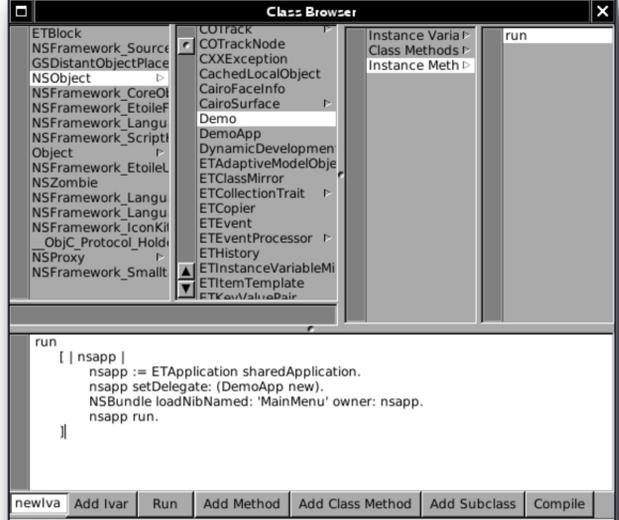


"Interoperability isn't just a technical problem, it's also a user interface problem. We have to make it easy to call between languages, so that users can always pick the most appropriate tool for the job at hand."

example [

```
"This shows how to call a C function from Smalltalk"
C objc_setAssociatedObject: {
  self.
  42.
  'fishes'.
  C enumValue: OBJC_ASSOCIATION_ASSIGN }.
"Note how C enumerations can be accessed. When encountering this construct,
the Pragmatic Smalltalk compiler will generate exactly the same code as an
Objective-C compiler would for this line:
objc_setAssociatedObject(self, 42, @"fishes", OBJC_ASSOCIATION_ASSIGN);
It will get the types for the function by parsing the header and
automatically map Smalltalk types to C and Objective-C types. The
programmer doesn't need to write any bridging or foreign function interface
code."
(C objc_getAssociatedObject: {self . 42 }) log.
]
```



The demo shown at FOSDEM this year showed fully dynamic development spanning Objective-C and Smalltalk. The developer can inspect the existing class hierarchy and the code for any methods implemented in Smalltalk. It is also possible to add, modify, or replace methods, add instance variables and classes at run time. Invoking a nonexistent class or method pops up a dialog asking the user to implement it, all from within a running app. The final version is statically compiled, with no explicit user intervention required.

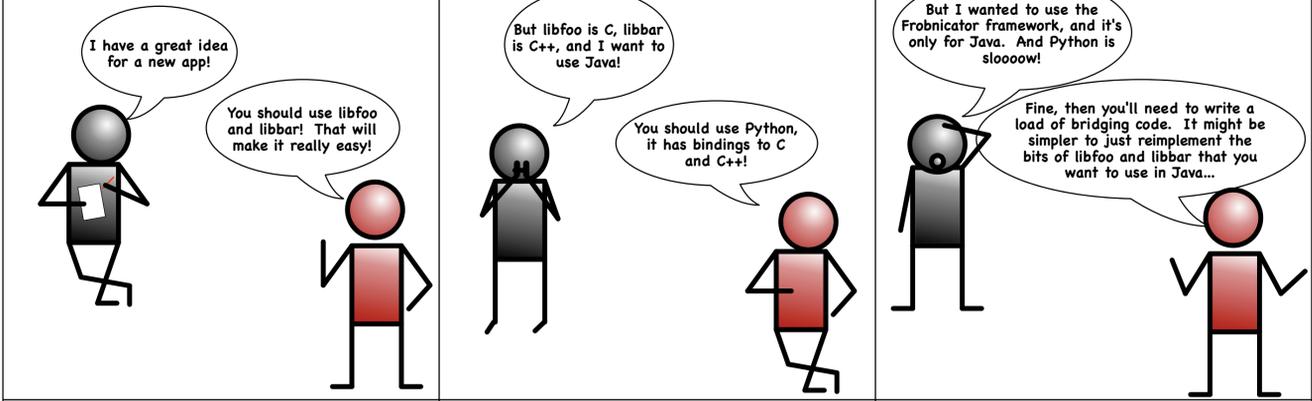
# Cross-Language Interoperability for Fast, Easy, and Maintainable Code

