Eclipse OMR

A Modern Toolkit for Building Language Runtimes

Mark Stoodley
OMR Project Technical Lead
About me

• Mark Stoodley
  mstoodle@ca.ibm.com
  @mstoodle

• Senior software developer in Toronto, Canada working for IBM Runtime Technologies

• Spent the last 14 years building JIT compilers in JVMs

• OMR Project Technical Lead
Eclipse OMR is Live !!

http://www.eclipse.org/omr
https://github.com/eclipse/omr

Interact with team via mailing list or GitHub Issues
(omr-dev@eclipse.org)

Dual License:
Eclipse Public License V1.0
Apache 2.0

Contributors: please sign Eclipse CLA 😊 !
http://www.eclipse.org/legal/CLA.php
- Initial contribution from IBM on March 7 is about 200KLOC
- Project team is very excited to be moving forward in the open!
OMR is...
OMR is...

for Building Language Runtimes
OMR is...

for Building Language Runtimes
OMR is...

A Modern Toolkit for Building Language Runtimes
What does

Building Language Runtimes mean?
What’s in a Language Runtime for Java?
What’s in a Language Runtime for Ruby?

Ruby Source -> Ruby Bytecode Compiler -> Ruby Bytecode Interpreter -> Ruby Diagnostic Services -> Ruby Garbage Collector -> Ruby Just-In-Time Compiler

Ruby Platform Abstraction Layer
What’s in a Language Runtime for **Python**?

- Python Source
- Python Bytecode Compiler
- Python Bytecode Interpreter
- Python Diagnostic Services
- Python Just-In-Time Compiler
- Python Garbage Collector
- Python Platform Abstraction Layer
But all language runtimes are completely different implementations

- Investment in one runtime has no leverage in other runtimes

- Cloud => many languages => expensive to advance platform
OMR started from IBM J9 Java...
By concentrating “Java”ness into a “glue” layer for each core component
1. OMR has *no* language semantics

2. OMR is *not* a language runtime

3. OMR can become part of *any* language runtime
...toolkit doesn’t seem quite right...
...toolkit not quite right...

OMR provides
High Quality Language Runtime
Ingredients
Ingredients Now Available!

<table>
<thead>
<tr>
<th>Ingredient</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>port</td>
<td>platform abstraction (porting) library</td>
</tr>
<tr>
<td>thread</td>
<td>cross platform pthread-like threading library</td>
</tr>
<tr>
<td>gc</td>
<td>garbage collection framework for managed heaps</td>
</tr>
<tr>
<td>vm</td>
<td>APIs to manage per-interpreter and per-thread contexts</td>
</tr>
<tr>
<td>omrtrace</td>
<td>tracing library for communication with IBM Health Center monitoring tools</td>
</tr>
<tr>
<td>omrsigcompat</td>
<td>signal handling compatibility library</td>
</tr>
<tr>
<td>example</td>
<td>demonstration code to show how a language runtime might consume some OMR components, also used for testing</td>
</tr>
<tr>
<td>fvttest</td>
<td>a language independent test framework so that OMR components can be tested outside of a language runtime</td>
</tr>
<tr>
<td>+ a few others</td>
<td></td>
</tr>
</tbody>
</table>

...a little less than 200KLOC at this point
More ingredients coming soon!

diag: more diagnostic support for runtimes to aid runtime developers and users

hcagentcore: core IBM Health Center agent code to connect with a runtime

gc: adding generational and other policies / technologies

jit: Just In Time compiler and native code generation tools

docs: <gasp> we need more overview documentation

tests: more tests to ensure high quality contributions
How do you test

language runtime components

with no language semantics?
Mock it up!

• fvtest directory
  – Leverages Google test unit test framework
  – Component tests for:
    • Port, gc, signals, threads, vm fork, trace, agents
    • Utilities and data structures, method dictionary
    • etc.

• “make test” forms the basis for high quality contributions
  – Contributions cannot cause new failures in “make test”
Can it really work?

We asked ourselves the same question...
So we put it to the test!

Ruby+OMR
Python+OMR
SOM++ (Smalltalk) +OMR

...also used in IBM SDK for Java 8
...and consumed daily as we build the next IBM SDK for Java

Goal: test drive integration into existing runtime communities
COMPATIBILITY is critical
Ruby + OMR Technology Preview

• We ported OMR into CRuby
  – Focus on easy integration and compatibility
  – All C extensions work as-is
  – It runs Rails!

• Presented at Ruby Kaigi 2015, FOSDEM & jFokus 2016

• Available on github:
  – Download docker images for 64-bit X86, OpenPOWER, and LinuxONE @ https://goo.gl/P3yXuy
Diagnostic Tooling
Health Center

• Provides a live view of runtime details
• Works with IBM JDK, IBM Node.js, and Ruby+OMR Technology Preview
• http://goo.gl/u3VITI
Health Center – CRuby Method Profiling
The number of collections increased by 126% in the last third of the log compared to the middle third. However, the change in the heap usage was 0%, which suggests that an increase in application activity or fragmentation rather than a memory leak may be the problem. If the workload is not constant then the change in the frequency of collections may be nothing to worry about.

The mean occupancy is 41% which is close to optimal.
Garbage Collection and Memory Visualizer

- Provides a graphical details on GC events post mortem from verbose:gc logs
- Works with IBM JDK, IBM Node.js and Ruby+OMR Technology Preview

https://goo.gl/YwNrmI
Garbage Collection and Memory Visualizer
Performance

Up to 2.5X faster on some benchmarks with Ruby+OMR JIT!
Is Ruby Fast Yet?

