SweetRules:
Tools for RuleML Inferencing and Translation

Benjamin Grosof
MIT Sloan School of Management
Information Technologies group
http://ebusiness.mit.edu/bgrosof

Slides of invited presentation at Developers Day, Rules on the Web track,
at 13th International Conference on the World Wide Web

5/22/2004 Copyright 2004 by Benjamin Grosof. All Rights Reserved
Outline

• Concept, Architecture, and Goals
• Rule and Ontology Languages/Systems involved
• Capabilities and Components Today
• More about Combining Rules with Ontologies
• Application Scenarios and Examples
• Status and Plans
• Motivations, revisited: Conclusions and Directions
• Acknowledgements
• Resources
Context and Players

• Part of SWEET = “Semantic WEb Enabling Tools” (2001 – )
  – Other parts:
    • SweetDeal for e-contracting
      – Which uses SweetRules

• Cross-institutional. Collaborators invited!
  – Originated and coordinated by MIT since 2001
  – Code by MIT, UMBC, U. Karlsruhe, U. Zurich
  – Uses code by IBM, SUNY Stonybrook, Sandia Natl. Labs, Helsinki
  – More loosely, several other institutions cooperating: BBN, NRC/UNB, Stanford
  – Many more are good targets: subsets of Flora, cwm, Triple, Hoolet, Jena, DRS, ROWL, KAON (main), JTP, SWI Prolog, ...
Concept, Architecture, and Goals

• Concept and Architecture: Tools suite for Rules and RuleML
  – Translation and interoperability between heterogeneous rule systems (forward- and backward-chaining) and their rule languages/representations
  – Inferencing including via translation between rule systems
  – Authoring and testing of rulebases
  – Open, lightweight, extensible, pluggable architecture overall

• Goals:
  – Research vehicle: embody ideas, implement application scenarios (e.g., contracting, policies)
    • Situated Courteous Logic Programs (SCLP) KR
    • Description Logic Programs (DLP) KR which is a subset of SCLP KR
  – Proof of concept for feasibility, including of translations between heterogenous families of rule systems
    • Encourage others: researchers; industry esp. vendors
SweetRules Overview

Key Ideas:

– Unite the commercially most important kinds of rule and ontology languages via a new, common knowledge representation (SCLP) in a new standardized syntax (RuleML), including to cope with heterogeneity and resolve contradictory conflicts.
  • Capture most of the useful expressiveness, interoperably and scalably.
– Combine a large distributed set of rule and ontology knowledge bases that each are active: each has a different associated engine for reasoning capabilities (inferencing, authoring, and/or translation).
– Based on recent fundamental KR theory advances, esp. Situated Courteous Logic Programs (SCLP) and Description Logic Programs.
  • Plus semantics-preserving translations between different rule languages/systems/families

Application Areas (prototyped scenarios):

– Policies and authorizations; contracting, supply chain management; retailing, customer relationship management; business process automation and e-services; financial reporting and information; etc.
RuleML KR Expressiveness

- SweetRules supports: RuleML in its highly expressive Situated Courteous Logic Programs (SCLP) extension, V0.8
  - Horn LP …
  - + Negation-As-Failure = “Ordinary” LP (OLP)
  - + Courteous feature: prioritized conflict handling (partially ordered priorities, mutual exclusion integrity constraints, e.g., for partial-functionality; limited classical negation of atoms, e.g., p vs. not-p in heads)
  - + Situated feature: procedural attachments
    - Sensors: external queries when rule body atoms are tested
      - Built-ins in SWRL V0.6 correspond to sensors.
    - Effectors: external actions triggered when rule head atoms are concluded
- RuleML also supports referencing OWL/DAML+OIL ontologies
  - URI predicate name (in RuleML rule) refers to class or property (in OWL axioms)
    - This was pioneered in SweetDeal using SweetRules
    - The same approach was then taken in SWRL V0.5+
Rule and Ontology Languages/Systems That Interoperate via SweetRules and RuleML, Today

1. RuleML
   - SCLP extension, V0.8

2. XSB (the pure subset of it = whole Ordinary LP)

3. Jess (a pure subset of it = a large subset of Situated Ordinary LP)
     - Uses recent novel theory for translation between SOLP and Production Rules.

4. IBM CommonRules (whole = large subset of stratified SCLP)
   - Implements the Courteous Compiler (CC) KR technique.
     - which reduces (S)CLP to equivalent (S)OLP, tractably.
   - Includes bidirectional translators for XSB, KIF, Smodels.
   - Its overall concept and design was point of departure for several aspects of SweetRules
5. Knowledge Interchange Format (KIF) (a subset of it = an extension of Horn LP)
   - First Order Logic (FOL). Semi-standard, morphing into Simple Common Logic ISO standard. Several tools support, e.g., JTP. Research language to date.
     • Note: FOL is superset of DLP and of SWRL’s fundamental KR.

