OWL-S(ervices)

Planners with manners

(no more deadplan humor)

The Semantic Web

- "Next generation", "machine friendly" Web
  - Interlinked information for programs
- The "original vis…*Wait this sounds familiar!*
- Two steps beyond traditional Web content
  - Past Web *data* (XML)
  - To Web *knowledge* (RDF, OWL, and Beyond)
- But what about Web *behavior* (programs!)?
  - Java applets, Javascript, CGIs, etc.

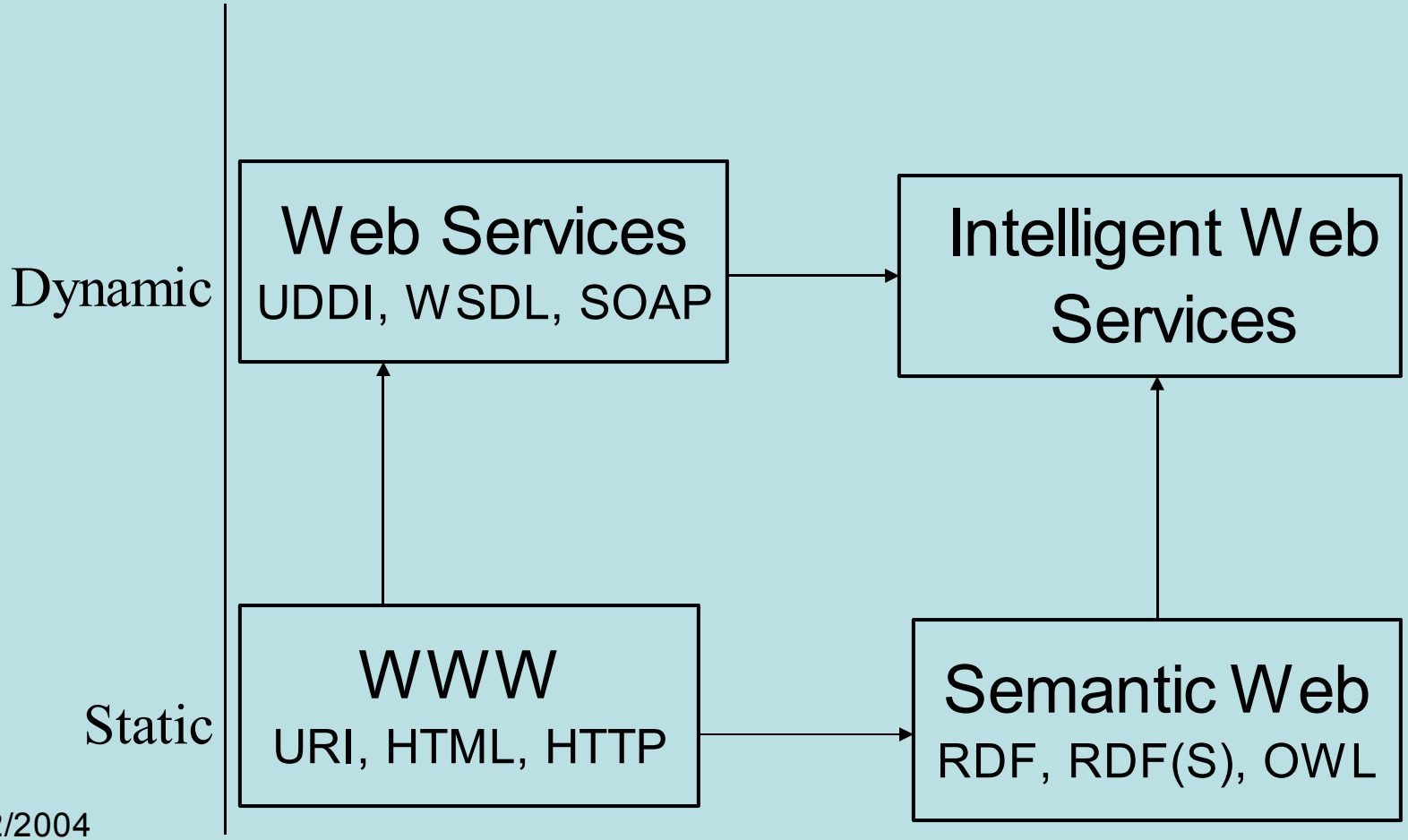
Web Services

- A competing vision
  - Programs who need programs
  - Components
  - Discover, manipulate, interact, react, etc. to *functionality*
- Data interoperability via XML and related standards
- Language/system/etc. interop achieved by *very* loose coupling

(Semantic (Web) Services)

- Services for the Semantic Web
  - Reasoners, datastores, planners, schedulers, etc.
- Semantic Web enabled Web Services
  - Services are complex entities
  - Automated manipulation of services requires rich descriptions and flexible “understanding” of the service, as well as ultimate users goals, preferences, etc.
- Agents reborn?

