Overview of Semantic Web Services

Presentation for
Semanticc Web Services Coalition (a.k.a. Initiative)
March 18, 2003, by teleconference

Prof. Benjamin Grosof

MIT Sloan School of Management

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

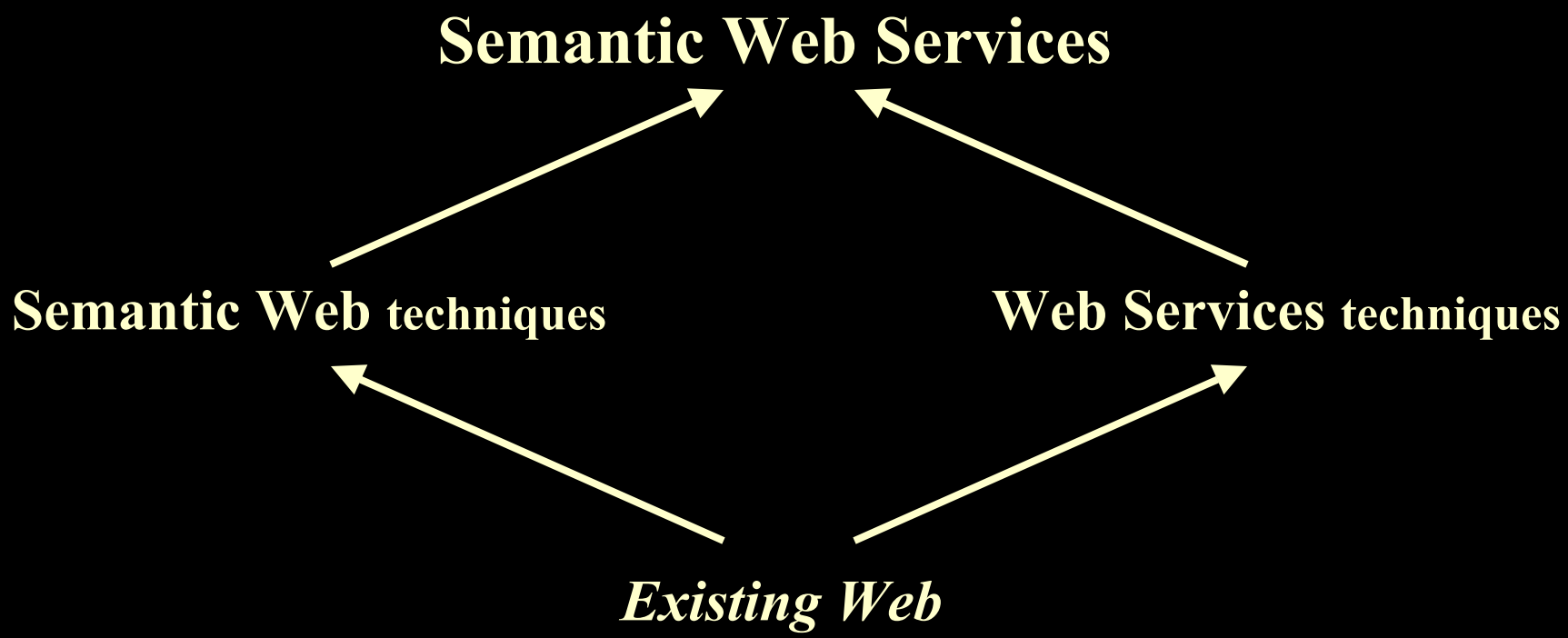
Outline

- Semantic Web Services (SWS) overview:
  - definitional, tasks, visions *(10 slides)*
- OPTIONAL Slides -- overviews of:
  - Rule-based SWS
  - Web Services
  - Semantic Web
  - RuleML

