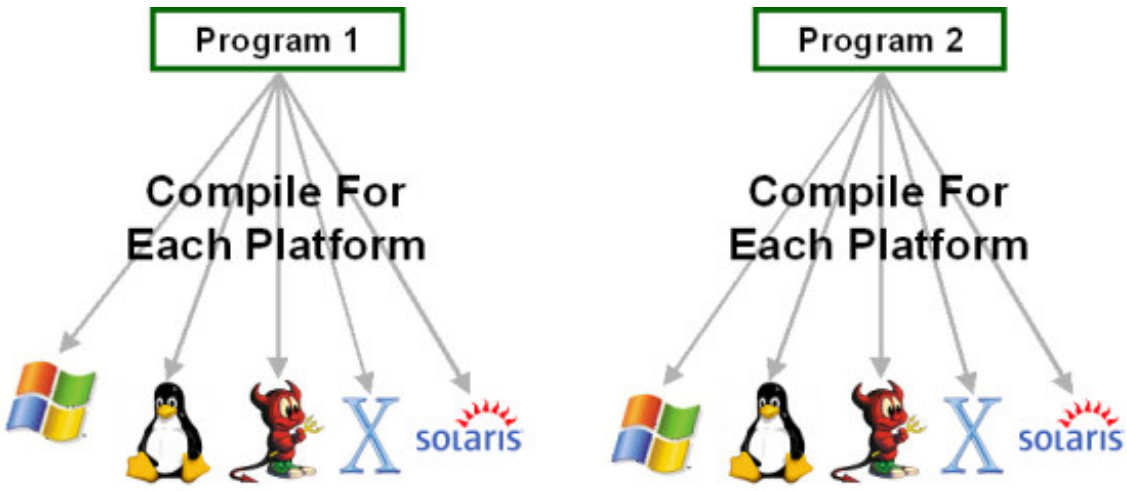
Bytecode Translation: From The .NET CLR To Parrot

Jonathan Worthington
University Of Cambridge

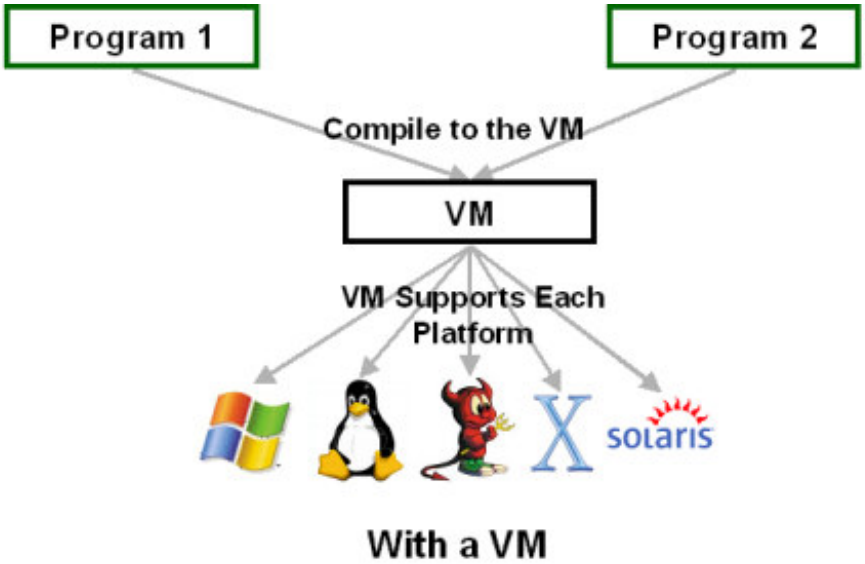
.NET To Parrot Translation

Virtual Machines Rock

Virtual machines for high level languages hide away the details of the underlying hardware and OS, making deployment simpler.



Without a VM



With a VM

.NET To Parrot Translation

Re-using code rocks too, but…

- We all use libraries rather than writing stuff from scratch.
- Libraries in native code just had to use a standard calling interface.
- In a world with multiple VMs, it is possible you'd want to use a library compiled to run on VM A from a program running on VM B.
- But VM A can't understand VM B's code. Argh! What to do?