6. OWL (the Description Logic Programs subset)
   - Description Logic ontologies. W3C standard. Several tools support, e.g., FACT, RACER, Jena, Hoolet, etc.
   - Uses recent novel DLP theory for translation between Description Logic and Horn LP.

7. Process Handbook (large subset = subset of SCLP)
     • Uses recent novel SCLP representation of Frames with multiple default inheritance.

8. Smodels (NB: somewhat old version; large subset = finite OLP)
Capabilities and Components Today

- **Translators** in and out of RuleML:
  - RuleML ↔ {XSB, Jess, CommonRules, KIF, Smodels}
  - RuleML ← {OWL, Process Handbook} (one-direction only)
  - SOLP RuleML ← SCLP RuleML (Courteous Compiler)

- **Inferencing engines** in RuleML via translation:
  - Simple drivers translate to another rule system, e.g., CommonRules, Jess, or XSB, then run inferencing in that system’s engine, then translate back.
  - Observation: Can easily combine components to do other kinds of inferencing, in similar indirect style, by combining various translations and engines.

- **Authoring and Testing** front-end: currently rudimentary, partial
  - Command-line UI + Dashboard GUI with set of windows
  - Edit in RuleML. Edit in other rule systems’ syntaxes. Compare.
  - View human-oriented presentation syntax. View XML syntax. (Future: RDF.)
Capabilities and Components Today, cont. ’d

- **Uses Courteous Compiler** to support Courteous feature (prioritized conflict handling) even in systems that don’t directly support it, as long as they support negation-as-failure
  - E.g., XSB Prolog, Jess, Smodels
  - Uses Courteous Compiler component from IBM CommonRules
- **Uses IBM CommonRules translators:** CommonRules ↔ {XSB, KIF, Smodels}
- **Some components have distinct names** (for packaging or historical reasons):
  - **SweetJess** translation and inferencing RuleML ↔ Jess
    - Available upon request free for research use as download.
  - **SweetOnto** translation RuleML ↔ OWL
    - Available currently as part of KAON open-source code base, called “DLP” component there
- **Code base:** Java, XSLT, shell scripts (for testing drivers)
More about Combining Rules with Ontologies

There are several ways to use SweetRules to combine rules with ontologies:

1. **By reference**: via URI as name for predicate
2. **Translate DLP subset of OWL into RuleML**
   - Then can add SCLP rules
     - E.g., add Horn LP rules and built-in sensors
     - \( \Rightarrow \) interesting subset of the SWRL V0.6 KR
     - E.g., add default rules or procedural attachments
3. **Translate non-OWL ontologies into RuleML**
   - E.g., object-oriented style with default inheritance
     - E.g., Courteous Inheritance for Process Handbook ontologies
4. **Use RuleML Rules to map between ontologies**
   - E.g., in the spirit of the Extended COntext Interchange (ECOIN) approach/system.
   - SWRL V0.6 good start for mapping between non-DLP OWL ontologies.
Venn Diagram: Expressive Overlaps among KR’s

First-Order Logic

Description Logic

Horn Logic Programs

Logic Programs

Description Logic Programs

(Negation As Failure)

(Procedural Attachments)
Translating a Fact from RuleML to Jess

<damlRuleML:fact>
  <damlRuleML:_rlab>fact8962</damlRuleML:_rlab>
  <damlRuleML:_head>
    <damlRuleML:atom>
      <damlRuleML:_opr>
        <damlRuleML:rel>shopper</damlRuleML:rel>
      </damlRuleML:_opr>
      <damlRuleML:ind>Debbie</damlRuleML:ind>
    </damlRuleML:atom>
  </damlRuleML:_head>
</damlRuleML:fact>

equivalent in JESS:
  (assert (shopper Debbie) )
Translating a Rule from RuleML to Jess

```xml
<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>
</damlRuleML:imp>
```
Continued: Translating a **Rule** from RuleML to Jess

```xml
<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>

Equivalent in JESS:
(defrule steadySpender
  (shopper ?Cust)
  (spendingHistory ?Cust loyal)
  =>
  (assert (giveDiscount percent5 ?Cust) ) )
```
Some New Research Application Scenarios for Rule-based Semantic Web Services

- **SweetDeal** [Grosof & Poon WWW-2003] configurable reusable e-contracts:
  - Represents modular modification of proposals, service provisions
  - LP rules as KR. E.g., prices, late delivery exception handling.
  - On top of DL ontologies about business processes from MIT Process Handbook
  - Evolved from EECOMS pilot on agent-based manufacturing SCM
    ($51M NIST ATP 1996-2000 IBM, Boeing, TRW, Vitria, others)

- **Financial** knowledge integration (ECOIN) [Firat, Madnick, & Grosof 2002]
  - Maps between contexts using LP rules, equational ontologies, SQL DB’s.

