Porting the GNAT Tasking Runtime System to the Java Virtual Machine

Laurent Millet (ENST), Ted Baker (FSU)

talk for Ada-Europe'98

9 June 1998

Outline

1. Background
2. Mapping GNARL to JVM
3. Prototyping using JNI
4. Conclusions

Background

- Java Virtual Machine (JVM)
- Precedent: Intermetrics Ada-to-JVM compiler
- GNAT: Ada compiler
- GNARL: tasking runtime system for GNAT
- GNULLI: low-level task primitives for GNARL
- Java port of tasking = retarget of GNULLI
- Java Native Interface (JNI)
  allows calling Java methods from C-language code

GNULLI Task Primitive Operations

- `Create_Task`
- `Abort_Task`
- `Self`
- `Write_Lock`, `Unlock`
- `Sleep`, `Timed_Sleep`, `Wakeup`
- `Set_Priority`
- `Yield`

Implement these in terms of JVM objects and operations.

`Create_Task`

- use constructor of `java.lang.Thread`
  or any class with `run` method
- Ada Task Control Block (ATCB) is ordinary Java object
  derived from `java.lang.Object`
- available methods on ATCB include synchronization

`Abort_Task`

- JVM has no preemptive abort operation
- if body of `Abort_Task` is null
  abort occurs only at abort completion polling points

`Self`

- `java.lang.Thread.currentThread`
- Ada task = Java thread
  with data component that points to ATCB

`Write_Lock, Unlock`

- `monitorenter`, `monitorexit` apply to all objects
  hence to ATCB
- no priority ceiling locking in Java
- GNULL emulates using `Thread.setPriority`

`Sleep, Timed_Sleep, Wakeup`

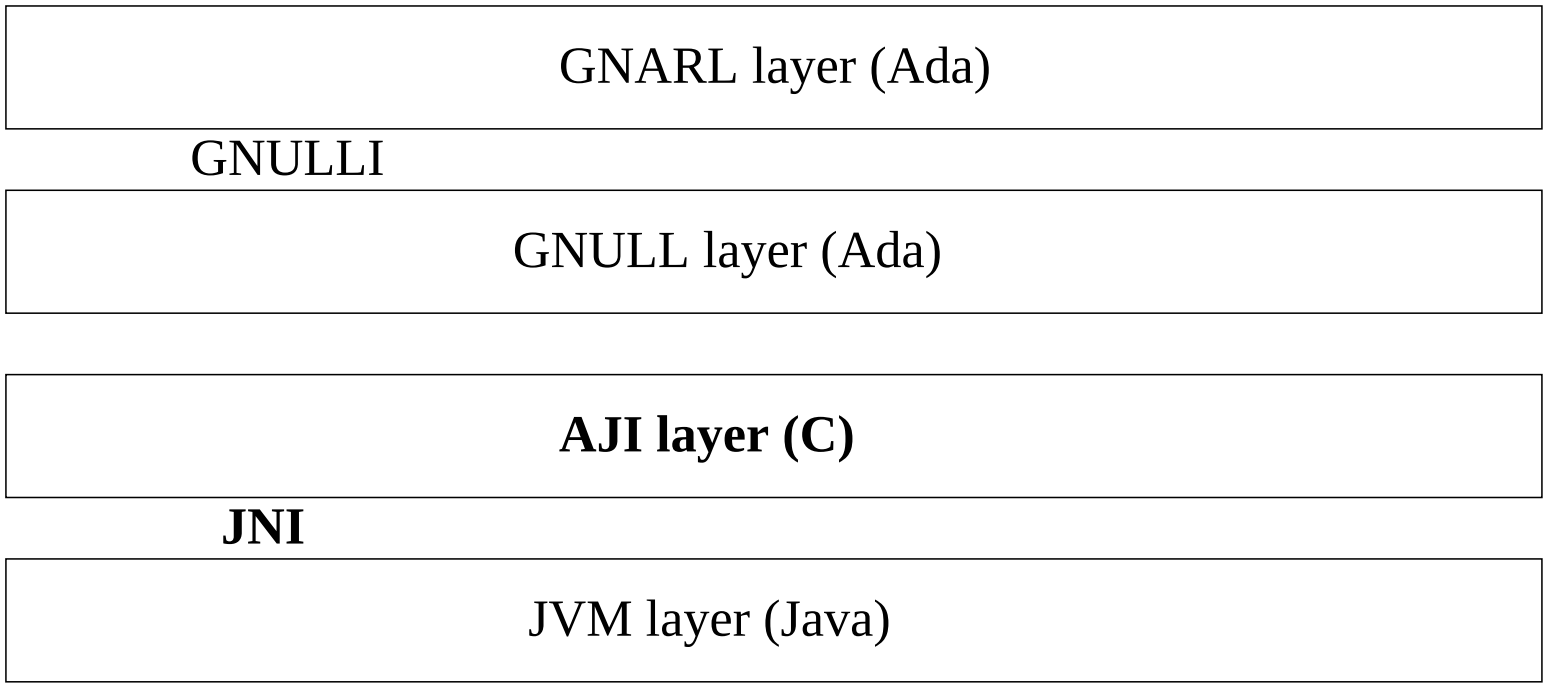
- `java.lang.Object.Wait` ≈ `Sleep`
  atomically releases lock
- `java.lang.Object.Notify` ≈ `Wakeup`
- apply to all Java objects, including ATCBs

Scheduling Support

- `java.lang.Thread.setPriority` ≈ `Set_Priority`
- Java does not require priority scheduling support, or precisely define the effects of priorities, though
- `java.lang.Thread.yield` ≈ `Yield`

Prototyping using JNI

How to test tasking port without JVM byte-code code compiler?



This is only a temporary implementation, for testing.

Java Environment Pointer

- parameter of JNI functions
- gives thread context in which to execute method

Task Creation

```
package AJI;
public class Wrapper extends Thread {
  private int proc, self_id;
  public Wrapper (int proc, int self_id) {
    this.proc     = proc;
    this.self_id  = self_id;
  }
  public void run () {
    wrap (proc, self_id);
  }
  private native void wrap (int proc, int self_id);
  static {
    System.loadLibrary ("wrapper");
  }
}
```

Conclusion

- Prototype implementation works
- Ada programs are compiled to native SPARC object code
- Tasking "system calls" are all through the JVM, *via* JNI
- As of December, passed all basic tasking-related ACVC tests
  - 189 Ada ACVC 83 tests
  - 49 Ada 95 tests
  - Annex D tests

Further Conclusions

- Demonstrates feasibility of Interface via JNI to Java libraries, for native-code Ada applications
- A Java-compatible tasking runtime system, such as this one, is a prerequisite