Magik on Java™

Phil Robinson
Consulting Engineer, Asset Management
Digital Energy

imagination at work

Introduction

GE is making a radical technology shift to its geospatial technology platform to bring the following business benefits to customers:

- A 3 to 5 times improvement in code performance
- Ability to leverage all of the capability of the Java ecosystem
- Provide improved tools to manage and analyze the performance of a deployed system

GE is doing so by utilizing the power of the Java Platform to run the Magik environment. In particular, GE is using the new capabilities that Oracle has added to the Java language and the Java Virtual Machine (JVM) to support languages like Magik. Oracle JVM Architect John Rose pointed out, "When we added new bytecode features to Java SE 7, we were aiming to help implementers of dynamic and domain-specific languages, giving them full access to our deep technology investments in the JDK and JVM. GE's work with Magik is exactly the type of software engineering we want to enable."

Taking this approach allows customers to maintain their investment in this technology while at the same time being able to use the full breadth of the Java ecosystem.

This technical paper explains the opportunities driving this change, the subsequent benefits and how GE is leveraging Java technology to bring them to the Smallworld™ product suite.

Business Challenges for Geospatial Solutions

Customers working on solving business problems with a geospatial element have several requirements relating to the extensibility of the solutions they choose.

- A powerful and flexible development language that has specific strengths around solving geospatial business problems
- A robust and scalable geospatial database able to manage complex, interconnected networks
- Ability to maximize the utilization of underlying hardware infrastructure

These requirements are met by the Smallworld Magik platform and its Version Managed Data Store (VMDS) which together provide highly scalable geospatial solutions with version management capability. GE has been using this technology with great success for more than 20 years. The introduction of this technology shift is being done without losing these key benefits.

Another challenge for businesses is finding the right resources to extend their solutions when business needs require it. Flexibility and choice of technology to do this would present a key advantage for customers in this position. This multi-language paradigm (i.e., the possibility of using more than one programming language in a single execution environment) provides exactly this flexibility.

An example of how the multi-language paradigm provides this flexibility relates to the different types of programming language. Some programming languages lend themselves to rapid development more than others – good examples are the so-called dynamic languages (i.e., Magik, Ruby, Python and others). Other languages are good at creating more structured environments where programming errors can be identified at compile time – examples of these are the statically typed languages (i.e., Java, C and others). Depending on the situation, one type may be more productive than the other.

Another example is that the particular skill and experience of a developer may lie in one language or another. By having a multi-language paradigm, the pool of potential resources (codebase, tools, people) that could be used to make the changes is a lot larger than if only one language is supported. This is a tremendous business benefit and is part of the value proposition for a multi-language paradigm, and GE fully embraces it to bring these benefits to our customers.

Augmenting a Technology Platform without Disruption

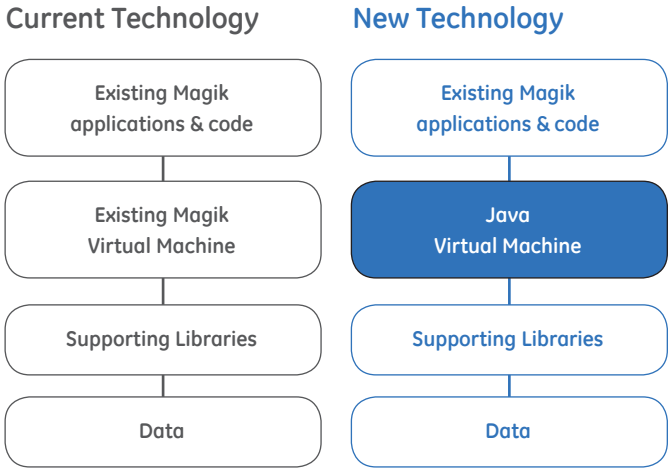
Background to the Magik Programming Environment

The Smallworld desktop solutions are extended by using a dynamic programming language called Magik. Magik is extremely powerful while at the same time insulating developers from certain types of development overhead that can lead to software failure. As a result of these two qualities, Smallworld products have a superior extensibility capability which has been a large part of the success of that product line to date.

Magik runs by executing instructions on its own virtual machine.1 A virtual machine is software that simulates a computer processor in order to provide a level of abstraction from the underlying hardware architecture. Common examples are the Oracle® Java Virtual Machine used by the Java language (and others), the Common Language Runtime (CLR) by Microsoft® used by the .NET environment and GE's Magik Virtual Machine.

