WELCOME! to the ISWC-2009 Tutorial

"Semantic Rules on the Web"

by Benjamin Grosof, Mike Dean, and Michael Kifer

INSTRUCTIONS! All participants, please:

- Download the tutorial slideset

at http://www.mit.edu/~bgrosof/#ISWC2009RulesTutorial

- Sign in on the participants list (hard copy sheet)

with your name, organization, email;

optionally also add your interests, homepage URL

Top-Level Outline of Tutorial

A. Introduction, Overview, and Uses

B. Concepts and Foundations

C. Conclusions and Directions

+ Appendix: References and Resources

Background Assumed:

- basic knowledge of first-order logic, relational databases, XML, RDF(S), OWL

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing, ordering policies, contracts
   - E-science: ecological process, mechanics context
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

NB: (2.)-(4.) are interleaved.

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Outline of Part C. Conclusions & Directions

1. More about Tools

2. Conclusions

3. Directions for Future research

Appendix: References and Resources

(General Discussion)

Rough Schedule, Overall

~14:00-15:00 Part A: Intro & Uses

~15:00-15:45 Part B: Concepts & Foundations

~15:45-16:15 Coffee Break

~16:15-17:30 Part B, continued: Concepts & Foundations

~17:30-18:00 Part C: Conclusions & Directions

PART A. SLIDES FOLLOW

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing, ordering policies, contracts
   - E-science: ecological process, mechanics context
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

NB: (2.)-(4.) are interleaved.

Learning Goals for Tutorial

1. Overview of current state of logical KR theory, applications, languages, standards, tools/systems, market

2. Relationship to Web and Semantic Tech, overall

3. Introduction to the research issues

"Semantic" Technology

- "Semantic" in "semantic web" and "semantic rules" means:
  - 1. Knowledge-based

    … and …
  - 2. Having meaning independent of algorithm and implementation
  - Equipped with an interoperable conceptual abstraction

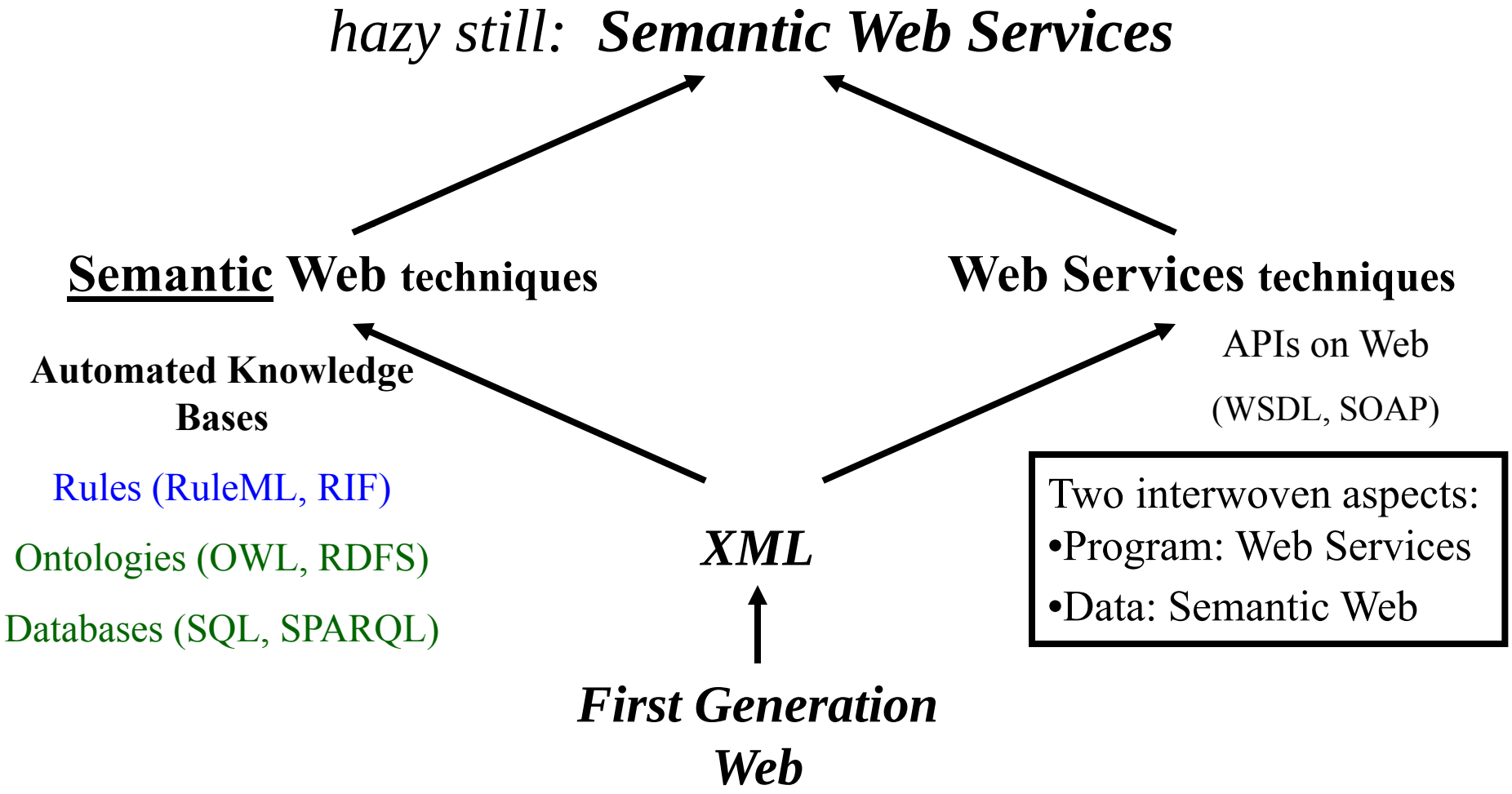
    … based on declarative knowledge representation (KR)

    = Shared principles of what inferences are sanctioned from a given set of premises

What are Rules on the Web

- Convergence of three streams is well along the way
  1. Using Web for interchange of rules, even pre-Web legacy kinds
     - XML syntax for rules. Transcend organizational silos.
  2. Rules working in Web context, using:
     - Web data, schemas, ontologies; Web services, queries, databases
  3. Rules using semantic knowledge representation (KR)
     - Semantics are required for effective sharing of knowledge and tools
- Web as scope for rule-based structured knowledge
  - Enrich the Web as a knowledge platform – public and intranets
  - Collaborative knowledge acquisition (KA), e.g., Wiki's
  - Web-located knowledge bases (KBs) and KR services
- ⇒ Semantic rules on the Web
  - Standardization is a key activity currently

Semantic Web in context of Web



Semantic Web: concept, approach, pieces

- Shared semantics when interchanging data ∴ knowledge
- Knowledge Representation (cf. AI, DB) as approach to semantics
  - Standardize KR syntax, with KR theory/techniques as backing
- Web-exposed Databases: relational and XML/RDF data/queries
  - Challenge: share database schemas via meta-data
  - RDF = "Resource Description Framework" W3C standard
- Ontology = formally defined vocabulary
  - OWL: "Web Ontology Language" W3C standard
    - Taxonomic class/property hierarchy, property-value restrictions, decidable subset of FOL
  - Ex.: Lions are a subcategory within felines
  - Ex.: Every health care visit has a required copayment amount
- Rules = if-then logical implications, facts ~subsumes relational DBs
  - RIF: "Rule Interchange Format" W3C standard (Candidate Recommendation)
    - Based on Logic Programs (LP) Knowledge Representation
    - Based on RuleML (Rule Markup & Modeling Language) standards design
    - Production rule languages
  - Ex.: Any student who has abused printing privileges is prohibited from using color printers
  - Ex.: AAA members get a weekend discount of 20% on suites, at hotel chain X
  - Ex.: During the mitosis phase of an animal cell's lifecycle, all DNA is replicated

Flavors of Rules Commercially Most Important today in E-Business

- E.g., in OO applications, DBs, workflows.

- Relational databases, SQL: Views, queries, facts are all rules.
  - SQL99 even has recursive rules.
- Production rules (OPS5 heritage): e.g.,
  - Jess, ILOG, Blaze, Haley: rule-based Java/C++ objects.
- 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.*
- *Emerging: Semantic-based technology*

Above are "Currently Commercially Important (CCI)"

Commercial Applications of Rules today in E-Business

- There are many. An established area since the 1980's.
  - Expert systems, policy management, workflow, systems management, financial & insurance, e-commerce, trust, personal messaging, defense intelligence, ….
  - Far more applications to date than of Description Logic.

- Advantages in systems specification, maintenance, integration.

- Market momentum: moderately fast growing
  - Fast in early-mid 1980's.
  - Slow late 1980's-mid-1990's.
  - Picked up again in late 1990's. (Embeddable methodologies.)
  - Accelerating in 2000's.

Vision: Uses of Rules in E-Business

- Rules are 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 widely used: e.g., SQL views, queries.
- Rules in communicating applications, e.g., embedded intelligent agents.

Semantic Rules: Differences from Rules in the 1980's / Expert Systems Era

- Get the KR right (knowledge representation)
  - More mature research understanding
  - Semantics independent of algorithm/implementation
  - Cleaner; avoid general programming/scripting language capabilities
  - Highly scaleable performance; better algorithms; choice for interoperability
  - Highly modular wrt updating; use prioritization
  - → Highly dynamic, scaleable rulebase authoring: distributed, integration, partnering
- Leverage Web, esp. XML
  - Interoperable syntax
  - Merge knowledge bases
- Embeddable
  - Into mainstream software development environments (Java, C++, C#); not its own programming language/system (cf. Prolog)
- Knowledge Sharing: intra- or inter- enterprise
- Broader set of Applications

Value of Rules as form of KR

- Rules as a form of KR (*knowledge representation*) are especially useful
  - relatively mature from basic research viewpoint
  - good for prescriptive specifications (vs. descriptive)
    - a restricted programming mechanism
  - integrate well into commercially mainstream software engineering, e.g., OO and DB
    - easily embeddable; familiar
    - vendors interested already: Webizing, application development tools
- ⇒ Identified as part of mission of the W3C Semantic Web Activity, in about 2001

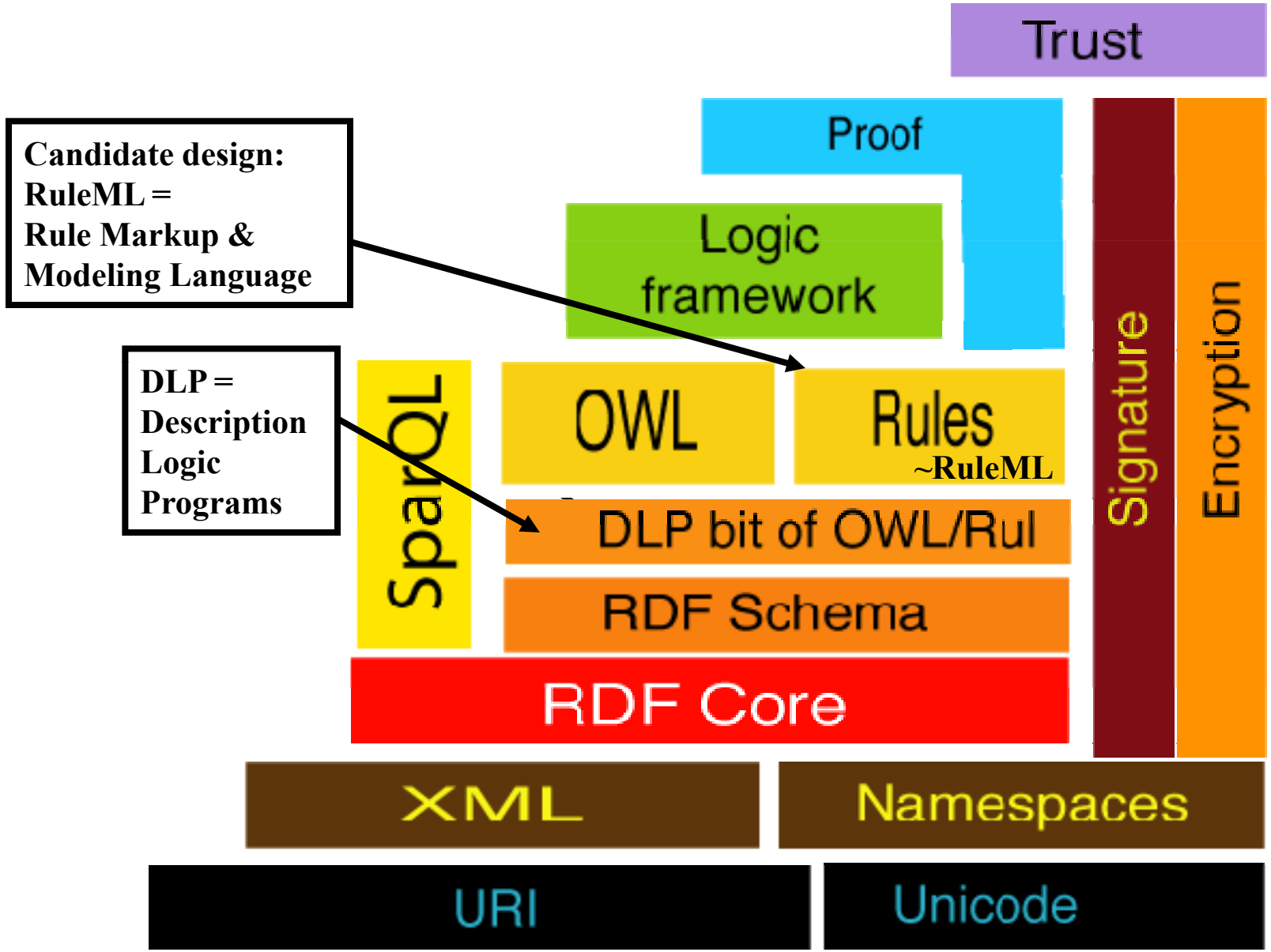
LP is <u>the</u> Core KR in today's world … <u>including</u> the Semantic Web

- **LP is the core KR of structured knowledge management today**
  - **Databases**
    - Relational / SQL
    - XML semi-structured / XQuery
    - RDF semi-structured / SPARQL (triple stores)
  - **Semantic Rule Standards**
    - RuleML standards design
    - Rule Interchange Format (RIF)*
  - **Semantic Ontologies**
    - Most commercial implementations of OWL are based on semantic rules: Description Logic Programs (DLP) + moderate extensions. E.g., Oracle.
    - OWL 2** standard includes the RL Profile, i.e., its Rules subset
- **The Semantic Web today is mainly based on LP KR**
  - … and thus essentially equivalent to semantic rules
  - **You might not have realized that!**

* W3C Candidate Recommendation

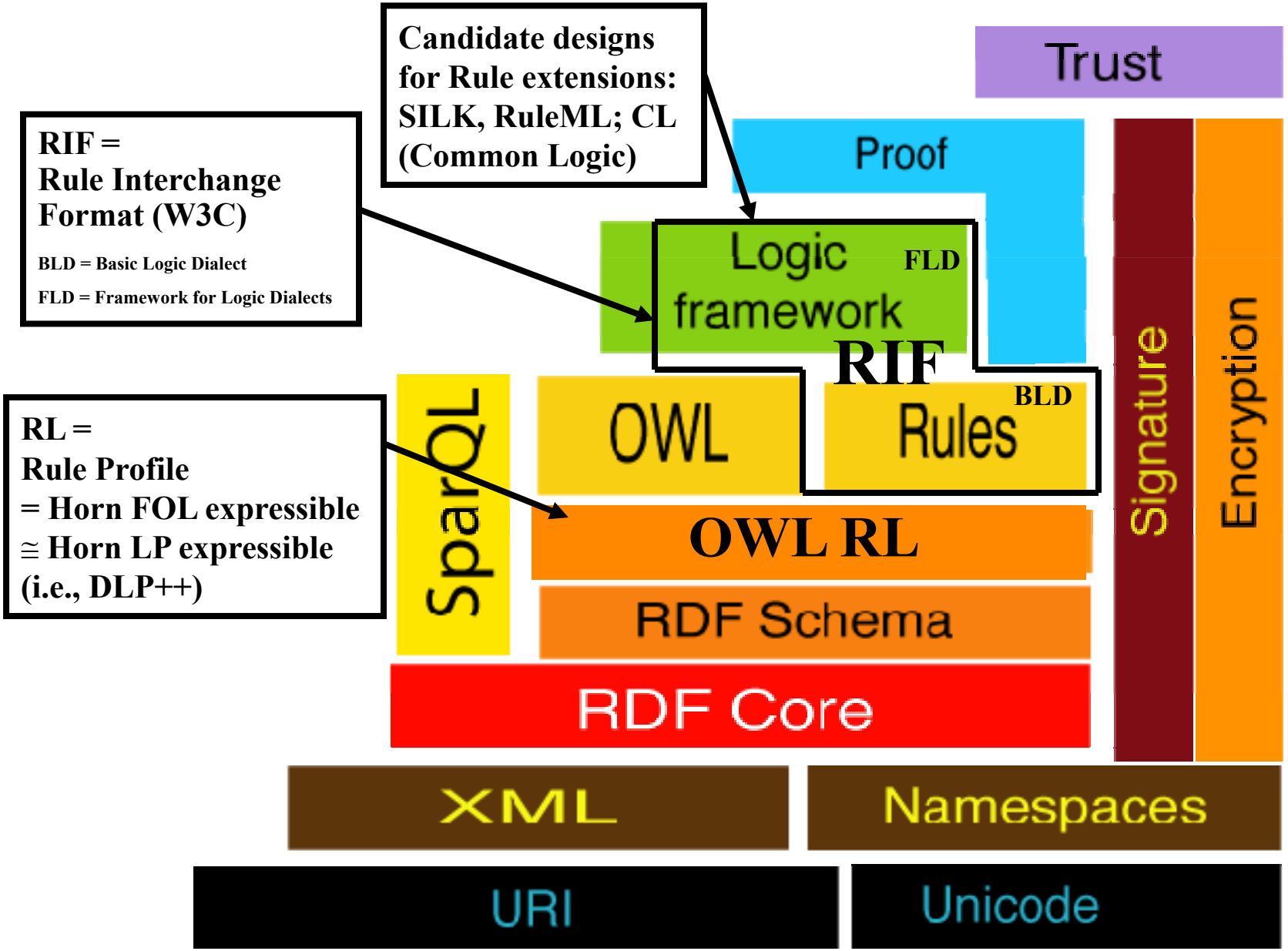
** W3C Proposed Recommendation

08-2005 W3C Semantic Web "Stack": Standardization Steps



Modification of slide by W3C (just added annotation)

Updated: 10-2009 Semantic Web "Stack"



Modified from slide by W3C (just added annotation)

Overview of Key Languages & Standards

1. Database Queries are Rules
   - SQL, SPARQL, XQuery
2. Rule Markup/Modeling Language (RuleML)
   - Main focus is LP, with extensions; FOL too incl. SWRL
   - Web Services modeling: SWSL, WSML
3. W3C Rule Interchange Format (RIF)
   - Basic LP (no defaults or actions)
   - Framework for extensions (defaults and much more)
   - Production rules
4. SILK: Hyper Logic Programs – advanced expressiveness
5. Rules in, and for, W3C OWL (and RDFS) ontologies
   - SWRL. RIF+OWL/RDF combinations.
6. OMG Production Rule Representation (PRR)
7. ISO Common Logic (successor to KIF): classical logic
8. OMG Semantics of Business Vocabulary & Business Rules (SBVR)
9. JSR94 Rule Management APIs

Overview of Key Tools

1. Rule systems designed to work with RDF/OWL
   - Commercial-world: Jena; Oracle; others
   - Research-world: SweetRules; cwm; others
2. Prolog and Production Rule systems
   - XSB; Jess; others
3. Advanced Expressiveness
   - FLORA-2 and SILK; IBM CommonRules
4. Rules in Semantic Wikis
   - Semantic MediaWiki
5. Some Available Large Rule Bases
   - OpenCyc, Process Handbook, OpenMind

Need for Other Kinds of Ontologies besides OWL

- Forms of ontologies practically/commercially important in the world today*:
  - SQL DB schemas
  - "Conceptual models" in UML and E-R (Entity-Relationship)
  - OO inheritance hierarchies, procedural interfaces, datatype declarations
  - XML Schema
  - OWL is still emerging, wrt deployed usage – dwarfed by all the above
  - LP/FOL/BRMS predicate/function signatures
  - Builtins (e.g., SWRL/RuleML)
  - Equations and conversion-mapping functions
- Overall relationship of OWL to the others is as yet largely unclear
  - There are efforts on some aspects, incl. UML
- OWL cannot represent the nonmon aspects of OO inheritance
- OWL does not yet represent, except quite awkwardly:
  - n-ary relations
  - ordering (sequencing) aspects of XML Schema

- (*NB: Omitted here are statistically flavored ontologies that result from inductive learning and/or natural language analysis.)

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing, ordering policies, contracts
   - E-science: ecological process, mechanics context
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

NB: (2.)-(4.) are interleaved.