A Picture


Dynamic
Static
Web Services
UDDI, WSDL, SOAP
Intelligent Web Services
WWW
URI, HTML, HTTP
Semantic Web
RDF, RDF(S), OWL

Web Service Tasks

- Discovery and Selection
- Negotiation and Contracting
- Coordination
- Composition
- Execution, Monitoring, Simulation

OWL-S

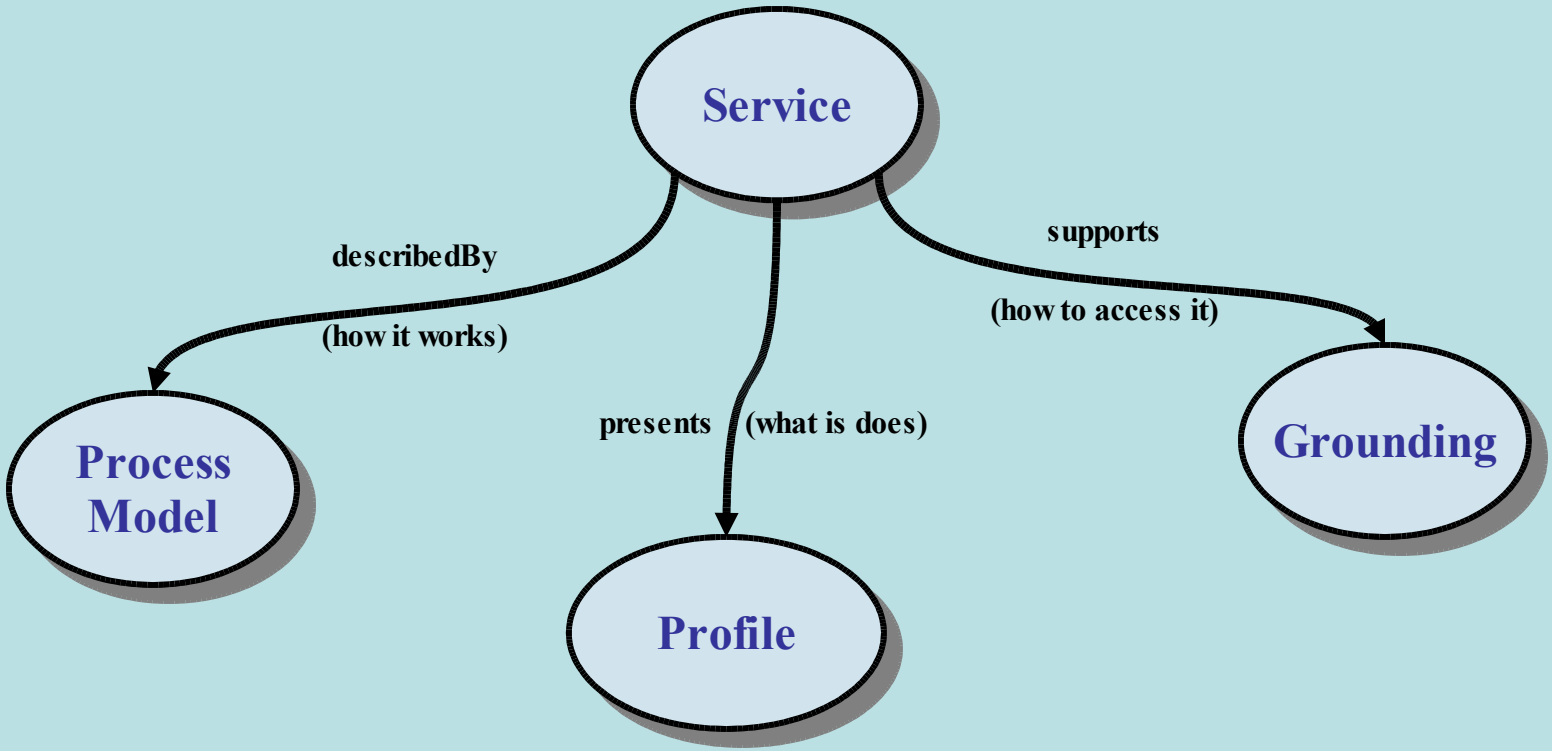
- A collection of foundational ontologies
  - *Intended* to be a framework
  - In practice, it has made significant choices
    - And what flexibilty has come from underspecification
  - OWL centered
    - Fluctuating between Full and DL
    - Quickly embracing much more expressivity (e.g., SWRL)
- *Encodes* what OWL cannot express
  - In particular, the Process Model
  - But plenty of other things!