David Chisnall

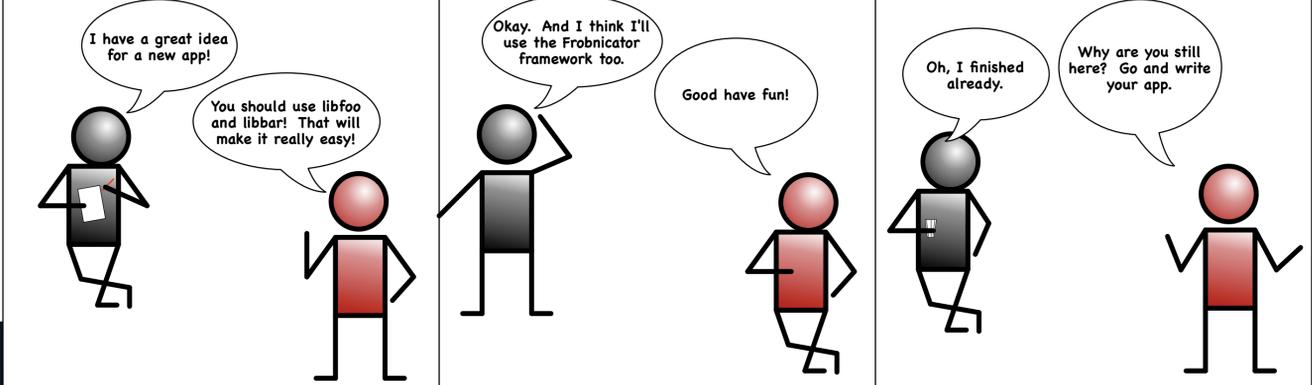
Our approach to cross-language interoperability involves compiling multiple languages down to a shared binary representation. We currently compile dialects of Smalltalk and JavaScript and an object-oriented parser generator down to the same object model as Objective-C. We can call C directly and use Objective-C as a hybrid language for more complex bridging. Objective-C effectively becomes a domain-specific language for calling C and C++ from high-level languages.

A single object may have methods implemented in different languages.

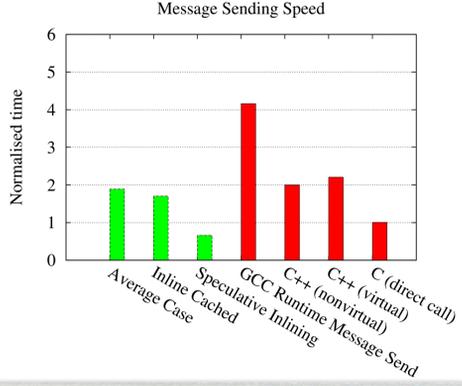
The World Today:



The World We're Building:



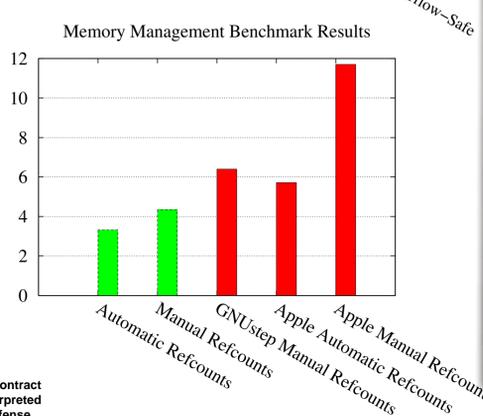
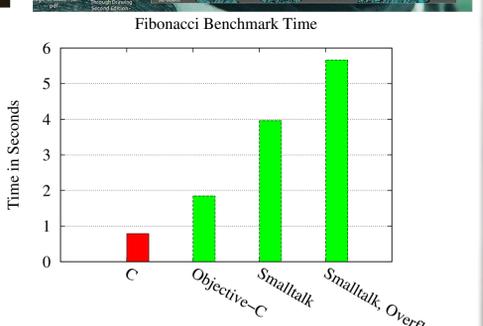
The 3E graphics editor uses C++ for high-performance 3D modelling code the Smalltalk, Objective-C for model descriptions, and Smalltalk for procedural geometry generation and scripting.



The Objective-C implementation developed as part of this work is used in a number of open source and commercial games and applications, with millions of installs, including Android ports of iOS apps.



One unfortunate side effect of mixing languages is that your code is as safe as the least-safe language you use. This applies even when you use a JVM written in C++. We are currently exploring sinking enforcement of high-level language isolation properties (such as object encapsulation) into the hardware, with the CHERI CPU designed as part of the DARPA-funded CTSRD project. We are already able to enforce object isolation in a dialect of C running on this CPU.



Part of this work has focussed on optimising dynamic languages. The biggest single performance cost for dynamic languages is the cost of message sending (dynamic method invocation). We have significantly improved the performance of this, both in the best and common cases. We added inline caching and speculative inlining to the compiler and improved the lookup code in the runtime, making our average case twice as fast as GCC and our best case faster than a direct C call. We have shown that it's possible to use late-bound dynamic languages without sacrificing performance and that mixing languages doesn't need to be slow or difficult. Even though the implementation in Smalltalk is noticeably slower than the C version for the Fibonacci benchmark, it does have one advantage: it gives the correct answer for any input, whereas the C version silently overflows...

This research is sponsored by the Defense Advanced Research Projects Agency (DARPA) and the Air Force Research Laboratory (AFRL), under contract FA8750-10-C-0237. The views, opinions, and/or findings contained in this article/presentation are those of the author/presenter and should not be interpreted as representing the official views or policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the Department of Defense.