EECOMS Example of Conflicting Rules: Ordering Lead Time

- Vendor's rules that prescribe how buyer must place or modify an order:
- A) 14 days ahead if the buyer is a qualified customer.
- B) 30 days ahead if the ordered item is a minor part.
- C) 2 days ahead if the ordered item's item-type is backlogged at the vendor, the order is a modification to reduce the quantity of the item, and the buyer is a qualified customer.
- D) 45 days ahead if the buyer is a walk-in customer.

- Suppose more than one of the above applies to the current order? **Conflict!**
- Helpful Approach: **precedence** between the rules.
  - E.g., D is a catch-case: A > D , B > D , C > D
- Often only *partial* order of precedence is justified.
  - E.g., C > A , but no precedence wrt B vs. A, nor wrt C vs. B.

Ordering Lead Time Example in LP with Courteous Defaults

```
@{prefCust}   orderModifNotice(?Order,14days)   :-
                  preferredCustomerOf(?Buyer,SupplierCo),   purchaseOrder(?Order,?Buyer,SellerCo) ;
@{smallStuff}   orderModifNotice(?Order,30days)  :-
                  minorPart(?Buyer,?Seller,?Order),   purchaseOrder(?Order,?Buyer,SupplierCo) ;
@{reduceTight}   orderModifNotice(?Order,2days)   :-
                  preferredCustomerOf(?Buyer,SupplierCo) and
                  orderModifType(?Order,reduce) and
                  orderItemIsInBacklog(?Order) and
                  purchaseOrder(?Order,?Buyer,SupplierCo) ;
silk:overrides(reduceTight,  prefCust) ;    // reduceTight has higher priority than prefCust
// The below  exclusion constraint specifies that orderModifNotice is unique, for a given order.
silk:opposes(orderModifNotice(?Order,?X), orderModifNotice(?Order,?Y))   :-   ?X != ?Y ;
```

- Rule D, and prioritization about it, were omitted above for sake of brevity.
- Above rules are represented in Logic Programs KR, using the Courteous defaults feature
- Notation:
  - ":-" means "if". "@{…}" encloses a rule label. "?" prefixes a logical variable.
    "overrides" predicate specifies prioritization ordering.
    An exclusion constraint specifies what constitutes a conflict.
    "!=" means ≠ . "silk:" is a namespace prefix.

EECOMS Supply Chain: Early Commercial Implementation & Piloting

- EECOMS agile supply chain collaboration industry consortium including Boeing, Baan, TRW, Vitria, IBM, universities, small companies
  - $29Million 1998-2000; 50% funded by NIST ATP
  - application piloted IBM CommonRules and early approaches which led to SweetDeal, RuleML, SweetRules, RIF, and SILK
    - contracting & negotiation; authorization & trust

Example: E-Commerce Pricing Offer from SupplierCo to Buyer

@{usualPrice} price(per_unit, ?PO, $60) :-
purchaseOrder(?PO, supplierCo, ?AnyBuyer) and
quantity_ordered(?PO, ?Q) and (?Q ≥ 5) and (?Q ≤ 1000) and
shipping_date(?PO, ?D) and (?D ≥ "2000-04-24") and (?D ≤ "2000-05-12") ;

@{volumeDiscount} price(per_unit, ?PO, $51) :-
purchaseOrder(?PO, supplierCo, ?AnyBuyer) and
quantity_ordered(?PO, ?Q) and (?Q ≥ 100) and (?Q ≤ 1000) and
shipping_date(?PO, ?D) and (?D ≥ "2000-04-28") and (?D ≤ "2000-05-12") ;

silk:overrides(volumeDiscount , usualPrice) ; // *volumeDiscount rule has higher priority*

// *The below exclusion constraint says the value of price is unique for a given PO*

silk:opposes(price(per_unit, ?PO, ?X), price(per_unit, ?PO, ?Y)) :- ?X != ?Y ;

...

- Notation:
  "@{…}" encloses a rule label. "?" prefixes a logical variable.
  "overrides" predicate specifies prioritization ordering.
  An exclusion constraint specifies what constitutes a conflict.
  "!="means ≠ . "silk:" is a namespace prefix.

Pricing Example --

XML Encoding of Rules in RuleML

```
<rulebase>
 <imp>
  <rlab>usualPrice</rlab>
  <head>
   <cslit>
    <opr><rel>price</rel></opr>
     <ind>per_unit</ind>
     <var>PO</var>
     <ind>$60</ind>
   </cslit>
  </head>
  <body>     …  (see next page, if included)    </_body>
 </imp>
…
</rulebase>
```

- *NB: This uses an older version of RuleML markup syntax. RIF syntax is similar, but RIF Basic Logic Dialect cannot express defaults.*

Ecology Ex. of Causal Process Reasoning (in SILK)

```
/*    Toxic discharge into a river causes fish die-off.    */
/* Init. facts, and an "exclusion" constraint that fish count has a unique value */
  occupies(trout,Squamish);
  fishCount(s0,Squamish,trout,400);
  silk:opposes(fishCount(?s,?r,?f,?C1), fishCount(?s,?r,?f,?C2)) :- ?C1 != ?C2;
/* Action/event description that specifies causal change, i.e., effect on next state */
@{tdf1} fishCount(?s+1,?r,?f,0) :- occurs(?s,toxicDischarge,?r) and occupies(?f,?r);
/* Persistence ("frame") axiom */
  @{pef1}  fishCount(?s+1,?r,?f,?p) :- fishCount(?s,?r,?f,?p);
/* Action effect axiom has higher priority than persistence axiom */
 silk:overrides(tdf1,pef1);
/* An action instance occurs */
  @{UhOh}  occurs(s0+1,toxicDischarge,Squamish);
```

As desired: **|= fishCount(s0+1,Squamish,trout,400);**
fishCount(s0+2,Squamish,trout,0);

Notes: @{…} encloses a rule label. ? prefixes a variable. :- means if. != means ≠. opposes indicates an exclusion constraint between two literals, which means "it's a conflict if".

E-Commerce Ex. of Causal Process Reasoning (in SILK)

```
/*    E-commerce delivery logistics. */
/* Initial fact, and prevention constraint that location is unique */
  loc(s0,PlasmaTV46,WH_LasVegasNV);
  silk:opposes(loc(?s,?item,?posn1), loc(?s,?item,?posn2))   :-   ?posn1 != ?posn2;
/* Action/event description that specifies causal change, i.e., effect on next state */
  @{mov1}  loc(?s+1,?item,?addr) and neg loc(?s+1,?item,?warehouse) :-
        shipment(?s,?item,?warehouse,?addr) and loc(?s,?item,?warehouse);
/* Persistence ("frame") axioms about location */
  @{pel1}  loc(?s+1,?item,?posn) :- loc(?s,?item,?posn);
/* Action effect axiom has higher priority than the persistence axioms */
  silk:overrides(mov1,pel1).
  silk:overrides(mov1,pel2);
/* An action instance occurs */
  @{deliv57}  shipment(s0+1, PlasmaTV46, WH_LasVegasNV, 9_Fog_St_SeattleWA);
```

As desired: **|= loc(s0+2, PlasmaTV46, 9_Fog_St_SeattleWA);**

~~|=~~ loc(s0+2, PlasmaTV46, WH_LasVegasNV);

Trust Mgmt. Ex. of Higher-Order Defaults (in SILK)

illustrating also basic Knowledge-level Communication, and Frame syntax

In Frame syntax: subject[property -> object] *stands for* property(subject,object).

```
/*    Trust policy administration by multiple agents, about user permissions */
/* Admin. Bob controls printing privileges including revocation (neg). */
  Bob[controls -> print];   Bob[controls -> neg print];    /* neg print means it is disallowed.*/
  Cara[controls -> ?priv];  /* Cara is the most senior admin., so controls all privileges. */
/* If an administrator controls a privilege and states at a time (t) that a user has a privilege,
   then the user is granted that privilege. Observe that ?priv is a higher-order variable. */
  @{grant(?t)} ?priv(?user) :- ?admin[states(?t) -> ?priv(?user)] and ?admin[controls(?priv)];
/* More recent statements have higher priority, in case of conflict.  */
  silk:overrides(grant(?t2), grant(?t1)) :- ?t2 > ?t1;
/* Admins Bob and Cara make conflicting statements over time about Ann's printing */
  Cara[states(2007) -> print(Ann)];   Cara[states(2007) -> webPage(Ann)];
   Bob[states(2008) -> neg print(Ann)];
```

As desired: **|= neg print(Ann); webPage(Ann);**

/* Currently, Ann is permitted a webpage but not to print. */

Notes: @{…} encloses a rule label. ? prefixes a variable. :- means if. != means ≠. neg is strong negation. There is an implicit exclusion (silk:opposes) between P and neg P, for every literal P.

Physics Ex. of Contextual Assumptions (in SILK)

```
/* "P8: Joe drops a glove from the top of a 100m cliff.
        How long does the fall take in seconds?" */
// Initial problem-specific facts
AP_problem(P8);  fall_event(P8);  P8[height->100];
// Action description that specifies causal implications on the continuous process
?e[time->((2 * ?h / ?n)^0.5)] :- fall_event(?e) and ?e[height->?h  and  net_accel->?n];
?e[net_accel->(?g - ?a)] :- fall_event(?e) and
                                ?e[gravity_accel->?g  and  air_resistance_accel->?a];
 // Other facts
?e[gravity_accel->9.8] :- loc(?e, Earth);
?e[gravity_accel->3.7] :- loc(?e, Mars);
// Contextual assumptions for answering Advanced Placement exam (AP) problems
@{implicit_assumption} loc(?e, Earth) :- AP_problem(?e);
silk:opposes(loc(?e, Earth), loc(?e, Mars));
@{implicit_assumption} ?e[air_resistance_accel->0] :- AP_problem(?e);
silk:overrides(explicitly_stated, implicit_assumption);
```

As desired: `|=  P8[net_accel->9.8  and  time->4.52];  // 4.52 = (2*100/9.8)^0.5`

Physics Ex. of Contextual Assumptions (in SILK)

/* "P8: Joe drops a glove from the top of a 100m cliff on Mars.
How long does the fall take in seconds?" */
/* Initial problem-specific facts*/
AP_problem(P8); fall_event(P8); P8[height->100];
@{explicitly_stated} loc(P8,Mars);

…

As desired: **|= P8[net_accel->3.7 and time->7.35]; // 7.35 = (2*100/3.7)^0.5];**

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing, ordering policies, contracts
   - E-science: ecological process, mechanics context
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

NB: (2.)-(4.) are interleaved.

Challenge: Capturing Semantics around Policies

- Deep challenge is to capture the semantics of data and processes:
  - To represent, monitor, and enforce policies – e.g., trust and contracts
  - To map between definitions of policy entities, e.g., in financial reporting
  - To integrate policy-relevant information powerfully

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

Trust Policies and Compliance in US Financial Industry Today

- Ubiquitous high-stakes Regulatory Compliance requirements
  - Sarbanes Oxley, SEC (also in medical domain: 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

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

Verticals that appear good candidates for Early Adoption of SW Rules for Privacy

- Financial
  - Cf. discussion earlier in this talk
  - Historically, an early adopter of information technology overall esp. for integration
  - Large sector of global economy
  - Privacy/trust policies very important, distributed & heterogeneous
- Medical
  - Privacy/trust policies very important, distributed & heterogeneous
  - Expecting help on privacy from information technology
  - Large sector of global economy
- Police/Military
  - Privacy/trust policies very important, distributed & heterogeneous
  - Looking for help on privacy from information technology
  - Major funder of SW basic research to date, e.g., DARPA Agent Markup Language program 2000-2005
- In many other realms, there is a large gap between revealed vs. avowed preferences for value of privacy/confidentiality.

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 and enforcement
  - Provable guarantees of implementation behavior
- 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

Some Answers to: "Why does SW/SWS Matter to Business?"

- 1. "Death. Taxes. Integration." - They are always with us.
- 2. "Business processes require communication between organizations / applications." - Data and programs cross org./app. boundaries, both intra- and inter- enterprise.
- 3. "It is the *automated knowledge* economy, stupid!" - The world is moving towards a knowledge economy. And it is moving towards deeper and broader automation of business processes. The first step is automating the use of structured knowledge.
  - Theme: *reuse* of knowledge across multiple tasks/apps/orgs

SW Early Adoption Candidates: High-Level View

- "Death. Taxes. Integration."
- Application/Info Integration:
  - Intra-enterprise
    - EAI, M&A; XML infrastructure trend
  - Inter-enterprise
    - E-Commerce: procurement, SCM
  - Combo
    - Business partners, extranet trend

SW Adoption Roadmap: Strategy Considerations

- Likely first uses in a lot of B2B interoperability or heterogeneous-info-integration intensive applications (e.g., finance, travel)
  - Actually, probably 1st intra-enterprise, e.g., EAI
- Reduce costs of communication in procurement, operations, customer service, supply chain ordering and logistics
  - increase speed, create value, increase dynamism
  - macro effects create
    - stability sometimes (e.g., supply chain reactions due to lag; other negative feedbacks)
    - volatility sometimes (e.g., perhaps financial market swings)
  - increase flexibility, decrease lock-in
- Agility in business processes, supply chains

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing/ordering policies, contracts
   - E-science: ecological process
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

NB: (2.)-(4.) are interleaved.

Ontology Translation Via Rules

- Use rules to represent mappings from data source to domain ontologies
  - Rules can be automatically or manually generated
  - Can support unit of measure conversion and structural transformation
- Example using SWRL
  - http://www.daml.org/2004/05/swrl-translation/Overview.html
- http://snoggle.semwebcentral.org

Uses of Semantic Rules for XBRL

- Ontology mappings: contextual, reformulation
  - *Examples*:
    - Price with vs. without shipping, tax
    - Earnings last 4 qtrs vs.{last 3 qtrs + forecast next qtr}
    - Profit with vs. without depreciation
    - Historical info when statutory treatment changes
    - Implicit context: use a typical definition of revenue
  - Your vs. my pro-forma or analytic view
    - Between companies, governmental jurisdictions
  - Exception handling, special cases, one-time events
    - Footnotes – "where the real action is"
    - *Example:* Revenue includes sale of midtown NYC headquarters bldg

Example: Exception in Ontology Translation (in SILK)

```
/* Company BB reports operating earnings using R&D operating cost which includes
   price of a small company acquired for its intellectual property. Organization GG
   wants to view operating cost more conventionally which excludes that acquisition
   amount. We use rules to specify the contextual ontological mapping. */
@{normallyBringOver} ?categ(GG)(?item) :- ?categ(BB)(?item);
@{acquisitionsAreNotOperating} neg ?categ(GG)(?item) :-
     acquisition(GG)(?item) and (?categ(GG) ## operating(GG));
overrides(acquisitionsAreNotOperating, normallyBringOver); /* exceptional */
acquisition(GG)(?item) :- price_of_acquired_R_and_D_companies(BB)(?item);
R_and_D_salaries(BB)(p1001); p1001[amount -> $25,000,000];
R_and_D_overhead(BB)(p1002); p1002[amount -> $15,000,000];
price_of_acquired_R_and_D_companies(BB)(p1003); p1003[amount -> $30,000,000];
R_and_D_operating_cost(BB)(p1003); /* BB counts the acquisition price item in this category */
R_and_D_operating_cost(GG) ## operating(GG);
Total(R_and_D_operating_cost)(BB)[amount -> $70,000,000]; /* rolled up by BB cf. BB's definitions */
Total(R_and_D_operating_cost)(GG)[amount -> ?x] :- … ; /* roll up the items for GG cf. GG's definitions */
```

As desired: |= R_and_D_salaries(GG)(p1001); …
neg R_and_D_operating_cost(GG)(p1003); /* GG doesn't count it */
Total(R_and_D_operating_cost)(GG)[amount -> $40,000,000];

Notation: @{…} encloses a rule label. ? prefixes a variable. :- means if. X ## Y means X is a subclass of Y. overrides(X,Y) means X is higher priority than Y.

Equational Ontological Conflicts in Financial Reporting

# of customers = # of end_customers + # of distributors	# of customers = # of end_customers + # of prospective customers
Gross Profit = Net Sales – Cost of Goods	Gross Profit = Net Sales – Cost of Goods – Depreciation
P/E Ratio = Price / Earnings(last 4 Qtr)	P/E Ratio = Price/ [Earnings(last 3 Qtr) + Earnings(next quarter)]
Price = Nominal Price + Shipping	Price = Nominal Price + Shipping + Tax

" heterogeneity in the way data items are *calculated* from other data items *in terms of definitional equations*"

Slide also by Aykut Firat and Stuart Madnick

EOC in Primark Databases

Top 25 US Co. by Net Sales (Disclosure DB)

Rank	Company	Net Sales (000's)	Date
1	General Motors Corp	168,828,600	12/31/95
2	Ford Motor Co	137,137,000	12/31/95
3	Exxon Corp	121,804,000	12/31/95
4	Wal Mart Stores Inc	93,627,000	01/31/96
5	AT&T	79,609,000	12/31/95
6	Mobil Corp	73,413,000	12/31/95
7	International Business M	71,904,000	12/31/95
8	General Electric Co	70,028	
...	...	...	

Top 25 International Co. by Net Sales (Worldscope DB)

Rank	Company	Net Sales (000's)	Date
1	Mitsubishi Corporation	165,848,468	03/31/96
2	General Motors Corp	163,861,100	12/31/95
...	...	...	...
8	Exxon Corp	107,893,000	12/31/95
...	...	...	...
16	International Business M	71,940,000	12/31/95
17	General Electric Co	69,948,000	12/31/95
20	Mobil Corp	64,767,000	12/31/95
...	...	...	...

Primark was a company that owned:

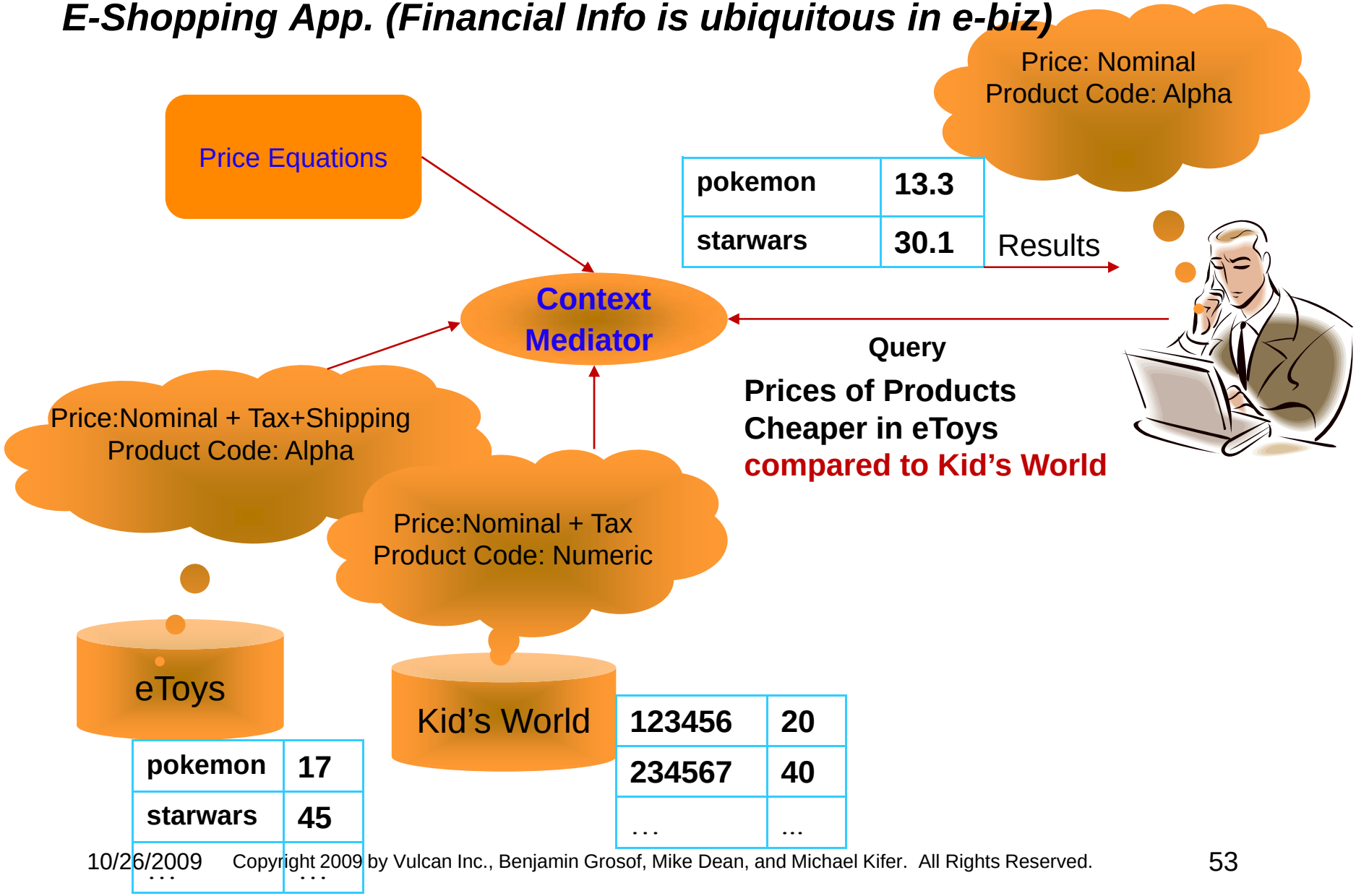
- Disclosure
- Worldscope
- DataStream

Information services

Solution Approach: ECOIN

Extended COntext INterchange MIT Sloan prototype

E-Shopping App. (Financial Info is ubiquitous in e-biz)



ECOIN Approach, continued

- Context-based loosely-coupled integration
- Symbolic Equation Solving combined with LP

XBRL INTERNATIONAL MEMBERS LOG-IN | SEARCH THE S

Home SITE GUIDE EVENTS CONTACT US XBRL AROUN WORLD

- WHAT IS XBRL
- Benefits and Uses
- How XBRL Works
- FAQ
- LATEST NEWS
- XBRL IN ACTION
- XBRL AND BUSINESS
- TECHNICAL INDEX
- TAXONOMIES
- SPECIFICATIONS
- BEST PRACTICES
- TECHNICAL SUPPORT
- EDUCATION AND TRAINING
- ABOUT THE ORGANISATION
- HOW TO JOIN
- **MEMBERS' AREA**
- INFORMATION FOR MEMBERS
- e-GROUPS
- SUPPLY CHAINS

An Introduction to XBRL

XBRL is a language for the electronic communication of business and financial data whicl revolutionising business reporting around the world. It provides major benefits in the preparation, analysis and communication of business information. It offers cost savings, greater efficiency and improved accuracy and reliability to all those involved in supplying using financial data.