Replacing the Runtime

GE is augmenting Magik by using the Java Virtual Machine (JVM) as a replacement for the runtime executing the Magik environment (the Magik Virtual Machine) as shown in Figure 1.



Why now?

This has been made possible by the introduction of new language features in the JVM at Java 7, specifically the invokedynamic bytecode.2

This new instruction was introduced by Oracle specifically to support dynamic languages like Magik running on the JVM.5 Very Furthermore, it allows programming languages like Magik to take advantage of JVM's optimization capabilities and can dramatically improve performance.

Because of these changes to the JVM, the time is right for GE to take advantage of this new capability for its development environment.

Utilizing JVM to run Magik also enables GE to leverage the breadth of platform and OS support that Java provides. Java is supported on all major operating systems, such as Windows®, Linux®, Solaris and Mac OS® X, and on platforms ranging from small embedded devices to large servers.5

Minimizing Disruption

One of GE's key objectives is to maintain a very high degree of compatibility with existing Magik code in order to avoid a potentially large cost of porting code to a new form. As can be seen from Figure 1, this is achieved by focusing change only to replacing the existing virtual machine and its interfaces to surrounding applications and support structures. By maintaining the behavior of the Magik runtime, the introduction of the Java Virtual Machine is practically transparent to existing applications.

Benefits of Using the Java Virtual Machine

By using the Java Virtual Machine, the Smallworld technology is underpinned by the engine driving the world's second most popular programming language (Java).3

As a result, the Java Virtual Machine is one of the world's most optimized virtual machines. It is able to "tune" software as it runs, which provides superior performance. Magik running on the JVM can result in 3–5x performance improvement in the execution of Magik code.

Another key benefit is interoperability with the Java language and libraries. Because both Magik and Java are compiled down to the same byte code and are running in the same execution environment, it is possible for both types of code to interact. This means that it is possible for Magik to utilize Java code and for Java code to utilize Magik. As a result, new functionality can be written in the Java programming language, and existing functionality can leverage the enormous range of software available in the Java ecosystem. This, of course, is exactly the multi-language paradigm mentioned previously in action.

Finally, it is possible via the technology to use Java tools against Magik programs. For example, it is now possible to diagnose performance issues within a Magik application using Java profiling tools like VisualVM. Commercial profilers and debuggers are also available. This allows businesses running production solutions from GE to gain insight into the operation of the system and, therefore, manage them more effectively.

Summary

The Smallworld Magik desktop platform has been enormously successful for GE and its customers. GE is now enhancing this platform to enable the Magik platform to make full use of the benefits of the Java Virtual Machine and the Java ecosystem without the cost of having to port existing code to another form.

The benefits of this approach are:

- Superior Magik execution performance
- Java–Magik interoperability
- 64-bit computing
- Access to the full range of Java tooling
- Compatibility with existing applications and Magik code
- Support for a large set of operating systems running on anything from embedded devices, desktops, or servers.

GE is using key changes that Oracle has made to the JVM to provide these benefits. Georges Saab, Oracle Vice President of Java SE said, "We are extremely pleased to see GE effectively using the technologies introduced into Java SE 7 for supporting dynamic languages on the JVM. This is a great validation of our strategy of enabling customers to create powerful and performant domain-specific languages on the JDK to provide real business benefit through technology. We look forward to more success with GE and the Java ecosystem in the future.

References

1. Virtual machine – Wikipedia
   http://en.wikipedia.org/wiki/Virtual_machine
2. JSR 292 Specification. http://jcp.org/en/jsr/detail?id=292
3. Java Virtual Machine support for Non-Java Languages.
   http://docs.oracle.com/javase/7/docs/technotes/guides/vm/multiple-language-support.html
4. "Looking Ahead to Java SE 7 and 8: A Discussion with Oracle's Java Language Architect, Brian Goetz".
   http://www.oracle.com/technetwork/articles/java/briangoetzchief-188795.html
5. Oracle JDK 7 and JRE 7 Certified System Configurations.
   http://www.oracle.com/technetwork/java/javase/config-417990.html

About GE
GE (NYSE: GE) is the world's Digital Industrial Company, transforming industry with software-defined machines and solutions that are connected, responsive and predictive. GE is organized around a global exchange of knowledge, the "GE Store," through which each business shares and accesses the same technology, markets, structure and intellect. Each invention further fuels innovation and application across our industrial sectors. With people, services, technology and scale, GE delivers better outcomes for customers by speaking the language of industry. www.ge.com

Contact information
Americas: 1-855-YOUR1GE (1-855-968-7143)
gedigital@ge.com

www.ge.com/digital

imagination at work