Service

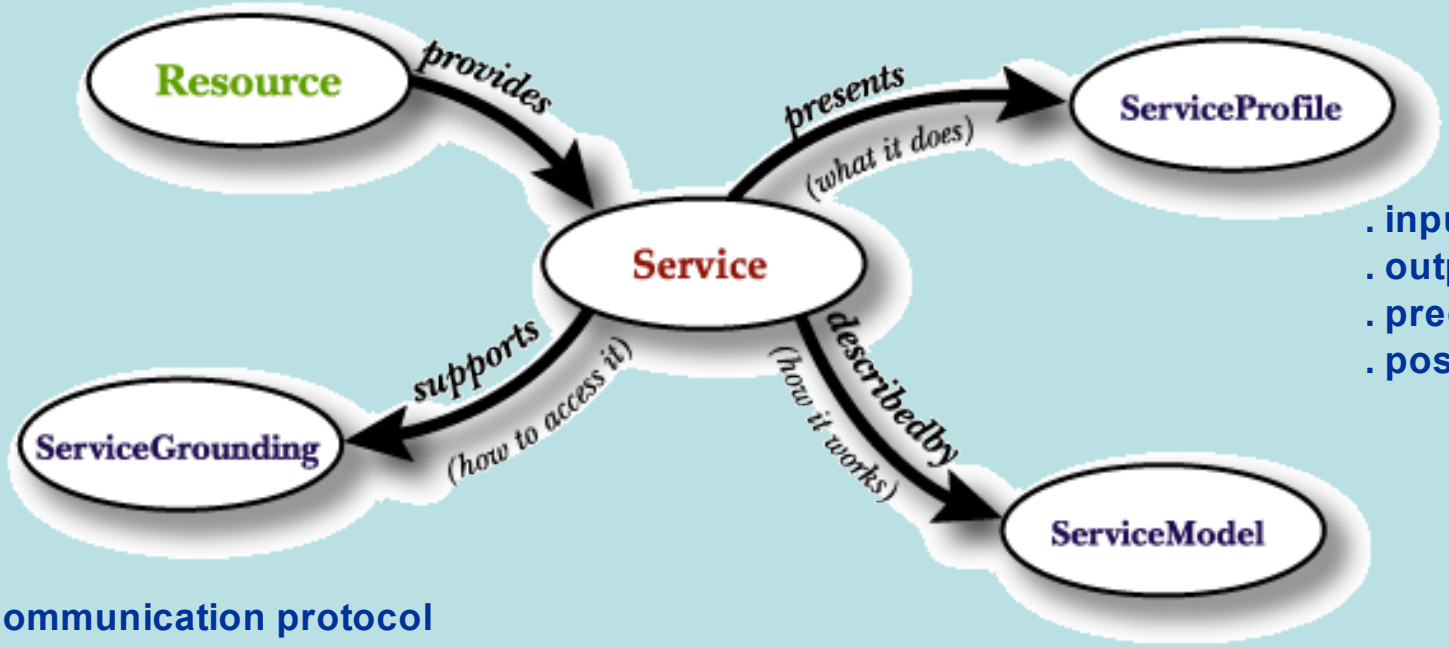
- Service is a specific functionality
  - Which might be quite complex
- In the ontology, Service is the “hook” to connect the various parts
  - And it’s the parts that are of interest
- To my knowledge, no one has used the Service Class in any significant way
- I find that curious

OWL-S Service Description

Three components of OWL-S descriptions



Someone Else's Drawing



. input types
. output types
. preconditions
. postconditions

. communication protocol
(RPC, HTTP, …)
. port number
. marshalling/serialization

• process flow
• composition hierarchy
• process definitions

Service Profile

- High-level description of a service
- Used for advertisements and requests
- A profile contains:
  - a human readable description of the service
  - functional attributes
    - Inputs, outputs, preconditions, effects
  - “non-functional” attributes
    - guarantees of response time or accuracy, cost of the service, etc.
- A profile is a *view* of the service

IOPEs

- Inputs, Outputs, Preconditions, and Effects
- Part, but not all, of the "behavioral signature" of the service
- Most (current) matchmaking done on IO, described with OWL Classes
- PE language just arriving
  - Preconditions: What must be true before I can invoke the service
  - Effects: Things the service makes true
  - Used in most planning
    - For goal directed planning (regression or progression)
    - For task directed planning (to guide decomposition)

IOs cont.