XBRL stands for eXtensible Business Reporting Language. It is one of a family of "XML" languages which is becoming a standard means of communicating information between businesses and on the internet.

XBRL is being developed by an international non-profit consortium of approximately 450 r companies, organisations and government agencies. **It is an open standard, free of lice fees.** It is already being put to practical use in a number of countries and implementatio XBRL are growing rapidly around the world.

This site provides information about the nature, uses and benefits of XBRL. It explains h individuals and companies can join the effort to move forward and make use of the langu

A Simple Explanation

The idea behind XBRL, eXtensible Business Reporting Language, is simple. Instead of treating financial information as a block of text - as in a standard internet page or a print document - it provides an identifying tag for each individual item of data. This is comput readable. For example, company net profit has its own unique tag.

The introduction of XBRL tags enables automated processing of business information by computer software, cutting out laborious and costly processes of manual re-entry and comparison. Computers can treat XBRL data "intelligently": they can recognise the information in a XBRL document, select it, analyse it, store it, exchange it with other computers and present it automatically in a variety of ways for users. XBRL greatly incre the speed of handling of financial data, reduces the chance of error and permits automati checking of information.

Companies can use XBRL to save costs and streamline their processes for collecting and reporting financial information. Consumers of financial data, including investors, analysts financial institutions and regulators, can receive, find, compare and analyse data much m rapidly and efficiently if it is in XBRL format.

XBRL can handle data in different languages and accounting standards. It can flexibly be adapted to meet different requirements and uses. Data can be transformed into XBRL by suitable mapping tools or it can be generated in XBRL by appropriate software.

The How XBRL Works page gives further explanation of XBRL, while Benefits and Uses s out how different types of organisation can gain from the standard.

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing, ordering policies, contracts
   - E-science: ecological process, mechanics context
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

NB: (2.)-(4.) are interleaved.

Contracts in E-Commerce Lifecycle

- Discovery, advertising, matchmaking
  - Search, sourcing, qualification/credit checking
- Negotiation, bargaining, auctions, selection, forming agreements, committing
  - Hypothetical reasoning, what-if'ing, valuation
- Performance/execution of agreement
  - Delivery, payment, shipping, receiving, notification
- Problem Resolution, Monitoring
  - Exception handling

Approach: *Rule*-based Contracts for E-commerce

- Rules as way to specify (part of) business processes, policies, products: as (part of) contract terms.
- Complete or partial contract.
  - As default rules. Update, e.g., in negotiation.
- 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.

Semantic Web Services

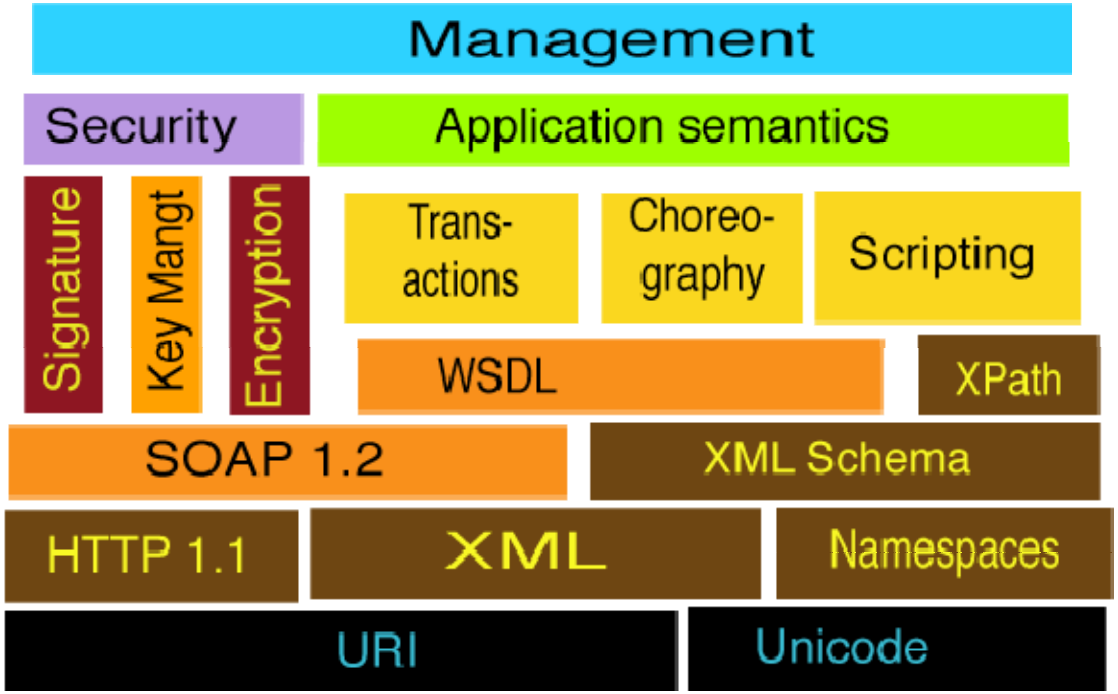
- Convergence of Semantic Web and Web Services
- Consensus definition and conceptualization still forming
- Semantic (Web Services):
  - Knowledge-based service descriptions, deals
    - Discovery/search, invocation, negotiation, selection, composition, execution, monitoring, verification
    - Advantage: **reuse** of knowledge across apps, these tasks
  - Integrated knowledge
- (Semantic Web) Services: e.g., infrastructural
  - Knowledge/info/DB integration
  - Inferencing and translation

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

W3C Web Services Stack (2004)

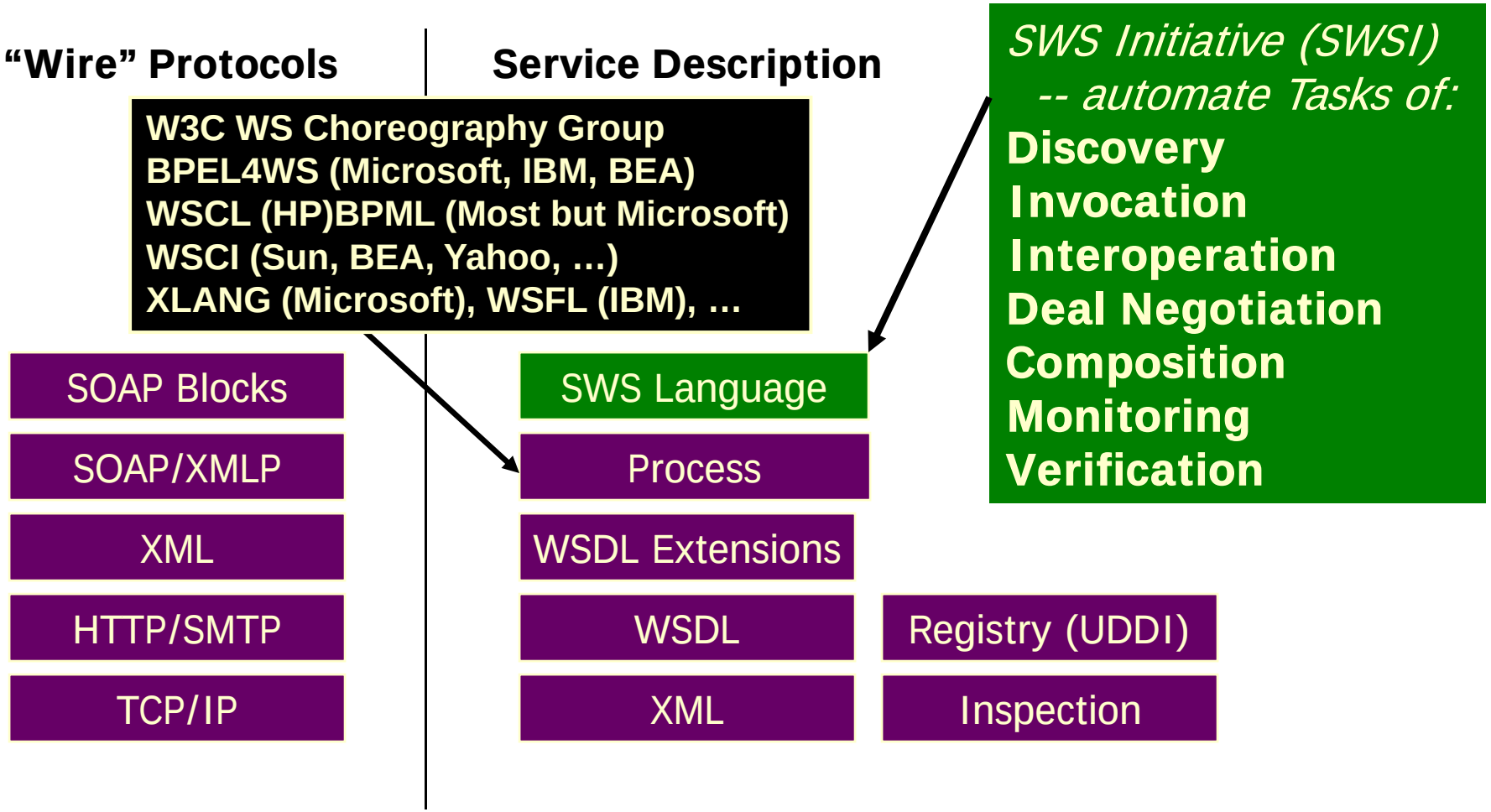


NOTES:

WSDL is a Modular Interface spec
SOAP is Messaging and Runtime
Also:
- UDDI is for Discovery
- BPEL4WS, WSCI, …
  are for transactions
- Routing, concurrency, …

Diagram courtesy Tim Berners-Lee: http://www.w3.org/2004/Talks/0309-ws-sw-tbl/slide6-0.html

SWS Language effort (2005), on top of Web Services Standards Stack



[Slide authors: Benjamin Grosof (MIT Sloan), Sheila McIlraith (Stanford) , David Martin (SRI International), James Snell (IBM)]

Semantic Web Services Framework (SWSF)

- By Semantic Web Services Initiative (SWSI) http://www.swsi.org
  - Coordinated global research and standards design in SWS during 2002-2005
  - Researchers from universities, companies, government
  - Industrial partners; DAML and WSMO backing
  - Collaborators: OWL-S, WSMO, RuleML, DAML
- Designed SWSF in 2005: http://www.daml.org/services/swsf/1.0/
  - Rules & FOL language (SWSL/RuleML)
  - Ontology for SWS (SWSO)
    - Drawn largely from OWL-S and PSL
  - Application Scenarios
  - *Also: requirements analysis*
- Influential, explored the issues
  - ⇒W3C SAWSDL – Semantic Annotations for WSDL and XML Schema
    - Extension mechanism – a hook – with shallow semantics in itself

SWS Tasks Form 2 Distinct Clusters, each with associated Central Kind of Service-description Knowledge and Main KR

1. Security/Trust, Monitoring, Contracts, Advertising/Discovery, Ontology-mapping Mediation
   - Central Kind of Knowledge: Policies
   - Main KR: Nonmonotonic LP (rules + ontologies)

2. Composition, Verification, Enactment
   - Central Kind of Knowledge: Process Models
   - Main KRs: FOL + **Nonmonotonic LP**

Rule-based Semantic Web Services

- Rules/LP in appropriate combination with DL as KR, for RSWS
  - DL good for categorizing: a service overall, its inputs, its outputs

- Rules to describe service process models
  - rules good for representing:
    - preconditions and postconditions, their contingent relationships
    - contingent behavior/features of the service more generally,
      - e.g., exceptions/problems
  - familiarity and naturalness of rules to software/knowledge engineers

- Rules to specify deals about services: cf. e-contracting.

Outline of Part A. Intro & Uses

1. Overview of tutorial, and get acquainted
2. What are: Rules on the Web, Semantic Rules/Web/Tech
3. Uses and Kinds of rules
   - Commercial, web. Current, envisioned.
   - Requirements. Business value, IT lifecycle.
   - Strategic roadmapping of future adoption
4. Example Use Cases
   - E-commerce: pricing, ordering policies, contracts
   - E-science: ecological process, mechanics context
   - Trust: compliance, policies, e.g. financial services
   - Info integration, ontology mapping, business reporting
   - Processes: policy-based workflow, causal action effects, Semantic Web Services

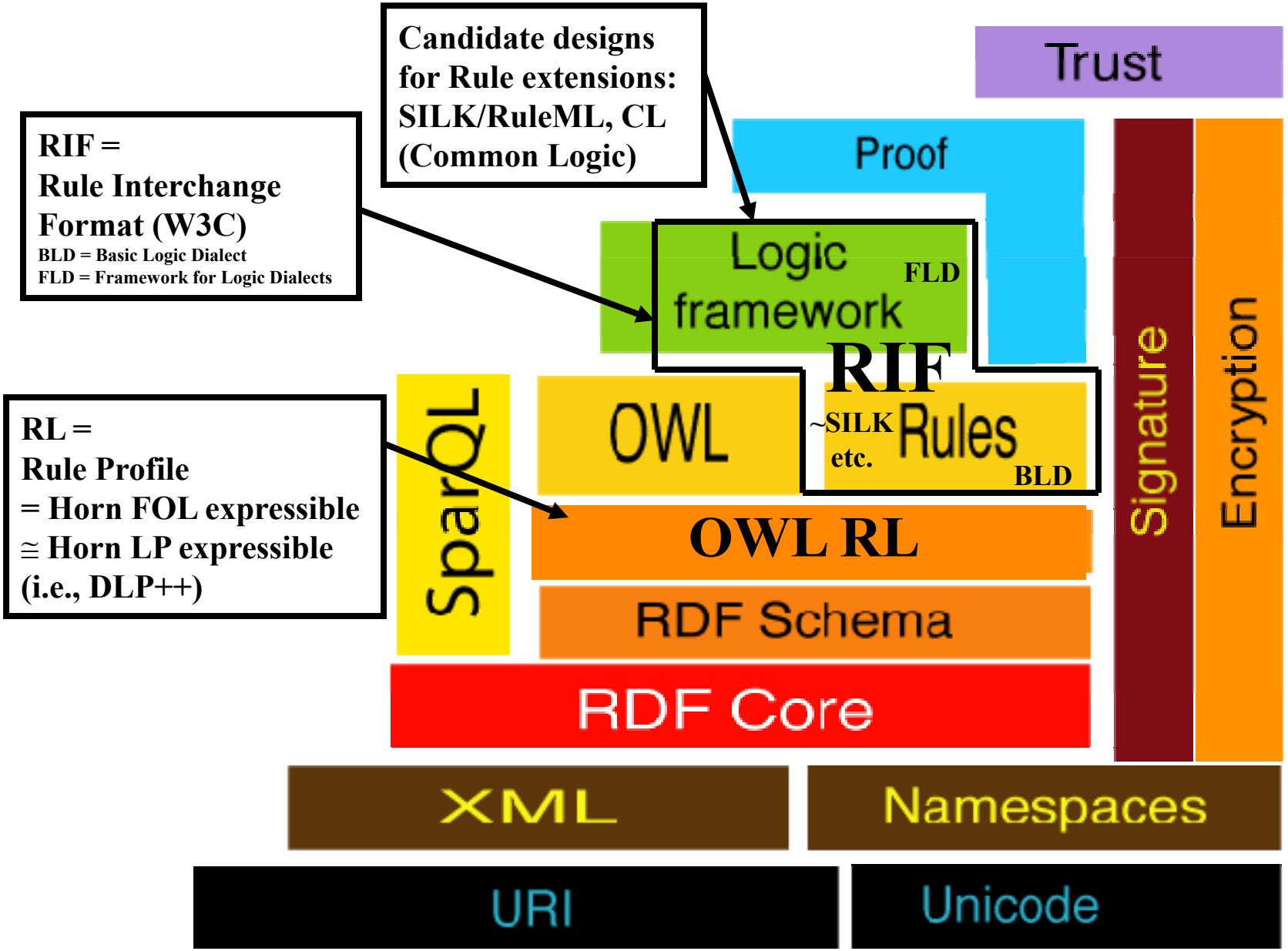
NB: (2.)-(4.) are interleaved.

PART B. SLIDES FOLLOW

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Updated: 10-2009 Semantic Web "Stack"



Modified from slide by W3C (just added annotation)

Concept of KR

- A KR S is defined as a triple (LP, LC, |=), where:
  - LP is a formal language of sets of premises (i.e., premise expressions)
  - LC is a formal language of sets of conclusions (i.e., conclusion expressions)
    - *Remark: In declarative logic programs KR, LC is a subset of LP*
  - |= is the entailment relation.
    - Conc(P,S) stands for the set of conclusions that are entailed in KR S by a set of premises P
    - We assume here that Conc is a functional relation.

- Typically, e.g., in FOL and LP, entailment is defined formally in terms of models, i.e., truth assignments that satisfy the premises and meet other criteria.

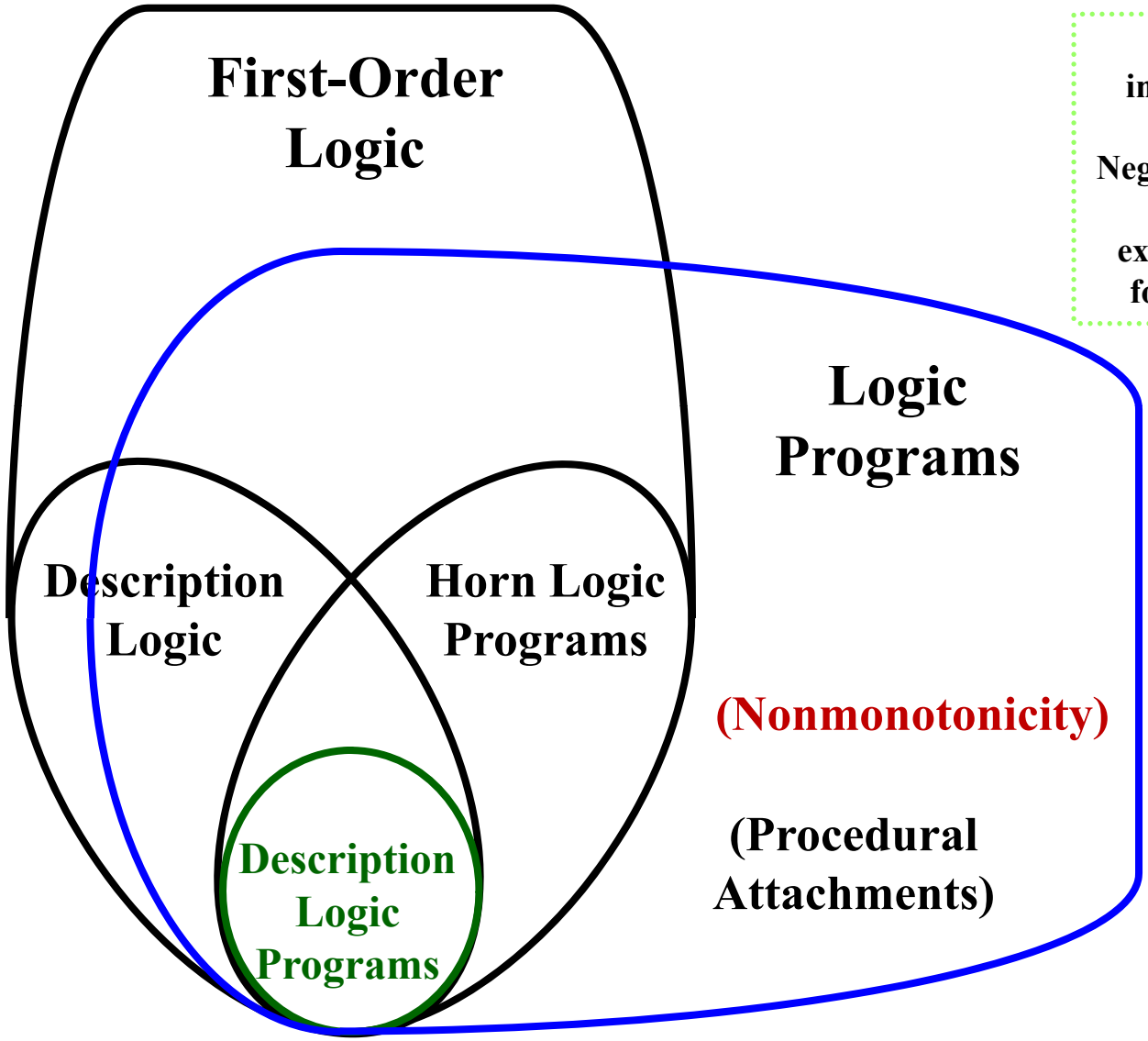
Knowledge Representation: What's the Game?

