DAML Rules
Report for PI Mtg. May 2004

Benjamin Grosof, co-chair
MIT Sloan School of Management,
http://ebusiness.mit.edu/bgrosof

with Mike Dean, co-chair
BBN Technologies

Presented at DAML PI Mtg., May 25, 2004, New York City
OTHER PRESENTATIONS ON RULES IN TODAY’S SESSIONS

- SWRL V0.5 overview by Peter Patel-Schneider
- SWRL V0.6 overview by Mike Dean
- SWRL Implementation (incl. Hoolet) by Ian Horrocks
- WWW-2004 DevDay Rules Track Overview by Harold Boley
Usage Comments about SWRL V0.6

Benjamin Grosof
MIT Sloan School of Management, http://ebusiness.mit.edu/bgrosof

with Mike Dean, co-chair
BBN Technologies

Presented at DAML PI Mtg., May 25, 2004, New York City
Usage Comments about SWRL V0.6

• Outline:
  – Expressiveness
  – “Warning Label”
  – *Later today:* Implementation strategy
Expressiveness of SWRL (V0.6)

SWRL expressiveness =
1. OWL-DL (i.e., SHOIQ Description Logic (DL) which is an expressive subset of FOL)
2. + Horn FOL rules, with no logical functions, where each predicate may be:
   - OWL named class (thus arity 1)
     - More generally, may use a complex class, but this is expressively inessential – can just replace by a named class and define that named class as equivalent to the complex class.
   - OWL property (thus arity 2)
   - OWL data range (thus arity 1)
     - RDF datatype
     - set of literal values, e.g., \{3\} or \{1,2,3,4,5\} or \{"Fred","Sue"\}
3. + some built-ins (mainly XML-Schema datatypes and operations on them)
   - This is new with V0.6
   - (All have arity 1 or 2.)
   - Plan: the set of built-ins is extensible

- The fundamental KR is an expressive subset of FOL
  - We’ll call it “DH” here. (It doesn’t have a real name yet.)
  - Its expressiveness is equivalent to: DL + function-free Horn.
Venn Diagram: Expressive Overlaps among KR’s

First-Order Logic

Description Logic

Horn Logic Programs

Logic Programs (Negation As Failure)

Description Logic Programs (Procedural Attachments)
“Warning Label”

1. The Theory of DH is Little Explored Territory as a KR.
   • In its full generality, DH is a relatively understudied fragment of FOL.
   • Its worst-case computational complexity is undecidable and is not known to be better than that of full FOL (e.g., for the propositional case).
   • There are not yet efficient algorithms known for inferencing on it “natively” as a KR.

2. To ensure extensibility of SWRL rulebases to include LP features that go beyond Horn expressiveness, restrict the OWL ontologies used within SWRL to be in the DLP subset of OWL-DL. E.g.:
   • If you want to use nonmonotonicity / negation-as-failure / priorities in your rules
   • If you want to use procedural attachments that go beyond the SWRL built-ins
   • E.g., effectors/actions with side effects

5/28/2004 Copyright 2004 by Benjamin Grosof. All Rights Reserved
Venn Diagram: Expressive Overlaps among KR’s

- **First-Order Logic**
  - Description Logic
  - Horn Logic Programs
    - (Negation As Failure)
    - (Procedural Attachments)

DH KR’s rough position. Subsumes DLP, DL, and part of Horn. Subsumed by FOL.
Alternative points in design space:

1. partial LP + full DL  =  SWRL V0.6

versus

2. full LP + partial DL  =  SCLP RuleML V0.8+
   (with DLP OWL2RuleML)

(SCLP = Situated Courteous Logic Programs KR)
DAML
Tools for Rules
Next-Phase Plan

Benjamin Grosof
MIT Sloan School of Management,
http://ebusiness.mit.edu/bgrosof

Presented at DAML PI Mtg., May 25, 2004, New York City
WWW-2004 DevDay last week

• Way cool!
• Lots of tools and use cases now!