- Ruby 2.2.2: 112.84
- Ruby 2.2.2 + OMR: 122.57
The chart illustrates the performance of different Ruby versions and configurations for calculating `Pow(2,N)` in terms of Time (seconds). The configurations include:

- Ruby 2.2.2
- Ruby 2.2.2 + OMR
- Ruby 2.2.2 + OMR + JIT
- OMR II Builder

The results are as follows:

- Ruby 2.2.2: 28.4 seconds
- Ruby 2.2.2 + OMR: 17.2 seconds
- Ruby 2.2.2 + OMR + JIT: 5.5 seconds
- OMR II Builder: 1.1 seconds
• We know it works; we tried it!

• IBM Java SDK will continue to consume OMR technology and work directly in open OMR project

• Project is open to all; we welcome others
  – New contributors!
  – New platforms!
  – New languages!
  – New tool interfaces!
  – New technology to help build runtimes!
Wrap Up

• Eclipse OMR: high quality language runtime components
  1. has *no* language semantics
  2. is *not* a language runtime
  3. can become part of *any* language runtime

• We are live at Eclipse as an incubator!
  – [https://github.com/eclipse/omr](https://github.com/eclipse/omr)
  – [omr-dev@eclipse.org](mailto:omr-dev@eclipse.org)
  – Contact us via mailing list or GitHub issues

• Dual Eclipse 1.0 + Apache 2.0 license

• We would love to see you all there!
A Parting Thought

• Almost no one starts a new project saying:
  – First, I’ll write the firmware from scratch …
  – First, I’ll write the file system from scratch …
  – First, I’ll write the display drivers from scratch …
A Parting Thought

• Almost no one starts a new project saying:
  – First, I’ll write the firmware from scratch …
  – First, I’ll write the file system from scratch …
  – First, I’ll write the display drivers from scratch …

• We would like to make these statements just as unlikely:
  – First, I’ll write the cross platform port library from scratch …
  – First, I’ll write the garbage collector from scratch …
  – First, I’ll write the JIT compiler from scratch …
BoF session tonight at 7pm in Lake Anne A

Native code generation for dynamic languages

Learn a bit more about the OMR JIT and its native code generation services
Learn more about OMR!

- A VM is a VM is a VM: The Secret Path to High Performance Multi Language Runtimes (Mark Stoodley, JVM Languages Summit 2015)
  https://www.youtube.com/watch?v=kOnyJurioyw
- Beyond the Coffee Cup: Leveraging Java Runtime Technologies for the Polyglot (Daryl Maier, Java One 2015)
  http://www.slideshare.net/0xdaryl/javaone-2015-con7547-beyond-the-coffee-cup-leveraging-java-runtime-technologies-for-polyglot
- What’s in an Object? Java Garbage Collection for the Polyglot (Charlie Gracie, Java One 2015)
  http://www.slideshare.net/charliegracie1/javaone-whats-in-an-object
- Building Your Own Language Runtime (Angela Lin, Robert Young, Craig Lehmann, Xiaoli Liang, CASCON 2015)
  https://ibm.box.com/s/7xdg25we2ezmdjbjqdyss30d7dl1iyo49
- It’s Dangerous to GC alone. Take this! (Robert Young and Craig Lehmann, Ruby Kaigi 2015)
- Experiments in sharing Java VM Technology with CRuby (Matthew Gaudet, Ruby Kaigi 2015)
  http://www.slideshare.net/MaBhewGaudet/experiments-in-sharing-java-vm-technology-with-cruby
- Ruby and OMR: Experiments in utilizing OMR technologies in Ruby (MRI) (Charlie Gracie, FOSDEM 2016)
  http://www.slideshare.net/CharlieGracie/ruby-and-omr
- A JVMs Journey into Polyglot Runtimes (Charlie Gracie, jFokus 2016)
  https://t.co/efCKf6aCB4
- OMR: A modern toolkit for building language runtimes (Mark Stoodley, EclipseCon March 2016)
  [ You’re reading it right now! ]
- Ruby+OMR Technology Preview github project with docker images for Linux on LinuxONE, OpenPOWER, and X86
  https://github.com/rubyomr-preview/rubyomr-preview

- Contact:
  - Mark Stoodley (mstoodle@ca.ibm.com, @mstoodle) OMR Project Technical Lead
IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM’s sole discretion. Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision. The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
Additional Important Disclaimers

- THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY.
- WHILST EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED.
- ALL PERFORMANCE DATA INCLUDED IN THIS PRESENTATION HAVE BEEN GATHERED IN A CONTROLLED ENVIRONMENT. YOUR OWN TEST RESULTS MAY VARY BASED ON HARDWARE, SOFTWARE OR INFRASTRUCTURE DIFFERENCES.
- ALL DATA INCLUDED IN THIS PRESENTATION ARE MEANT TO BE USED ONLY AS A GUIDE.
- IN ADDITION, THE INFORMATION CONTAINED IN THIS PRESENTATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM, WITHOUT NOTICE.
- IBM AND ITS AFFILIATED COMPANIES SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION.
- NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, OR SHALL HAVE THE EFFECT OF:
  - CREATING ANY WARRANT OR REPRESENTATION FROM IBM, ITS AFFILIATED COMPANIES OR ITS OR THEIR SUPPLIERS AND/OR LICENSORS