- Conceived as “knowledge” Pes
  - Things told to and from a service
- Typically corresponding to in and out messages (or some decomposition or synthesis of such)
- OWL classes as type system
  - Can use XML Schema, but discouraged
  - Indeed, XML types are treated as wire format
- Big issues with using OWL this way

Decker Problems

- OWL is based on open world assumptions
  - Just because you don't know, don't mean it's false
  - Absence of information doesn't cause (necessarily) cause problems
- OWL is first order and inference directed
  - No bound on the "relevant" information
- No data validation!
  - Easy to have too much or too little information!

PEs encore

- Preconditions and effects are described by *formulas* in some logic language
  - The 1.1 default is SWRL
  - But not quite SWRL
    - A precondition (or effect) expression is a conjunction of SWRL atoms (I.e., 1- and 2-place predicates with variables)
    - The variables can be bound in funny ways
    - Deletion is problematic
      - Inferred assertions
      - Even if deleted, perhaps merely unknown
    - Query is expensive
      - DL satisfiabilty test (potentially) for each permuation of individuals!
      - Expect very large KBs
  - Not clear service descriptions need PEs

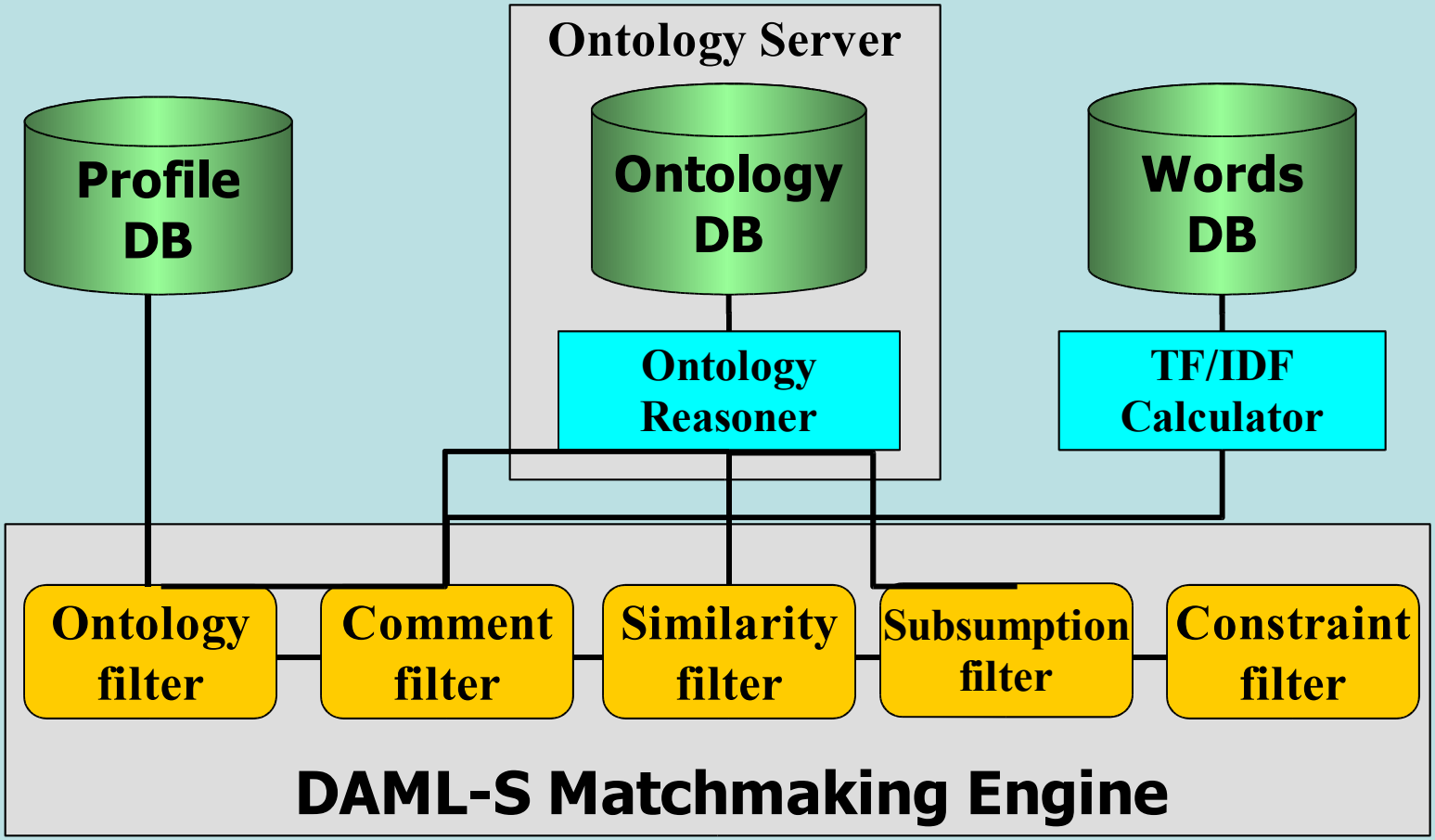
ServiceParamters

- Top Down vs. Bottom up
  - Describe the services with their properties
  - Build up requests as class descriptions
  - Can have many class hierarchies and organizations
  - Can be “local” in both the service descriptions and the class expressions
- Taxonomies require global organization
  - DL classes are more like self-organizing queries
- BUT! Matchmaking can be tricky

Matchmaking

- Primary current mechanism is subsumption
  - Or, perhaps, other DL based inferences
- Should the request be more or less general than the match?
- Standard categorization
  - Exact
  - Plug-in
  - Relaxed