• Themes:
  – Mostly LP/RuleML expressible rules
  – Many combine LP or HornFOL rules with OWL ontologies or OO syntax
DAML Rules Plan Overview

- **Vision:** studio for developers, studio for rule authors and users
- **Approach:** Composable Tools Suite supporting RuleML/SWRL
  - inferencing, translation/interoperability, authoring, testing
- **Infrastructure:** SemWebCentral, SWeDE
- **MIT Sloan (Benj. Grosof lead):** SweetRules RuleML tools:
  - translation and inferencing; architecture for suite integration
- **BBN (Mike Dean lead):** SWRL tools:
  - translator to Jena2; editor; validator
- **Stanford (Mark Musen lead):** Protégé support for rule authoring
- **More about Implementing SWRL**
- **Later:** More about FOL
- **Others Very Much Invited!**
  - some good candidates: those presented at WWW-2004 Developers Day Rules on the Web Track
MIT Sloan Plan:

SweetRules:
Tools for RuleML Inferencing and Translation
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
• 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, Hoolet, ROWL, Triple, Jena, DRS, 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.

Distributed Active Knowledge Bases
- heterogeneous rules / ontologies
- with associated inferencing, authoring, translation capabilities

Inferencing + Translation

Authoring + Testing

New Integration Capabilities

Reasoning Capabilities to Support Applications
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

5/28/2004 Copyright 2004 by Benjamin Grosof. All Rights Reserved
Rule and Ontology Languages/Systems That Interoperate via SweetRules and RuleML, Today, continued

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
     - ⇒ 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

Description Logic Programs (Negation As Failure)

Logic Programs (Procedural Attachments)
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]:
SweetRules Tools Available Now

- Available currently:
  - SweetJess
  - SweetOnto = KAON’s DLP component

- Rest of Suite being updated and prepared for release on SemWebCentral
SweetRules Plans

• Update, integrate, and polish suite overall
• Support latest versions of RuleML and CommonRules
• Open source on semwebcentral.org
• Scenarios: Explore applications in SW Services, e.g., trust policies, contracting, monitoring, semantic interoperability mappings
• Requirements analysis
SweetRules Plans, cont.’d

• Pluggable architecture for Rules tools
  – SemWebCentral aspects
  – SWeDE aspects
  – Eclipse wrappers for tools
  – Ontology of tools
  – Composition patterns, high-level interfaces design
SweetRules Plans, cont.’d

• Additional Goals:
  – *Via suite integration:* More interoperability between SWRL and RuleML
  – *Ongoingly:* Update RuleML spec in synch with SWRL spec (in RuleML Initiative, Joint Committee)
  – *Via suite integration:* More authoring/UI capabilities
SweetRules Plans, longer-term

– *Later*: Justifications and proof, e.g., via suite integration with InferenceWeb

– *Later*: More wrt additional kinds of rule systems:
  • **ECA** rules, **SQL** (needs some theory work, e.g., events for ECA)
  • **RDF-Query** and **XQuery**

– *Later*: More wrt connections-to / support-of web services:
  • Importing knowledge bases / modules, procedural attachments, translation/inferencing, events, …
SweetRules Groups/People

- Collaborators: Said Tabet, RuleML; Mike Dean, BBN; Mark Musen, Stanford; Harold Boley, NRC/UNB

- More Collaborators Invited!
  - Many more rule/ontology systems are good targets for interoperation/translation:
    - Flora, cwm, ROWL, Hoolet, Triple, DRS, KAON, JTP, SWI Prolog, …
Resources

• See papers, talk slides, and links at http://ebusiness.mit.edu/bgrossof
• ../#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

5/28/2004  Copyright 2004 by Benjamin Grosos. All Rights Reserved
DAML Rules Plan Overview

- **Vision:** studio for developers, studio for rule authors and users
- **Approach:** Composable Tools Suite supporting RuleML/SWRL
  - inferencing, translation/interoperability, authoring, testing
- **Infrastructure:** SemWebCentral, SWeDE
- **MIT Sloan (Benj. Grosof lead):** SweetRules RuleML tools:
  - translation and inferencing; architecture for suite integration
- **BBN (Mike Dean lead):** SWRL tools:
  - translator to Jena2; editor; validator
- **Stanford (Mark Musen lead):** Protégé support for rule authoring
- **More about Implementing SWRL**
- **Later: More about FOL**
- **Others Very Much Invited!**
  - some good candidates: those presented at WWW-2004
  - Developers Day Rules on the Web Track

5/28/2004 Copyright 2004 by Benjamin Grosof. All Rights Reserved
More about Implementing SWRL
Venn Diagram: Expressive Overlaps among KR’s

- First-Order Logic
- Description Logic
- Horn Logic Programs

DH KR’s rough position. Subsumes DLP, DL, and part of Horn. Subsumed by FOL.
Design Perspective

Alternative points in design space:

1. partial LP + full DL = SWRL V0.6

versus

2. full LP + partial DL = SCLP RuleML V0.8+
   (with DLP OWL2RuleML)

(SCLP = Situated Courteous Logic Programs KR)
More SWRL Implementation Strategy

- **Named-classes-only restriction** on SWRL rules simplifies implementation including translation to rule systems (e.g., RuleML, Jess, XSB), yet does not sacrifice fundamental expressiveness.
  - Both current implementations of SWRL do this.