- Expressiveness: useful, natural, complex enough
- Reasoning algorithms
- Syntax: encoding data format -- here, in XML
- Semantics: principles of sanctioned inference, independent of reasoning algorithms
- Computational Tractability (esp. worst-case): scale up in a manner qualitatively similar to relational databases: computation cycles go up as a polynomial function of input size

Overview of Logic Knowledge Representation (KR) and Markup Standards

- First Order Logic (FOL). Also called "classical logic", as is HOL (below).
  - Standards efforts:
    - ISO Common Logic (CL); FOL RuleML
  - Restriction: Horn FOL
  - Restriction: Description Logic (DL) – overlaps with Horn
    - Standards: W3C OWL DL; W3C RDF Schema (expressive subset)
  - Extension: Higher Order Logic (HOL)
    - Hilog = higher order syntactically, but reducible to first order
- Logic Programs (LP)
  - (Here: in the *declarative* sense.)
  - Standard (Candidate Recommendation): W3C Rule Interchange Format (RIF)
  - Standard designs for additional expressiveness: RuleML / SWSL / SILK
  - Extension features: Hilog; also:
    - Nonmonotonicity: Negation, Defaults (cf. Courteous)
    - Procedural attachments for external queries, events, actions
  - Restriction: Horn LP
  - Restriction: Description Logic Programs (DLP) – overlaps with DL

Venn Diagram: Expressive Overlaps among KRs



NB: Nonmon LP, including Courteous, relies on Default Negation as fundamental underlying KR expressive mechanism for nonmonotonicity

Description Logic: KR Expressiveness, in brief

- Restriction of First Order Logic (FOL)
  - Strongest restriction is on the patterns of variable appearances
    - Cannot represent many kinds of chaining (joins) among predicates
  - No logical functions
- Allows:
  - Class predicates of arity 1
  - Property predicates of arity 2
  - Membership axioms: foo instanceOf BarClass
  - Inclusion axioms between classes (possibly complex)
    - C1 subclassOf C2
    - I.e., x instanceOf C1 $\Rightarrow$ x instanceOf C2
  - Complex class expressions, e.g.
    - Electrical device that has two speakers and a 120V or 220V power supply
  - Indirectly can represent n-ary predicates
- Good for representing:
  - Many kinds of ontological schemas, including taxonomies
  - Taxonomic/category subsumptions (with strict inheritance)
  - Some kinds of categorization/classification and configuration tasks

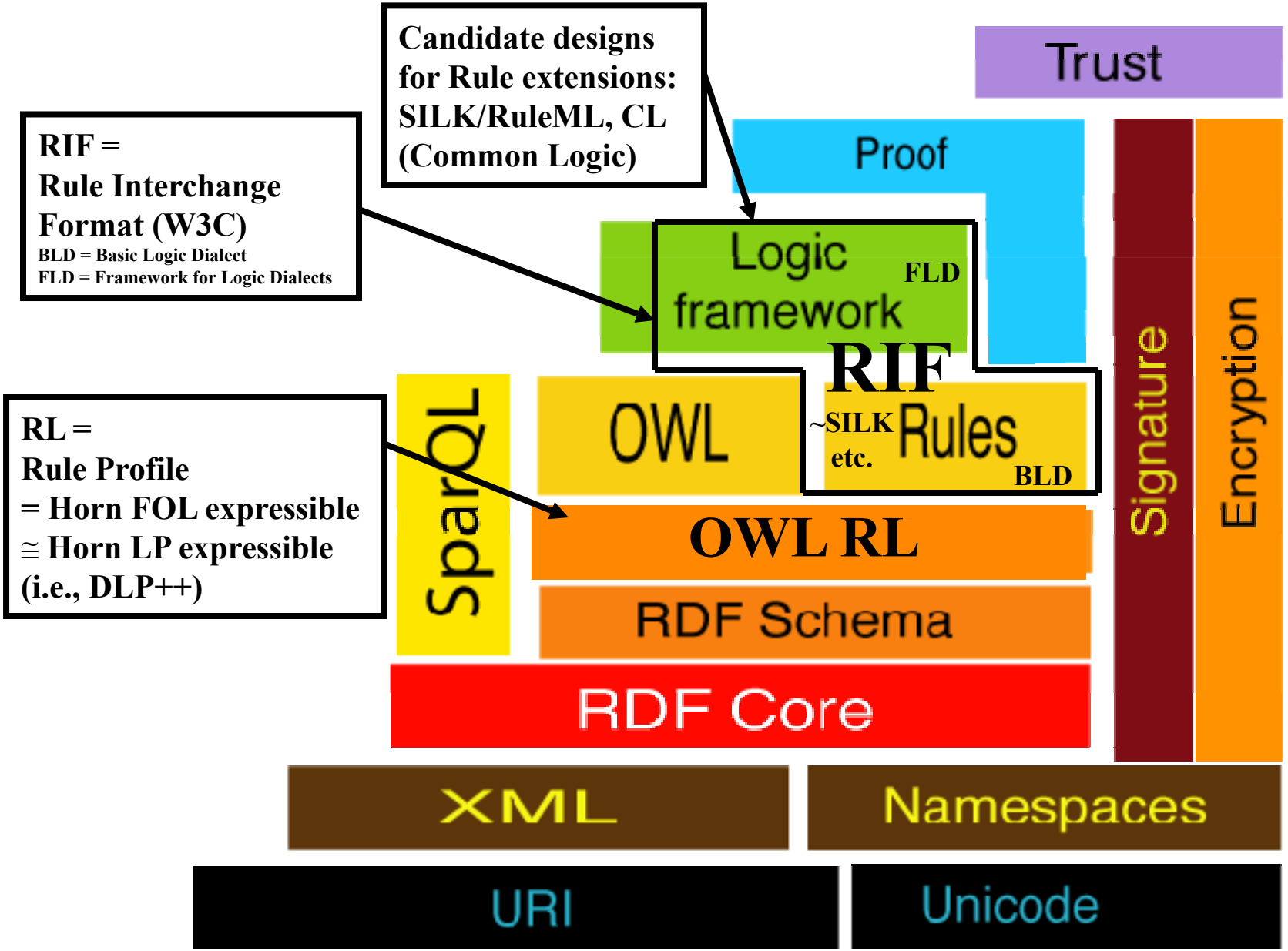
Summary of Computational Complexity of KRs

- *For task of inferencing, i.e., answering a given query.*
  - *Tractable = time is polynomial in n, worst-case; n = |premises|*
- First Order Logic (FOL)
  - Intractable for Propositional (co-NP-complete)
  - Undecidable in general case
  - Decidable but intractable for Description Logic
- Logic Programs (LP) with extensions for negation, defaults, Hilog, frames, attached procedures, …
  - Tractable for broad cases; same as Horn
    - $O(n^2)$ for Propositional with negation and defaults
    - Complexity qualitatively similar to Relational DBs
    - Truly Web-scaleable, therefore
  - Undecidable in general (cause: infinite recursion through functions)

More on Computational Complexity of LP

- O(n) for propositional Horn. (Ditto in FOL.)
- O(n·m) for propositional with negation (well-founded), where m = # atoms ($m \leq n$)
  - Defaults add no increase in the complexity bound (reducible linearly to NAF)
- Typically-met restrictions:
  - Constant-bounded number of distinct variables per rule (VB restriction)
    - In DL form of DLP, VB ≡ constant-bounded number of distinct DL quantifiers (incl. min/max cardinality) in class descriptions per inclusion axiom
  - Time per attached procedure call is tractable (AT restriction)
- Most feature extensions can be added to LP without affecting tractability
- Key restriction to ensure tractability (or decidability) is to:
  - Avoid blow-up from recursion through logical functions (of arity > 0)
    - ⇒ Keep the relevant set of ground atoms tractable (or finite)
    - Here, recursion means dependency cycles among rules
  - E.g., function-free is a simple sufficient condition
    - Then # of ground atoms = $O(n^{v+1})$, where v is the bound in VB
  - *More research on detailed theory and algorithms is needed, however*

Updated: 10-2009 Semantic Web "Stack"



Modified from slide by W3C (just added annotation)

KR View of Semantic Web related standards

Hazy wrt Standardization: more Framework

– Uncertainty (probabilistic, fuzzy); Provenance (proof, trust)

Logical Framework standards/designs: RIF-FLD, RuleML, SILK

LP (Logic Programs)

- **Umbrella standards/designs**
  - SILK
  - RuleML-LP
- **Database Query Standards***
  - SQL
  - SPARQL
  - XQuery
- **Business Rules Families***
  - Production
    - RIF-PRD
  - ECA (Event-Condition-Action)
  - Prolog

FOL (First Order Logic)

- **Umbrella standards/designs:**
  - CL (ISO Common Logic)
  - RuleML-FOL
- **Semantic/Web Standards (other)**
  - RDF
  - RDFS (Schema)
  - OWL RL (Rule Profile)
  - RIF-BLD (Basic Logic Dialect)
    - (and SWRL)
  - OWL DL (Description Logic)
  - OWL Full
  - SBVR (OMG Semantic Business Vocabulary and Rules)

*Via KR mapping to LP, maybe with restrictions

KR View of Semantic Web related standards

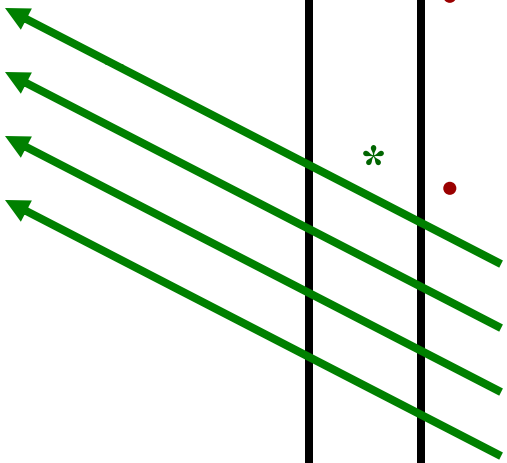
Hazy wrt Standardization: more Framework

– Uncertainty (probabilistic, fuzzy); Provenance (proof, trust)

Logical Framework standards/designs: RIF-FLD, RuleML, SILK

LP

- **Horn**
- **Rest**



FOL

- **Umbrella standards/designs:**
  - CL (ISO Common Logic)
  - RuleML-FOL
- **Semantic/Web Standards (other)**
  - RDF
  - RDFS (Schema)
  - OWL RL (Rule Profile)
  - RIF-BLD (Basic Logic Dialect)
    - (and SWRL)
  - OWL DL (Description Logic)
  - OWL Full
  - SBVR (OMG Semantic Business Vocabulary and Rules)

*Via KR mapping to LP (sound, nearly complete)

KR View of Semantic Web related standards

Hazy wrt Standardization: more Framework

– Uncertainty (probabilistic, fuzzy); Provenance (proof, trust)

Logical Framework standards/designs: RIF-FLD, RuleML, SILK

LP

- **Umbrella standards/designs**
  - SILK

FOL

*

Sound, but incomplete

➢ lack disjunctiveness

(no reasoning-by-cases)

*Via KR mapping to LP (hypermonotonic)

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Horn FOL

- The Horn subset of FOL is defined relative to clausal form of FOL
- A Horn clause is one in which there is at most one positive literal.

It takes one of the two forms:

1. $H \vee \neg B1 \vee \ldots \vee \neg Bm$. A.k.a. a definite clause / rule
   - Fact H . is special case of rule (H ground, m=0)
2. $\neg B1 \vee \ldots \vee \neg Bm$. A.k.a. an integrity constraint

where $m \geq 0$, H and Bi's are atoms. (An atom = pred(term_1,…,term_k) where pred has arity k, and functions may appear in the terms.)

- A definite clause (1.) can be written equivalently as an implication:
  - Rule := $H \Leftarrow B1 \wedge \ldots \wedge Bm$. where $m \geq 0$, H and Bi's are atoms

    head if body ;
- An integrity constraint (2.) can likewise be written as:
  - $\bot \Leftarrow B1 \wedge \ldots \wedge Bm$. A.k.a. empty-head rule ($\bot$ is often omitted).

For refutation theorem-proving, represent a negated goal as (2.).

Horn LP Syntax and Semantics

- Horn LP syntax is similar to implication form of Horn FOL
  - The implication connective's semantics are a bit weaker however. We will write it as ← (or as :-) instead of ⇐.
  - Declarative LP with model-theoretic semantics
  - Same for forward-direction ("derivation" / "bottom-up") and backward-direction ("query" / "top-down") inferencing
  - Model M(P) = a set of (concluded) ground atoms
    - Where P = the set of premise rules
- Semantics is defined via the least fixed point of an operator T_P. T_P outputs conclusions that are immediately derivable (through some rule in P) from an input set of intermediate conclusions I_j.
  - $I_{j+1} = T_P(I_j)$; $I_0 = \varnothing$ (empty set)
    - I_{j+1} = {all head atoms of rules whose bodies are satisfied by I_j}
  - M(P) = LeastFixedPoint(T_P) ; where LFP = the I_m such that $I_{m+1} = I_m$
  - Simple algorithm: DO {run each rule once} UNTIL {quiescence}

Example of Horn LP vs. Horn FOL

- Let P be:
  - DangerousTo(?x,?y) ← PredatorAnimal(?x) ∧ Human(?y);
  - PredatorAnimal(?x) ← Lion(?x);
  - Lion(Simba);
  - Human(Joey);
- I1 = {Lion(Simba), Human(Joey)}
- I2 = {PredatorAnimal(Simba),Lion(Simba), Human(Joey)}
- I3 = {DangerousTo(Simba,Joey), PredatorAnimal(Simba),Lion(Simba), Human(Joey)}
- I4 = I3. Thus M(P) = I3.

- Let P' be the Horn FOL rulebase version of P above, where ⇐ replaces ←.
- Then the ground atomic conclusions of P' are exactly those in M(P) above.
- P' also entails various non-ground-atom conclusions, including:
  1. Non-unit derived clauses, e.g., DangerousTo(Simba,?y) ⇐ Human(?y).
  2. All tautologies of FOL, e.g., Human(?z) ∨ ¬Human(?z).
  3. Combinations of (1.) and (2.), e.g., ¬Human(?y) ⇐ ¬DangerousTo(Simba,?y).

Horn LP Compared to Horn FOL

- Fundamental Theorem connects Horn LP to Horn FOL:
  - M(P) = {all ground atoms entailed by P in Horn FOL}

- Horn FOL has additional non-ground-atom conclusions, notably:
  - non-unit derived clauses; tautologies
- Can thus view Horn LP as the f-weakening of Horn FOL.
  - "f-" here stands for "fact-form conclusions only"
  - A restriction on form of conclusions (not of premises).
- Horn LP – differences from Horn FOL:
  - Conclusions Conc(P) = essentially a set of ground atoms.
    - Can extend to permit more complex-form queries/conclusions.
  - Consider Herbrand models only, *in typical formulation and usage.*
    - P can then be replaced equivalently by {all ground instantiations of each rule in P}
    - But can extend to permit: extra unnamed individuals, beyond Herbrand universe
  - Rule has non-empty head, *in typical formulation and usage.*
    - Can extend to detect violation of integrity constraints

The "Spirit" of LP

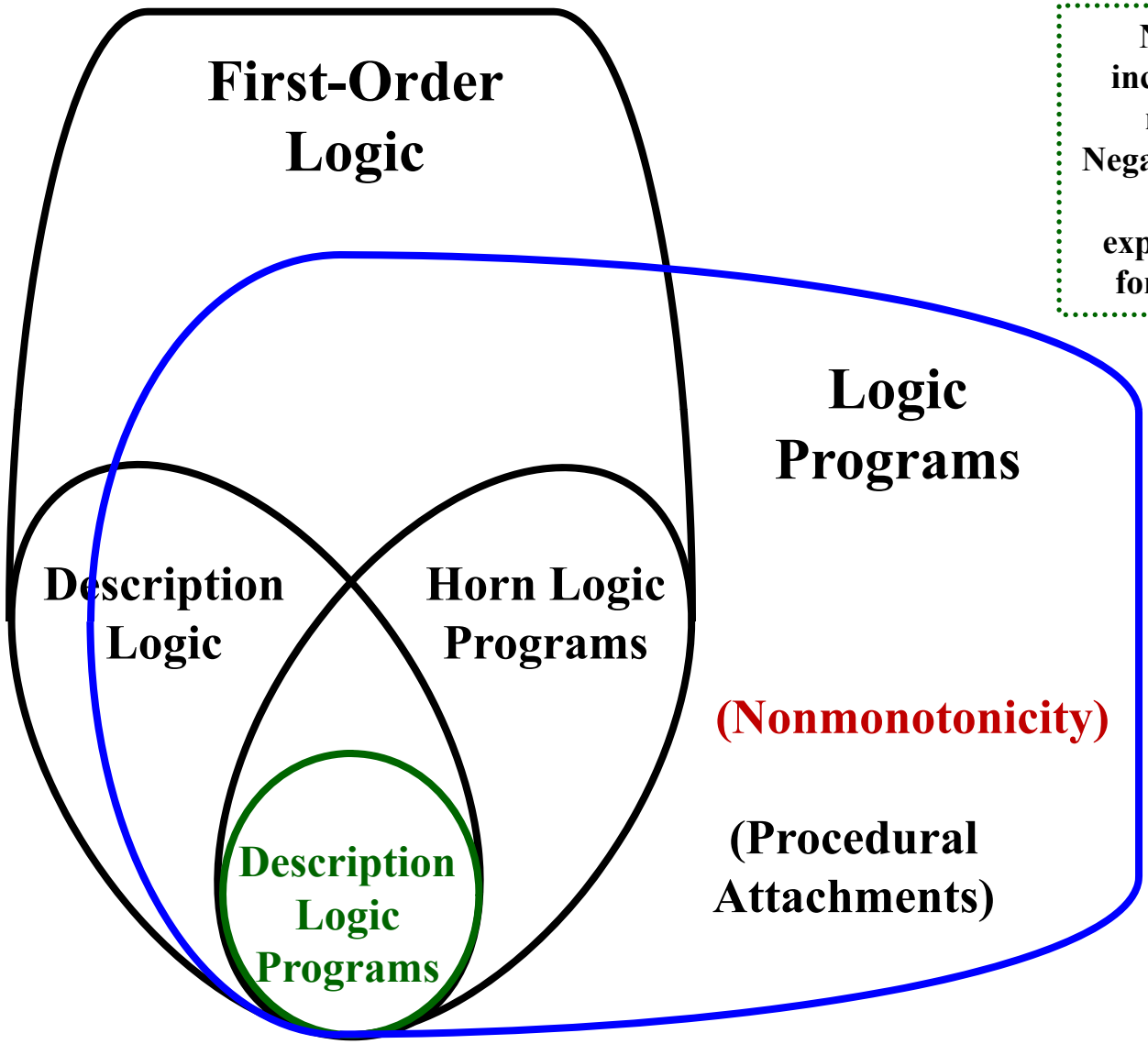
The following summarizes the "spirit" of how LP differs from FOL:

- "Avoid Disjunction"
  - Avoid disjunctions of positive literals as expressions
    - In premises, intermediate conclusions, final conclusions
    - (conclude (A or B)) only if ((conclude A) or (conclude B))
  - Permitting such disjunctions creates exponential blowup
    - In propositional FOL: 3-SAT is NP-hard
    - In the leading proposed approaches that expressively add disjunction to LP with negation, e.g., propositional Answer Set Programs
  - No "reasoning by cases", therefore
- "Stay Grounded"
  - Avoid (irreducibly) non-ground conclusions

LP, unlike FOL, is straightforwardly extensible, therefore, to:

- Nonmonotonicity – defaults, incl. NAF
- Procedural attachments, esp. external actions

Venn Diagram: Expressive Overlaps among KRs



NB: Nonmon LP, including Courteous, relies on Default Negation as fundamental underlying KR expressive mechanism for nonmonotonicity

Requirements Analysis for Logical Functions

- Function-free is a commonly adopted restriction in practical LP/Web rules today
  - DB query languages: SQL, SPARQL, XQuery
  - RDFS
  - Production rules, and similar Event-Condition-Action rules
  - OWL

- BUT functions are often needed for Web (and other) applications. Uses include:
  - Hilog and reification – higher-order syntax
    - For meta- reasoning, e.g., in knowledge exchange or introspection
      - Ontology mappings, provenance, KB translation/import, multi-agent belief, context
      - KR macros, modals, reasoning control, KB modularization, navigation in KA
      - Meta-data is important on the Web
  - Skolemization – to represent existential quantifiers
    - E.g., RDF blank nodes
  - Convenient naming abstraction, generally
    - steering_wheel(my_car)

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Frame Syntax and F(rame)-Logic

- An object-oriented first-order logic
- Extends predicate logic with
  - Objects with complex internal structure
  - Class hierarchies and inheritance
  - Typing
  - Encapsulation
- A basis for object-oriented logic programming and knowledge representation

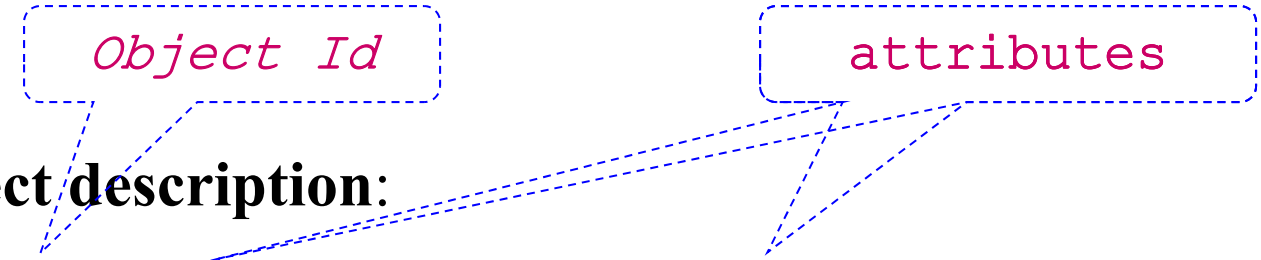
$$\frac{\text{O-O programming}}{\text{F-logic}} = \frac{\text{Relational programming}}{\text{Predicate calculus}}$$

- Background:
  - Basic theory: [Kifer & Lausen SIGMOD-89], [Kifer, Lausen, Wu JACM-95]
  - Path expression syntax: [Frohn, Lausen, Uphoff VLDB-84]
  - Semantics for non-monotonic inheritance: [Yang & Kifer, ODBASE 2002]
  - Meta-programming + other extensions: [Yang & Kifer, Journal on Data Semantics 2003]

Major F-logic Based Languages

- *FLORA-2* - an open source system developed at Stony Brook U.
- *Ontobroker* – commercial system from Ontoprise.de
- *WSMO* (Web Service Modeling Ontology) – a large EU project that developed an F-logic based language for Semantic Web Services, *WSML-Rule*
- *SWSI* (Semantic Web Services Initiative) – an international group that proposed an F-logic based language *SWSL-Rules* (also for Semantic Web Services)
- RuleML supports it as an included extension, developed in collaboration with SWSI
- *TRIPLE* – an open source system for querying RDF
- *SILK*

F-logic Examples



Object description:

John[*name* –> 'John Doe' and *phones* -> {6313214567, 6313214566},
children -> {Bob, Mary}]

Mary[*name* –>'Mary Doe', *phones* -> {2121234567, 5129297945},
children -> {Anne, Alice}]

Structure can be nested:

Sally[*spouse* -> John[*address* -> '123 Main St.']]

