TRANSLATOR:
A TRANSlator from LAnguage TO Rules

David Hirtle

Faculty of Computer Science
University of New Brunswick

March 24, 2006

Outline

- Introduction
- Input: Attempto Controlled English
- Translation
- Output: Rule Markup Language
- Future Work
- Conclusion

Introduction

- Semantic Web
  - machine understandability
  - standards: RDF, OWL … RIF
  - hasn't quite caught on
    - barrier to entry too high?

Users are expected to work with tools or rule languages which are transformed to and from this format.

- RIF Working Group Charter

Introduction

- “people axis”
  - vast majority are non-experts
  - user-friendly format needed
    - natural language, e.g., English
      - easy
      - familiar
      - expressive
      - *ambiguous*
    - controlled natural language to the rescue
      - formal, yet natural

Introduction

- TRANSLATOR
  - translates (controlled) English sentences into a formal representation for the Semantic Web
    - input: Attempto Controlled English (ACE)
    - output: Rule Markup Language (RuleML)
  - Java (Web Start)
  - uses Prolog-based APE web service

Attempto

- research project for past 10+ years
  - Dept. of Informatics, University of Zurich
  - still highly active
    - EU Network of Excellence REWERSE
- ACE, APE, RACE (Reasoning in ACE)
- more information
  - http://www.ifi.unizh.ch/attempto

Attempto Controlled English

- seems like English
- actually a formal language, like RDF
  - tractable subset of English
    - all ACE sentences are English, but not vice versa
  - unambiguously translatable into logic

Attempto Controlled English

- handling ambiguity
  - exclude imprecise phrasings
    - e.g., *Students hate annoying professors.*
  - interpretation rules
    - e.g., *The student brings a friend who is an alumnus and receives a discount.*
      - Who receives the discount?
        - » in ACE, student does (by default)
        - » repeat relative pronoun for other interpretation i.e., *The student brings a friend who is an alumnus and **who** receives a discount.*

Attempto Controlled English

- what kind of rules can be expressed?
  - what *is* a rule*?*
    - facts
      - e.g., *John is human.*
        - » Prolog representation: **`human(John).`**
    - rules
      - e.g., *All humans are mortal.*
        - » Prolog: **`mortal(X) :- human(X).`**
        - » “X is mortal if X is human”
      - allows inferring new facts
        - » e.g., **`mortal(John).`**

Attempto Controlled English

- what kind of rules can be expressed?
  - in natural language, many different forms
    - e.g., *Everyone is mortal.*
      All humanity is mortal.
      Every human being is mortal.
      For each person the person is mortal.
      If there is a member of the human race then he/she is mortal.
  - all above are valid ACE
  - further embellishment
    - negation, relative clauses, etc.

Attempto Controlled English

- what kind of rules can be expressed?
  - **not** “infix” implication
    - e.g., *The student is happy **if** there is no class.*
    - but, TRANSLATOR supports it
      - swap condition(s) and conclusion(s) before sending to APE
        - » result: *If there is no class then the student is happy.*
  - mostly derivation rules
    - not yet reaction rules, integrity constraints, etc.
    - same with RuleML

Discourse Representation Structures

- basis: Discourse Representation Theory
  - formal method of dealing with contextual meaning across multiple sentences
  - Hans Kamp (1981)
- DRS is syntactic variant of first-order logic
- APE outputs extended form of DRS
  - e.g., relations as arguments
- required for translation to RuleML

(ACE)

Every honest student who does not procrastinate receives a good mark and easily passes the course.