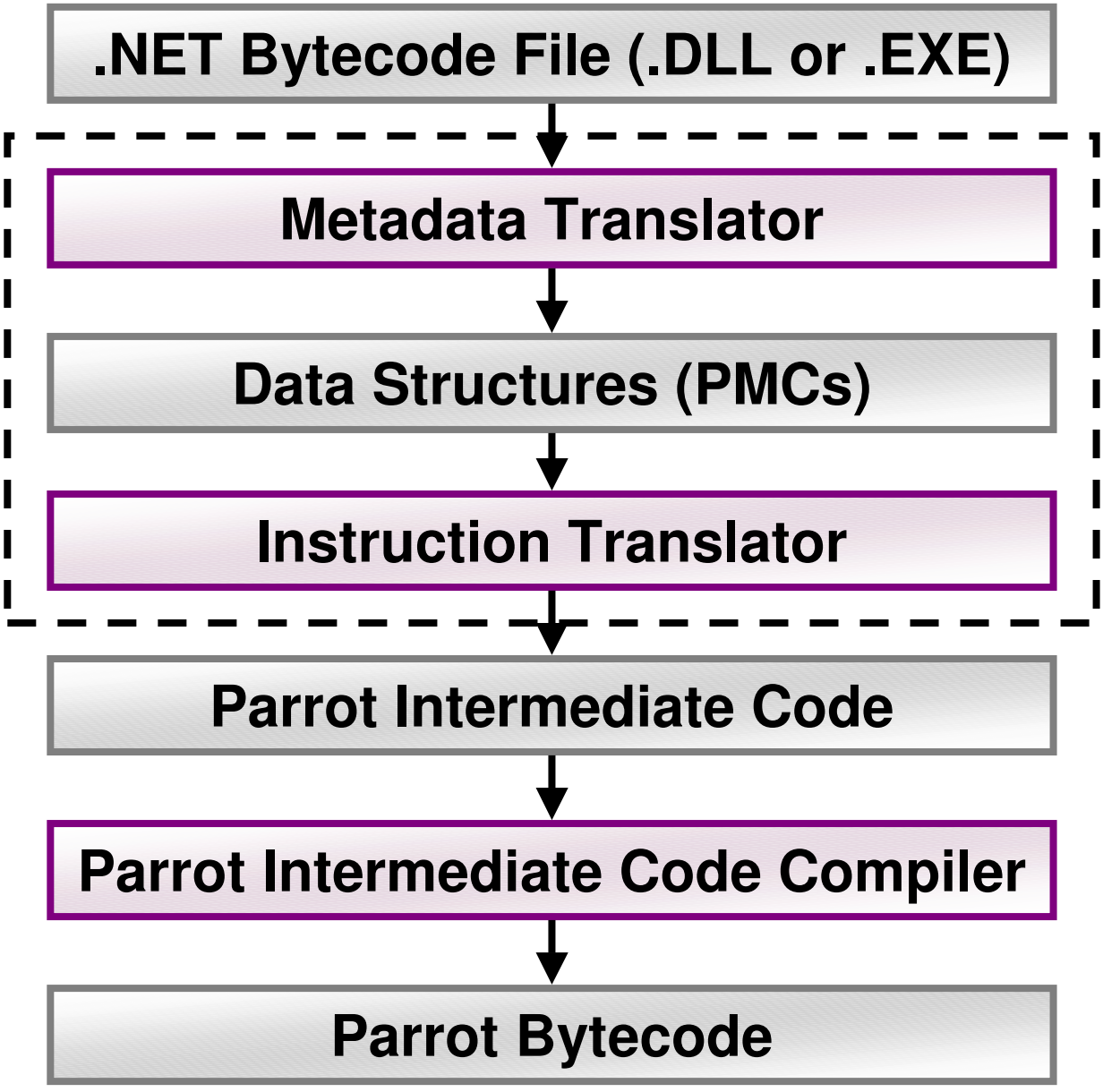
Translation to the rescue!

- Translate code that runs on VM A to use VM Bs instruction set, etc.
- Wins support for all languages that compile to VM A.

.NET To Parrot Translation

Designing the translator

The translator breaks down into a series of modules that execute one after the other, like a compiler.



.NET To Parrot Translation

Writing repetitive code is boring

- Code for translating diffferent instructions has a lot in common.
- Write translation rules to describe how to translate instructions.

```
[add]
code = 58
class = op
pop = 2
push = 1
instruction = ${DEST0} = ${STACK0} + ${STACK1}
```

- Then use a script to build the instruction translator from them.

Stack to register mapping

- .NET is a stack based VM, Parrot is register based. Many ways to handle this, from easy to hard.
- Pluggable mapping modules allow multiple solutions to be explored.

.NET To Parrot Translation

What can be translated so far?

- Arithmetic, logical, branching and comparison ops
- Integer, floating point and managed pointer types
- Instance and static fields, methods and constructors
- Instantiation, method calling, inheritance and interfaces
- Array related instructions

Conclusions

- Bytecode translation still feels like a good idea.
- Regression testing is valuable.
- Generate repetitive code.