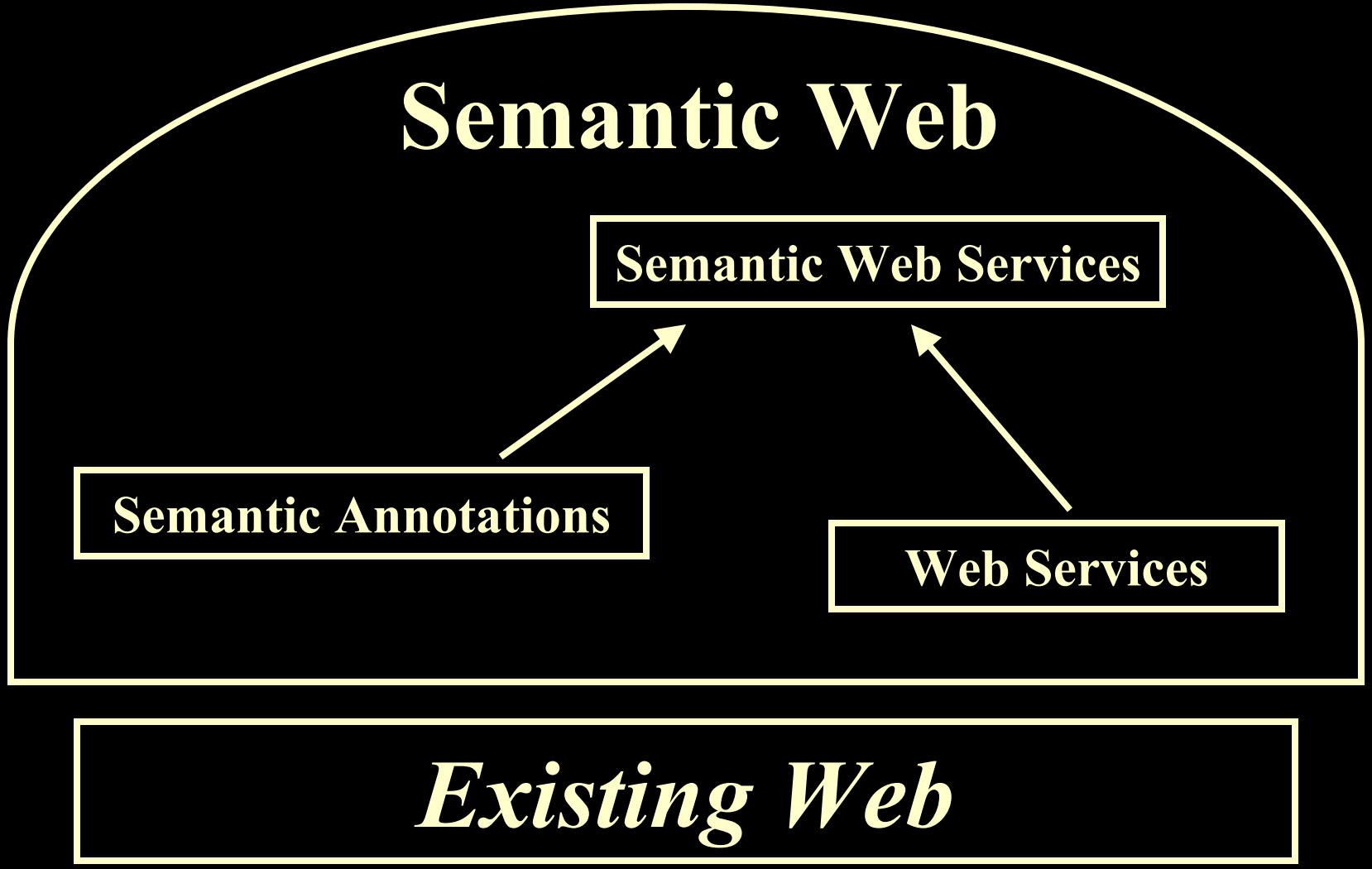
Semantic Web Services

- 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
  - Integrated knowledge
- (Semantic Web) Services: e.g., infrastructural
  - Knowledge/info/DB integration
  - Inferencing and translation

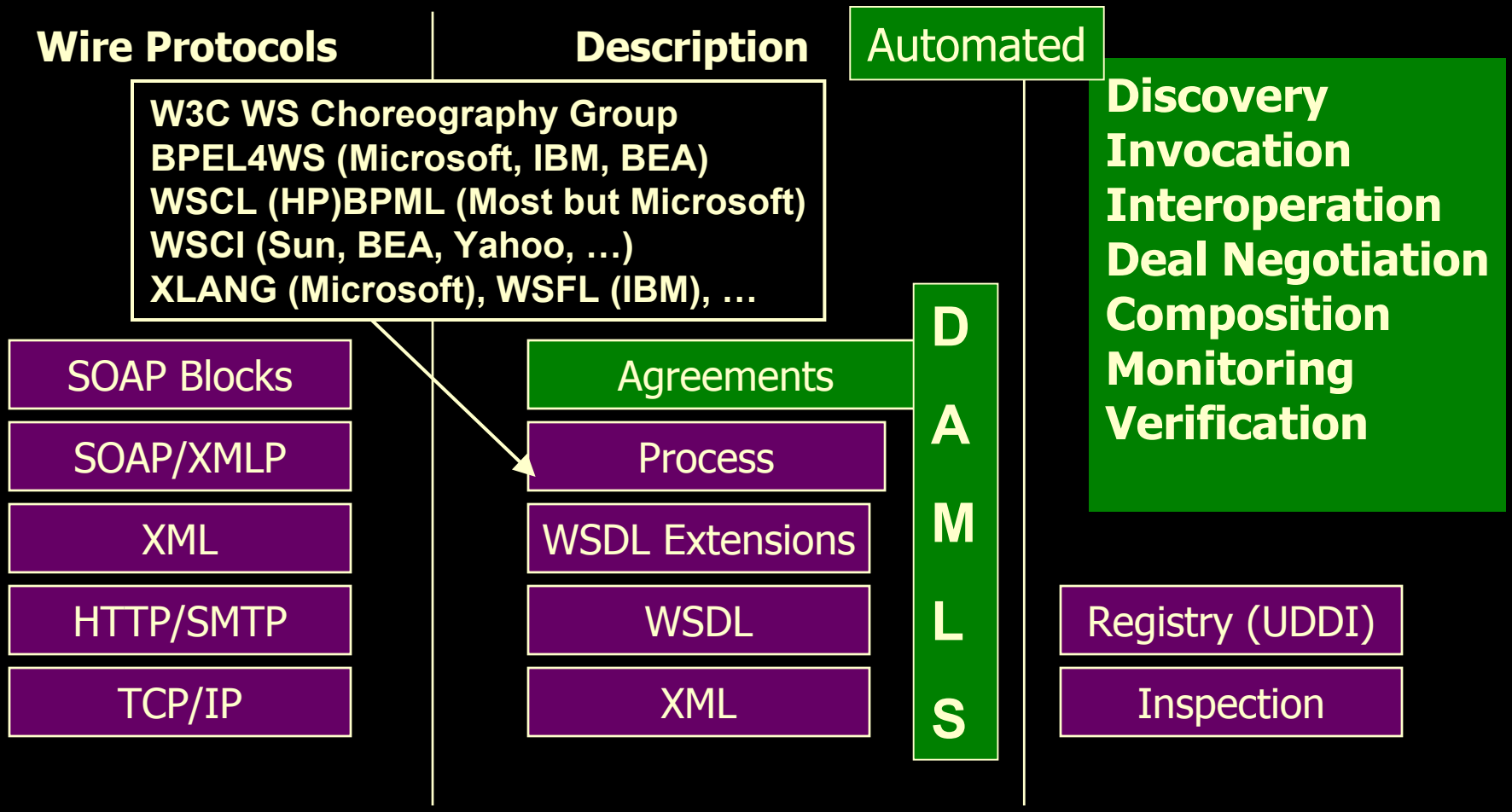
Next Generation Web



Vision of Semantic Web Includes Services



Current Web Services Standards Stack; Context for Semantic Web Services



[Slide co-authors: Sheila McIlraith (Stanford) , David Martin (SRI International), James Snell (IBM)]

SWS Tasks at higher layers of WS stack

Automation of:

- Web service discovery

  Find me a shipping service that will transport frozen vegetables from San Francisco to Tuktoyuktuk.