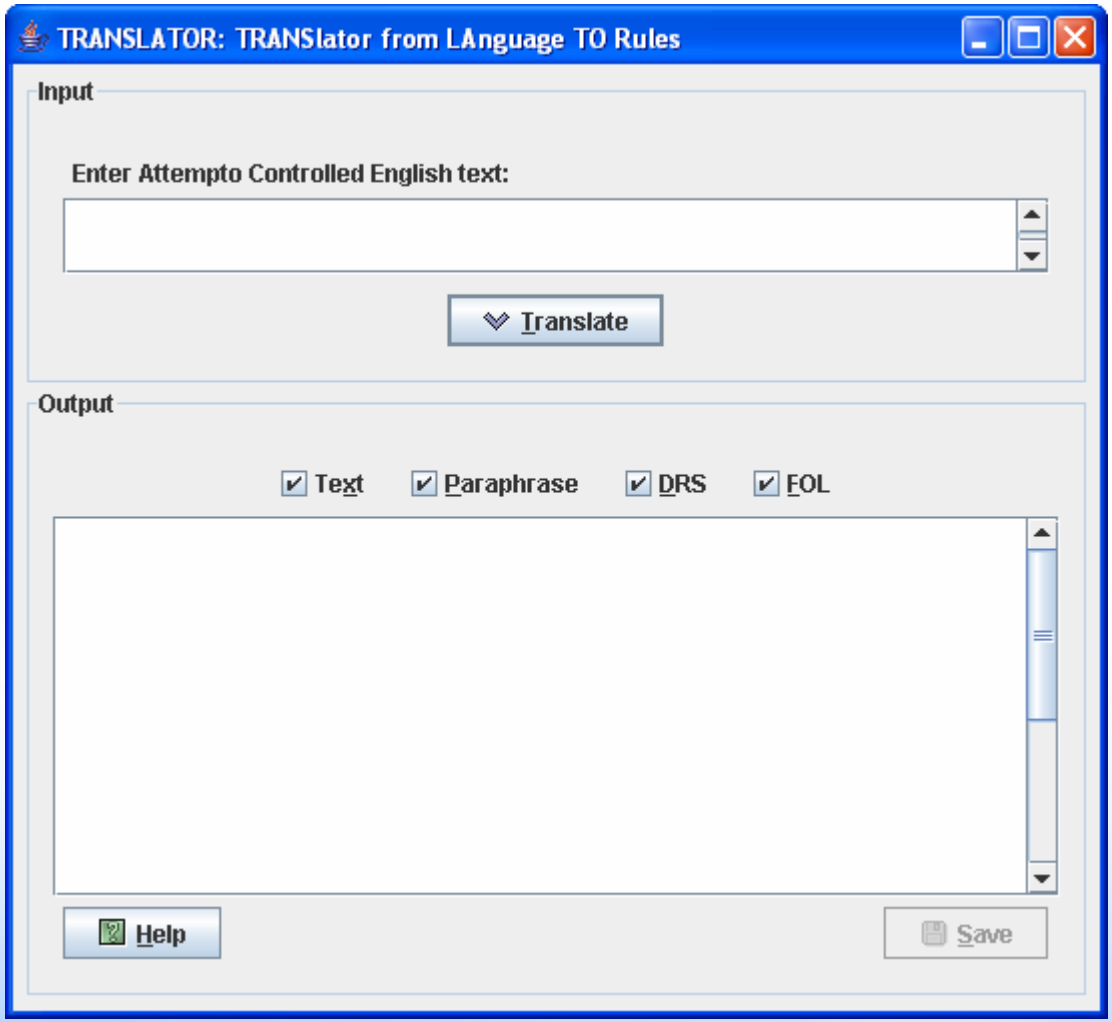
(DRS)

```
[]
   [A, B]
   object(A, student, person)-1
   quantity(A, cardinality, count_unit, B, eq, 1)-1
   structure(A, atomic)-1
   property(A, honest)-1
      NOT
      [C]
      predicate(C, unspecified, procrastinate, A)-1
   =>
   [D, E, F, G, H, I]
   structure(H, atomic)-1
   quantity(H, cardinality, count_unit, I, eq, 1)-1
   object(H, course, object)-1
   object(D, mark, object)-1
   quantity(D, cardinality, count_unit, E, eq, 1)-1
   structure(D, atomic)-1
   property(D, good)-1
   predicate(F, unspecified, receive, A, D)-1
   predicate(G, unspecified, pass, A, H)-1
   modifier(G, manner, none, easily)-1
```

Translation

1. accept ACE input from user
2. preprocess
   - "Y if X" → "if X then Y"; add "." if necessary
3. send query (input) to APE web service
   - threaded GUI remains responsive
4. retrieve and parse result from APE
5. traverse DRS to build RuleML/XML
   - simplified by JavaCC (open source tool)
6. display (user configurable) results

Translation



http://www.ruleml.org/translator/translator.jnlp

Translation

- DRS-RuleML mapping
  - direct
    - preserve “augmented” logic
    - RuleML syntax
      - positional (not “slotted”)
      - compact (“role-skipped”) form
  - explicit
    - e.g., quantifers: **`<Forall>`**, **`<Exists>`**
      - existential unless in the body of an implication
  - reversible
    - no loss of information
      - later: RuleML → ACE

Every honest student who does not procrastinate receives a good mark and easily passes the course.

(ACE)

```
[]
   [A …]
   …
   object(A, student, person)
   property(A, honest)
      NOT

      [C]
      predicate(C, … procrastinate, A)


   =>

   [D, … F, G, H …]
   …
   object(H, course, object)
   object(D, mark, object)
   property(D, good)
   predicate(F, … receive, A, D)
   predicate(G, … pass, A, H)
   modifier(G, manner, … easily)
```

```
…
<Forall>
 <Var>A</Var> …
 <Implies>
  <And> …
   <Atom><Rel>object</Rel>…<Ind>student</Ind>…</Atom>
   <Atom><Rel>property</Rel>…<Ind>honest</Ind></Atom>
   <Neg>
    <Exists>
     <Var>C</Var>
     <Atom><Rel>predicate</Rel>…<Ind>procrastinate</Ind> …
    </Exists>
   </Neg>
  </And>
  <Exists>
   <Var>D</Var>…<Var>F</Var><Var>G</Var><Var>H</Var> …
   <And> …
    <Atom><Rel>object</Rel>…<Ind>course</Ind>…</Atom>
    <Atom><Rel>object</Rel>…<Ind>mark</Ind>…</Atom>
    <Atom><Rel>property</Rel>…<Ind>good</Ind></Atom>
    <Atom><Rel>predicate</Rel>…<Ind>receive</Ind>…</Atom>
    <Atom><Rel>predicate</Rel>…<Ind>pass</Ind>…</Atom>
    <Atom><Rel>modifier</Rel>…<Ind>easily</Ind></Atom>
   </And>
  </Exists>
 </Implies>
</Forall>
…
```