- Can translate full SWRL / DH $\Rightarrow\Rightarrow$ FOL for which “native” (general-purpose) reasoners are indeed available.
  - E.g., OTTER or Simple Common Logic / KIF
  - The Manchester implementation of SWRL does this.
  - **Drawbacks**:
    - General-purpose FOL reasoners are often not very efficient.
    - Today, they also usually don’t directly support Webized syntax.
More SWRL Implementation Strategy, cont.’d

• Can translate subset of SWRL / DH into a KR for which “native” reasoners are indeed available. E.g.:

1. Horn LP expressible subset $\Rightarrow$ LP, e.g., RuleML, Jess, XSB
   – E.g., Horn LP SWRL rules + DLP OWL ontologies
     • Horn LP restriction on the SWRL rules means that:
       – rules are named-classes-only (no complex class expressions appear)
       – rules are definite (consequent is non-empty); and
       – ground atomic conclusions suffice.
   • The BBN implementation does this (Horn rules $\Rightarrow$ Jess)

2. DL-expressible subset of DH $\Rightarrow$ DL, e.g., OWL
   – E.g., DLP SWRL rules + any OWL-DL
     • E.g., SWRL rules are used to define some ontologies
     • No implementation of this is yet available.
More about SWRL V0.6 Built-Ins

• The built-ins (3.) can be viewed as predicates/relations that have a fixed extension.
  – Alternatively, the set of tuples satisfied by calls to the built-ins can be viewed as corresponding to a virtual fact set adjoined to the FOL theory.

• These are similar to sensors in Situated Logic Programs RuleML.

• The built-ins can be implemented via procedural attachments that are purely informational (free of side effects)
  – Intuitively, they are typically evaluated when rule body is tested.
Punchline on Near-term Implementation Strategy

• (Unless you can invent a whole new technique…)

1. If you want full SWRL expressiveness, translate to some FOL syntax and then use a FOL theorem-prover to do inferencing.

2. If you want to translate to LP to exploit one of the many LP rule engines available (e.g., RuleML, Jess, XSB), or to exploit beyond-Horn LP expressive features (e.g., nonmon or actions), then restrict the SWRL ontologies to DLP.
   • RuleML is the obvious choice of translation target: it’s SWRL’s extension in direction of fuller LP expressiveness, and facilitates translations to multiple other rule languages’ engines (e.g., Jess, XSB).
   • SweetOnto tool (a.k.a. KAON DLP package) translates DLP OWL to RuleML. (There are other DLP implementations too.)
DAML Rules Plan Overview

- **Vision:** studio for developers, studio for rule authors and users
- **Approach:** Composable Tools Suite supporting RuleML/SWRL
  - inferencing, translation/interoperability, authoring, testing
- **Infrastructure:** SemWebCentral, SWeDE
- **MIT Sloan (Benj. Grosof lead):** SweetRules RuleML tools:
  - translation and inferencing; architecture for suite integration
- **BBN (Mike Dean lead):** SWRL tools:
  - translator to Jena2; editor; validator
- **Stanford (Mark Musen lead):** Protégé support for rule authoring
- **More about Implementing SWRL**
- **Later:** More about FOL
- **Others Very Much Invited!**
  - some good candidates: those presented at WWW-2004
    Developers Day  Rules on the Web Track
Getting Involved!