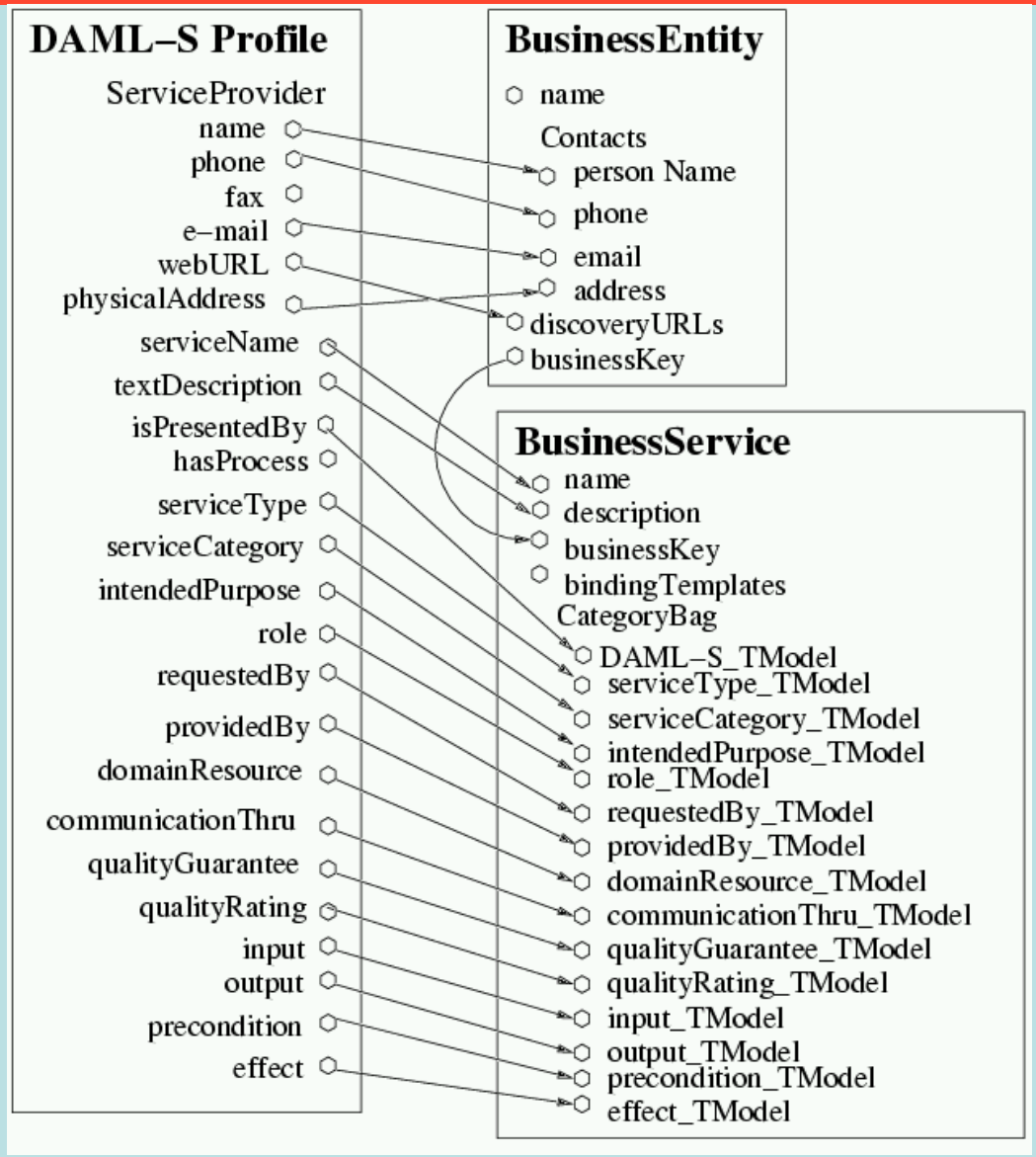
DAML-S Matchmaker



UDDI Components

- White Pages
  - contains business name, text description, contact info and other related info.
- Yellow Pages
  - contains classification information about the business entity and types of the services the entity offers
- Green Pages
  - contains information about how to invoke the offered services

