TechnicalFAQ

What software is required to write a J2SE implementation?

At least the following:

- A JVM
- The Java class libraries
- Command line tools

What language will the Harmony JVM be in?

One of the goals of the Harmony project is to create a design modular enough so that parts of the VM can be implemented in different languages.

Is that even possible?

Hopefully! Some reading that may be of interest includes:

- Andrew Gray's work on using MMTk (the Jikes RVM Memory Management ToolKit) with the Rotor CLI VM
- Experience Integrating a New Compiler and a New Garbage Collector Into Rotor
- Porting the JMTk memory management toolkit - about integrating JMTk/MMTk into GCJ
- Google Scholar Search for Modular Virtual Machines

Can a JVM written in Java perform as well as one written in C/C++?

There is evidence to show that it is possible, and some research shows specific areas where a Java JVM can outperform a statically compiled JVM.

Jikes RVM developer Steve Blackburn wrote the following (slightly edited for clarity):

In a nutshell, for a high performance VM, the single biggest issue is the quality of your compiler/s (which dictates the performance of user code)---this is true regardless of the implementation language. Very little runtime is actually due to the execution of the VM per se (perhaps 10%) and that is primarily just the execution of the JIT and the GC. The remaining 90% is user code (and libraries) whose performance rests largely on the quality of your JIT. We've demonstrated the performance of MMTk and I've already said many times on this list why the lack of implementation/implemented language impedance helps us so much. As for the JIT, we have the "eat your own dogfood" phenomena: if the JIT is any good, then writing the JIT in Java won't be a performance problem. We've seen this with the Jikes RVM project. Dave Grove has a work list as long as his arm of things we could do to improve in the optimizing compiler, but we lack the resources to get them done. That is why Jikes RVM is currently lagging the commercial JITs where it once was competitive. I think this is why the harmony project holds a lot of promise: it will sharpening the community's focus and lead to much better targeting of resources.

We don't have any published performance comparisons. I can only tell you that for a long time our comparison point was the IBM product VM (on PPC) and for a while we were competitive. Since we no longer have full time IBM staff working on the project (they've been diverted to the product 😉, we've started to slip. The story on the IA32 is less pretty mainly because our primary focus for much of the time was on the PPC, so a lot of trivial (but time consuming) IA32 optimizations are yet to be done.

[on using Jikes RVM as a Serverside JVM] _there is nothing about the Java-in-Java VM design which is preventing it from being used "to run high volume production code". Jikes RVM performs very well in this context and that is exactly where it first made its mark (outscaling commercial VMs running JBB on large SMP boxes)._

The two primary issues holding it back today are

- *keeping the compiler up to speed with the commercial VMs,*
- *completeness of the VM.*

To the extent that either are limitations, they are due to resource constraints and have nothing whatsoever to do with the Java-in-Java implementation technology. To the contrary: in my view there is no way we'd have achieved as much as we have with the resources we've had if we did not have the considerable software engineering advantages of a strongly typed language.

See also JvmInJava

What does this doobaflacky thing people keep talking about mean?

You may find an answer in the Terminology section of this Wiki