• Please contact Benjamin Grosof and Mike Dean (DAML Rules co-chairs) with your rules …
• Tools
• Ideas
• Rulebases
• Use cases
• Other resources
• Relevant plans
SWSI Rules

Benjamin Grosof
MIT Sloan School of Management,
http://ebusiness.mit.edu/bgrosof

Presented at DAML PI Mtg., May 25, 2004, New York City
**SWSL Plan includes large role for Rules**

- LP Rules together with Ontologies, for “SCAMP” group of tasks:
  - Trust Policies representation, enforcement: Security, privacy, authorization, access control
  - Contracting: contracts, advertising and some matchmaking, proposals, requests for proposals, some negotiation (modification of proposals)
  - Monitoring: exception handling, compliance, problem resolution, compliance

- With Trust policies or Contracts

- LP or FOL Rules together with Ontologies for Semantic Interoperability: data mappings, ontology translation

- LP or FOL Rules together with Ontologies, for Process Models
  - OWL-S Preconditions and Effects
  - PSL-style Process Models
Outbrief from SWSL group

at SWSI F2F

May 24, 2004
Deliverable

Single document covering both:
1. OWL-S Profile + Atomic Process + Grounding, enhanced with Rules
2. Process model with concepts from the core of PSL that replaces the OWL-S (composite) Process model

Target date: September 30, 2004
Target place: W3C (e.g., Member Submission)
The Why and How of Near-term Impact in SWS’s

- Policies in Security/Trust, Contracts, Advertising, Monitoring
  - Combine rules + ontologies in LP
  - Extend OWL-S profile
- Verification of process properties, compatibility; and enactment
  - Combine ordering constraints with pre-conditions/effects as in PSL
  - Extend OWL-S grounded atomic processes
  - Longer term: (semi-)automated composition
SCAMP drill down: Goals of Version 1

- **Key foci**
  - Policy specification and enforcement
    - Trust: policies for security authorization, access, privacy/confidentiality
    - Contracts: pricing, delivery, refunds, cancellations, non-performance, …
      - Contract agreements, proposals, requests for proposals, advertisements
    - Monitoring: task of enforcing policies (e.g., for trust or contracts), policies to handle exceptions & non-compliance (compare results to promises)
    - Borrow from ebXML, EDI, XACML, P3P, LegalXML, …??
  - Start from spirit and particulars of OWL-S Profile
    - Add more particular “service ontologies”
  - Choosing good rule language
    - RuleML with extensions, e.g., ontology import/incorporation (DLP OWL and later OO with default inheritance), HiLog, and F-Logic syntax.
    - Need a surface syntax
  - Framework for negotiation
- **Primary deliverable:** technical document - proposal & rationale
- **Later deliverable:** illustrative application scenario examples
- **Defer:** Complex discovery/matchmaking
SCAMP drill down, cont’d

• Develop upper and middle ontology in selected areas
  – Borrow from ebXML, EDI, XACML, P3P, LegalXML,…??
• Simple advertising/discovery
  – E.g., based on keywords and simple ontology
  – More complex dynamic discovery not focus of version 1
1. Policies for security and monitoring and contracts would meet immediate needs in WS today
   – Want them checked at run time
   – Ensuring compliance with trust policies has become high-priority in many areas of business today:
     • USA: Sarbanes-Oxley (financial reporting liability), HIPAA (patient records privacy)
     • EU: privacy reg’s
   • Yet to a great extent they can be specified and enforced using a relatively simple and mature technology: LP rules.
     – Most trust policy languages / engines today are based on, or equivalent to, rules (+ DLP-expressible ontologies).
     – Ditto for Web standards for trust policies e.g., XACML, P3P both have (prioritized) rules.
More about Game Plan, cont.'d

• Have more in the way of formal coordination with W3C and Oasis etc.
  – Liaison members officially in relevant W3C and Oasis etc. working groups:
    • W3C: WSDL, WS Choreo, SWS Interest Group, WS Policy; P3P, Semantic Web activity incl. www-rdf-rules
    • Oasis: WS Security, XACML, Legal XML, ?ebXML,
    • RuleML; ISO Common Logic
    • ?RosettaNet; ? UN CEFACT EDI / UBL
Policies and Compliance in US Financial Industry Today

- Ubiquitous high-stakes Regulatory Compliance requirements
  - Sarbanes Oxley, SEC, HIPAA, etc.
- Internal company policies about access, confidentiality, transactions
  - For security, risk management, business processes, governance
- Complexities guiding who can do what on certain business data
- Often implemented using rule techniques

- Often misunderstood or poorly implemented leading to vulnerabilities
- Typically embedded redundantly in legacy silo applications, requiring high maintenance
- Policy/Rule engines lack interoperability
# Example Financial Authorization Rules