F-Logic Examples (cont.'d)

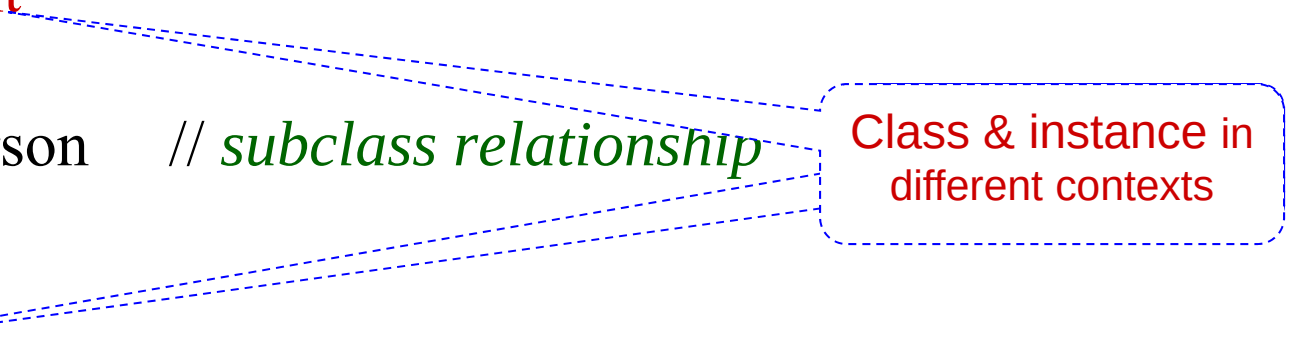
ISA hierarchy:

John # Person // *class membership*
Mary # Person
Alice # Student

Student ## Person // *subclass relationship*

Student # EntityType

Person # EntityType



F-Logic Examples (cont.'d)

"Methods": like attributes, but take arguments

?S[*professor*(?Course) –> ?Prof] **: -**

?S:student[*took*(?Semester) –>?Course[*taught*(?Semester)–> ?Prof]];

- *professor*, *took*, *taught* – 1-argument methods
- object attributes can be viewed as 0-ary methods

Queries :

?– Alice[*professor*(?Course) –> ?P], ?Course # ComputerScienceCourse;

Alice's CS professors.

F-Logic Examples (cont.'d)

Browsing the IsA hierarchy:

```
?- John # ?X ;        // all classes of which John is a member
?- Student ## ?Y;     // all superclasses of class student
```

Defining a virtual class:

*Rule defining a **virtual** class of red cars*

```
?X # RedCar  :-  ?X # Car  and  ?X[color -> red];
```

Complex meta-query about schema:

Rule defining a method that returns attributes whose range is class ?Class

```
?O[attributesOf(?Class) -> ?Attr]  :-
        ?O[?Attr ->?Value]  and  ?Value # ?Class;
```

Remark: Semantics for HiLog & F-Logic

- The F-logic and HiLog semantics & proof theory
  - Generalize terms and literals
  - Not limited to rules/LP
  - Apply also to classical logic (FOL) – and other logics
  - Sound & complete

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

HiLog

- A higher-order extension of predicate logic, which has a tractable first-order syntax
  - Allows certain forms of logically clean, yet tractable, meta-programming
  - Syntactically appears to be higher-order, but semantically is first-order and tractable
- Appears promising for OWL Full and its use of RDF [Kifer; Hayes]

- Implemented in FLORA-2 and SILK
  - Also partially exists in XSB, Common Logic, others
- [Chen, Kifer, Warren, "HiLog: A Foundation for Higher-Order Logic Programming", J. of Logic Programming, 1993]

Examples of HiLog

Variables over predicates and function symbols:

p(?X,?Y) **: -** ?X(a,?Z) and ?Y(?Z(b));

Variables over atomic formulas (*reification*):

p(q(a)).

r(?X) **: -** p(?X) and ?X;

A use of HiLog in FLORA-2 and SILK **(**e.g., even more complex schema query**):**

?Obj[*unaryMethods*(?Class) –> ?*Method*] **: -**

?Obj[?*Method*(?Arg) –> ?Val] and ?Val # ?Class;

***Meta-variable**: ranges over unary method names*

Reification

- Blending HiLog with F-logic also allows **reification** – making objects out of formulas:

john[believes -> ${mary[likes -> bob]}]

- Introduced in [Yang & Kifer, ODBASE 2002]
- Rules can also be reified

Object made out of the formula mary[likes -> bob]

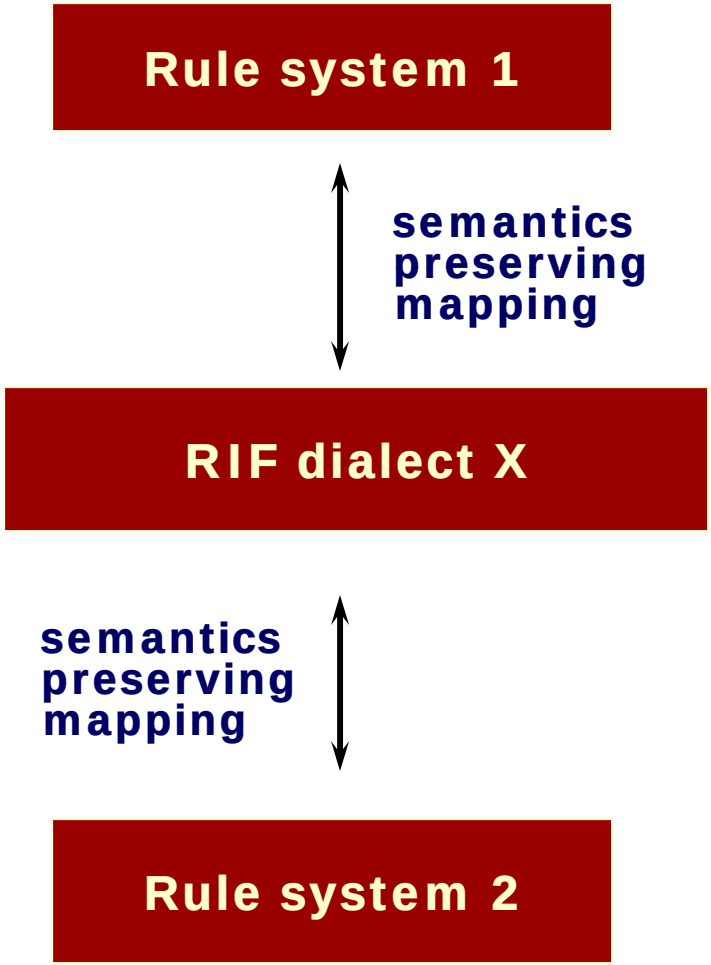
Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

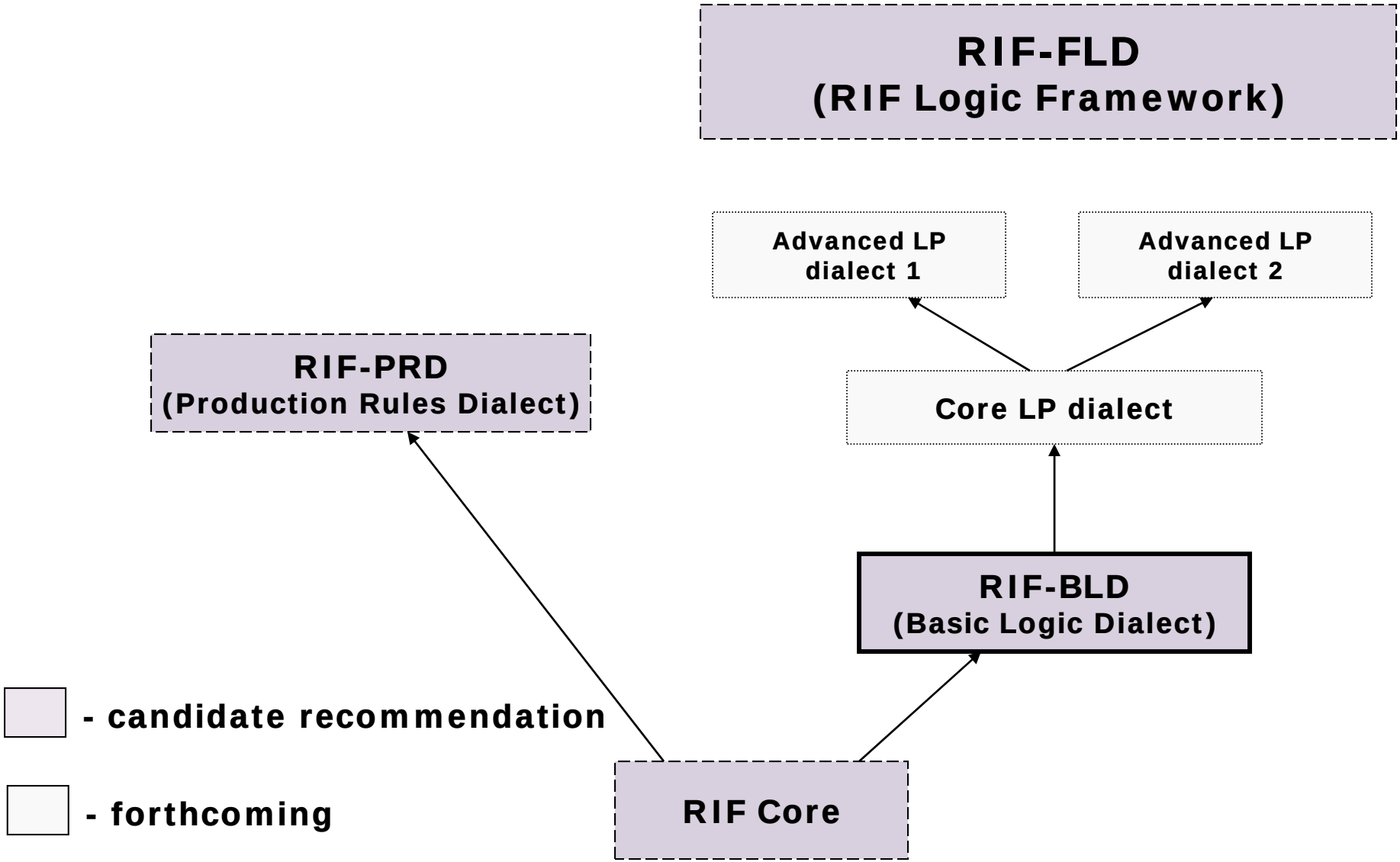
What is RIF?

- A collection of *dialects* (rigorously defined rule languages)
- Intended to facilitate rule sharing and exchange
- XML is medium of exchange
- Dialect consistency

  Sharing of RIF machinery:
  - XML
  - syntactic elements
  - elements of semantics



Current State of RIF



The Basic Logic Dialect (BLD)

- Basically Horn rules (no negation) plus
  - Frames
  - Predicates/functions with named arguments
  - Equality both in rule premises and conclusions
- Web-ized
  - XML data types
  - IRIs throughout
- Semantic Web integration
  - Can import RDF and OWL
  - BLD + OWL ⊃ SWRL

RIF-CORE and RIF-PRD

- RIF-Core is defined by restricting BLD
  - No function symbols
  - Equality only in rule body
  - Decidable (module the built-ins)
- RIF-PRD – a separate branch of dialects
  - Contains RIF-Core
  - Procedural, not logic-based
  - Shares much of the notational machinery with BLD

Why RIF Framework (RIF-FLD)?

- Too hard to define dialects from scratch
  - RIF-BLD is just a tad more complex than Horn rules, but requires more than 30 pages of dense text
- Instead: define dialects by *specializing* from another dialect
  - RIF-BLD can be specified in < 3pp in this way
- A "*super-dialect*" is needed to ensure that all dialects use the same set of concepts and constructs
- RIF Framework is intended to be just such a super-dialect
- The forthcoming LP dialects will be defined by specializing RIF-FLD
- Even RIF-BLD was initially defined by specialization from RIF-FLD

RIF-FLD Features

- Not a completely specified logic by itself : dialects are required to specify a number of parameters (to specialize)
- Highly extensible syntax and semantics
- Supports most forms of non-monotonic reasoning (e.g., various forms of negation, defaults)
- … And classical logic

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Concept of Logical Monotonicity

- A KR S is said to be <u>logically monotonic</u> when in it:

$$P1 \subseteq P2 \quad \Rightarrow \quad Conc(P1,S) \subseteq Conc(P2,S)$$

- Where P1, P2 are each a set of premises in S
- I.e., whenever one adds to the set of premises, the set of conclusions non-strictly grows (one does not retract conclusions).

- *Monotonicity is good for pure mathematics.*
  - *"Proving a theorem means never having to say you are sorry."*

Nonmonotonicity – its Pragmatic Motivations

- Pragmatic reasoning is, in general, nonmonotonic
  - E.g., policies for taking actions, exception handling, legal argumentation, Bayesian/statistical/inductive, etc.
  - Monotonic is a special case – simpler in some regards
- Most commercially important rule systems/applications use nonmon
  - A basic expressive construct is ubiquitous there:
    - Default Negation a.k.a. Negation-As-Failure (NAF)
      - BUT with varying semantics – often not fully declarative cf. LP
        - Primarily due to historical hangovers and lack of familiarity with modern algorithms
  - Another expressive construct, almost as ubiquitous there, is:
    - Priorities between rules
- Such nonmonotonicity enables:
  - Modularity and locality in revision/updating/merging

Default Negation: Intro

- Default negation is the most common form of negation in commercially important rule and knowledge-based systems.
- Concept/Intuition for ~q ; ~ stands for default negation
  - q is not derivable from the available premise info
  - fail to believe q
  - … but might also not believe q to be false
  - A.k.a. "*weak*" negation, or NAF.
- Contrast with: ¬q ; ¬ stands for strong negation
  - q is believed to be false
  - A.k.a. "*classical*" negation

LP with Negation As Failure

- Normal LP (NLP), a.k.a. *Ordinary* LP (a.k.a. "general" LP)
  - Adds NAF to Horn LP
- Syntax: Rule generalized to permit NAF'd body literals:
  - $H \leftarrow B_1 \wedge \ \ldots \wedge B_k \wedge \sim B_{k+1} \wedge \ldots \wedge \sim B_m$;

where $m \geq 0$, H and Bi's are atoms

- Semantics has subtleties for the fully general case.
  - Difficulty is interaction of NAF with "recursion", i.e., cyclic dependencies (thru the rules) of predicates/atoms.
  - Lots of theory developed during 1984-1994
  - Well-understood theoretically since mid-1990's

Semantics for LP with Default Negation

- For fully general case, there are two major alternative semantics
- Both agree for a broad restricted case: *stratified* ordinary LP
- *Well Founded* Semantics (WFS): popular, widely used
  - Tractable for the propositional case. Often linear, worst-case quadratic.
  - Major commercial focus. E.g., XSB, Ontobroker.
  - Employs a 3rd truth value u ("undefined"), when non-stratified ("unstratified")
  - Definition uses iterated minimality: Horn-case then close-off; repeat til done.
  - Major limitation: cannot reason by cases
- *Answer Set Programs (ASP)*: popular as research topic
  - Enables a limited kind of disjunction in heads, conclusions
  - Good for combinatorial KR problems requiring nonmonotonicity
  - Only 2 truth values ⇒ sometimes ill-defined: no set of conclusions
    - Generalizes earlier "*stable model semantics*"
  - Can reason by cases! ⇒ Intractable for propositional case

Basic Example of LP with NAF

- RB1: (NB: this example is purely fictional.)
  - price(Amazon, Sony5401, ?day, ?cust, 49.99)
    ← inUSA(?cust) ∧ inMonth(?day, 2004_10) ∧ ~onSale(?day);
  - price(Amazon, Sony5401, ?day, ?cust, 39.99)
    ← inUSA(?cust) ∧ inMonth(?day, 2004_10) ∧ onSale(?day);
  - inMonth(2004_10_12, 2004_10);
  - inMonth(2004_10_30, 2004_10);
  - inUSA(BarbaraJones);
  - inUSA(SalimBirza);
  - onSale(2004_10_30);
- RB1 entails: (among other conclusions)
  1. Price(Amazon, Sony5401, 2004_10_12, BarbaraJones, 49.99)
  2. Price(Amazon, Sony5401, 2004_10_30, SalimBirza, 39.99)
- RB2 = RB1 updated to add: onSale(2004_10_12);
- RB2 does NOT entail (1.). Instead (nonmonotonically) it entails:
  3. Price(Amazon, Sony5401, 2004_10_12, BarbaraJones, 39.99)

Brief Examples of Non-Stratified Normal LP

- RB3:
  - a;
  - c ← a ∧ ~b;
  - p ← ~p;
- Well Founded Semantics (WFS) for RB3 entails conclusions {a,c}.
  p is not entailed. p has "*undefined*" (u) truth value (in 3-valued logic).
- ASP Semantics for RB3: ill defined; there *is no* set of conclusions.
  - *(NOT there is a set of conclusions that is empty.)*

- RB4:
  - a;
  - c ← a ∧ ~b;
  - p ← ~q;
  - q ← ~p;
- WFS for RB4 entails conclusions {a,c}. p,q have truth value u.
- ASP Semantics for RB4 results in two alternative conclusion sets: {a,c,p} and {a,c,q}. Note their intersection {a,c} is the same as the WFS conclusions.

Computing Well Founded Semantics for LP

- Always exactly one set of conclusions (entailed ground atoms)
- Tractable to compute all conclusions, for broad cases:
  - $O(n^2)$ for Propositional case of Normal LP
  - $O(n^{2v+2})$ for VB Datalog case (v = max # vars per rule)
  - NAF only moderately increases computational complexity compared to Horn (frequently linear, at worst quadratic)
- *By contrast, for Stable Semantics:*
  - *There may be zero, or one, or a few, or very many alternative conclusion sets*
  - *Intractable even for Propositional case*
- Proof procedures are known that handle the non-stratified general case
  - backward-direction: notably, SLS-resolution
    - Fairly mature wrt performance, e.g., tabling refinements
  - forward-direction
    - Reuse insights from backward-direction. Restrict to function-free.
    - Fairly mature wrt performance. Room to improve: esp. for updating.