- Web service invocation

  Buy me "Harry Potter and the Philosopher's Stone" at www.amazon.com

- Web service deals, i.e., contracts, and their negotiation

  Propose a price with shipping details for used Dell laptops to Sue Smith.

- Web service selection, composition and interoperation

  Make the travel arrangements for my WWW11 conference.

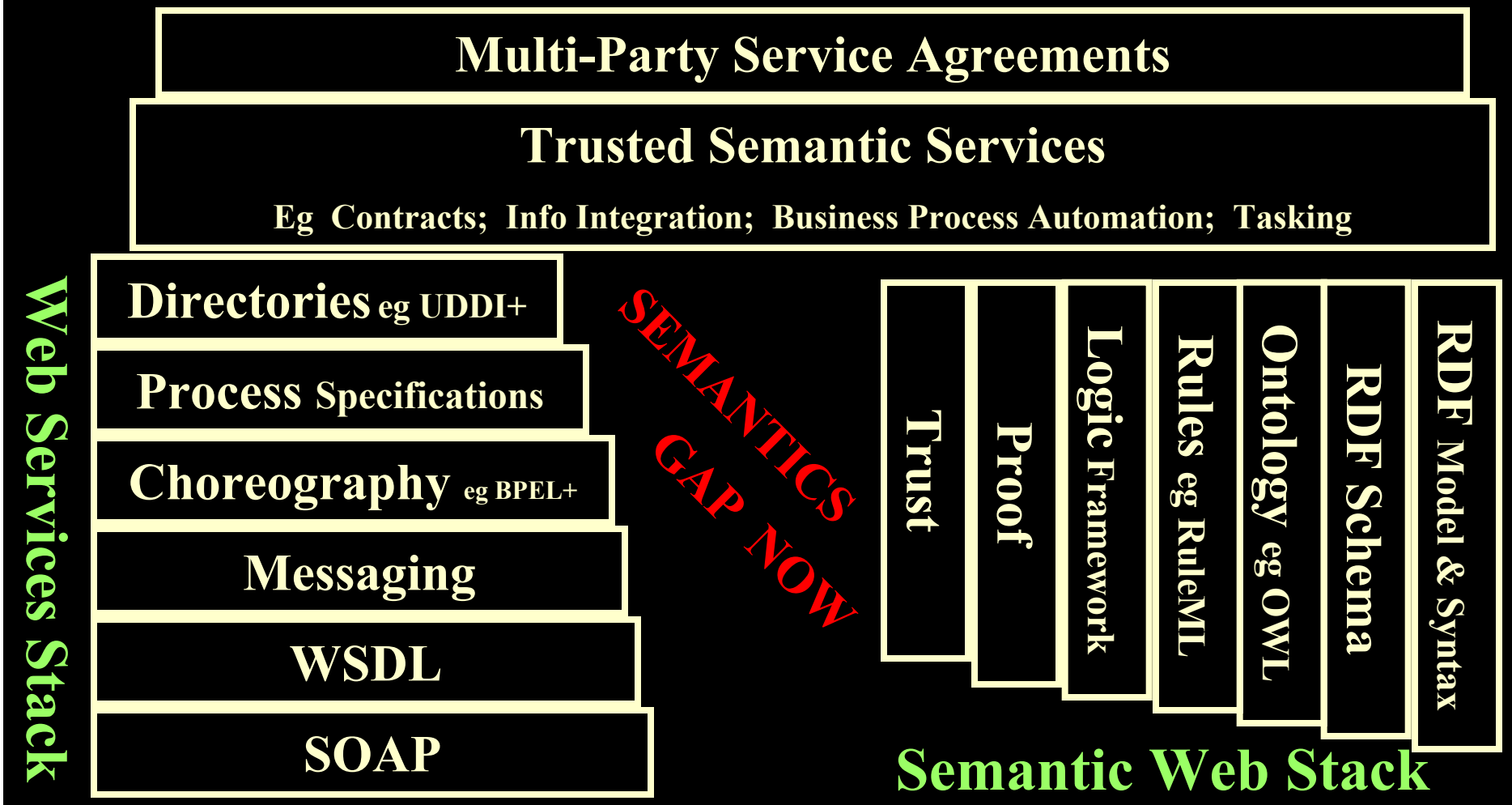
[Modification of slide also by Sheila McIlraith (Stanford) and David Martin (SRI International)]

SWS Tasks at higher layers of WS stack, continued

- Web service execution monitoring and problem resolution
  Has my book been shipped yet? … [NO!] Obtain recourse.
- Web service simulation and verification
  Suppose we had to cancel the order after 2 days?
- Web service executably specified at "knowledge level"
  The service is performed by running the contract ruleset through a rule engine.

