Shape Expressions: An RDF validation and transformation language

Eric Prud'hommeaux[1], Jose Emilio Labra Gayo[2], and Harold Solbrig[3]

[1] World Wide Web Consortium (W3C),
MIT, Cambridge, MA, USA eric@w3.org
[2] University of Oviedo
Spain labra@uniovi.es
[3] Division of Biomedical Statistics and Informatics
Mayo Clinic College of Medicine, Rochester, MN, USA

1 Introduction

1.1 RDF definitions

An RDF triple is a three-tuple $\langle s, p, o \rangle \in (\mathcal{I} \cup \mathcal{B}) \times \mathcal{I} \times (\mathcal{I} \cup \mathcal{B} \cup \mathcal{L})$, where $\mathcal{I}$ is a set of IRIs, $\mathcal{B}$ a set of blank nodes, and $\mathcal{L}$ a set of literals. The components s, p, o are called, the subject, the predicate, and the object of the triple, respectively. An RDF graph $\mathcal{G}$ is a set of RDF triples with the following operations:

$\circledcirc$	=	Empty graph (empty set of triples)
$[t]$	=	Singleton set with triple t
$t \rtimes g$	=	Result of adding triple t to graph g
$g_1 \oplus g_2$	=	Union of graphs g_1 and g_2

1.2 Shape Expression Abstract Syntax

New semantics of ShEx inspired by RelaxNG semantics.

The following grammar defines the Shape Expression abstract syntax. This is a simplified version of the language.

Shape	::= (label:Label, rule:Rule)
Rule	::= (Arc | And | Or | OneOrMore | Empty)
And	::= (rule1: Rule, rule2: Rule)
Or	::= (rule1: Rule, rule2: Rule)
OneOrMore	::= (rule:Rule)
Arc	::= (n:NameClass, v:ValueClass)
Empty	::= ()
Label	::= IRI | BlankNode
IRIstem	::= (i:IRI, isStem:Bool)
NameClass	::= (NameTerm | NameAny | NameStem)
NameTerm	::= (t:IRI)
NameAny	::= (excl:Set(IRIstem))
NameStem	::= (s:IRI)
ValueClass	::= ValueType | ValueSet | ValueAny | ValueStem | ValueReference
ValueType	::= type:IRI[4]
ValueSet	::= (s:Set(IRI))
ValueAny	::= (IRIstem)
ValueStem	::= (s:IRI)
ValueReference	::= (l:Label)
Action	::= (label:Label, code:String)
Schema	::= (rules:Set((Label, Rule)))

1.3 ShEx inference semantics

Figure 1 represents the inference rules of ShEx rules where:

$\Gamma \vdash g \simeq_r r$ = Graph g matches rule r in context Γ

$\Gamma \vdash p \simeq_n n$ = Predicate p matches nameValue n in context Γ

$\Gamma \vdash o \simeq_v v$ = Object o matches valueClass v in context Γ

Notice that there is no need to define optional and star as they can be defined in terms of OneOrMore (+) and EmptyRule (ε):

```
Optional(rule)     = Or(rule,Empty)
ZeroOrMore(rule)   = Or(OneOrMore(rule),Empty)
```

$$Or_1 \frac{\Gamma \vdash g \simeq_r r_1}{\Gamma \vdash g \simeq_r \text{Or } r_1\ r_2} \qquad Or_2 \frac{\Gamma \vdash g \simeq_r r_2}{\Gamma \vdash g \simeq_r \text{Or } r_1\ r_2}$$

$$And \frac{\Gamma \vdash g_1 \simeq_r r_1 \quad \Gamma \vdash g_2 \simeq r_2}{\Gamma \vdash g_1 \oplus g_2 \simeq_r \text{And } r_1\ r_2}$$

$$Empty \frac{}{\Gamma \vdash \circledcirc \simeq_r \text{Empty}}$$

$$OneOrMore_1 \frac{\Gamma \vdash g \simeq_r \text{r}}{\Gamma \vdash g \simeq_r \text{OneOrMore } r} \quad OneOrMore_2 \frac{\Gamma \vdash g_1 \simeq_r \text{r} \quad \Gamma \vdash g_2 \simeq_r \text{OneOrMore } r}{\Gamma \vdash g_1 \oplus g_2 \simeq_r \text{OneOrMore } r}$$

$$Arc \frac{\Gamma \vdash t.pred \simeq_n \text{n} \quad \Gamma \vdash t.obj \simeq_v \text{v}}{\Gamma \vdash [t] \simeq_r \text{Arc } n\ v}$$

Fig. 1. Inference rules for simplified ShEx