with Exceptions using XML Rules, Ontologies, and Process Descriptions". Proc. Intl. Wksh. on Rule Markup Languages for Business Rules on the Semantic Web, held 6/02 at the 1st Intl. Semantic Web Conf. (ISWC-2002). *Describes SweetDeal.*
- Grosof, Benjamin and Gandhe, Mahesh and Finin, Tim. "SweetJess: Translating DamlRuleML to Jess". Proc. Intl. Wksh. on Rule Markup Languages for Business Rules on the Semantic Web, held 6/02 at the 1st Intl. Semantic Web Conf. (ISWC-2002).
- Levy, Alon and Rousset, Marie-Christine. "CARIN: A Representation combining Horn Rules and Description Logic." *Artificial Intelligence* 104(1-2), 1998. (Note: Alon has since changed his surname to "Halevy".)
- Li, Ninghui and Grosof, Benjamin and Feigenbaum, Joan. "Delegation Logic: A Logic-based Approach to Distributed Authorization". Forthcoming, ACM Trans. on Information Systems Security (TISSEC) journal.

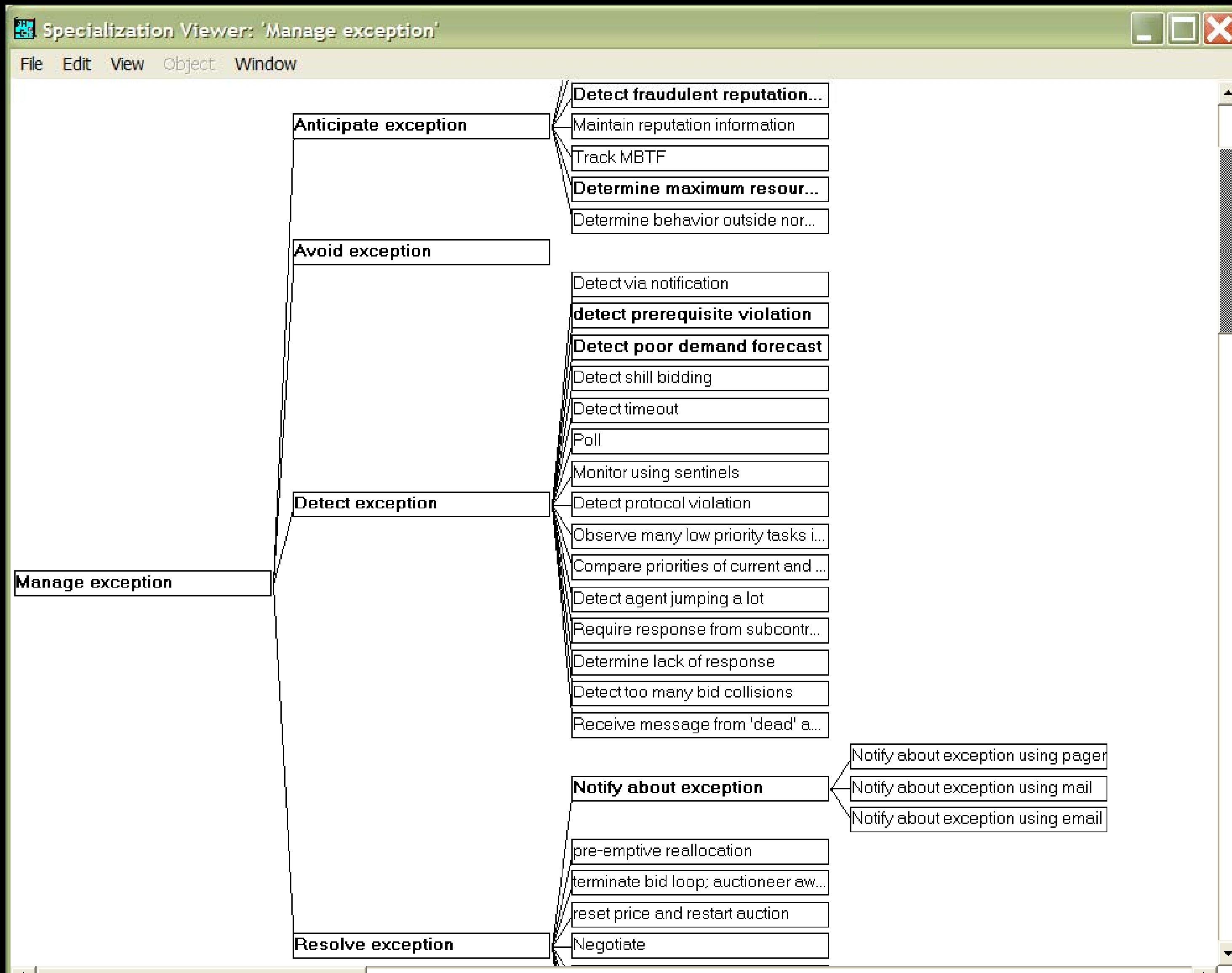
- Thanks!
- Questions?
- Comments? Pointers?
- For More Info:
  - http://www.mit.edu/~bgrosof/