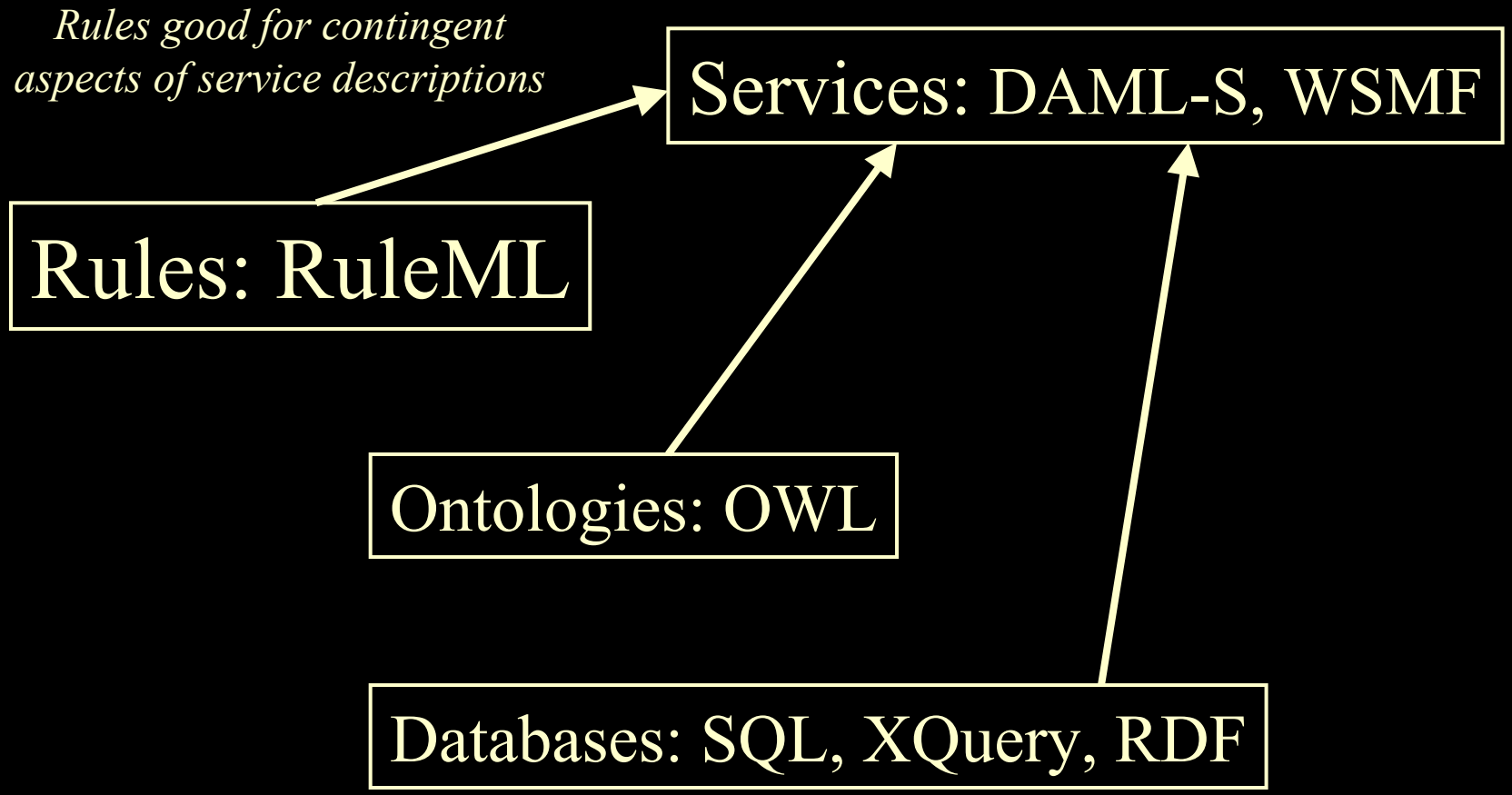
[Modification of slide also by Sheila McIlraith (Stanford) and David Martin (SRI International)]

Semantic Web Services Stack Diagram



3/16/2003

Vision: Semantic Web and Web Services Use DB's, Ontologies, and Rule Systems



Background – Vision:

Web is becoming XML → the Semantic Web

- XML (vs. HTML) offers much greater capabilities for structured detailed descriptions that can be processed automatically.
  - Eases application development effort for assimilation of data in inter-enterprise interchange
  - A suite of open standards both current and emerging
  - … including for knowledge-level SEMANTICS
- *Soon, Agents will Talk according to these standards…*
  - ∴ potential to revolutionize *interactivity* in Web marketplaces
    - B2B, …
- HTML itself is becoming XHTML: just a special case of XML

Background: Vision of Evolution: Agents in Knowledge-Based E-Markets

Coming soon to a world near you:…

- billions/trillions of agents (= k-b applications)
- …with smarts: knowledge gathering, reasoning, economic optimization
- …doing our bidding
  - but with some autonomy
- *A 1st step: ability to communicate with sufficiently precise shared meaning… via the SEMANTIC WEB*

OPTIONAL SLIDES FOLLOW:

Overview of Rule-based Semantic Web Services

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.

Rule-based Semantic Web Services

- Rules often good to executably specify service process models
  - e.g., business process automation using procedural attachments to perform side-effectful/state-changing actions ("effectors" triggered by drawing of conclusions)
  - e.g., rules obtain info via procedural attachments ("sensors" test rule conditions)
  - e.g., rules for knowledge translation or inferencing
  - e.g., info services exposing relational DBs
- Infrastructural: rule system functionality as services:
  - e.g., inferencing, translation

Application Scenarios for Rule-based Semantic Web Services

- SweetDeal [Grosof & Poon 2002] configurable reusable e-contracts:
  - LP rules about agent contracts with exception handling
  - … on top of DL ontologies about business processes;
  - *a scenario motivating DLP*
- Other:
  - Trust management / authorization (Delegation Logic) [Li, Grosof, & Feigenbaum 2000]
  - Financial knowledge integration (ECOIN) [Firat, Madnick, & Grosof 2002]
    - Rule-based translation among contexts / ontologies
    - Equational ontologies
  - Business policies, more generally, e.g., privacy (P3P)

MORE OPTIONAL SLIDES FOLLOW:

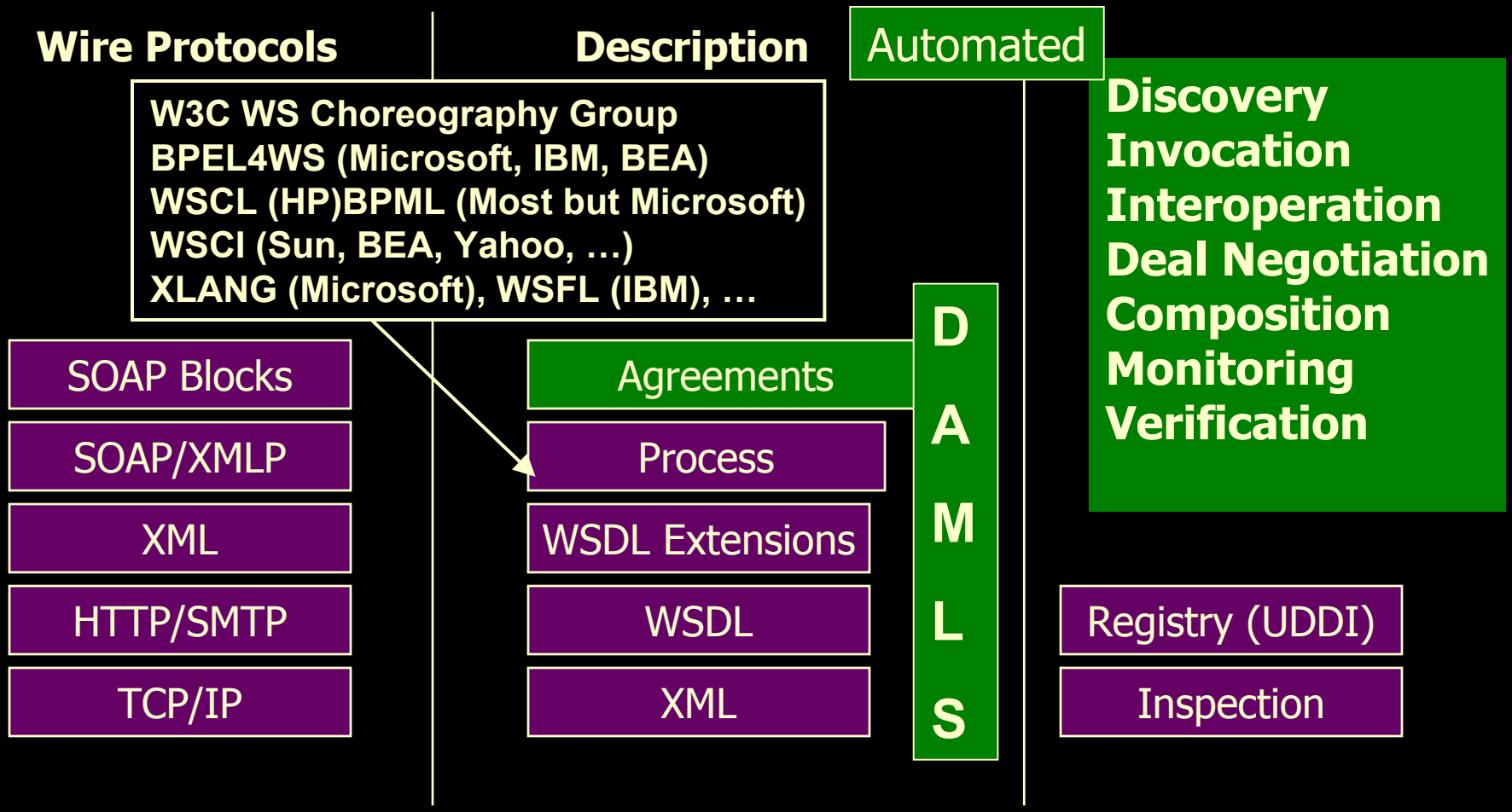
Overview of Web Services

Web Service -- definition

- *(For purposes of this talk:)*

- A procedure/method that is invoked through a Web protocol interface, typically with XML inputs and outputs

Current Web Services Standards Stack; Context for Semantic Web Services



[Slide co-authors: Sheila McIlraith (Stanford) , David Martin (SRI International), James Snell (IBM)]

WS Stack: some Acronym Expansion

- SOAP = simple protocol for XML messaging
- WSDL = protocol for basic invocation of Web Services, their input and output types in XML
- Choreography = higher-level application interaction protocols in terms of sequences of exchanged message types, contingent branching
  - Currently morphing into a W3C activity
- "Agreement" here = agreement between invoker and provider of the service, described at knowledge level
- *Overall: lots of proprietary jockeying and de-facto mode testing/pressuring of the open-consortial standards bodies (e.g., of W3C) "riding the tiger"*

WS Players

- Basically, all the major software vendors
  - Biggies: Microsoft, IBM, Oracle, Sun, SAP, …
  - Webserver/XML ebiz space: BEA, CommerceOne, Ariba, …
  - Niche offerings, e.g., travel agent services, weather, …
- Standards bodies: W3C; Oasis incl. Security
- Overall: lots of proprietary jockeying and *de-facto* mode testing/pressuring of the open-consortial standards bodies (e.g., of W3C) "riding the tiger"
- Still low-level in terms of application abstractions

MORE OPTIONAL SLIDES FOLLOW:

Overview of Semantic Web

The Semantic Web

The 1st generation, the Internet, enabled disparate machines to exchange data.

•The 2nd generation, the World Wide Web, enabled new applications on top of the growing Internet, making enormous amounts of information available, in human-readable form, and allowing a revolution in new applications, environments, and B2C e-commerce.

•The next generation of the net is an "agent-enabled" resource (the "Semantic Web") which makes a huge amount of information available in machine-readable form creating a revolution in new applications, environments, and B2B e-commerce.

…by enabling "agent" communication at a Web-wide scale.

Web is becoming XML → the Semantic Web

- XML (vs. HTML) offers much greater capabilities for structured detailed descriptions that can be processed automatically.
  - Eases application development effort for assimilation of data in inter-enterprise interchange
  - A suite of open standards both current and emerging
  - … including for knowledge-level SEMANTICS
- *Soon, Agents will Talk according to these standards…*
  - ∴ potential to revolutionize *interactivity* in Web marketplaces
    - B2B, …
- HTML itself is becoming XHTML: just a special case of XML