<table>
<thead>
<tr>
<th>Classification</th>
<th>Application</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Merchant</td>
<td>Purchase Approval</td>
<td>If credit card has fraud reported on it, or is over limit, do not approve.</td>
</tr>
<tr>
<td>Mutual Funds</td>
<td>Rep trading</td>
<td><em>Blue Sky</em>: State restrictions for rep’s customers.</td>
</tr>
<tr>
<td>Mortgage Company</td>
<td>Credit Application</td>
<td>TRW upon receiving credit application must have a way of securely identifying the request.</td>
</tr>
<tr>
<td>Brokerage</td>
<td>Margin trading</td>
<td>Must compute current balances and margin rules before allowing trade.</td>
</tr>
<tr>
<td>Insurance</td>
<td>File Claims</td>
<td>Policy States and Policy type must match for claims to be processed.</td>
</tr>
<tr>
<td>Bank</td>
<td>Online Banking</td>
<td>User can look at own account.</td>
</tr>
<tr>
<td>All</td>
<td>House holding</td>
<td>For purposes of silo (e.g., statements or discounts), aggregate accounts of all family members.</td>
</tr>
</tbody>
</table>
Advantages of Standardized SW Rules

• Easier Integration: with rest of business policies and applications, business partners, mergers & acquisitions
• Familiarity, training
• Easier to understand and modify by humans
• Quality and Transparency of implementation in enforcement
  – Provable guarantees of behavior of implementation
• Reduced Vendor Lock-in
• Expressive power
  – Principled handling of conflict, negation, priorities
Advantages of SW Rules, cont’d: 

Loci of Business Value

- Reduced system dev./maint./training costs
- Better/faster/cheaper policy admin.
- Interoperability, flexibility and re-use benefits
- Greater visibility into enterprise policy implementation => better compliance
- Centralized ownership and improved governance by Senior Management
- Rich, expressive trust management language allows better conflict handling in policy-driven decisions
Policies for Compliance and Trust Mgmt.: Role for Semantic Web Rules

• Trust Policies usually well represented as rules
  – Enforcement of policies via rule inferencing engine
  – E.g., Role-based Access Control
    • This is the most frequent kind of trust policy in practical deployment today.
  – W3C P3P privacy standard, Oasis XACML XML access control emerging standard, …

• Ditto for Many Business Policies beyond trust arena, too
  – “Gray” areas about whether a policy is about trust vs. not: compliance, regulation, risk management, contracts, governance, pricing, CRM, SCM, etc.
  – Often, authorization/trust policy is really a part of overall contract or business policy, at application-level. Unlike authentication.
  – Valuable to reuse policy infrastructure
Deeper Research Directions

Benjamin Grosof
MIT Sloan School of Management,
http://ebusiness.mit.edu/bgrosof

Presented at DAML PI Mtg., May 25, 2004, New York City

5/28/2004 Copyright 2004 by Benjamin Grosof. All Rights Reserved
Basic Design-Space Points for Rules

- FOL -- ISO Common Logic
- LP -- RuleML
- OWL + Horn -- SWRL

- Pick your favorite for your application!
  - But some are Webized better

5/28/2004  Copyright 2004 by Benjamin Grosof. All Rights Reserved
More Research Challenges: Core

• Integrating rules with ontologies
  – Rules refer to ontologies (e.g., in RuleML)
  – Rules to specify ontologies (e.g., Description Logic Programs)
  – Rules to map between ontologies (e.g., ECOIN)
  – Combined rules + ontologies knowledge bases (e.g., RuleML + OWL)

• Describing business processes & web services via rules + ontologies
  – Rules query web services (e.g., in RuleML Situated feature)
  – Rules trigger actions that are web services (e.g., ditto)
  – Capture object-oriented process ontologies
    • Default inheritance via rules (e.g., Courteous Inheritance)
    • Wrapper/transform to legacy C++, Java, UML
    • Develop open source knowledge bases (e.g., MIT Open Process Handbook Initiative)
  – Event triggering of rules (e.g., capture ECA rules in RuleML)
  – Rules in process models, e.g., cf. OWL-S, PSL
More Research Challenges: Business Policies

- Apply advanced rule and ontology representation to business policies in compliance, trust, contracts, etc.
  - Application scenarios for compliance checking/support services intra- and inter-enterprise
  - Policy language & engines on top of rule language & engines
  - In/with existing/emerging standards: XBRL, XACML, P3P, ebXML, EDI, Legal XML, …
  - Strategy and roles in the market ecology: regulators, communal repositories, service providers, etc.
  - Embedding into the bigger pictures of financial services, e-commerce, semantic web services, business process automation
Some Interesting Directions for DAML Rules
(- but most of it beyond Program End)

• **Preamble:** ...