OWL-S to Tmodels



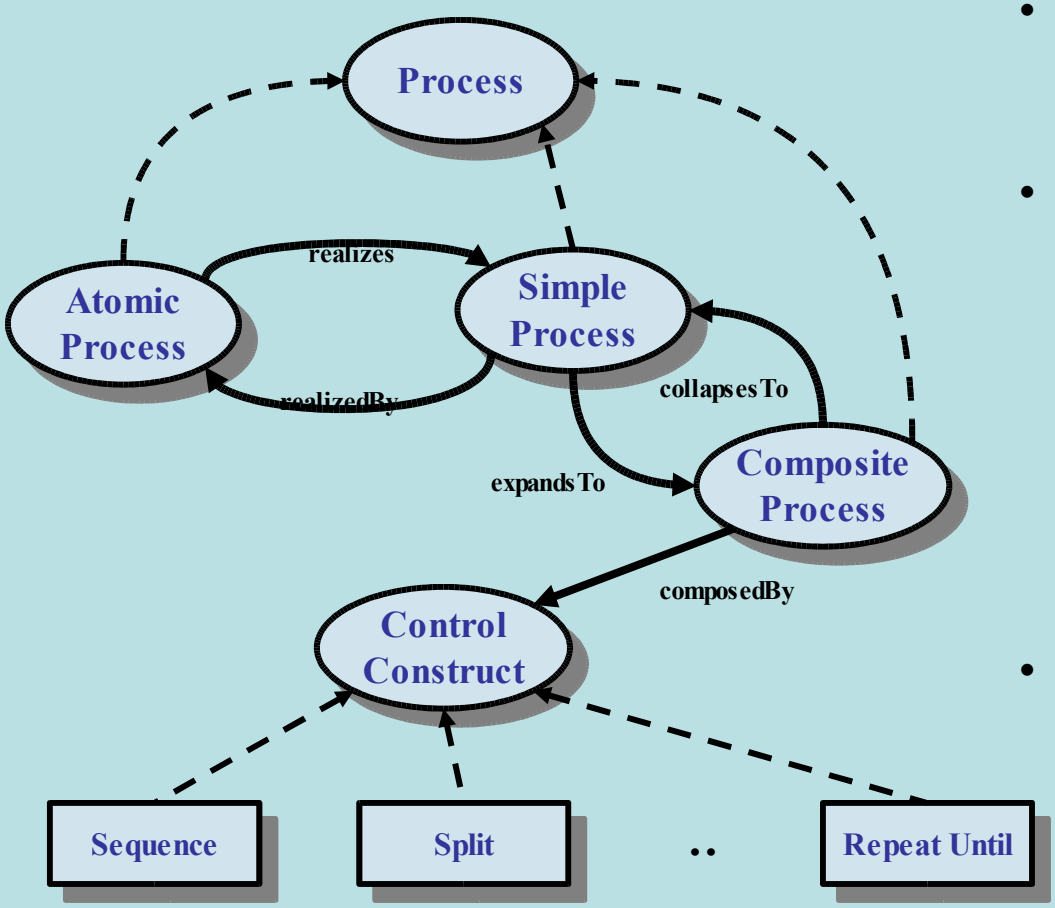
Matchmaking/discovery encore

- Feels like a big, easy win
  - UDDI TC solicit input
  - RDF & OWL all about metadata, right?
- But where is the win?
  - Serious dearth of success stories
  - Matching algorithms perhaps overhyped
    - Similarity measures seem more appropriate
  - Matching over what parts of the Service description?
- Negotiation seems critical