Some Implementations of Unstratified LP

- Well Founded:
  - XSB (research / commercial; open source)
  - Ontoprise (commercial)
  - Intellidimension (commercial)
  - SweetRules (research; open source)
  - SILK (research / commercial)

- Answer Set Programs:
  - Smodels (research)
  - DLV (research / commercial)
  - Clasp (research)

- *There are a number of others, esp. research*

Negation-As-Failure Implementations: Current Limitations in Many Systems

- Practice in Prolog and other currently commercially important (CCI) rule systems is often "sloppy" (incomplete / cut-corners) relative to canonical semantics for NAF
  - in cases of recursive rules, WFS algorithms required are more complex
  - ongoing diffusion of WFS theory & algorithms, beginning in Prologs

- Current implemented OLP inferencing systems often do not handle the fully general case in a semantically clean and complete fashion.
  - Many are still based on older algorithms that preceded WFS theory/algorithms

- Other CCI rule systems' implementations of NAF are often "ad hoc"
  - Lacked understanding/attention to semantics, when developed

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Ubiquity of Priorities
in Commercially Important Rules -- and Ontologies

- Updating in relational databases
  - more recent fact *overrides* less recent fact
- Static rule ordering in Prolog
  - rule earlier in file *overrides* rule later in file
- Dynamic rule ordering in production rule systems (OPS5)
  - "meta-"rules can specify agenda of rule-firing sequence
- Event-Condition-Action rule systems rule ordering
  - often static or dynamic, in manner above
- Exceptions in default inheritance in object-oriented/frame systems
  - subclass's property value *overrides* superclass's property value, e.g., method redefinitions
- All lack Declarative KR Semantics

Defeasible Reasoning

- **Rules can be true by default but may be defeated**
  - A form of commonsense reasoning
- **Application domains:**
  - policies, regulations, and law
  - actions, change, and process causality
  - Web services
  - inductive/scientific learning
  - natural language understanding
  - …
- **Existing approaches:**
  - Courteous Logic Programs (Grosof , 1997)
    - The main approach used commercially (IBM Common Rules, 1999)
  - Defeasible logic (Nute, 1994) *[similar to Courteous LP]*
  - "Prioritized defaults" (Gelfond & Son, 1997)
  - Preferred answer sets (Brewka & Eiter, 2000)
  - Compiling preferences (Delgrande et al., 2003)
  - …

Semantical KR Approaches to Prioritized LP

The currently most important for Semantic Web are:

1. Courteous LP
   - KR extension to Ordinary LP
   - In RuleML, since 2001
   - Commercially implemented and applied
     - IBM CommonRules, since 1999
2. Defeasible Logic
   - Closely related to Courteous LP
     - Less general wrt typical patterns of prioritized conflict handling needed in e-business applications
     - In progress: theoretical unification with Courteous LP

Courteous LP: the What

- **Updating/merging of rule sets: is crucial, often generates conflict.**
- **Courteous LP's feature prioritized handling of conflicts.**
- **Specify scope of conflict via a set of exclusion constraints**
  - Each is a preventive spirit integrity constraint on a set of competing literals
    - It says that not all of the competing literals can be entailed as true.
    - opposes(p, q) ≈ (⊥ :- p and q) // Case of 2 competing literals
  - opposes(discount(?product,"5%"), discount(?product,"10%"));
  - opposes(loyalCustomer(?cust,?store), premiereCustomer(?cust,?store));
- **Permit strong negation of atoms: (NB: a.k.a. (quasi-) "classical" negation.)**
    - ¬p means p has truth value *false* . ¬p is also written as: neg p in ASCII.
    - implicitly, for every atom p: opposes(p, ¬p);
- Priorities between rules: partially-ordered.
  - Represent priorities via reserved predicate that compares rule labels:
    - overrides(rule1, rule2) means rule1 is higher-priority than rule2.
    - Each rule optionally has a rule label whose form is a functional term.
    - overrides can be reasoned about, just like any other predicate.

Priorities are available and useful

- Priority information is naturally available and useful. E.g.,
  - recency: higher priority for more recent updates.
  - specificity: higher priority for more specific cases (e.g., exceptional cases, sub-cases, inheritance).
  - authority: higher priority for more authoritative sources (e.g., legal regulations, organizational imperatives).
  - reliability: higher priority for more reliable sources (e.g., security certificates, via-delegation, assumptions, observational data).
  - closed world: lowest priority for catch-cases.

- Many practical rule systems employ priorities of some kind, often implicit. E.g.,
  - rule sequencing in Prolog and production rules.
    - Courteous LP subsumes this as a special case (totally-ordered priorities)
    - Also Courteous LP enables: merging, more flexible & principled treatment.

Courteous LP: Advantages

- Facilitate updating and merging, modularity and locality in specification.
- Expressive: strong negation, exclusions, partially-ordered prioritization, reasoning to infer prioritization.
- Guarantee consistent, unique set of conclusions.
  - **Exclusion is enforced**. E.g., never conclude discount is both 5% and that it is 10%, nor conclude both p and ¬p.
- Scaleable & Efficient: low computational overhead beyond ordinary LPs.
  - Tractable given reasonable restrictions (VB Datalog):
    - extra cost is equivalent to increasing v to (v+2) in Ordinary LP, worst-case.
  - By contrast, more expressive prioritized rule representations (e.g., Prioritized Default Logic) add NP-hard overhead.
- Modular software engineering:
  - Transform: CLP $\rightarrow \rightarrow$ OLP. Via simple "argumentation theory" approach.
    - Add-on to variety of OLP rule systems, with modest effort.

EECOMS Example of Conflicting Rules: Ordering Lead Time

- Vendor's rules that prescribe how buyer must place or modify an order:
- A) 14 days ahead if the buyer is a qualified customer.
- B) 30 days ahead if the ordered item is a minor part.
- C) 2 days ahead if the ordered item's item-type is backlogged at the vendor, the order is a modification to reduce the quantity of the item, and the buyer is a qualified customer.
- D) 45 days ahead if the buyer is a walk-in customer.

- Suppose more than one of the above applies to the current order? **Conflict!**
- Helpful Approach: **precedence** between the rules.
  - E.g., D is a catch-case: A > D , B > D , C > D
- Often only *partial* order of precedence is justified.
  - E.g., C > A , but no precedence wrt B vs. A, nor wrt C vs. B.

Ordering Lead Time Example in LP with Courteous Defaults

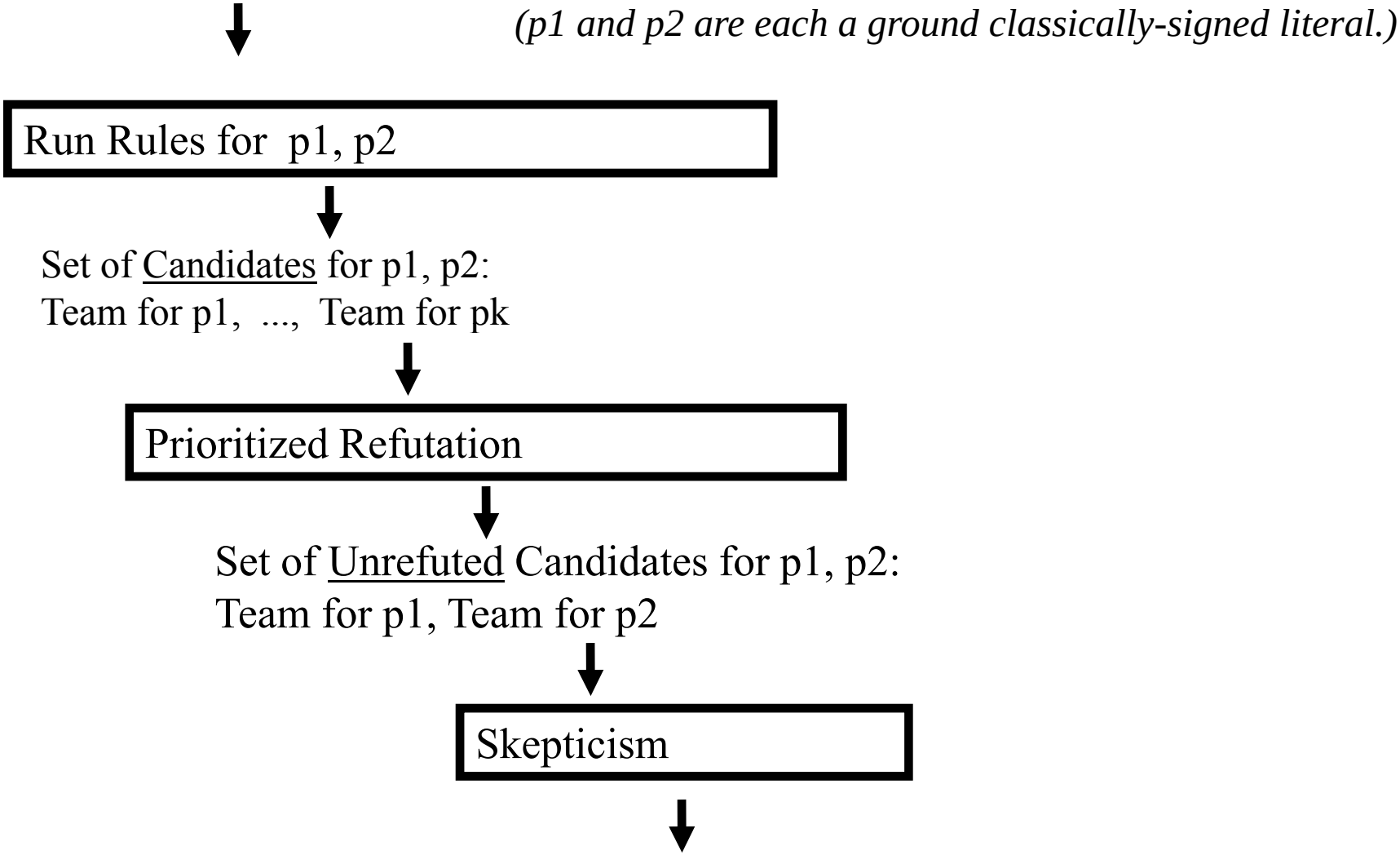
- @prefCust orderModifNotice(?Order,14days)
  ← preferredCustomerOf(?Buyer,SupplierCo) ∧
  purchaseOrder(?Order,?Buyer,SellerCo) ;
- @smallStuff orderModifNotice(?Order,30days)
  ← minorPart(?Buyer,?Seller,?Order) ∧
  purchaseOrder(?Order,?Buyer,SupplierCo) ;
- @reduceTight orderModifNotice(?Order,2days)
  ← preferredCustomerOf(?Buyer,SupplierCo) ∧
  orderModifType(?Order,reduce) ∧
  orderItemIsInBacklog(?Order) ∧
  purchaseOrder(?Order,?Buyer,SupplierCo) ;
- overrides(reduceTight , prefCust) ;
- opposes(orderModifNotice(?Order,?X), orderModifNotice(?Order,?Y)) :- ?X ≠?Y ;

- NB: Rule D, and prioritization about it, were omitted above for sake of brevity.

Courteous LP Semantics: ***Prioritized argumentation in an exclusion locale.***

Conclusions from exclusion-locales previous to this exclusion-locale {p1, p2}

(p1 and p2 are each a ground classically-signed literal.)



Argumentation Theories approach to Defaults in LP

- **Combines Courteous + Hilog, and generalizes**
- **New approach to defaults: "argumentation theories"**
  - Meta-rules, in the LP itself, that specify when rules ought to be defeated
  - [Wan, Grosof, Kifer, *et al.* ICLP-2009]
- **Extends straightforwardly to combine with other key features**
  - E.g., Frame syntax, external Actions
- **Significant other improvements on previous Courteous**
  - Eliminates a complex transformation
  - Much simpler to implement
    - 20-30 background rules instead of 1000's of lines of code
  - Much faster when updating the premises
  - More flexible control of edge-case behaviors
  - Much simpler to analyze theoretically

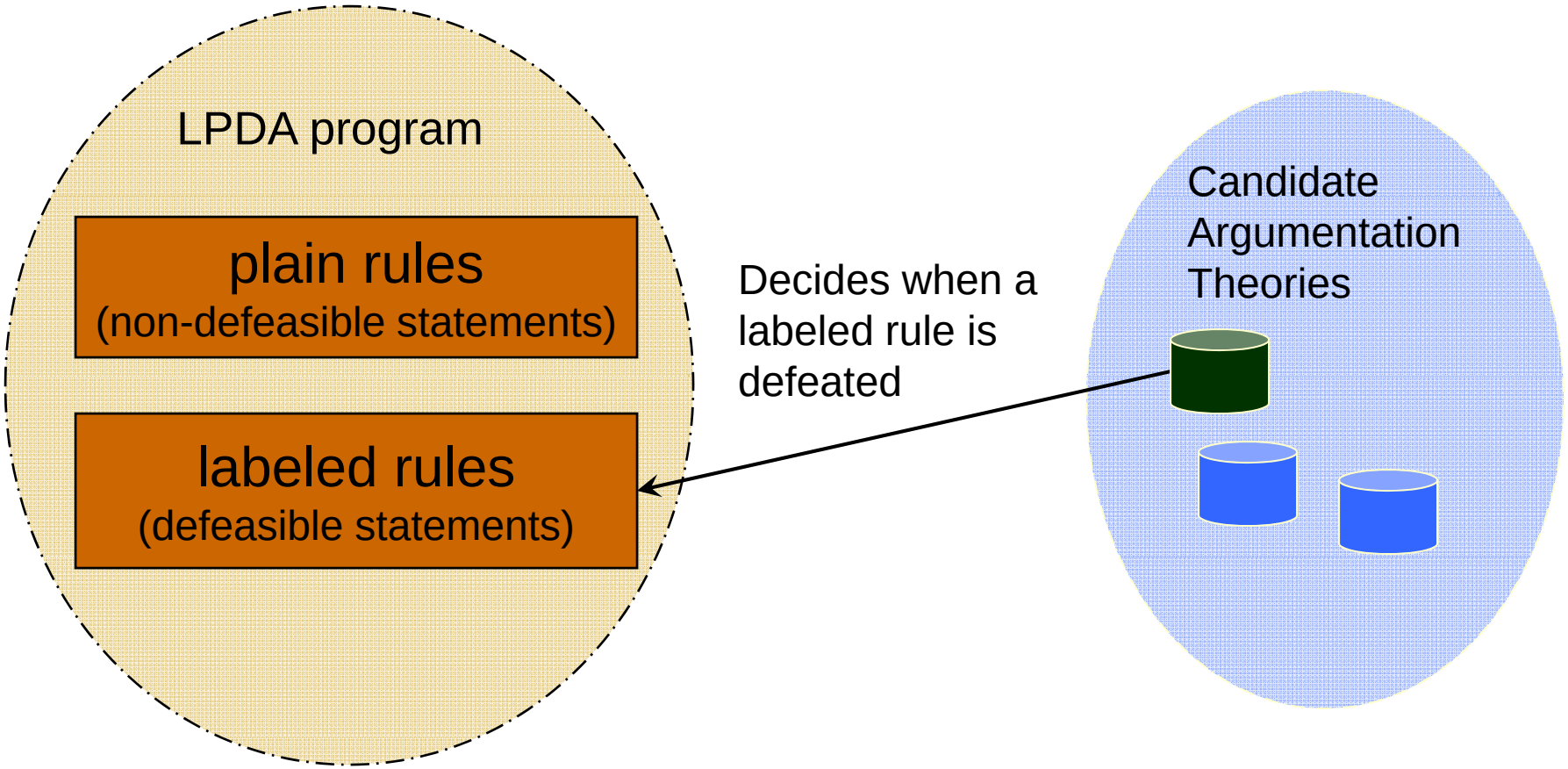
LPDA approach, continued

- **More Advantages**
  - 1st way to generalize defeasible LP, notably Courteous, to HiLog higher-order and F-Logic frames
  - Well-developed model theory, reducible to normal LP
  - Reducibility and well behavior results
  - Unifies almost all previous defeasible LP approaches
    - Each reformulated as an argumentation theory
  - Cleaner, more flexible and extensible semantics
    - Enables smooth and powerful integration of features
  - Leverages most previous LP algorithms & optimizations

- **Implemented** in SILK V1 via an extension of FLORA-2
  - Public release planned for approx. winter 2009-2010

LPDA Framework

- **L**ogic **P**rograms with **D**efaults and **A**rgumentation theories



Slide courtesy Hui Wan

Example – AT for Courteous ($\mathcal{AT}^{GCLP}$)

$defeated(?R) :- $defeats(?S, ?R).
$defeats(?R, ?S) :- $refutes(?R, ?S) or $rebuts(?R, ?S).

Prioritization (user specified)

$refutes(?R, ?S) :- $conflict(?R, ?S), overrides(?R, ?S).
$refuted(?R) :- $refutes(?R2, ?R).
$rebuts(?R, ?S) :- $conflict(?R, ?S),
not $refuted(?R), not $refuted(?S).

Default negation (NAF)

Meta predicates ("Reflection")

$candidate(?R) :- body(?R, ?B), call(?B).
$conflict(?R, ?S) :- $candidate(?R), $candidate(?S),
opposes(?R, ?S).
opposes(?R, ?S) :- opposes(?S, ?R).
opposes(?L1,?L2) :- head(?L1, ?H), head(?L2, neg ?H).

Exclusion (user specified)

Explicit negation

Slide courtesy Hui Wan

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Basic Hypermonotonic Mapping from Clausal FOL to/from NAF-Free Courteous LP

- **An FOL clause C:**

  L1 or L2 or … or Lk.

  is mapped to k directed clauses, one for each choice of head literal:

  L1 :- neg L2 and neg L3 and … and neg Lk.

  L2 :- neg L1 and neg L3 and … and neg Lk.

  …

  Lk :- neg L1 and neg L2 and … and neg Lk-1.

- **This is called the *omnidirectional ruleset* for C, a.k.a. the *omni***
- **Conversely, a naf-free Courteous LP rule is mapped to FOL as a material implication, thus clausal. (It is fairly easy to stick to naf-free.)**
- **A KR S *behaves hypermonotonically* == S is nonmonotonic and when its premises are viewed classically, then entailment in S is sound but incomplete w.r.t. classical**
  - Incompleteness is desirable when there is conflict
- **The mapping from definite Horn FOL to Horn LP is just a special case**
  - Exactly one directed clause has a positive head; just keep that one
  - E.g., in DLP, OWL RL, RIF BLD, many rule implementations

Examples of Basic Hypermonotonic mapping

- **/* SBVR Car rental: A driver ?p is Approved only if ?p has a Validated application.*/**
  - **/* FOL: */ forall ?p. Validated(?p) <== Approved(?p).**

  becomes the ff. omnidirectional ruleset in Hyper LP:
  - **neg Approved(?p) :- neg Validated(?p) . /* Exploit strong negation feature (neg). */**
  - **Validated(?p) :- Approved(?p).**

- **/* OWL 2 DL beyond RL: The classes Cat and Bird are disjoint. */**
  - **/* FOL */ forall ?x. neg (Cat(?x) and Bird(?x)).**

  becomes the ff. omnidirectional ruleset in Hyper LP:
  - **neg Cat(?x) :- Bird(?x).**
  - **neg Bird(?x) :- Cat(?x).**

- **/* Scheduling: Joe's meeting will be at 3pm or 4pm or 5pm today. */**
  - **/* FOL source: */ mtg(3p) or mtg(4p) or mtg(5p).**

  becomes the ff. omnidirectional ruleset in Hyper LP:
  - **mtg(5p) :- neg mtg(3p) and neg mtg(4p).**
  - **mtg(4p) :- neg mtg(3p) and neg mtg(5p).**
  - **mtg(3p) :- neg mtg(4p) and neg mtg(5p).**

Hypermon Mapping from FOL++ to LP

- **Covers unrestricted FOL clauses, plus skolemization, thus full FOL**
- **Can further add Frames and Hilog (and deontic etc. modals, esp. using Hilog)**
- **Thus can cover full OWL/RDF and Common Logic, most of SBVR**
- **Give up disjunction / reasoning by cases, so is weakened**
- **Greatly generalizes the approach of Description LP and OWL 2 RL**
- **Leverages Courteous feature of Hyper LP**

Remedying FOL Semantics' Lack of Scalability

- **Hyper LP handles conflict robustly**
  - Whereas FOL is "a glass bubble" – it is perfectly brittle semantically in face of contradictions from
    - Quality problems/errors in the data and knowledge
    - Conflict when merging KBs

    E.g., OWL beyond -RL

- **A VLKB with a million or billion axioms formed by merging from multiple Web sources, is unlikely to have zero KB/KA conflicts from:**
  - Human knowledge entry/editing
  - Implicit context, cross-source ontology interpretation
  - Updating cross-source
  - Source trustworthiness
- ***Weakening provides a critical advantage for VLKB scalability***
  - ***semantically, as well as computationally***

FOL: A Glass Bubble

Extreme sensitivity to conflict limits its scalability in # of axioms and # of merges

Keijo Kopra from Finland as he competes in the Iittala Cup glass-blowing competition June 7, 2008. (Reuters)

KR Conflict Handling – A Key to Scalability

BEFORE ⇒ **AFTER**

KR: Classical Logic (FOL, OWL) ⇒ **KR: LP with Defaults (Courteous-style)**

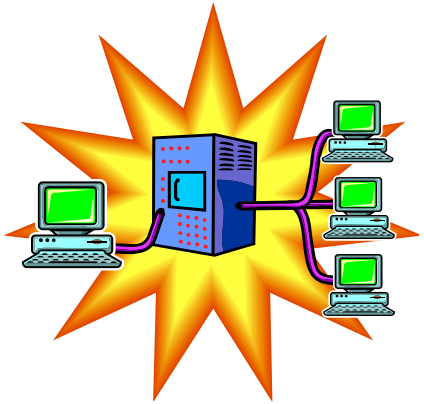
Contradictory conflict is globally contagious, invalidates all results.