• **These directions are for both RuleML and SWRL.**

• **CAUTION:** Most of these directions have time horizon beyond the end of the DAML Program.
Some Interesting Directions for DAML Rules
- some of it nearer term

- Alternative syntaxes
  - Presentation Syntax for human authoring
    - Draw upon ideas in Prolog, N3, HiLog/F-Logic, XQuery, RDF-Query
  - RDF syntax for RuleML

- Extend SWRL and RuleML towards FOL
  - Focus: define syntax
  - Coordinate with Simple Common Logic, DRS

- Application scenarios, use cases
  - E.g., Services SCAMP
  - E.g., ontology translation / data mappings
Some Interesting Directions for DAML Rules
- some of it nearer-term

- Inferencing techniques, with associated theory and complexity

- Translation mappings and techniques b/ rule systems
  - More rule systems/languages, esp. of types important commercially
  - ↔ Ontology systems too

- More implementation experience, generally
- Refine application ⇒ technical requirements/focus, generally
  - where’s the business/social value
Some Interesting Directions, cont.’d

• Combine SWRL with Nonmon
  – A requirement from SWSI Rules
  – Negation-As-Failure, Priorities; Aggregations (require closing)
  – It’s already available in RuleML
    • So one obvious approach is to translate SWRL to RuleML
      – … using DLP OWL2RuleML translator (e.g., SweetOnto)

• Extend SWRL to OWL-Full
  – How much immediate demand is there for this?
  – Can use HiLog techniques (e.g., in Flora-2)
Some Interesting Directions for DAML Rules

• More expressiveness in direction of existentials (head/outer).
  – E.g., simpler semantics/theory for anonymous existentials, bnodes, and their relationship to skolemization (cf. recent Yang & Kifer work)

• More about attached procedures cf. Situated LP and Jess/production rules, and some policy languages:
  – Dynamic sensing, e.g., query a web service
  – Actions/effectors with side effects
  – Develop use cases to start, e.g., in SCAMP

• Justifications, proofs, explanations – interchangeably
  – E.g., use and extend InferenceWeb

• Extension toward HiLog limited higher-order expressiveness (esp. LP)

• Extension toward Lloyd-Topor style syntactic sugar (esp. LP)

• Extension towards F-Logic extension, esp. in presentation syntax
Overview of Approaches to Implementing SWRL

- Translate full SWRL to FOL (then use FOL theorem-provers)
- Translate subsets
  - Horn-expressible to LP
  - DL-expressible to OWL
  - In more detail:
    - Horn subset of SWRL to LP-type rules – e.g. RuleML, Jess, XSB (then use rule engine)
      - E.g., just translate subset of definite Horn rules
      - E.g., translate definite named-classes-only SWRL rules and DLP subset of OWL
      - E.g., further use DLP techniques on complex classes within SWRL rules
    - Translate to DL subset to OWL and use DL inferencing engines
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)
More Details on SWSI Rules

from SWSI F2F Discussion

on May 24, 2004
New Tasks for SWSL Requirements from SWSA Requirements Analysis

• “New” here = wrt current emphasis in SWSL Requirements doc
• Tasks focus to add to requirements:
  – Security
    • Esp. policy / decision-making aspects
  – Semantic Interoperability
    • Esp. mapping outputs of one service to inputs of another service
      – E.g., Semantic Web based “glue” processes/services
• These were of broadest need according to the SWSA scenarios requirements analysis
  • (Presented by Mark Burstein at Oct. 2003 SWSI F2F)
Focus Tasks for nearer-term

- Focus on pieces to support particular set of tasks
  - Which need little or no procedural process modeling, temporality, or planning.
  - Start with what rules + ontologies alone can handle

1. Policies for trust:
   - For task of security/privacy/authorization

2. Mapping-type mediation, e.g., of input and output info, or very light workflow:
   - for task of semantic interoperability
Focus Tasks for nearer-term, cont.'d