- Simple query broading seems useulf

From Discovery to Composition

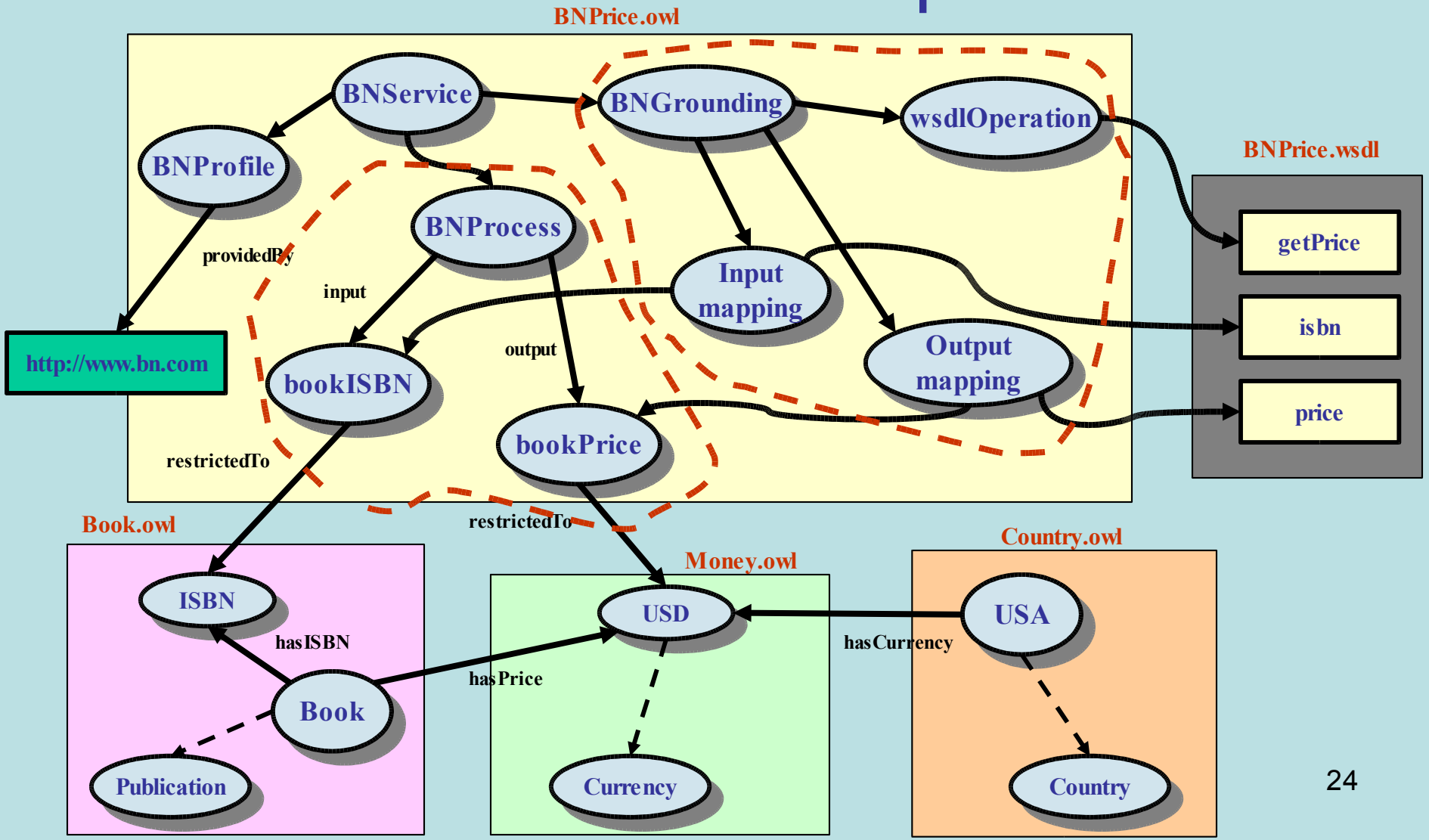
- Thus far, discussed finding existing, complete, existing services
  - But what if there is no service that does what you want?
  - But you have a rich description of what you want?
  - And there are combinations of services that achieve your desires?
- Discovering *virtual*, dynamically composed services