- **Business Policies:**
  - Trust management (Delegation Logic) [Li, Grosof, & Feigenbaum 2003]:
Status and Plans

• Downloadable currently:
  – SweetJess
  – SweetOnto  = KAON’s DLP component
• Rest of suite is currently being updated to support:
  – latest versions of RuleML and CommonRules
  – hosting on open-source site (plan: on semwebcentral.org)
• Recent project emphasis overall on papers and specifications.

• Planned: part of new phase of DAML Rules tools effort
  – Generally polish, integrate
  – Collaborators: Said Tabet, RuleML; Mike Dean, BBN; Mark Musen, Stanford; Harold Boley, NRC/UNB
**Status and Plans, continued**

**Additional Goals:**
- More meat to pluggable architecture
- More authoring/UI capabilities
- More SWRL support
- More wrt additional kinds of rule systems:
  - **ECA rules, SQL** (needs some theory work, e.g., events for ECA)
  - RDF-Query and XQuery
- More wrt connections-to / support-of web services:
  - Importing knowledge bases / modules, procedural attachments, translation/inferencing, events, …
- Explore applications in services, e.g., policies, contracts

**More Collaborators Invited!**
- Many more rule/ontology systems are good targets for interoperation/translation:
  - Flora, cwm, Triple, Hoolet, Jena, DRS, ROWL, KAON, JTP, SWI Prolog, …
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.,
• 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.)
Summary of Objectives Motivating SweetRules: Integrating Distributed Rules and Ontologies

Address “the 5 D’s” of real-world reasoning ⇒ desired improvements:

1. **Diversity** – Existing/emerging kinds of ontologies and rules have heterogeneous KR's. *Handle more heterogeneous systems.*

2. **Distributedness** - of ownership/control of ontology/rule active KB's. *Handle more source active KB’s.*

3. **Disagreement** - Conflict (contradiction) will arise when merging knowledge. *Handle more conflicts.*

4. **Dynamism** - Updates to knowledge occur frequently, overturning previous beliefs. *Handle higher rate of revisions.*

5. **Delay** - Computational scaleability is vital to achieve the promise of knowledge integration. *Achieve Polynomial-time ( ~ databases).*
Contradictory conflict is contained locally, indeed tamed to aid modularity. 

Contradictory conflict is globally contagious, invalidates all results. 

Knowledge integration tackling the 5 D’s (esp. diversity and distributedness) is labor-intensive, slow, costly. 

Knowledge integration is highly automated, faster, cheaper.
Acknowledgements

• Many people have contributed to SweetRules
  – Can’t mention everyone here
• Thanks especially to:
  – Hoi Chan, IBM (CommonRules lead implementer)
  – Terrence Poon, MIT (translators implementer)
  – Mahesh Gandhe, UMBC ⇒ IBM (SweetJess lead implementer)
  – Timothy Finin, UMBC (SweetJess)
  – Abraham Bernstein, U. Zurich (Process Handbook)
  – Boris Motik, U. Karlsruhe (SweetOnto lead implementer)
  – Earl J. Wagner, MIT (authoring lead implementer)
  – Ian Horrocks, U. Manchester (DLP theory)
  – Raphael Volz, U. Karlsruhe (DLP, SweetOnto)
  – M. Youssef Kabbaj, MIT (translators implementer)
Resources

• See papers, talk slides, and links at http://ebusiness.mit.edu/bgrosof
• ../#RecentSoftware : Links to SweetJess, SweetOnto, CommonRules (where can download)
• ../#RecentPapersByTopic : (for most below, there are earlier versions too)
  – "Description Logic Programs: Combining Logic Programs with Description Logic", WWW-2003.