Vision of Evolution: Agents in Knowledge-Based E-Markets

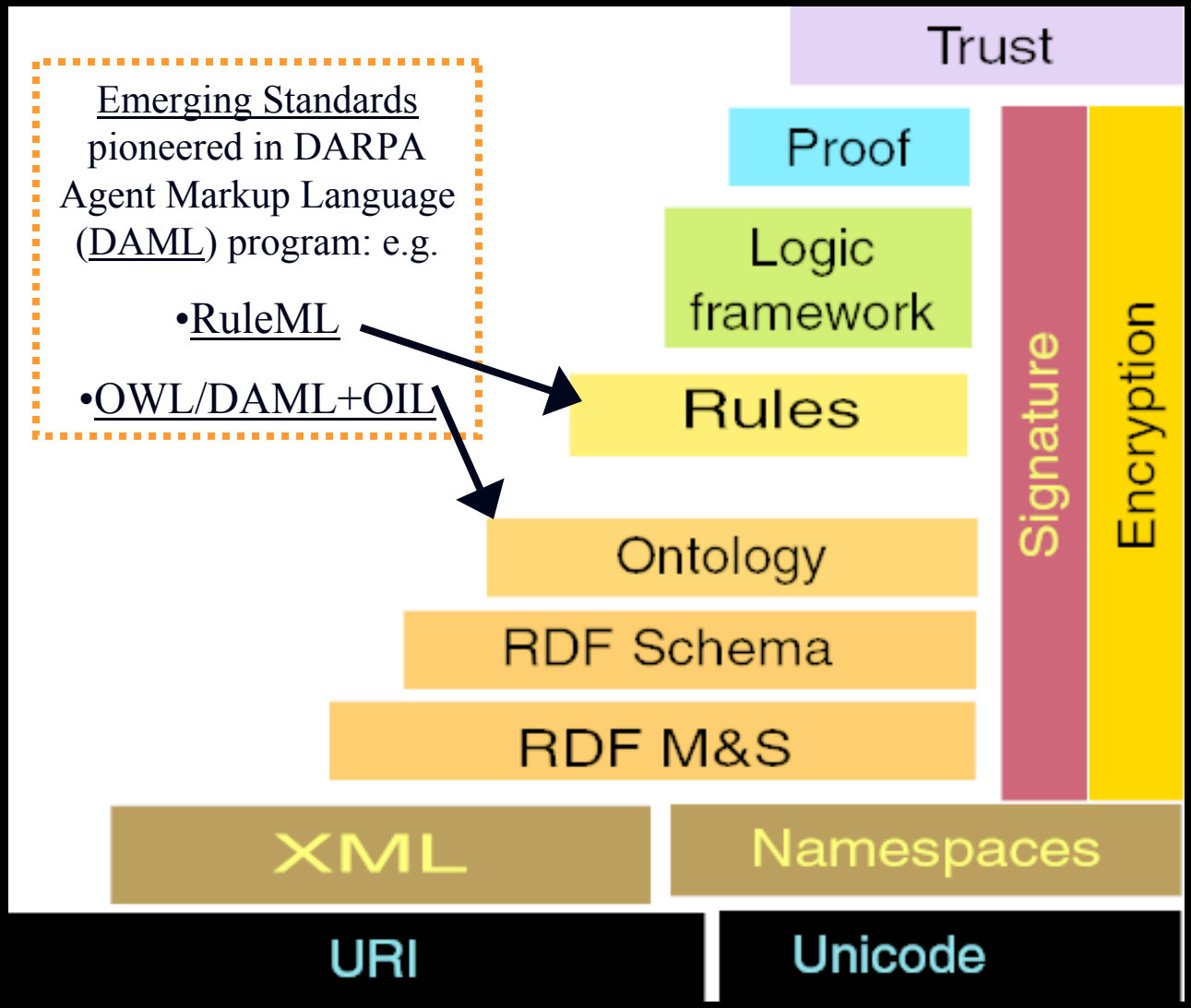
Coming soon to a world near you:…

- billions/trillions of agents (= k-b applications)
- …with smarts: knowledge gathering, reasoning, economic optimization
- …doing our bidding
  - but with some autonomy
- *A 1st step: ability to communicate with sufficiently precise shared meaning… via the SEMANTIC WEB*

SW: Research Players

- US: DARPA Agent Markup Language Program (DAML) program
- EU: OntoWeb program
- @MIT:
  - Sloan IT group: Grosof, Madnick, Firat, Klein, *et al*
  - LCS / W3C advanced-dev.: Berners-Lee, *et al*
- Number of companies:
  - HP, IBM, Adobe, Oracle, …

Semantic Web "Stack": Standardization Steps



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

SW Stack: Acronym Expansion

- W3C = World Wide Web Consortium: umbrella standards body
- XML-S: XML Schema, i.e., basic XML spec
- RDF: Resource Description Framework:
  - W3C Working Group
  - Labelled directed graph syntax
  - Good for building knowledge representation on top of: simpler, more powerful than basic XML
  - M&S = Model and Syntax
  - RDF Schema = extension: simple class hierarchies
- Ontology = formally defined vocabulary & class hierarchy, generalizes Entity-Relationship models
  - OWL = W3C Web Ontologies Working Language
  - … based closely on DAML+OIL

SW: Standards Players

- US-EU Joint Committee:
  - Early standards drafting
  - 1st focus: ontologies: DAML+OIL → W3C OWL
  - 2nd focus (current): rules: RuleML
- W3C: Semantic Web Activity
- Oasis: various incl. Security
- New efforts (currently in formation):
  - US-EU Joint Committee on Semantic Web Services
  - ISO: CommonLogic first-order logic (formerly KIF)

RDF vs. XML

- RDF original goals were: (1) to represent open meta-data created by multiple authors over the Web to describe and annotate arbitrary Web resources, e.g., whole websites, e.g., for use by digital librarians; and (2) to facilitate logical KR / SW. Lots of emphasis on an abstract data model.
- XML original goals largely to facilitate human-read document processing, then later to accommodate structured data cf. databases. More initial focus on syntax than on its own data model.

RDF vs. XML, continued

- XML document is a: labelled directed graph that is also:
  - Ordered (sequence of children matters)
  - Tree (a restriction!)
- RDF analogue of an XML document is a set of arcs ("triples"); this is a labelled directed graph that is:
  - Unordered (but can declare explicit order where need)
    - Good for general data modeling, e.g., in mainstream software engineering
  - NB: Cycles permitted (no tree restriction)
- RDF also permits:
  - "Reification", i.e., naming of an RDF triple so that it can be a node in another RDF triple.
- RDF encourages the nodes and arc labels to be URI's themselves. XML less general and open in this regard – it's clumsier, is one way to view it.

