RuleML2TPTP Project
Part 1

Changyang Liu
Changyang.liu AT unb.ca

Nov. 17th, 2014

Professor: Dr. Harold Boley
Advisor: Dr. Tara Athan

Agenda

- What is the project about?
- How did we develop this project?

Introduction

- In this project, a translator in XSLT 2.0 is developed to convert Datalog+ Deliberation RuleML 1.01 in XML format to an equivalent representation in a subset of the TPTP (Thousands of Problems for Theorem Provers) language.

Datalog+ RuleML

- Rule Markup Language in XML format
- Standard Web rule knowledge representation
- Datalog+ Deliberation RuleML 1.01

Datalog+ Deliberation RuleML 1.01

- Existential Rules, where variables in rule conclusions are existentially quantified.
- Equality Rules, where the binary “Equal” predicate is applied in rule conclusions.
- Integrity Rules, which use the empty “Or” in rule conclusions to provide a convenient way for expressing falsity.

TPTP

- Comprehensive library, and the language, of the ATP (automated theorem proving) test problems
- Supports the testing and evaluation of ATP systems
- Standard input (the TPTP language) and output formats are enforced

XSLT (Extensible Stylesheet Language Transformations)

- XSLT stands for XSL Transformations
- Transform XML documents into other XML, non-XML formats, or into plain text
- XSLT 2.0 is the version being used

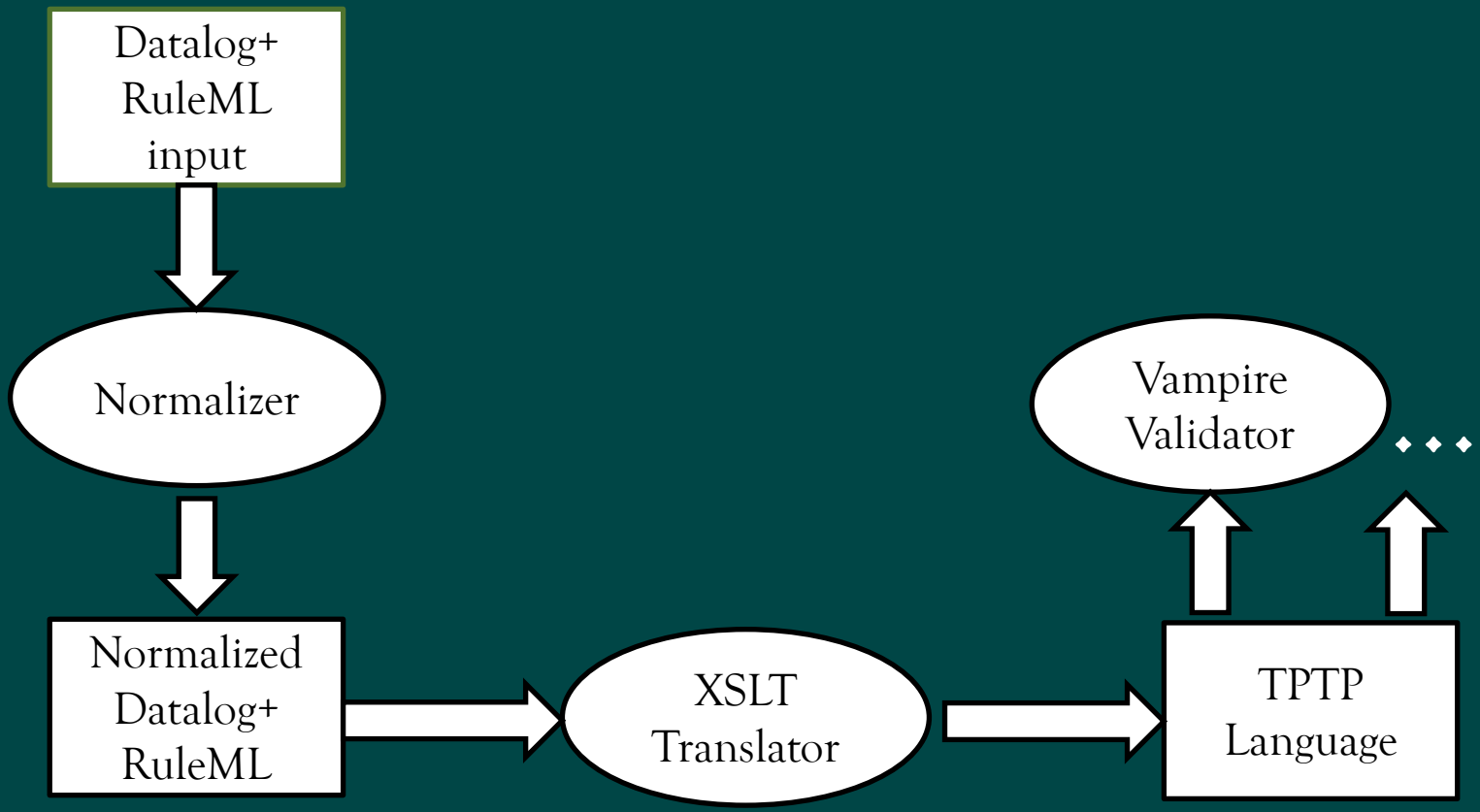
Procedure

- Phase Ⅰ : Preparation
- Phase Ⅱ : Coding
- Phase Ⅲ: Testing

Preparation

- Learn the grammars and structures of Datalog+ Deliberation RuleML and TPTP
- Learn how to use XSLT to do the translation
- Design the procedure of the project
- Decide the tools we use

Flow Diagram



Tools

- XSLT processor: Saxon
- Datalog+ RuleML Normalizer: 101_nafneghornlogeq_normalizer.xslt
- TPTP Validator: Vampire system

Example

```
<?xml version="1.0"?>
<RuleML xmlns="http://ruleml.org/spec">
 <Assert>
<Forall>
    <Var>H</Var>
    <Implies>
     <Atom>
      <Rel>human</Rel>
      <Var>H</Var>
     </Atom>
     <Exists>
      <Var>M</Var>
      <Atom>
       <Rel>hasMother</Rel>
       <Var>H</Var>
       <Var>M</Var>
      </Atom>
     </Exists>
    </Implies>
  </Forall>
 </Assert>
```

```
<?xml v e r s i on ="1.0" encoding="UTF    8"?>
<RuleML xmlns="http : / / ruleml . org / spec">
 <act index="1">
  <Assert mapMaterial="yes " mapDi rect ion=" bidirectional ">
   <formula>
    <Forall>
     <declare>
      <Var>H</Var>
     </declare>
     <formula>
      <Implies material="yes " direction=" bidirectional ">
       <Exists>
        <declare>
         <Var>M</Var>
        </declare>
        <formula>
         <Atom>
          <op><Rel>hasMother</Rel></op>
          <arg index="1"> <Var>H</Var> </arg>
          <arg index="2"> <Var>M</Var> </arg>
         </Atom>
        </formula>
       </Exists>
       <if >
        <Atom>
         <op> <Rel>human</Rel> </op>
         <arg index="1"> <Var>H</Var> </arg>
        </Atom>
       </ if >
      </Implies>
     </formula>
    </Forall>
   </formula>
  </Assert>
 </act>
```

Output

```
fof ( exampleExistential , axiom , (
   ! [H] : ( human(H) => ? [M] : hasMother (H, M) )
 ) ) .
```

Testing

- Use different source input to do the testing
- Use the normalizer to make the source input mainly striped
- Use the translator to generate the TPTP output
- Validate the output by using the Vampire system
- Find bugs in the program and fix them

Thank you