(DRS)

(RuleML)

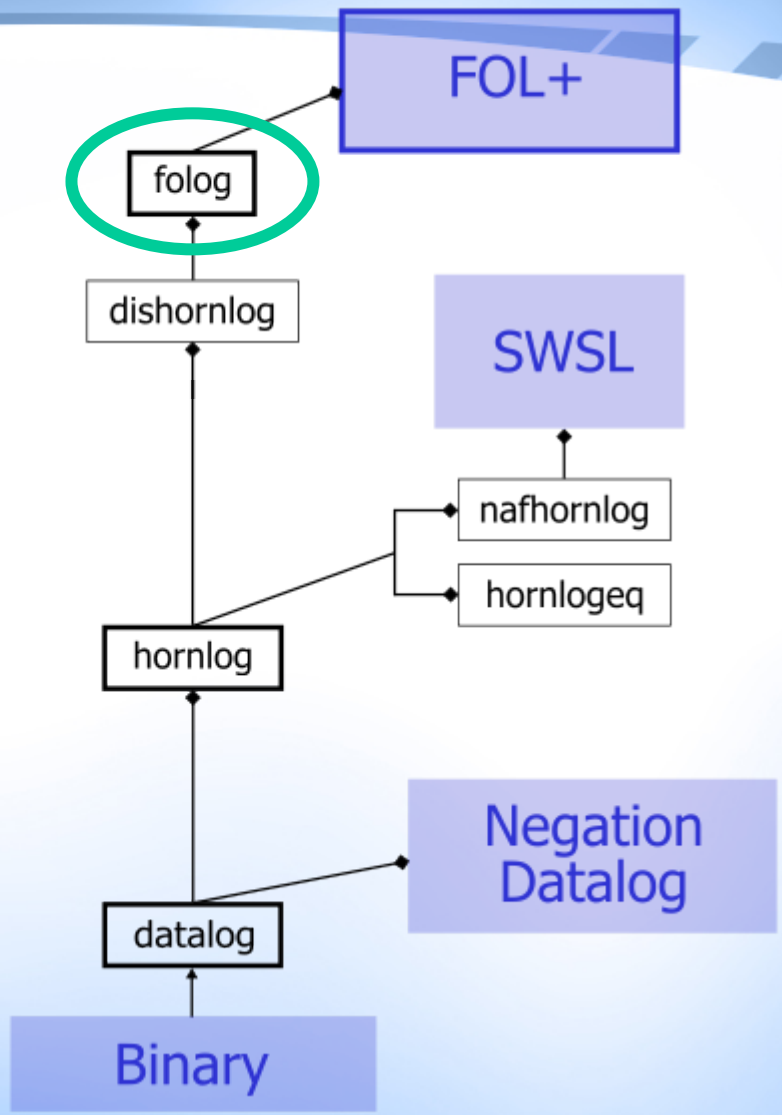
Rule Markup Language

- international initiative since 2000
  - now developed at NRC-IIT, e-Business
- canonical language for interoperable rule markup
  - XSLT translators to other Semantic Web languages
- collaboration with standards bodies
  - W3C, OMG, OASIS
- more information
  - http://www.ruleml.org

Rule Markup Language

- family of sublanguages
  - realized with modular XML Schemas
    - current version: 0.9
  - each represents well-known rule system
    - accommodate diverse needs of users
    - e.g., datalog, hornlog
    - TRANSLATOR uses folog (first-order logic)
      - FOL RuleML

Rule Markup Language



http://www.ruleml.org/modularization

Rule Markup Language

- advantages
  - interchange (XML)
  - compatibility with RDF, OWL and SWRL
    - also major input to W3C's upcoming RIF
  - tools
    - engines
      - OO jDREW, Mandarax, NxBRE…
    - XSLT translators
  - features
    - negation-as-failure, data types, weights, etc.

Future Work

- query support
  - **<Query>** vs. **<Assert>** currently undecidable
    - DRS of “Does the man enter a card?” same as that of “The man enters a card.”
- verbalization
  - RuleML → DRS
  - DRS → ACE with DRACE tool
- Attempto
  - negation-as-failure, modality, …

Conclusion

- TRANSLATOR allows anyone to write facts and rules in formal representation for use on the Semantic Web
  - critical factor in success of original Web?
- ACE → DRS → RuleML
- future collaboration with Attempto team

http://www.ruleml.org/translator

http://www.ifi.unizh.ch/attempto/tools