OPTIONAL SLIDES FOLLOW

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



Some exception handlers in the MIT Process Handbook



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) ) )
```

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.

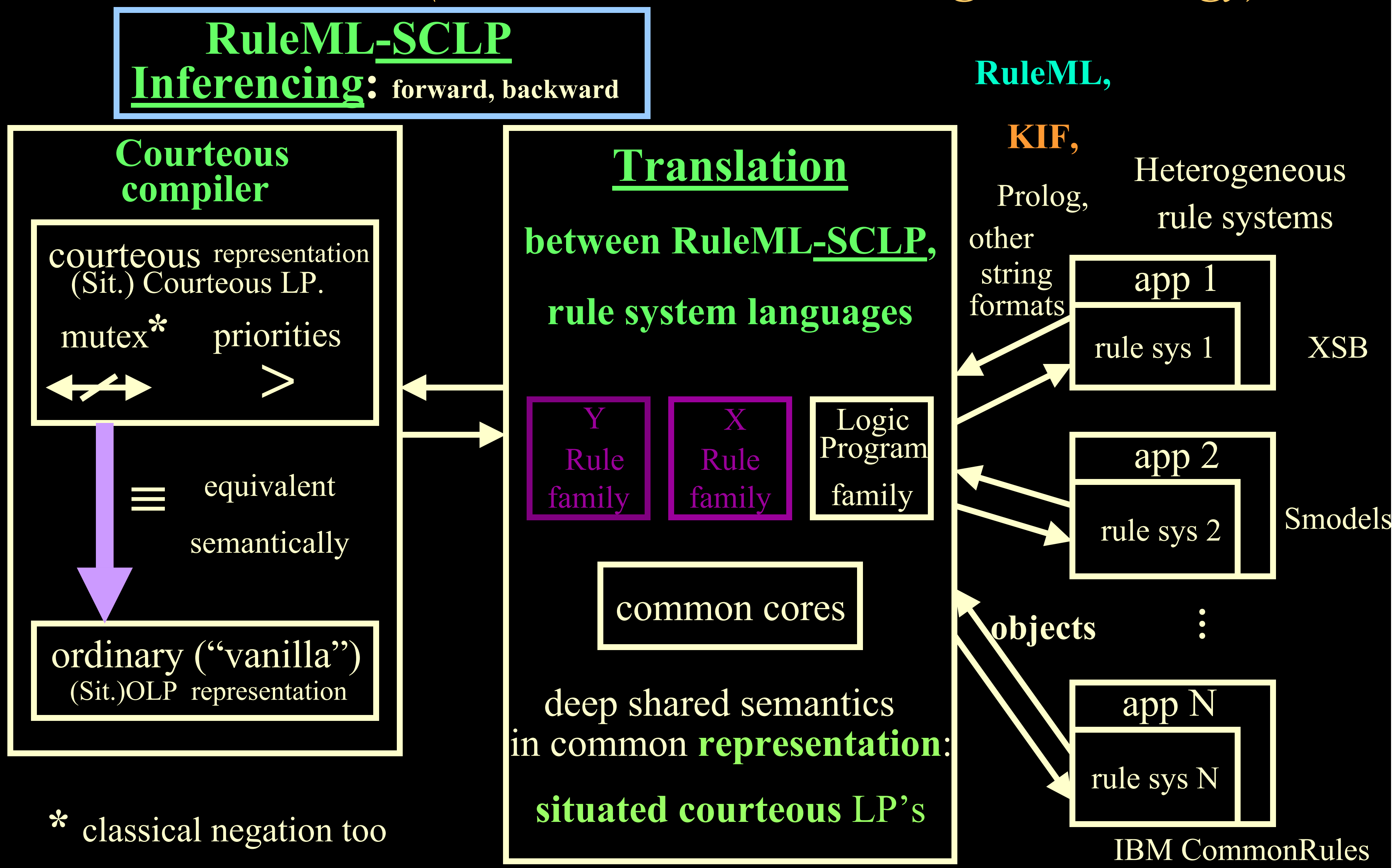
jproc = Java attached procedure.

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

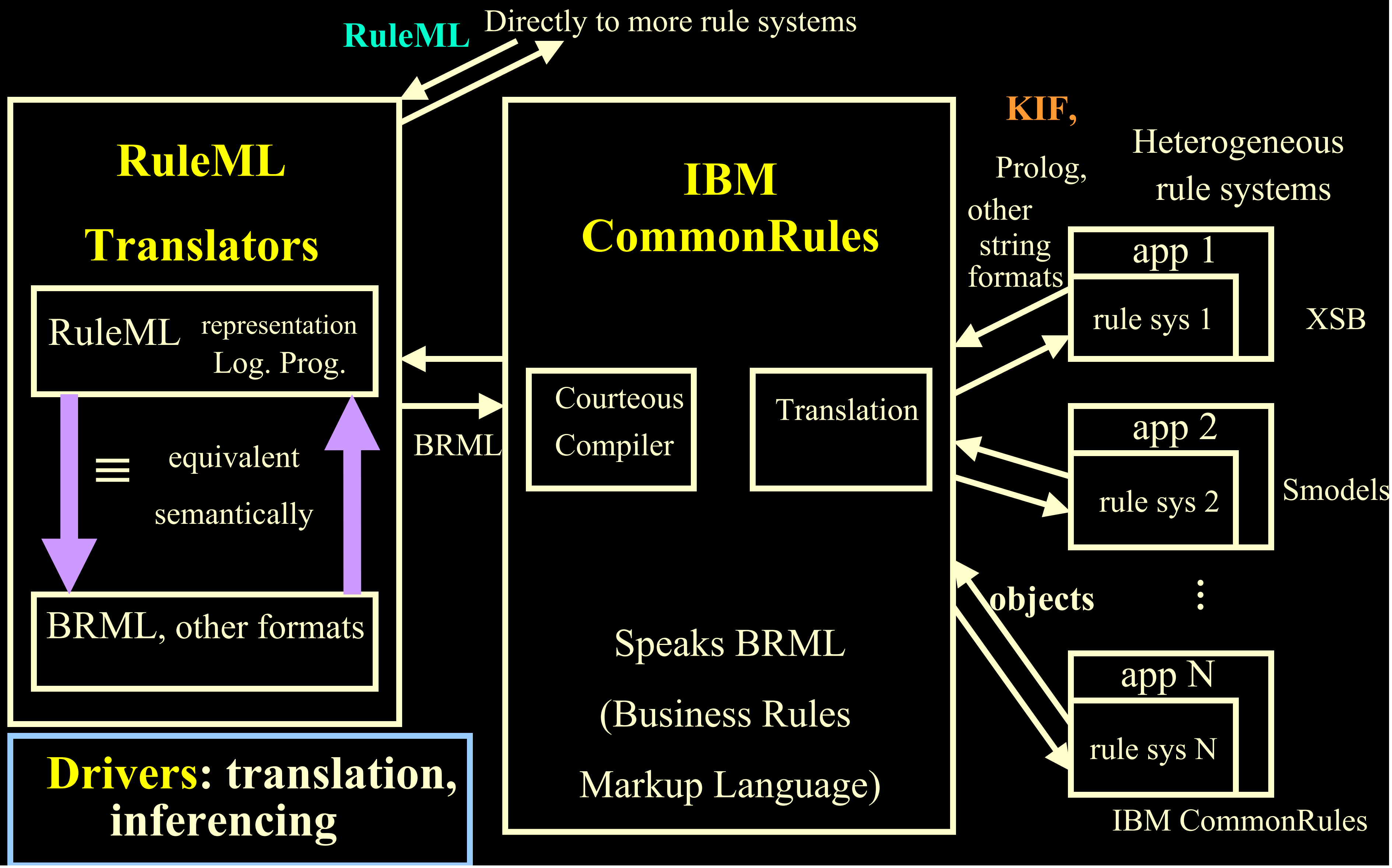
Equivalent in JESS: key portion is:

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

Functionality: SWEETRules Prototype
(Semantic WEb Enabling Technology)



Dec.-2001 Architecture: SWEETRules Prototype
(Semantic WEb Enabling Technology)



Speaker Bio

- Benjamin Grosof is Douglas Drane Assistant Professor in Information Technology at MIT Sloan School of Management. His research is to create and study knowledge-based information technology for e-commerc eapplications, including high-level business/agent communication,information integration, contracts/negotiation, trust, product descriptions, business rules/policies, Web services, and e-marketplaces. The pioneer of inter-operable XML business rules and of their application to contracting, he co-leads the RuleML emerging industry standards effort on inter-operable XML/RDF rules. He is PI currently for a project in the DARPA Agent Markup Language (DAML) initiative, to create Semantic Web technology and explore its business applications.

- Previously, he was a senior research scientist at IBM T.J. Watson Research Center (12 years there), where most recently he conceived and led IBM CommonRules (V3.0 currently on IBM alphaWorks) and co-led its application piloting for rule-based XML agent contracting in EECOMS, a $29Million NIST industry consortium project on manufacturing supply chain management. His notable technical contributions also include fundamental advances in rule-based security authorization, conflict handling for rules, rule-based intelligent agents, and integration of rules with machine learning. He is author of over 30 refereed publications, two major software releases, and a patent. His recent service includes co-chairing the AAAI (National Conference on Artificial Intelligence) Workshops on AI in E-Commerce (1999) and Knowledge-Based E-Markets (2000). His background includes 2 years in software startups, PhD in Computer Science (specialty AI) from Stanford University, and BA in Applied Mathematics from Harvard University.