⇒

Contradictory conflict is contained locally, indeed tamed to aid modularity.

Knowledge integration involving conflict is labor-intensive, slow, costly.

⇒



Knowledge integration involving conflict is highly automated, faster, cheaper.

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Heavy Reliance on Procedural Attachments in Currently Commercially Important Rule Families

- E.g., in OO applications, DBs, workflows.

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

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

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

Additional Motivations in Semantic Web for Procedural Attachments

- Query over the web
- Represent services
- Shared ontology of basic built-in purely-informational operations on XML Schema datatypes
  - E.g., addition, concatenation
  - E.g., in RuleML & SWRL, N3.
- Hook rules to web services, generally

Providing Declarative Semantics for Procedural Attachments

- **Procedural attachments historically viewed in KR theory as … well … *procedural* ;-) … rather than declarative.**
  - Not much theoretical attention
- Needed for Semantic Web: a *declarative* KR approach to them
- **Production LP is probably the most important approach today**
  - E.g., SILK, RuleML, SweetRules, IBM Common Rules, predecessors
    - Formerly called ***Situated*** LP
  - Provides disciplined expressive abstraction for two broad, often-used categories of procedural attachments:
    - Purely-informational Tests
    - Side-effectful Actions
  - Makes restrictions: assumptions become explicit
  - Declarative semantic guarantees, interoperability
  - Embodies primarily analytical insight, initially
  - Provides also: expressive generalizations, algorithms/techniques

Production LP: Overview II

- Point of departure: LPs are pure-belief representations, but most practical rule systems want to invoke external procedures.
- Production/Situated LP's feature a semantically-**clean** kind of **procedural attachments**. I.e., they hook beliefs to drive procedural APIs outside (a.k.a. "external" to) the rule engine.
- Procedural attachments perform
  - **external queries** ("sensing") when testing a body atom
  - **external actions** ("effecting") upon concluding a head atom

  The attached procedure is invoked during inferencing.
- A procedural attachment associates an "internal" predicate/atom with an "external" procedural call pattern, e.g., a Java method. Such associations are specified as part of the extended KR.

Production LP: Overview III

- phoneNumberOf(?person,?num) :- BoeingBluePages.getPhoneMethod(?person,?num);
  // internal predicate/fact inferred based on external query that invokes attached procedure
- ATTMobile.sendTextMethod(?num,?string) :- shouldSendTextMsg(?num,?string);
  // external action that invokes attached procedure is inferred based on internal conclusion fact

- Specify binding-signature for each sensoring attached procedure
  - For each argument ?xi: whether ?xi is an input ("bound") vs. an output arg.
    - Simplest signature is that all args are input args
  - OK to declare multiple binding signatures per sensoring attached procedure.
- Also specify datatypes of arguments in attached procedures signatures

- Attached procedures can be invoked/loaded remotely (e.g., Java, web services)

- Overall: cleanly separate out the procedural semantics as a declarative extension of the pure-belief declarative semantics. Easily separate chaining from action. (Declarative = Independent of inferencing control.)

Production LP: Overview IV

- PLP is KR for Hooking Rules to Services
  - With ontologies
  - Esp. Web services
  - Declaratively
- Rules use services
  - E.g., to query, message, act with side-effects
- Rules constitute services executably
  - E.g., workflow-y business processes

Semantics of Production LP I

- Definitional: complete inferencing+action occurs during an "episode" – intuitively, run all the rules (including invoking effectors and sensors as we go), then done
- Effectors can be viewed as all operating/invoked after complete inferencing has been performed
  - Independent of inferencing control
  - Separates pure-belief conclusion from action

Semantics of Production LP II

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

Semantics of Production LP III

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

Updating & Events in Production LP

- "Event" is a set of facts/rules, constituting an update to KB
- An interesting kind of thing to do with a Production LP is to update its premises, and perform incremental inferencing+action.
  - new PLP *P2* = (update *U2*) ∪ (previous *P1*)
  - Incremental inferencing+action is defined as:
    - Generate the inferences that are novel

      NovelConclusions = Conclusions(*P2*) – Conclusions(*P1*)

    - Perform the external actions (effecting) associated with NovelConclusions
- Extension to PLP:
  - An event channel is an attached procedure that delivers events as updates
    - Listening to an event channel can be viewed as a persistent external query

Algorithms for PLP Implementation

- The most complicated aspect of implementing the Production feature of LP is to ensure sensor-safeness, i.e., that sensing is attempted only after sufficient bindings are available (for a given atom being tested/queried, in a given rule).
- This is roughly similar to implementing safe negation (NAF) in Normal LP, but somewhat more complicated conceptually and algorithmically.
- It is more similar to some of the techniques developed in bottom-up evaluation, magic sets, relational database tabling, etc., of OLP's where binding signatures (a.k.a. "modes") are considered.

Production Rules (PR)

- Big sector commercially
  - Jess semi-open Java tool, popular among researchers
  - Drools open source C tool, got popular in last 2 yrs
- PR2LP, LP2PR: via SweetRules approach (2002, 2005)
  - Horn: fairly simple; several systems implement it now
  - External actions and queries: use PLP restrictions
  - NAF: use insights of stratification and well-founded semantics & proof theory, PR salience and modules
- ECA (Event-Condition Action rules) are similar to PR
- RIF PRD (Production Rules Dialect)
  - procedural operational semantics, leverages RIF-Core (subset of RIF-BLD)
- OMG Production Rules Representation: meta-model

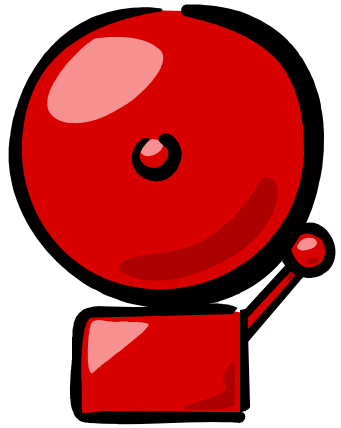
Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

Integrity Constraints

Two styles with quite different semantics:

1. Alarm: Rule that detects a violation
   - Typical: the rule reports/notifies that constraint is violated
     - Other rules infer resulting actions to take
   - E.g., many BRMS, SILK



…VERSUS…

2. Model-cutting: Rule that forces global contradiction when axiom is violated
   - Typical: no model, lose all useful entailments!!
   - E.g., FOL

Lloyd-Topor Expressive Features

- Via the Lloyd-Topor transformation, it is straightforward to extend the expressiveness of LP with additional FOL-type connectives and quantifiers, as syntactic sugar: [Lloyd 1987]
  - $\vee,\exists,\forall,\leftarrow$ in body; $\wedge,\forall,\leftarrow$ in head
    - Freely nested within body or within head
    - Negation is freely nested in body, too
  - *Stays tractable!*
- Disallowed: $\vee,\exists$ in head (these are disjunctive)
- Some features are monotonic (do not rely on NAF):
  - $\vee,\exists$ in body; $\wedge,\forall,\leftarrow$ in head
  - These can be applied as syntactic sugar to Horn LP
- Other features are nonmonotonic (do rely on NAF):
  - $\forall,\leftarrow$ in body
- Many rule systems and languages support a subset of Lloyd-Topor features
  - E.g., RIF, RuleML, SILK, Prolog, Jess, CommonRules

Default Inheritance cf. OO

- Ubiquitous in object-oriented languages & applications
- Defaults naturally increase reuse, modularity
- OWL and FOL cannot represent defaults (they are monotonic)
- Requirement for semantic web service process ontologies
  - Need to jibe with mainstream web service development methodologies, based on Java/C#/C++ etc.

- Approach: Represent OO default-inheritance ontologies using nonmon LP rules
  1. [Grosof & Bernstein 2003] Courteous Inheritance approach
     - Transforms inheritance into Courteous LP (in RuleML, using SweetRules)
     - Represents MIT Process Handbook (ancestor of PSL)
       - 5,000 business process activities; 38,000 properties/values
       - Linear-size transform (n + constant).
  2. [Yang & Kifer, 2006] approach
     - Transform inheritance into essentially Normal LP (using FLORA-2)

Additional Expressive Features in Rules & LP, e.g., SILK

- Explicit equality (and equivalence) reasoning
  - In head of non-fact rules, therefore derived
  - Interaction with nonmonotonicity
  - Key characteristic: substitutivity of equals for equals
  - Related to Herbrand aspect of LP semantics
- Existentials, skolemization
  - RDF blank-nodes, anonymous individuals [Yang & Kifer]
  - Related to Herbrand aspect of LP semantics
- Aggregation (operate on entailed lists): count, total, min, max, etc.
  - Depends on nonmonotonicity, stratification
- Datatypes – they are basic but fairly straightforward
- "Constraints" (e.g., equation/inequality systems)
  - Commonly: via external query/assert to specialized solver
- *Also:* Reasoning within the KR about the results of side-effectful actions
  - E.g., Transaction Logic [Kifer *et al*], Golog [Reiter, Lin, *et* al]
    - These are research-world, not commercial, today

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

SILK & Hyper LP: Overview

- **A KR Language and KR System with reasoner, UI, interchange**
  - Syntax & semantics, forward & backward inferencing, Java API, translators
- **Goal: Expressiveness + Semantics + Scalability + Web**
- **Focus: Defaults and Processes**
- **Largest rule research program in the US** (that we are aware of)
  - Begun in 2008, part of Vulcan's Project Halo, primarily via contractors
- ***Hyper* LP KR combines new features**
  - Defaults and Weakened Classical, cf. generalized Courteous LP and Hypermon. map.
  - External Actions and Events and Queries, cf. generalized Production LP

  with previous advanced features
  - Higher-order and Frames, cf. Hilog and F-Logic
  - Webized syntax, cf. RIF/RuleML and OWL/RDF
  - Closed-World, cf. well-founded unstratified NAF
  - Good Efficiency of reasoner performance
  - Equality, Functions, and misc. other less glamorous features

- **Status: prototype engine, language, and theory for expressive core**
  - V1 adds Higher-Order Defaults to FLORA-2
  - Extensive requirements analysis, use cases, benchmarking; ReCyc translation
  - V2 in development adds more features and Java API (See ISWC-2009 Demo!)

SILK & Hyper LP Overview (cont.'d)

- **Radically extends the KR power of W3C OWL, SPARQL, and RIF — and of SQL**
  - Defaults and robust conflict handling – *cope with knowledge quality and context*
  - Higher-order and flexible meta-reasoning – *elevate meta-data to meta-knowledge*
  - Actions and events, cf. production rules and process models – *activate knowledge*
- **Raises the KR abstraction level for business users (SMEs) and NL KA/UI**
- **Use cases in business policies, ontology mapping, e-commerce, biomed, …**
- **Redefining the KR playing field for Semantic Web, business rules, and rule-based process management**
  - Defaults and Higher-Order – yet retain computational web scalability
  - Escape from Glass Mountain – yet retain grade-AAA model-theoretic semantics
- **Motto: "Transforming Knowledge"**
  - Composes a set of KR transformations for …
  - Expressive extensions – language and semantics
  - Translations between KRs/syntaxes, for interchange
  - Reuse of previous algorithms and implementations
- **http://silk.semwebcentral.org**

Semantic Rules KR: Features Comparison

Level ("generation")	Groups of features	*SILK*	*FLORA-2*	*RIF-BLD*
1G. Basic	ie: Horn, chaining, external queries, built-ins *(Level Summary)*	Y	Y	Y
2G. Advanced	*(Level Summary)*	**Most**	**lots**	**some**
	Equality (derived via non-fact rules)	Y	Y	Y
	Functions	Y	Y	Y
	Convenience Package: Frames, integrity constraints, skolemization	Y	Y	R. frames
	Closed-World: unstratified NAF, aggregates, Lloyd-Topor	Y	Y	N
	Higher-Order (incl. reification)	Y	Y	N
	Actions (external) (via procedural attachments)	Developing	N	N
	Base Defaults (prioritized, cf. Courteous)	Y	N	N
	Webized syntax (URI names and XML/RDF KBs)	Developing	N	Y
3G. Hyper	*(Level Summary)*	**Pioneer**	**N**	**N**
	Higher-Order Defaults	Y	N	N
	Weakened Classical (sound interchange with default rules)	Developing	N	N
Other Misc.		(NA)	(NA)	(NA)
	Other Expressive	Developing	Inheritance	-
	Reasoner Efficiency (upper-tier on OpenRuleBench)	good	good	NA (standard)

- Summarizes detailed analysis of 40 KR expressive features, 17 systems.
- Notes: R. = Restricted

Features Comparison – More Systems & Stds

Level	Groups of Features	*SILK*	*FLORA -2*	*RIF-BLD*	*Jena*	*Onto-broker*	*Jess*	*IBM C.R.*	*DLV*	*SQL*	*SPA-RQL*	*Common Logic*	*OWL2 RL*	*OWL2 DL*
Basic	Horn chain. etc.	Y	Y	Y	Y	Y	Y	Y	Y	R.	R.	Y	R.	R.
Advanced	*(Level summary)*	Most	lots	some	some	some	some	some	some	some	some	some	some	some
	Equality	Y	Y	Y	R.	R.	R.	N	Y	N	R.	Y	R.	Y
	Functions	Y	Y	Y	N	N	N	Y	Y	N	N	Y	N	N
	Frames etc.	Y	Y	R.	R.	Y	R.	R.	R.	R.	R.	R.	R.	R.
	Closed-World	Y	Y	N	N	Y	R.	R.	Y	R.	R.	N	N	N
	Higher-Order	Y	Y	N	N	N	R.	N	N	N	R.	Y	R. bit	R. bit
	Actions	Dev.	Y	N	N	N	Y	Y	N	N	N	N	N	N
	Base Defaults	Y	N	N	N	N	N	Y	N	N	N	N	N	N
	Webized	Dev.	R.	Y	Y	R.	R.	R.	R.	N	Y	Y	Y	Y
Hyper	*(Level summary)*	1st	N	N	N	N	N	N	N	N	N	N	N	N
	H-O. Defaults	Y	N	N	N	N	N	N	N	N	N	N	N	N
	Weak. Classical	Dev.	N	N	N	N	N	N	N	N	N	N	N	N
Misc.		NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
	Other Expres.	Dev.	Inherit-ance	-	-	-	events	-	disju.	R.	R.	classical	-	classic-al
	Efficiency	good	good	NA	fair	good	fair	poor	good	NA	NA	NA	NA	NA

- Summarizes detailed analysis of 40 KR expressive features, 17 systems.
- Notes: Dev. = Developing, R. = Restricted; C.R.= CommonRules; disju.=disjunctive.

Features Comparison – More Systems & Stds

Background on Systems and Standards:

- Jess is a representative commercial production rule (PR) system. PR was shown 5-7 years ago to have a semantic subset (based on the SweetRules translation). The currently most commercially important business rule management systems (BRMS) are based on PR or similar event-condition (ECA) action rules.

- W3C Rule Interchange Format (RIF)'s Basic Logic Dialect (BLD) is its main semantic part. There is also a framework for extensions. RIF is based largely on RuleML, except for RIF's Production Rule Dialect (PRD).

- W3C OWL 2 RL is OWL's Rules subset (based on Description LP).

- Jena is a popular open-source semantic web toolkit, incl. for rules.

- Ontobroker is a commercial forward-chaining LP system.

- IBM Common Rules (C.R.) introduced the base defaults feature.

- Common Logic (CL) is an ISO standard for classical logic, used also by OMG's Semantic Business Vocabulary and Rules (SBVR) standard.

- DLV is a disjunctive LP system, by Univ. of Calabria (it supports disjunction in rule heads)

163

More KR Rationale about SILK & Hyper LP

- **"*Hyper*" since it is Web (*hyper*text) centric, and it behaves *hyper*monotonically**
- **It integrates several major LP extensions never previously combined:**
  - Higher-order and Frames and Skolemization, cf. F-Logic
  - + Defaults, cf. Courteous LP (and Defeasible Logic)
    - Newly generalized and modified approach using Argumentation Theories
    - Sound Interchange with Full Classical Logic, via Hypermonotonic mapping
      - Unrestricted clauses, plus skolemization: greatly generalizes DLP, OWL RL
      - Behaves robustly in the face of knowledge quality errors and conflicting merging
    - Pervasively combined with all other KR features
- **Give up reasoning by cases**
  - Source of exponential worst-case complexity in classical, disjunctive LP, stable LP
  - Can hope to reintroduce in restricted or altered form, or develop work-arounds, later
  - But there are many apps not requiring it, e.g., DBMS, BRMS
- **Can realistically hope to be web-scalable performance-wise, unlike highly expressive classical**
  - Polynomial computational complexity, under non-onerous restrictions
    - Same complexity as Horn rules!! (Must be careful of recursion through functions.)
  - Many optimizations available
  - Established track record of high scalability for relational databases

Sem Tech Industry Requirements targeted by SILK

- **Need to raise abstraction level, e.g., for SME and NL KA/UI**
  - (SME = Subject Matter Expert, a.k.a. Business User)
- **Need robustness & meta-reasoning for web KB integration**
  - Cope with conflict, mediation, context, knowledge quality
  - Defaults ⇒ robustness, modularity ⇒ scalability
  - Higher-order ⇒ puts the meta- deeply in knowledge not just data
- **Hope: be like advance of the Relational model in DBMS**
  - Will Hyper LP be to the 2010s what Relational was to 1970s-80s?
    - (NB: software industry clockspeed was slower back then)

Driving Requirements for SILK Expressiveness

- Processes *[For science, BPM. E.g., >50% of questions on Environmental Sci. AP.]*
  - Actions, Causality, Events, Reactivity, State Change
- Knowledge-level Communication *[Knowledge, science, & business are societal]*
  - I.e., Import and Merge of External Knowledge, incl. data/facts, ontologies, rules
  - Via Pull/Query, and Via Push/Events
  - From Web, built-ins, specialized reasoners, broad-purpose reasoners
  - **Mediate** ontologies and contexts
  - Interchange with **Classical** logic KR, as well as with LP/rules KR
  - *Uses for Classical include:*
    - *Background KBs, e.g., ontology, e.g., about processes*
    - *Existing techniques and KBs for equations, constraints, and processes*
    - *Common Logic (and KIF), SBVR, OWL, RDF*

Uses of *Major SILK Expressive Features*

- Defaults (beyond naf) *[For many purposes, pervasively]*
  - Exceptions, Priorities, Inheritance, Strong Negation, Preventive Integrity Constraints
  - *For OO, robust KB merging/updating, process causality, policy and regulation/law, natural language incl. KA, import of classical, argumentation, hypotheticals and counterfactuals*

- Higher-order, incl. for Meta-reasoning *[For many purposes, pervasively]*
  - *Convenient, concise abstraction for KR designers, and for KE/SME users*
  - *Many KRs have some of it, incl. RDF, OWL-Full, BRMS, Cyc. E.g., transitive_closure(?P).*
  - *Meta-reasoning uses include: KR macros, KB translation/import, ontology mappings, reasoning control, provenance, KB modularization, navigation in KA, multi-agent & nested belief, context, modals. Plus – the Web is about* meta-*data.*

Outline of Part B. Concepts & Foundations

1. Overview of Logical Knowledge Representations
   - Logic Programs (LP) and its relationship to First Order Logic (FOL)
   - Rule-based Ontologies: Description Logic, Description LP, OWL RL
2. Basics: Horn Case; Functions
3. F-Logic, Frame Syntax, Object Oriented Style
4. HiLog, Higher-Order Syntax, Reification, Meta-Reasoning
5. W3C Rule Interchange Format: Dialects, Framework
6. Nonmonotonicity: Defaults, Negation, Priorities; FOL's Glass Bubble
   - Semantics for Default Negation
   - Courteous LP, Argumentation Theories
   - Hypermonotonic Mapping: FOL $\leftrightarrow$ LP, Soundly
7. Procedural Attachments to Actions, Queries, Built-ins, and Events
   - Production/Situated LP, Production Rules
8. Additional Features: Integrity Constraints, Inheritance, Lloyd-Topor, Equality, Skolemization, Aggregation, Datatypes, "Constraints"
9. Hyper LP and SILK – Putting it all together

PART C. SLIDES FOLLOW

Outline of Part C. Conclusions & Directions

1. More about Tools

2. Conclusions

3. Directions for Future research

Appendix: References and Resources

(General Discussion)

More about Tools

1. *Rule systems designed to work with RDF/OWL:*
   Commercial-world: Jena

- Jena-2 SW suite has rule (and RDF/OWL) capabilities
  - Open source, popular, Java
  - Basic Horn-ish
  - Supports forward, backward, and mixed direction inferencing
  - Operates directly on RDF/OWL statements, without copying in/out
  - Works well with RDF(S). Suite includes OWL capabilities
  - Rules are used to implement RDFS and OWL reasoners

More about Tools

1. *Rule systems designed to work with RDF/OWL, continued:*
   Commercial-world: Oracle; other

- Oracle has rule capabilities in semantic tech suite, as part of its flagship database platform
  - Oracle Spatial RDF, now approaching its 3rd production release, motivated and (essentially) implements OWL 2 RL. It also supports user-defined rules using its own rule syntax.
  - Also has production-rule type products, including recently acquired Haley Ltd. – a leader in NL KA – and Ruleburst
  - In development: support for W3C RIF Basic Logic Dialect (BLD)
- Various others, e.g., Ontotext, Ontoprise, Versatile Information Systems

More about Tools

1. *Rule systems designed to work with RDF/OWL, continued:*
   Research-world: SweetRules; cwm; others

- SweetRules has semantic translator from DLP subset of OWL to LP Rules in RuleML and SWRL. Open source, Java. Not maintained.
- Cwm implements N3 – rules + RDF. N3 is a popular syntax for RDF. Semantically hazy in some regards, but overlaps a lot with LP. Open source, Python.
- SweetRules pioneered design and implementation of fully semantic interoperability of nonmon LP with Jess production rules, and generally supports Courteous Production LP
- KAON2 implements primarily monotonic rules in FOL & LP
- Numerous others
  - Protege 3 and 4, Pellet, KAON2, and others support SWRL
  - OWLJessKB was an early tool employing Jess to support a subset of OWL DL
  - Several systems combine SWRL with Jess, cf. SweetRules approach

More about Tools

2. *Prolog and Production Rule systems*
   - XSB: semantic, Prolog, full WFS negation, fast, C with available Java front end (Interprolog)
   - Jess: production rules, popular, Java, free for non-commercial use but not open source
   - YAP and SWI open source Prolog's are on a development trajectory towards WFS and SW
   - Benchmarking: OpenRuleBench
     - Open source tools for benchmarking rule systems
     - Benchmarking study: [S. Liang, M. Kifer, *et al.* WWW-2009]; extended report on website.
     - XSB, Ontobroker, YAP Prolog, DLV all did well
     - http://openrulebench.semwebcentral.org

More about Tools

3. *Advanced Expressiveness*

- FLORA-2: open source, built in/on XSB Prolog, has Hilog, Frame, reification, skolemization features
- SILK (in development): extends FLORA-2with Courteous defaults, attached procedures, hypermonotonic translation, APIs. Partly in Java. Planned to be free for non-commercial use. Release in 2010.
- IBM CommonRules (1999) supports Courteous Defaults and Production-LP style external actions. Cheap or free, Java.