Process Model



- Atomic processes
  - directly invocable
  - black box
- Composite processes
  - consists of other processes
  - defined by a control construct
    - Sequence
    - Split
    - …
    - RepeatUntil
- Simple processes
  - abstract views, not executable
  - atomic process without a grounding
  - simplified representation of a composite process

OWL-S Example



ParameterTypes

- Pushes us toward OWL Full
  - Processes are instances
  - They have parameters
  - Which “have” (are related to) types
  - Which are classes….oops
- Unclear how type checking should work
  - Preliminary efforts in terms of execution traces
  - Requires enormous amount of modeling
  - Perhaps requires rules or other extra DL reasoning behavior

CompositeProcesses

- Primarily derived from the Golog work from University of Toronto and Stanford
- Situation Calculus based
- Complex Actions/Processes are “macro” expansions for a large number of sitcalc axioms
- Compile those axioms to a Prolog program
- Bit like HTN, but rather idiosyncratic!

Orchestration or Choreography?

- Or both!
- There were presumptions and assumptions
- These Ps & As were more or less ratified
  - Partly inherited from WSDL of the RPC days
  - Planning tends to be “central control” oriented
    - But, really, that can work either way
- Trying to be a better BPEL
  - Is this a sane strategy?
- The Irreality of Control Constructs and CompositeProcesses

SimpleProcesses

- Dark Horse
  - OWL-S has at least 4 abstraction points
    - WSDL
    - Grounding
    - Profile
    - SimpleProcess
  - Probablly essential for Process Templetes
    - Can stand for Atomic or Composite process
    - Could have constraints and inferred replacements
    - Would need lots of stuff from Profile

Planning

- Given a state of the world, a goal, and a domain, find a sequence of actions that achieves the goal
- State of the world == (RDF/OWL KB)
- Goal == (RDF/OWL KB (but small))
- Domain
  - set of operators, i.e., primative tasks, i.e., AtomicProcesses
  - Set of methods, i.e., task *decompositions*, i.e., CompositeProcesses
- Planning proceeds by replacing tasks in the task lists with their decomposition until you have a list of primative tasks (the plan)

HTN Planning

- Given a state of the world, a task list, and a domain, find a sequence of actions that achieves the tasks
- State of the world == (RDF/OWL KB)
- Goal == (RDF/OWL KB (but small))
- Domain == set of operators, i.e., AtomicProcesses

Information Gathering

- During plan time or during execution time?
  - Planning for sensing
    - Contingency/conditional plans
    - On line planning
    - Recovery and replanning
  - Planning for infogathering at plan time
- Services seem more info/computational
  - Traditional operators are more physical
  - Perhaps rethink what it is to plan

Grounding

- Specifies how to execute a service
- Each AtomicProcess has a grounding
- Current specification
  - Mapping to WSDL
  - AtomicProcess -> Operation
  - Input/Output -> Message Parts
- (New, WSDL 2.0 OWL/RDF coming soon)

WSDL2DAMLS

Translation and Data integration

- Heterogenuous information formats
  - Not just data formats, but concepts
  - Plus the concept/data divide
    - Currently use XSLT to/from RDF/XML
    - Clunky and hardcoded but solves many Decker problems
  - Ontology mapping
    - By compution or by inference
  - A service based approach
- Active mediation
  - Send the translation/massage closer to the data

Metaservices

- Services for services
  - Creating, destroying, moving, managing, discovering
  - UDDI good example
  - Mobile services?
- Where to specify?
  - ServiceParmeters seem to be the catch all

Acknowledgements

- Evren Sirin, for lifesaving slides, a noble effort, and general fellowship
- Terry Payne for more stolen slides than expected
- Jen Golbeck for slides I didn't really use but did crib some stuff from
- #mindswap for listening to my meltdown
- Aditya for a valient effort
- Me! For not breaking anything and having old but somewhat useful slides