Revised and Extended Strawman for SWSL: ROSE: Rules and Ontologies for web SErVICES

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

Slides presented at SWSL F2F May 23, 2004
http://www.swsi.org

5/24/2004 Copyright 2002-2004 by Benjamin Grosof. All Rights Reserved.
• Ff. is included for RECAP …
Outline of Presentation

• Intro: Review the “NO Procedural Process Model” proposal and discussion from Oct. 2003 SWSI F2F (see separate text file)
• Rest is Further Thinking and clarification
  – KR Approach
  – What we’ll be able to do with it
  – Strategy for combining/extend it with other approaches
Outline of Proposed KR Approach

• LP Rules as core of near-term Knowledge-based Service Descriptions
  – + Procedural Attachments: Effectors, Sensors, Events
  – + DLP Ontologies
  – + OO default inheritance, e.g., using Courteous Inheritance
    • Model C++, Java, C#, UML
  – + Hilog/F-Logic-y “meta-”logical expressiveness
    • Close relationship to Flora, via underlying LP representation
• Other Aspects / Extensions – less immediately:
  – FOL
    • Constraints
    • (DL – DLP)
    • ? What else needed ?
  – Procedural Process Models
    • ? Which model ? Concurrent Transaction Logic? (Am open-minded.)
    • Best guess: Start with capabilities of BPEL, WS Choreography design
    • ? What will be the “extension points” of the KR / Process Model?
Goals wrt Key SWS Tasks

– The point of SWS is knowledge reuse
  • Especially the Knowledge-based service descriptions

– … Across the Key Tasks in our Requirements:
  • Advertising/Discovery/Matchmaking; Contracts (selection, negotiation); Enactment, Composition; Monitoring, problem resolution, exception handling; Verification; Trust/Security/Privacy Policies

– Underlying: Hypothetical Reasoning
Where Rules + Ontologies alone are useful

- LP Rules + ~DL Ontologies alone are useful -- enough to be worthwhile -- in almost all of the main Tasks areas, with reuse between Tasks as well as between Applications:
  - ADM: partial contracts, subsumption
    - E.g., see papers from WWW-2003 EC session
  - Contracts/selection/negotiation: pricing, policies, contingent provisions
    - E.g., cf. SweetDeal approach
  - Monitoring, problem resolution, exception handling
    - E.g., cf. SweetDeal approach
  - Enactment
    - Via procedural attachments, esp. effectors, events
  - Composition: e.g., via composing service-description knowledge bases by union’ing their rules/ontologies
  - Trust Policies:
    - Most major practical approaches are rule-based already:
      - RBAC, XACML, P3P, etc.
  - Underlying: Hypothetical Reasoning
    - A major strength of Rule-based KR
NEW SLIDES FOLLOW

• Ff. are further thoughts…
Outline of Revised & Elaborated Approach

– Design Philosophy
  • Where we have business value to offer to WS

– Tasks to Focus Upon
  • Security, Contracts, Advertising, access, authorization, mappings/mediation for semantic interoperability, Monitoring, privacy, and Policies (SCAMP)

– Technical Approach
  • logic program expressiveness for Rules, + generic and service Ontologies + extensions, for Services (ROSE)
  • Combine with much of the other proposed approaches

– A (placeholder) Name: ROSE

– Game Plan
Design Philosophy

– Focus on where's the business value of Semantic WS -- that we have to offer near term to mainstream WS / Web community
  • hence the tasks focus below

– Focus on tasks
  • What it takes to support a particular set of tasks
  • NOT (primarily) on "this captures what I know about a service“ in an undirected way

– Usually, any particular service description is incomplete
  • (Quite incomplete!)
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
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.
Business Value $\Rightarrow$ Strategy, cont.’d