4. *Rules in Semantic Wikis*

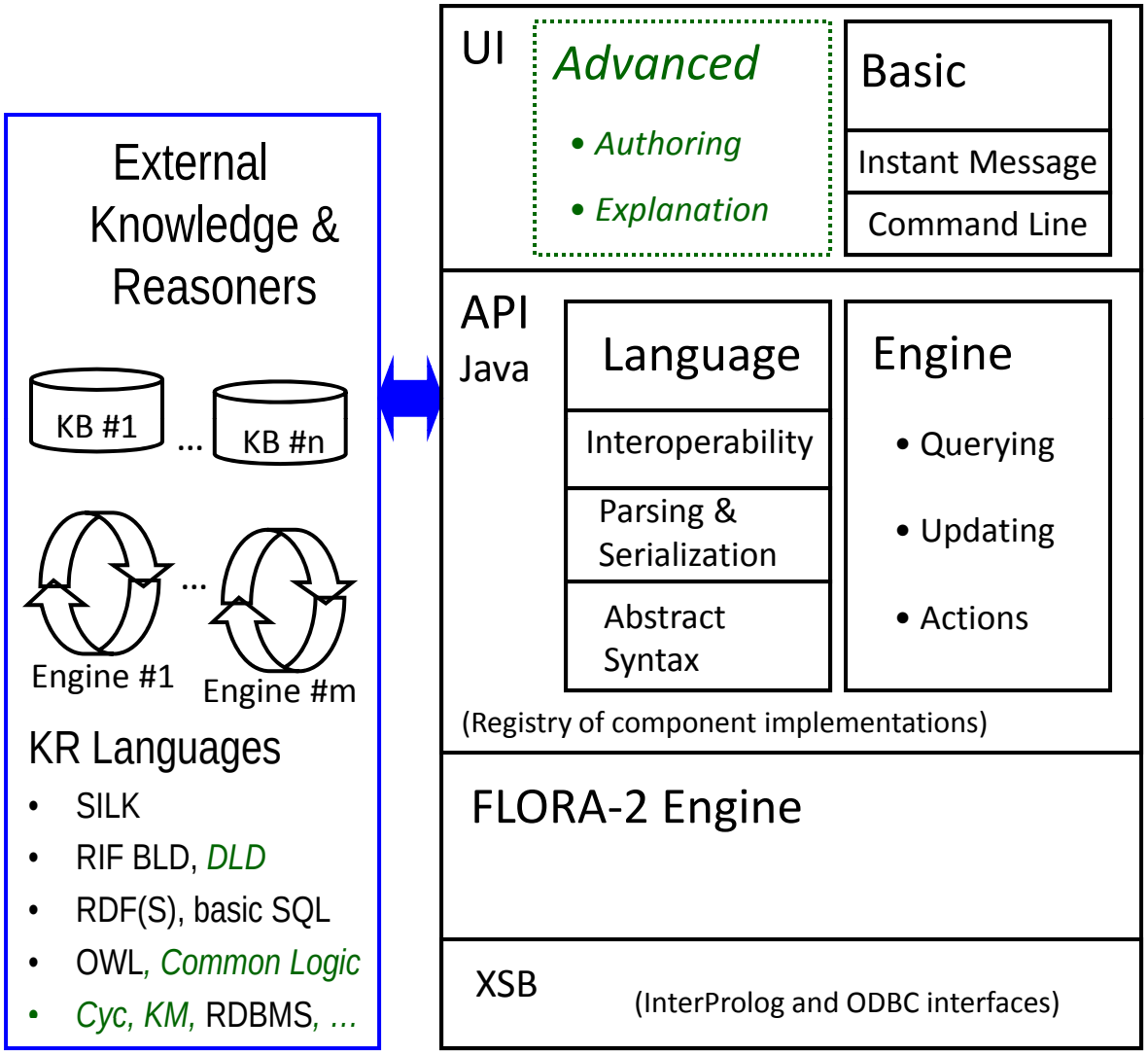
- Semantic MediaWiki+ (SMW+) Simple Rules extension is in development (release ~ end 2009). Basic Horn-ish LP.
- Open source, PHP. SMW+ is a leading Semantic Wiki that extends the software Wikipedia runs, by Vulcan/Ontoprise.

More about Tools

5. *Some Available Large Rule Bases*

- OpenCyc / ResearchCyc
  - Open source / free for non-commercial use
  - ~ 1 Million / 3 Million axioms. Large 25 year effort.
  - Idiosyncratic semantically, but overlaps with LP
  - ReCyc: translation to SILK is in development (by Vulcan with Cycorp/SRI)
- Open Process Handbook
  - Open source. Semantic Wiki–ish. http://ccs.mit.edu/ph
  - 5,000 business processes, each with ~10 axioms
  - Lots of text and links too. 15 year effort.
  - Translatable to Courteous LP, via approach along lines of SweetPH approach [A. Bernstein, B. Grosof 2003-2005 reports] http://www.mit.edu/~bgrosof/#SweetPH
- OpenMind – collaborative commonsense KB
  - Open source. ~1 Million axioms. Built by Web users.
  - Lacks declarative semantics
  - http://openmind.media.mit.edu

SILK V2 Architecture



- V2 Functionality
  - Higher-order defaults reasoning, combines many other advanced KR features
  - SILK and external KR language support integrated tightly with reasoning engine
- *Future Items*
  - *Meet Process requirements*
  - *More UI is key: graphical, limited NL*
  - *Integrate with AURA*
  - *SILK KR: truth maintenance, probabilistic & constraints, parallelization*
- Test Sets Focus
  - Defaults, Process
  - Biology (1st yr college level)

NB: *Italics* indicate future items beyond V2

BRMS Industry Roadmap: facing disruption

- **Semantic rules is a prospectively truly disruptive innovation for the existing business rules management systems (BRMS) industry sector**

- **See "The New Rules of Business" [Grosof EBRC-2007 keynote]**
  - Strategic analysis of evolving market dynamics and what players should do about it
    - Done with a Management professor hat on
  - http://www.mit.edu/~bgrosof/#EBRC2007Talk

Conclusions

1. Theme: Centrality to Web
   - More than most people realize, LP Rules are central to the Web, both current and future
   - Relational, XML, and RDF databases/querying is LP
   - Thriving commercial business rules market sector, based on production rules / event-condition-action rules, is moving to the Web, and translates largely to LP
   - Often used for ontologies: represent, implement, map
   - Semantic tech and semantic web is largely already LP-based

Conclusions, continued

2. Theme: Incremental Evolution
   - LP Rules, and Semantic Web overall, is incremental technologically wrt relational and Web DBMS

3. Theme on KR expressiveness: Reducibility
   - LP feature extensions built up in layers
   - E.g., Lloyd-Topor, Hilog, Frame syntax, Courteous Defaults each reduce tractably to Normal LP

Key Directions for Future Research

1. Expressiveness
   - Relationship between FOL and Default LP
   - Disjunction, Probabilistic, Abduction, Fuzzy
   - Misc. smaller issues, e.g., equality, aggregation
2. Reasoning performance
   - Forward-direction, truth maintenance, termination
   - Parallelization
3. Knowledge acquisition and UI
   - Explanation
   - Limited natural language
   - Business users / Subject Matter Experts (SMEs)
   - Collaboration
4. Applications and Tools
   - Build, experiment

ADDITIONAL REFERENCES & RESOURCES FOLLOW

References & Resources I: Standards on Rules and Ontologies

- http://www.ruleml.org RuleML *Includes links to some tools and examples.*
- http://www.w3.org/Submission/2004/SUBM-SWRL-20010521 SWRL
  - http://www.daml.org/committee Joint Committee. Besides SWRL (above) this includes:
    - http:///www.daml.org/2004/11/fol/ SWRL-FOL
    - http://www.ruleml.org/fol FOL RuleML (also see RuleML above)
  - http://www.daml.org/rules DAML Rules
- http://www.swsi.org Semantic Web Services Initiative. Especially:
  - Semantic Web Services Language (SWSL), incl. SWSL-Rules and SWSL-FOL and overall requirements/tasks addressed
- http://cl.tamu.edu Common Logic (successor to Knowledge Interchange Format)

- Also: Object Management Group (OMG) has efforts on rules and ontologies (cooperating with RuleML and W3C)
- Also: JSR94 Java API effort on Rules (cooperating with RuleML)

References & Resources II: Standards on Rules and Ontologies

- http://www.w3.org World Wide Web Consortium, esp.:
  - …/2005/rules/ Rule Interchange Format
  - …/2007/owl/ OWL 2 – see esp. OWL RL Profile
  - …/2001/sw/ Semantic Web Activity, incl RDF, OWL, SPARQL, and RIF
  - …/2002/ws/ Web Services Activity, incl. SOAP and WSDL
  - www-rdf-rules@w3.org Rules discussion mailing list
  - www-sws-ig@w3.org Semantic Web Services discussion mailing list
  - P3P privacy policies
  - XQuery XML database query

- http://www.oasis-open.org Oasis, esp. on web policy & web services:
  - XACML XML access control policies
  - ebXML e-business communication in XML
  - Legal XML
  - BPEL4WS Business Processes as Web Services
  - Web Services Security

Refs & Resources III: LP with Negation

- Przymusinski, T., "Well Founded and Stationary Models of Logic Programs", Annals of Artificial Intelligence and Mathematics (journal), 1994. *Constructive model theory, and proof theory, of well founded semantics for LP.*
- Van Gelder, A., Schlipf, J.S., and Ross, K.A., "The Well-Founded Semantics for General Logic Programs", Journal of the ACM 38(3):620-650, 1991. *Original theory of well founded semantics for LP.*
- Gelfond, M. and Lifschitz, V., The Stable Model Semantics for Logic Programming, Proc. 5th Intl. Conf. on Logic Programming, pp. 1070-1080, 1988, MIT Press. *Original theory of stable semantics for LP. Answer set programs extend this.*
- Lloyd, J.W., "Foundations of Logic Programming" (book), 2nd ed., Springer-Verlag, 1987. *Includes Lloyd-Topor transformation, and correspondence of semantics to FOL in definite Horn case. Reviews theory of declarative LP. Somewhat dated in its treatment of theory of NAF since it preceded well founded and stable semantics.*
- Baral, C., and Gelfond, M., "Logic Programming and Knowledge Representation", J. Logic Programming, 1994. *First and last parts review theory of declarative LP. Stronger on stable semantics than on well founded semantics.*
- Gelfond, M., "Answer Sets" (book chapter 7). In: Handbook of Knowledge Representation. Elsevier, 2007. *Up-to-date exposition of answer set programs.*

Resources IV: More Key LP Theory

- "Description Logic Programs: Combining Logic Programs with Description Logic", by B. Grosof, I. Horrocks, R. Volz, and S. Decker, Proc. 12th Intl. Conf. on the World Wide Web (WWW 2003), 2003. *On DLP KR and how to use it.*

"Logical Foundations of Object-Oriented and Frame-Based Languages", by M. Kifer, G. Lausen, and J. Wu, *J. ACM* 42:741-843, 1995.

-- "HiLog: A Foundation for Higher-Order Logic Programming", by W. Chen, M. Kifer, and D.S. Warren, *J. Logic Programming* 15(3):187-230, Feb. 1993.

-H. Wan, B. Grosof, M. Kifer, P. Fodor, S. Liang, Logic Programming with Defaults and Argumentation Theories, 25th International Conference on Logic Programming (ICLP 2009), Pasadena, California, July 2009. *On LP defaults approach.*

References & Resources V: Misc. on Rules and Ontologies

- http://ccs.mit.edu/ph MIT Process Handbook, incl. Open Process Handbook Initiative
- Bernstein, A. and Grosof, B. "Beyond Monotonic Inheritance: Towards Semantic Web Process Ontologies". Working reports, 2003-2005. http://www.mit.edu/~bgrosof/#SweetPH

"Semantic Web Services Framework" (SWSF), V1.0+, by Battle, S., Bernstein, A., Boley, H., Grosof, B., Gruninger, M., Hull, R., Kifer, M., Martin, D., McIlraith, S., McGuinness, D., Su, J., and Tabet, S. (alphabetic), May 2005. Technical Report (~200 pages).

-Grosof, B., "Representing E-Commerce Rules Via Situated Courteous Logic Programs in RuleML", Electronic Commerce Research and Applications (journal) 3(1):2-20, 2004. *On situated courteous LP KR, RuleML overview, and e-commerce applications of them.*

- Grosof, B. and Poon, T., "SweetDeal: Representing Agent Contracts with Exceptions using Semantic Web Rules, Ontologies, and Process Descriptions", Intl. Journal of Electronic Commerce 8(4):61-98, Summer 2004. *On SweetDeal e-contracting app.*
- Firat, A., Madnick, S., and Grosof, B., "Financial Information Integration in the Presence of Equational Ontological Conflicts", Proc. Workshop on Information Technologies and Systems, 2002. *On ECOIN. Also see A. Firat's PhD thesis, 2003.*

- J. Hebeler, M. Fisher, R. Blace, A Perez-Lopez, M. Dean, *Semantic Web Programming*, Wiley, 2009. *A whole book.*

Resources VI: DL Safe SWRL rules

- OWLED's DL Safe SWRL Rules Task Force [1] [2], whose proposals have already been implemented in Pellet and KAON2.
  - [1] http://wiki.webont.org/page/DL_Safe_SWRL_Rules
  - [2] http://code.google.com/p/owl1-1/wiki/SafeRules

References & Resources VII: Misc. on Rules and Ontologies

- Grosof, B., Gandhe, M., and Finin, T., "SweetJess: Translating DamlRuleML To Jess". Proc. Intl. Wksh. On Rule Markup Languages for Business Rules on the Semantic Web, 2002 (the 1st RuleML Workshop, held at ISWC-2002). See extended and revised working paper version, 2003. *On SweetJess translation/interoperability between RuleML and production rules.*
- Forgy, C.L., "Rete: A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem". Artificial Intelligence 19(1):17-27, 1982. *On the key Rete algorithm for production rules inferencing.*
- Friedman-Hill, E., "Jess in Action" (book), 2003. *On Jess and production rules.*
- Ullman, J., "Principles of Knowledge Base and Database Systems Vol. I" (book), 1988. *See esp. the chapter on Logic Programs, incl. algorithm for stratification.*
- http://xsb.sourceforge.net XSB Prolog. See papers by D. Warren *et al.* for theory, algorithms, citations to standard Prolog literature (also via http://www.sunysb.edu/~sbprolog)
- Horrocks, I. and Patel-Schneider, P., paper on OWL Rules and SWRL, Proc. WWW-2004 Conf. *On SWRL theory incl. undecidability.*
- Horrocks, I. and Bechhofer, S., paper on Hoolet approach to SWRL inferencing via FOL theorem-prover, Proc. WWW-2004 Conf. *On SWRL inferencing.*

References & Resources VIII: More Courteous and Situated

- Grosof, B., Labrou, Y., and Chan, H., "A Declarative Approach to Business Rules in Contracts", Proc. 1st ACM Conf. on Electronic Commerce, 1999, ACM Press. *On courteous LP KR with mutexes and its e-contracts applications.*
- Grosof, B., "Courteous Logic Programs: Prioritized Conflict Handling for Rules", Proc. Intl. Logic Programming Symposium., 1997. See extended version: IBM Research Report RC 20836, 1997. *Basic version courteous LP (since generalized).*
- Grosof, B., "A Courteous Compiler from Generalized Courteous Logic Programs To Ordinary Logic Programs", (IBM) research report extension to "Compiling Courteous Logic Programs Into Ordinary Logic Programs", 1999. Available via http://ebusiness.mit.edu/bgrosof or IBM incl. in CommonRules documentation. *Details on courteous compiler/transform.*
- Grosof, B., Levine, D.W., Chan, H.Y., Parris, C.J., and Auerbach, J.S., "Reusable Architecture for Embedding Rule-based Intelligence in Information Agents", Proc. Wksh. on Intelligent Information Agents, at ACM Conf. on Information and Knowledgte Management, ed. T. Finin and J. Mayfield, 1995. Available also as IBM Research Report RC 20305. *Basic situated LP paper. Also see 1998 patent.*
- Grosof, B., "Building Commercial Agents: An IBM Research Perspective (Invited Talk). Proc. 2nd Intl. Conf. on the Practical Applications of Intelligent Agents and Multi-Agent Technology (PAAM97), pub. The Practical Applications Company, 1997. Also available as IBM Research Report RC 20835. *Overview of situated LP.*

Resources IX: Misc. Papers

- "SWRL: A Semantic Web Rules Language Combining OWL and RuleML", V0.7+, by I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean, Nov. 2004. Technical Report.

- RuleML website, especially design documents and list of tools. Ed. by H. Boley, B. Grosof, and S. Tabet, 2001-present. http://www.ruleml.org

- "Web Service Modeling Ontology (WSMO)" by J. de Bruijn et al., 2005. Technical Report.

- "A Declarative Approach to Business Rules in Contracts: Courteous Logic Programs in XML", by B. Grosof et al., Proc. EC-99.

- "A Policy Based Approach to Security for the Semantic Web", by L. Kagal et al., Proc. ISWC-2003.

- "Financial Information Integration in the Presence of Equational Ontological Conflicts", by A. Firat et al., WITS 2002 conf.

- "Delegation Logic: A Logic-based Approach to Distributed Authorization", *ACM Trans. on Info. Systems Security (TISSEC)*, by N. Li et al., 2003

Resources X: SILK

- SILK project page: http://silk.semwebcentral.org/
  - RR-2009 keynote slideset, by B. Grosof
  - H. Wan, B. Grosof, M. Kifer, P. Fodor, S. Liang, Logic Programming with Defaults and Argumentation Theories, 25th International Conference on Logic Programming (ICLP 2009), Pasadena, California, July 2009. *On LP defaults approach.*
  - Also:
    - S. Liang, P. Fodor, H. Wan, M. Kifer, OpenRuleBench: An Analysis of the Performance of Rule Engines, 18th International World Wide Web Conference (WWW 2009), Madrid, Spain, April 2009.
    - B.Grosof, Opportunities for Semantic Web knowledge representation to help XBRL, Position Paper, Workshop on Improving Access to Financial Data on the Web, Arlington, Virginia, October 2009.

Thank You

Disclaimer: The preceding slides represent the views of the authors only.
All brands, logos and products are trademarks or registered trademarks of their respective companies.

END OF ALL SLIDES

Disclaimer: The preceding slides represent the views of the authors only.
All brands, logos and products are trademarks or registered trademarks of their respective companies.

194