metaruby & ruby2c

Implementing Ruby in Ruby
&
Automatic translation of ruby code to C.

by Seattle.rb's

Ryan Davis <ryand-ruby@zenspider.com>

&

Eric Hodel <drbrain@segment7.net>

Overview

- Background information and Goals
- Introduction to metaruby
- BFTS
- Ruby2c Design
- ZenObfuscate as forcing function
- Current Status

The Problem:

C Sucks

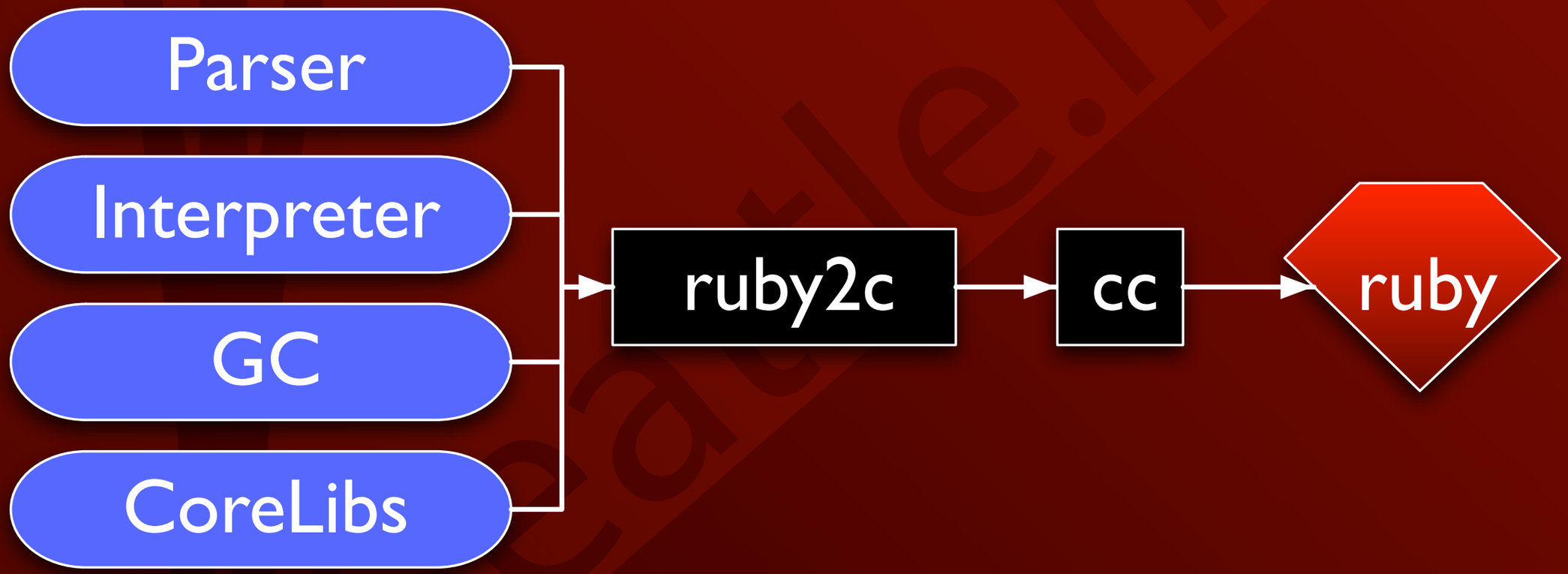
A Proposal

- Implement the whole thing in ruby, and translate to C.
  - No more mental context switching.
  - Able to test changes live in the system.
  - More understandable internals.
  - More accessible to others.
  - Must be in a subset of ruby that is easily translatable to C.

Metaruby

- Metaruby intends to implement ruby's internals in ruby itself.
- The metaruby implementation will use ruby2c to convert itself to C and bootstrap a new ruby binary.
- Metaruby should be fully compatible w/ Matz's ruby.

Basic Architecture



BFTS

- Big "Formal" Test Suite
- Unit tests for core library & internals
- Audited and up-to-date for ruby 1.8+
- RDoc for all tests creating a specification document.

A Simple ruby2c Example:

- Ruby parses code
- ParseTree extracts AST
- Rewriter doesn't touch it

  (SexpProcessor.process converts it to a Sexp)
- TypeChecker unifies it
- Ruby2C translates to C

Ruby

```
false and true
```

ParseTree

```
case NODE_AND:
  add_to_parse_tree(current, node->nd_1st);
  add_to_parse_tree(current, node->nd_2nd);
  break;
```

```
[:and, [:false], [:true]]
```

Rewriter

```
:and does not get rewritten
```

```
s(:and, s(:false), s(:true))
```

TypeChecker

```
def process_and(exp)
  rhs = process exp.shift
  lhs = process exp.shift
  rhs.sexp_type.unify lhs.sexp_type
  rhs.sexp_type.unify Type.bool
  return t(:and, rhs, lhs, Type.bool)
end
```

```
[:and, [:false, Type.bool], [:true,
       Type.bool], Type.bool]
```

Ruby2C

```
def process_and(exp)
  lhs = process exp.shift
  rhs = process exp.shift
  return "#{lhs} && #{rhs}"
end
```

```
Qfalse && Qtrue;
```

ZenObfuscate

- Obfuscates ruby code into binary form.
  - Builds on ParseTree/ruby2c platform.
- Got a real client early on
- Used concrete deliverables as a forcing function to get it done
  - Which was, in turn, a forcing function on the rest of the platform
    - Turned out to not be all that much extra work, just some edge-cases.

Current Status

- metaruby: Now using BFTS, core data types done, needs lots of other work.
- ruby2c: The design is fully implemented, we are expanding our supported subset of ruby.
- BFTS: Still auditing and expanding tests from rubicon.
- ZenObfuscate: Our first client went public with their RubyCocoa application last week.

Want to Help?

- We're recruiting for all aspects of this project.
- Tell us what you'd like to do.
- A ruby2c subset spec is coming soon.
- Lots to write, and much of it should be fun!