RDF vs. XML, continued more

- RDF adoption is much much less (yet) than XML.
- RDF is usually used with a particular XML syntax, but there are several for it.
- RDF's specification, and more importantly the theoretical understanding that underpins it, are not quite finished yet.
- RDF/RDF-Schema also includes some treatment of types and classes; XML Schema does too in a more practical manner.
- RDF and XML will probably be converged in the next several years – their data models are fairly close already. But XML has lots of inertia so far.

OWL: SW ontologies KR standard

- Draft Standard of W3C Web Ontologies Working Group (only about a year old), closely based on DAML+OIL precursor from research community. Uses RDF as syntax, extends RDF Schema.
- Based on Description Logic, a logical KR that has subset of expressiveness of first-order classical logic.
- Enables one to represent class hierarchies plus some more expressiveness, e.g., about cardinalities of properties and overlaps of classes.
- Still needs more theoretical and practical work to interoperate and bridge with conventional database schemas (e.g., Entity-Relationship (E-R) models and UML and SQL) and software engineering inheritance (e.g., class hierarchies in object-oriented (OO) langauges such as Java and C++).
- Description Logic's commercial adoption, deployment, and application is much much less (yet) than Rules', and hugely less than OO/E-R/UML/SQL.

SW: Standards Players

- US-EU Joint Committee:
  - Early standards drafting
  - 1st focus: ontologies: DAML+OIL → W3C OWL
  - 2nd focus (current): rules: RuleML
- W3C: Semantic Web Activity
- Oasis: various incl. Security
- New efforts (currently in formation):
  - US-EU Joint Committee on Semantic Web Services
  - ISO: CommonLogic first-order logic (formerly KIF)

SW-Related: XML Query Languages

- Goals
  - a data model for generic "natively" XML documents,
  - a set of query operators on that data model,
  - and a query language based on these query operators
  - Queries operate on single documents or fixed collections of documents.
- What SQL is for relational databases, XML Query languages are for collections of XML docs.
- There is a standard: W3C's XML Query Working Group
  - (W3C = World Wide Web Consortium)
- Oracle, IBM, Microsoft, etc. already support some
  - Not taking off quickly – complex spec

MORE OPTIONAL SLIDES FOLLOW:

Overview of RuleML

Flavors of Rules Commercially Most Important today in E-Business

- E.g., in OO app's, DB's, workflows.

- Relational databases, SQL: Views, queries, facts are all rules.
  - SQL99 even has recursive rules.
- Production rules (OPS5 heritage): e.g.,
  - Blaze, ILOG, 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.)*

Vision: Uses of Rules in E-Business

- Rules as an important aspect of coming world of Internet e-business: rule-based business policies & business processes, for B2B & B2C.
  - represent seller's offerings of products & services, capabilities, bids; map offerings from multiple suppliers to common catalog.
  - represent buyer's requests, interests, bids; → matchmaking.
  - represent sales help, customer help, procurement, authorization/trust, brokering, workflow.
  - high level of conceptual abstraction; easier for non-programmers to understand, specify, dynamically modify & merge.
  - executable but can treat as data, separate from code
    - potentially ubiquitous; already wide: e.g., SQL views, queries.
- Rules in communicating applications, e.g., embedded intelligent agents.

Why Standardize Rules Now?

- 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, app. dev. tools
- *⇒⇒ Identified as part of mission of the W3C Semantic Web Activity*

Overview of RuleML Today

- RuleML Initiative (2000--)
  - Dozens of institutions (~35), researchers; esp. in US, EU
  - Mission: Enable semantic exchange of rules/facts between most commercially important rule systems
  - Standards specification: 1st version 2001; basic now fairly stable
  - A number of tools (~12 engines, translators, editors), demo applications
  - Successful Workshop on Rules at ISWC was mostly about RuleML / LP
  - Has now a "home" institutionally in DAML and Joint Committee
    - Discussions well underway to launch W3C, Oasis efforts
- Initial Core: Horn Logic Programs KR

  …Webized (in markup)… and with expressive extensions

  URI's, XML, RDF, … *non-mon, actions, …*

Overview of RuleML Today, Continued

- Fully Declarative KR (not simply Prolog!)
  - Well-established logic with model theory
  - Available algorithms, implementations
  - Close connection to relational DB's; core SQL is Horn LP
  - *See [Baral & Gelfond '94] for good survey on declarative LP.*
- Abstract graph syntax
  - 1st encoded in XML…
  - … then RDF (draft), … then DAML+OIL (draft)
- Expressive Extensions incrementally, esp. already:
  - Non-monotonicity: Negation as failure; Courteous priorities
  - Procedural Attachments: Situated actions/effecting, tests/sensing
  - *In-progress:* Events cf. OPS5/Event-Condition-Action