- Pricing policies want/need nonmon, e.g., cf. scenarios in SweetDeal (B. Grosof et al.) and by M. Kifer et al.
- Many other policies are represented well as rules, e.g., cf. SweetRules e-commerce application scenarios (B. Grosof et al.), e.g., refunds, late deliveries, product/service goods descriptions, etc. Lawyers view contracts as default rules.
- … So those are tasks where Semantic Web / knowledge-based techniques can shine in near-term.
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
Business Value ⇒ Strategy, cont. ’d

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.
By contrast: more general Composition tasks (including Discovery matchmaking aspect) are hard

- There are many competing approaches to procedural process modeling
  - And composition is computationally intractable worst-case
- Requires a lot of completeness in the service descriptions to fully automate these
- Near term: programmers/developers do it (i.e., only lightly automated)
  - They expect to, and wouldn’t easily trust anyone else
    - Especially until more is available in way of standardized service contracts
  - It’s OK for them to be done at development time rather than run time

... So it’s an important topic for research rather than near-term standardization or development
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
Outline of Revised & Elaborated Approach

– Design Philosophy
  • Where we have business value to offer to WS

– Tasks to Focus Upon
  • Security, Contracts, Advertising, access, authorization, mappings/mediation for semantic interoperability, Monitoring, privacy, and Policies (SCAMP)

– Technical Approach
  • logic program expressiveness for Rules, + generic and service Ontologies + extensions, for SErvices (ROSE)
  • Combine with much of the other proposed approaches

– A (placeholder) Name: ROSE

– Game Plan
Details on Technical Approach I

• To start: in the KR itself, have little or no procedural process modeling, temporality, or planning.

• Some caveats:
  – To start, avoid that (or don’t rely on that) in the basic KR constructs.
    • So this is different than (much of the) CTR++ proposal.
  – Can have some temporal-process ontology, however.

  • Could combine with much of FLOWS ontology!
    – Should be relatively easy in principle to extend to include light workflow.
      • But maybe that’s not helpful compared to other available WS pieces such as WS Choreography.
Details on Technical Approach II

- Basic KR issue:
  - “LP $\uparrow$ FOL” Start with LP, then try to get as much FOL expressiveness (e.g., head disjunction) as possible
    - This proposal. Also (much of) CTR++.
    - Can be viewed as capturing computationally attractive subset of FLOWS – hence very compatible with FLOWS from that viewpoint.

Versus

- “FOL $\uparrow$ LP” Start with FOL, then try to get as much LP expressiveness (e.g., nonmon, side-effectful actions) as possible
  - FLOWS proposal, PSL

- How to view OWL-S in above terms? (Neutral?)
Venn Diagram: Expressive Overlaps among KR’s

First-Order Logic

Description Logic

Horn Logic Programs

Description Logic Programs

Logic Programs

(Negation As Failure)

(Procedural Attachments)
Technical Capabilities Enabled by DLP

• LP rules "on top of" DL ontologies.
  – E.g., LP imports DLP ontologies, with completeness & consistency
  – Consistency via completeness and use of Courteous LP

• Translation of LP rules to/from DL ontologies.
  – E.g., develop ontologies in LP (or rules in DL)

• Use of efficient LP rule/DBMS engines for DL fragment.
  – E.g., run larger-scale ontologies
  – ⇒ Exploit: Scaleability of LP/DB engines >> DL engines, as |instances| ↑.

• Translation of LP conclusions to DL.
• Translation of DL conclusions to LP.

• Facilitate rule-based mapping between ontologies / “contexts”
Details on Technical Approach III

• Coexists with PSL – insofar as have overlap in the basic KR (Horn + some more)

• In LP approach, one can use FLOWS/PSL ontology and styles of formulation,
  – But with some expressive limitations wrt disjunction etc.
  – Extends with nonmon and procedural attachments
Details on Technical Approach IV

- Wrt CTR++:
  - From Courteous LP: add courteous to subset of CTR++
    - Should be pretty straightforward using Courteous Compiler transformation approach
  - From CTR++: use its HiLog approach to get (bit limited) meta
    - Should be pretty straightforward
  - From CTR++: use its F-Logic approach for OO syntax where useful
    - Should be pretty straightforward
Details on Technical Approach V