• RuleML http://www.ruleml.org
• DAML Rules http://www.daml.org/rules
• Joint Committee http://www.daml.org/committee
• SemWebCentral http://www.semwebcentral.org

5/22/2004 Copyright 2004 by Benjamin Grosof. All Rights Reserved
Outline

• Concept, Architecture, and Goals
• Rule and Ontology Languages/Systems involved
• Capabilities and Components Today
• More about Combining Rules with Ontologies
• Application Scenarios and Examples
• Status and Plans
• Motivations, revisited: Conclusions and Directions
• Acknowledgements
• Resources
OPTIONAL SLIDES FOLLOW
Quickie Bio of Presenter

- MIT Sloan professor since 2000
- 12 years at IBM T.J. Watson Research; 2 years at startups
- PhD Comp Sci, Stanford; BA Applied Math Econ/Mgmt, Harvard
- Semantic web services is main research area:
  - Rules as core technology
  - Business Applications, Implications, Strategy:
    - e-contracting/supply-chain; finance; trust; …
  - Overall knowledge representation, e-commerce, intelligent agents
- Co-Founder, Rule Markup Language Initiative — the leading emerging standards body in semantic web rules (http://www.ruleml.org)
- Core participant in Semantic Web Services Initiative — which coordinates world-wide SWS research and early standards (http://www.swsi.org)
  - Area Editor for Contracts & Negotiation, Language Committee
  - Co-Chair, Industrial Partners program (SWSIP)
3 Areas of New Fundamental KR Theory that enable Key Technical Requirements for SWS

• 1. Description Logic Programs: [Grosof, Horrocks, Decker, & Volz WWW-2003]  
KR to combine LP (RuleML) rules on top of DL (OWL) ontologies, with:  
  – Power in inferencing (including for consistency)  
  – Scaleability of inferencing

• 2. Situated Logic Programs: [Grosof et al 1995; Grosof et al. 2002; Grosof ECRA 2004]  
KR to hook rules (with ontologies) up to (web) services  
  – Rules use services, e.g., to query, message, act with side-effects  
  – Rules constitute services executably, e.g., workflow-y business processes

• 3. Courteous Logic Programs: [Grosof ILPS-97; Grosof, Labrou, & Chan EC-99]  
KR to combine rules from many sources, with:  
  – Prioritized conflict handling to enable consistency, modularity; scaleably  
  – Interoperable syntax and semantics  
  – Well represents default inheritance in process ontologies (courteous inheritance)

• RuleML includes support for (1.)-(3.).
W3C Semantic Web “Stack”: Standardization Steps

Emerging Standards pioneered in DARPA Agent Markup Language (DAML) program:

- RuleML
- OWL

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

5/22/2004 Copyright 2004 by Benjamin Grososf. All Rights Reserved
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.
SweetDeal Approach: Rule-based Contracts for E-commerce

[Grosof, Labrou, & Chan EC-99; Wellman, Reeves, & Grosof CI ‘02; Grosof & Poon IJEC ’04]

• Rules as way to specify (part of) business processes, policies, products: as (part of) contract terms.
  – Combined with ontologies.
• Complete or partial contract.
  – As default rules. Update, e.g., in negotiation. Exceptions handling.
• 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.
• SWEET = Semantic WEb Enabling Technology
  – software components, theory, approach
  – pilot application scenarios, incl. contracting (SweetDeal)
Examples of Contract Provisions
Well-Represented by Rules in Automated Deal Making

- Product descriptions
  - Product catalogs: properties, conditional on other properties.
- Pricing dependent upon: delivery-date, quantity, group memberships, umbrella contract provisions
- Terms & conditions: refund/cancellation timelines/deposits, lateness/quality penalties, ordering lead time, shipping, creditworthiness, biz-partner qualification, service provisions
- Trust
  - Creditworthiness, authorization, required signatures
- Buyer Requirements (RFQ, RFP) wrt the above
- Seller Capabilities (Sourcing, Qualification) wrt the above
Where Rules Shine in Goals wrt Key SWS Tasks

- Knowledge reuse in knowledge-based service descriptions:
  - … Across the Key Tasks in our Requirements:
    - Contracts (proposals, request-for-proposals, selection, negotiation, advertising); Discovery; Enactment, Composition; Monitoring, Problem resolution, Exception handling; Verification
    - Business/Trust/Security/Privacy Policies
    - Semantic Interoperability (mappings, specializations)
    - Underlying: Hypothetical Reasoning
Where are the Holdups?  
... and Challenges for Research

- KR & standards to integrate Rules with Ontologies more expressively
- KR, & later standards, to represent Services descriptions using Rules and Ontologies.
  - A step is our SweetDeal approach; much current work in SWSI.
- KR & strategy to leverage legacy content, e.g., OO service/process ontologies
  - A rich research area. We are doing much current work on that.
    - Preliminary-version approach is available as paper “Beyond Monotonic Inheritance: Towards Semantic Web Process Ontologies” at http://ebusiness.mit.edu/bgrosof

- Procedural process models aspect of SWS, as underlying foundation
  - Messy, many competing conceptual approaches
  - Realm of slow progress; much energy in WS standards efforts:
    - Oasis WSBPEL, W3C WS Choreography