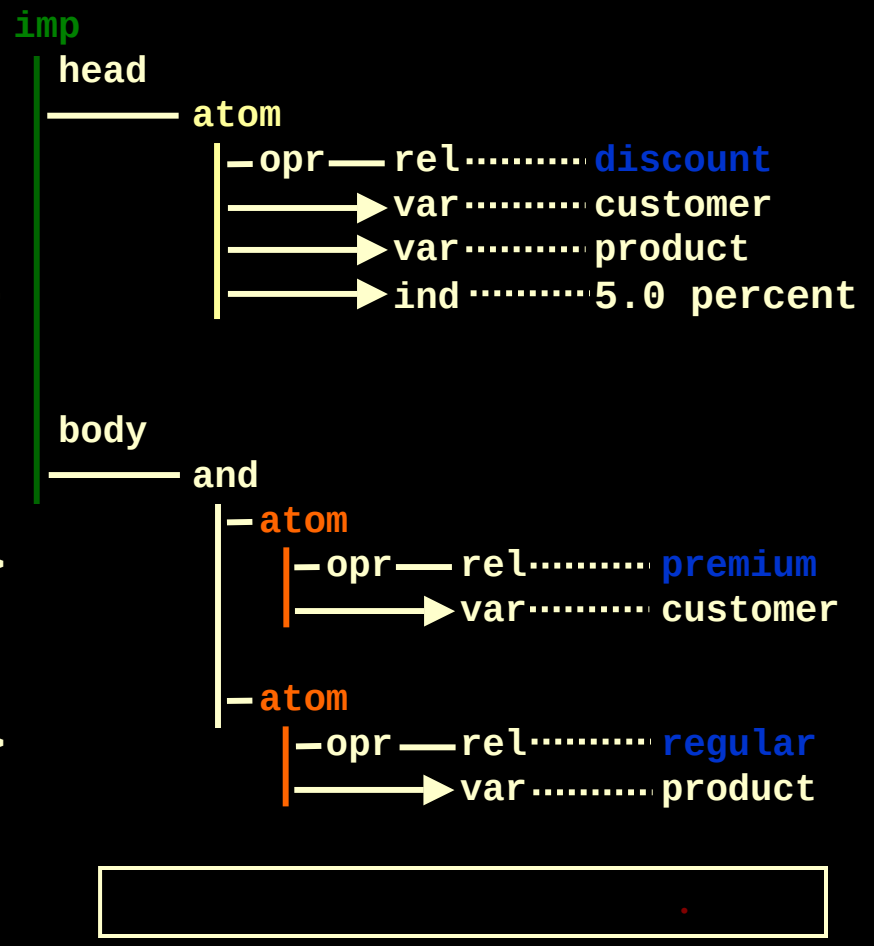
RuleML Example: Markup and Tree

"The **discount** for a *customer* buying a *product* is **5.0 percent**
if the *customer* is **premium** and the *product* is **regular**.',

discount(?customer,?product,"5.0 percent") ← premium(?customer) /\ regular(?product);

Markup Syntax

```
<imp>
  <_head>
    <atom>
     <_opr><rel>discount</rel></_opr>
     <tup><var>customer</var>
          <var>product</var>
          <ind>5.0 percent</ind></tup>
    </atom>
  </_head>
  <_body>
    <and>
      <atom>
        <_opr><rel>premium</rel></_opr>
        <tup><var>customer</var></tup>
      </atom>
      <atom>
        <_opr><rel>regular</rel></_opr>
        <tup><var>product</var></tup>
      </atom>
    </and>
  </_body>
</imp>
```



Technical Approach of RuleML: I

- 1. Expressively: Start with: Datalog Logic Programs *as kernel*
  - Rule := $H \leftarrow B1 \wedge \ldots \wedge Bk$; $k \geq 0$, H and Bi's are atoms.
    head if body ;
- Declarative LP with model-theoretic semantics
  - forward ("derivation"/ "transformation") and backward ("query") inferencing
- Rationale: captures well a simple shared core among CCI rule sys.
  - Tractable! (if bounded # of logical variables per rule)
- Horn LP -- differences from Horn FOL:
  - Conclusions are a set of ground atoms.
  - Consider Herbrand models only, *in typical usage.*
    - Can extend to permit equalities in rules/conclusions.
  - Rule has non-empty head, *in typical usage.*

Technical Approach of RuleML: II

- 2. Syntax: Permit rules to be labeled -- need names on the Web!
- 3. Syntax: Permit URI's as predicates, functions, etc. (names)
  - namespaces too
- 4. Expressively: Add: extensions cf. established research
  - negation-as-failure (well-founded semantics) -- in body *(stays tractable!)*
    - "Ordinary" LP (cf. declarative pure Prolog)
  - classical negation: limited to head or body atom – syntactic sugar
  - prioritized conflict handling cf. Courteous LP *(stays tractable!)*
    - modular rulesets; modular compiler to Ordinary LP
  - procedural attachments: actions, queries ; cf. Situated LP
  - 1st-order logic type expressiveness cf. Lloyd LP's – syntactic sugar
    - $\vee,\forall,\exists$ in body; $\wedge,\forall$ in head *(stays tractable!)*
  - logical functions (arity > 0)

Technical Approach of RuleML: III

- 5. Expressively: Add: restrictions cf. established R&D
  - E.g., for particular rule systems, e.g., Prolog, Jess, …
    - Also "pass-thru" some info without declarative semantics (pragmatic meta-data)
- 6. Syntax for XML:
  - Family of DTD's/Schemas:
    - a generalization-specialization hierarchy (lattice)
    - define DTD's modularly, using XML entities (~macros)
    - optional header to describe expressive-class using "meta-"ontology
- 7. Syntax: abstract unordered graph syntax (data model)
  - Support RDF as well as XML (avoid reliance on sequence in XML)
  - "Roles" name each child, e.g., in collection of arguments of an atom
  - Orderedness as optional special case, e.g., for tuple of arguments of an atom
- 8. Syntax: module inclusion: merge rulesets ; import/export
  - URI's name/label knowledge subsets

Tools: SweetRules, including SweetJess

- SweetRules V1 '01: RuleML inferencing and bi-directional translation with equivalent semantics via RuleML, between:
  - XSB Prolog: backward Ordinary Logic Programs (OLP)
  - Smodels: forward OLP
  - IBM CommonRules: forward Situated Courteous LP (SCLP)
  - Knowledge Interchange Format (KIF): First Order Logic interlingua
  - + *Design in principle for:* SQL
    - well-understood in theory literature: as OLP
  - + *Design in principle for:* production (OPS5), ECA
    - Based on Situated extension of LP, piloted in IBM Agent Building Environment '96 for info-workflow applications. Also piloted in EECOMS.
    - BUT: not much other literature/theory to support
    - HENCE motivation to "bring them to the party" … resulting in:
- …V2 '02: adds SweetJess as component:
  - Jess: production (OPS5) , close to ECA
    - popular, open-source, Java: it's useful in particular
    - expressive restriction: "all bound sensors"

SWEET =

Semantic WEb

Enabling Tools