3. Contracts, incl. advertising, request for proposals, proposals, selection
   – Focus on policy provisions/aspects and decision making in terms of those
   – For task of contracts and negotiation
   – For tasks of advertising and discovery

4. Monitoring and exception handling
   – Focus on contract/policy aspects
   – For task of monitoring and exception handling
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**
2. Semantic interoperability mappings between information models used by different services – e.g., output of one service to input of another service – would also meet an immediate need in WS today.
   – Rules + ontologies – e.g., SWRL – are good for doing such mappings.
     • More clean, and more cleanly expressive, than XSLT – the state of the art for XML stuff today.
     • Today’s thriving commercial vendors in the overall (not-necessarily-XML) space, such as Vitria, often use Rules heavily for this (e.g., Event-Condition-Action rules).
     • This is intrinsically “semantic” stuff where Semantic Web techniques can shine. It’s another WS niche we should “own” as SW’ers.
Summary of Technical Approach

• Basic KR foundation:
  – Start with LP expressiveness
  – Add nice generic LP extensions:
    • Courteous priorities
    • Situated procedural attachments for queries (sensing) and actions (effecting)
    • HiLog “higher-order” expressiveness
    • F-Logic syntax for 2-ary properties etc.
    • Etc.
• Generic ontology capabilities – from the basic KR foundation:
  – Expresses a considerable fragment of OWL: DLP+extensions
  – Can express OO process ontologies with default inheritance, cf.:
    • Process Handbook frames, C++, Java, UML
Summary of Technical Approach, cont. ’d

• Develop Service Ontologies – with associated definitional knowledge bases
  – Start with OWL-S (esp. its profiles aspect); draw also from FLOWS, (?)CTR++
  – In overall spirit of OWL-S profile, but can go further/deeper
  – Service Ontology here = talks about relevant aspects of services, e.g., activities, WSDL “interfaces”, WS-Choreo messages, profile aspects, etc.
  – Provide & use hooks to WSDL, WS-Choreo, ?BPEL, ?SOAP
  – Extend info models in those
  – Draw upon the LP-expressible subsets of the above
• Later: more extensions
  – E.g., for procedural process modeling, temporality, planning, etc.
  – E.g., hopefully to get more of “LP union FOL” as fundamental expressiveness
Example I – Credit Card Verification System

- Typical for eCommerce websites accepting credit cards – Visa, MC, Discover, Amex
- Rules for transaction authorization
  - Bank performs account limit, expiration, address and card code verification
  - A fraud alert service may flag a card
  - Service provider may blacklist customer
- Overrides, e.g., alert service over bank rules
CommonRules Implementation for Credit Card Verification Example

Sample Rule Listing

```xml
<bankResp>
  if checkTran(?Requester)
  then
    transactionValid(self,?Requester);
</bankResp>

<cardRules2>
  if checkCardDet(?Requester, ?accountLimit, ?exp_flag, ?cardholderAddr, 
                   ?cardholderCVC) and 
  checkTranDet(?Requester, ?tranAddr, ?tranCVC) and 
  notEquals(?tranCVC, ?cardholderCVC)
  then 
  CNEG transactionValid(self,?Requester);
...
overrides(cardRules2, bankResp);
checkTran(Joe);
checkCardDet(Joe, 50, "false", 13, 702);
checkTranDet(Joe, 13, 702);
cardGood(Fraudscreen.net,Joe,good);
customerRating(Amazon.com, Joe, good);
```

CommonRules translates straightforwardly ↔ RuleML.

We show its human-oriented syntax as a presentation syntax for RuleML.
Runtime Results for Credit Card Verification

Sample Output

SCLPEngine: Adorned Derived Conclusions:

CNEG transactionValid_c_3(self, Mary);
transactionValid_c_2(self, Joe);
transactionValid_c_2(self, Mary);
transactionValid_r_2(self, Mary);
transactionValid_u(self, Joe);
CNEG transactionValid_u(self, Mary);

transactionValid(self, Joe);
CNEG transactionValid(self, Mary);

Adorned conclusions represent intermediate phases of prioritized conflict handling in Courteous Logic Programs

CNEG = limited classical negation (which is permitted in Courteous LP)
CNEG p means p is (believed to be) false

Self = the agent making the authorization decision, i.e., the viewpoint of this local rulebase.
(This is as usual in trust management.)