Ripple: Functional Programs as Linked Data

Joshua Shinavier
josh@fortytwo.net

Soph-Ware Associates, Inc.,
624 W. Hastings Rd, Spokane, WA 99218 USA
http://www.soph-ware.com

Abstract. Ripple is a scripting language expressed in RDF lists. Its scripts both operate upon and are made up of RDF metadata, extending the idea of HTTP dereferenceability to computation. Ripple is a variation on the "concatenative" theme of functional, stack-oriented languages such as Joy and Factor, and distinguishes itself through a multi-valued, "pipeline" approach to query composition, as well as the inherent distributability of its programs. The Java implementation of Ripple includes a query engine, a provisonal assortment of primitive functions, and an interactive interpreter which parses commands and queries in a readable, Turtle-like format. A demo application can be found at: http://fortytwo.net/ripple.

Key words: RDF, scripting language, semantic web, linked data

1 Linked Data

"Linked data"[1] is the subset of the Semantic Web which associates statements about the "thing" identified by a particular URI with the corresponding web location. An appropriate HTTP request for such a URI should produce an RDF document containing statements about it. Ripple assumes, furthermore, that the set of statements in such a response is *complete*, and the software may reject statements about the URI from any other source. This restriction makes the language referentially transparent with respect to its one RDF query operation, forward traversal.

2 RDF Equivalence

Not only does Ripple *operate* on linked data, but its programs are meant to *be* linked data as well. Every Ripple expression is equivalent to a collection of type rdf:List. In the Java implementation, conversion between the RDF graph representation of a list and its more efficient linked list counterpart is transparent to the user application. RDF properties are identified with functions which map subjects to objects. For instance, in the following query, `dup`, `toString`,

[1] http://www.w3.org/DesignIssues/LinkedData.html

`sha1`, `swap`, and `equal` are all primitive functions, whereas `foaf:mbox` and `foaf:mbox_sha1sum` are properties:

```
@prefix :      <http://fortytwo.net/2007/03/ripple/demo#>.
@prefix foaf:  <http://xmlns.com/foaf/0.1/>.
@define goodMboxSha1Sum:
    /dup /foaf:mbox/toString/sha1 /swap /foaf:mbox_sha1sum /equal.
```

The `@define` directive creates a new list and identifies it with a URI, in this case: `http://fortytwo.net/2007/03/ripple/demo#goodMboxSha1Sum`. The list may then be applied as a function:

```
<http://www.w3.org/People/Berners-Lee/card#i>/:goodMboxSha1Sum.
```

The above is a program which dereferences Tim Berners-Lee's FOAF URI, finds the sha1 sum of his email address, and checks it against the stated value, yielding `true`. Characteristically of Ripple, it doesn't matter where or when a program is executed; provided that its URI is made to be dereferenceable (for instance, by exporting data to a public location, or eventually, by attaching the interpreter to a triple store with a SPARQL endpoint), the program itself is dereferenceable, and a Ripple query engine on a remote machine will draw it into its own execution environment as needed.

3 The Compositional Pipeline

If Joy[2] is a stack language, then Ripple is a "stream-of-stacks" language. Each of its functions consumes a series of stacks as input, and produces a series of stacks as output, which is how it gets away with using RDF properties as functions. This simple query, for instance, yields not one, but several values:

```
<http://www.w3.org/People/Berners-Lee/card#i>/foaf:knows/foaf:name.
```

To distribute operations over an arbitrary number of values, Ripple replaces functions with "instances" which behave like elements of a pipeline: receiving input, transforming it, and passing it on. Instances may have state; for example, instances of the `limit` primitive (which counts its input and stops transmitting them after a certain point) or of the `unique` primitive (which remembers its input, and will not transmit a duplicate stack).

4 Conclusion and Future Work

Ripple is an exploratory project, which is to say that further development will be driven by discoveries made along the way. Thus far, Ripple has been most useful for quickly picking out and exploiting useful patterns in linked data. If the hypertext web is any example, the web of data will be vast, complex, and overall, loosely structured. As it grows, we will need an equally sophisticated web of programs to keep pace with it.

[2] http://www.latrobe.edu.au/philosophy/phimvt/joy/j00ovr.html