• Wrt OWL:
  • can integrate OWL via DLP and also via info passing of facts etc. that are derived, e.g., via procedural attachments

• Wrt OWL-S:
  – Add rules to profiles etc.
  – Have more extension points (see next slide)
Wrt existing (non-”Semantic”) Web Services standards/pieces:

– Introduce mechanisms to define rule+ontology knowledge bases (small or large) for profiles, mappings, policies, contracts; then later for actions, sequencing

Work up from existing WSDL, and to lesser extent WS Choreography

– add OWL-S + rules + some service ontologies
More about Game Plan

• More Extensions:
  – Could create one or more languages with task-specific constructs, which compile down to, or extend from, a base LP rule+onto language

• Need to divvy up our intellectual territory, e.g.:
  – LP-oriented for SCAMP tasks
  – FOL-oriented for synthesizing compositions and other aspects (e.g., enactment) involving deeper procedural process modeling
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
Alternative Name: ROWS

- ROWS = Rules and Ontologies for Web Services
Outline of Revised & Elaborated Approach

– Design Philosophy
  • Where we have business value to offer to WS

– Tasks to Focus Upon
  • Security, Contracts, Advertising, access, authorization, mappings/mediation for semantic interoperability, Monitoring, privacy, and Policies (SCAMP)

– Technical Approach
  • logic program expressiveness for Rules, + generic and service Ontologies + extensions, for SErvices (ROSE)
  • Combine with much of the other proposed approaches

– A (placeholder) Name: ROSE

– Game Plan
OPTIONAL SLIDES FOLLOW
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>
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
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
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);
...
</cardRules2>
```

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.)
### Equational Ontological Conflicts in Financial Reporting

<table>
<thead>
<tr>
<th>Equation 1</th>
<th>Equation 2</th>
</tr>
</thead>
<tbody>
<tr>
<td># of customers = # of end_customers + # of distributors</td>
<td># of customers = # of end_customers + # of prospective customers</td>
</tr>
<tr>
<td>P/E Ratio = Price / Earnings(last 4 Qtr)</td>
<td>P/E Ratio = Price/ [Earnings(last 3 Qtr) +Earnings(next quarter)]</td>
</tr>
<tr>
<td>Price = Nominal Price + Shipping</td>
<td>Price = Nominal Price + Shipping + Tax</td>
</tr>
</tbody>
</table>

“heterogeneity in the way data items are calculated from other data items in terms of definitional equations”
**Solution Approach:** ECOIN

*Extended COntext INterchange MIT Sloan prototype E-Shopping App. (Financial Info is ubiquitous in e-biz)*

- **Price Equations**
- **Context Mediator**
- **eToys**
  - Price: Nominal + Tax + Shipping
  - Product Code: Alpha
- **Kid’s World**
  - Price: Nominal + Tax
  - Product Code: Numeric

**Query**
- Prices of Products
- Cheaper in eToys compared to Kid’s World

<table>
<thead>
<tr>
<th>Product Code</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>45starwars</td>
<td>17</td>
</tr>
<tr>
<td>pokemon</td>
<td>17</td>
</tr>
<tr>
<td>starwars</td>
<td>30.1</td>
</tr>
<tr>
<td>20123456</td>
<td></td>
</tr>
<tr>
<td>30.1starwars</td>
<td></td>
</tr>
<tr>
<td>40234567</td>
<td></td>
</tr>
<tr>
<td>234567</td>
<td></td>
</tr>
</tbody>
</table>

Copyright 2002-2004 by Benjamin Grosif. All Rights Reserved.
Approach: ECOIN

• Context-based loosely-coupled integration
  
  Extends the Context Interchange (COIN) framework developed at MIT

• Symbolic Equation Solving using Constraint Logic Programming
  
  Integrates symbolic equation solving techniques with abductive logic programming
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)
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