C++ *Has No Useful Purpose?*

Dr Russel Winder

Partner, Concertant LLP

russel.winder@concertant.com

Director, It'z Interactive Ltd

russel@itzinteractive.com

C++ *Has No Useful Purpose!*

Dr Russel Winder

Partner, Concertant LLP

russel.winder@concertant.com

Director, It'z Interactive Ltd

russel@itzinteractive.com

Aims of the Session

- To show that C++ is redundant in the world of systems development.
- To show that where C++ might be thought to have a role, there are better alternatives – far, far better alternatives.

The Prototype

- ‘C++: Why Bother’ ACCU London 2007-02-22.

Widely-used Programming Languages

- Fortran
- C
- C++
- Java
- C#
- *Visual Basic*
- *Cobol*
- Python
- Groovy
- Ruby
- JavaScript
- *Lua*
- Haskell
- Objective Caml
- SML/NJ

Which C++?

- GNU
- Comeau
- Microsoft
- RealView
- Greenhills
- IAR

- Standard, which standard?
  - C++
  - C++/.NET
  - EC++
  - MISRA C++
  - JSF C++

C++ *Is Now Too Complicated*

- 1980's C++ – a definite step forward
  - C with Classes
  - Emergence of object-oriented systems
  - But lack of parameterized types
- 1990's C++ – consolidation
  - Templates
- 2000's C++ – problems
  - STL error messages incomprehensible
  - It has all got too complicated

C Is Preferred Over C++

- C is simple and straightforward.
- Gnome, KDE:
  - C appears to be the language of choice:
    - C as portable assembler.
    - Minimizes library dependence.
  - C++ hasn't taken off.
  - Python being used more and more.

Is object-based as good as object-oriented?

An Example: GFontBrowser

- Gnome, Fontconfig, Cairo based system.
- C++ seemed the right way:
  - GTKmm, libgtkmm
  - Glade, libglademm
- Decision
  - Continue and suffer the hassles?
  - Rewrite in C?
  - Rewrite on Python?

The Rise of the Virtual Machine

- Python and Java reintroduced virtual machines.
- Portability, WORA.
- Commoditization of processors and operating systems.
- C and C++ have problems targeting different processors:
  - C, C++ have different model to processors.
  - Compiling. Autotools, SCons, Waf, Rant, etc.

Hegemony of the Virtual Machine

- Java and C# are the statically typed languages of choice.
- Python, Ruby, Groovy are the dynamically typed languages of choice.

The Problems of Small Systems

- Small embedded systems cannot support Java, C#, Python, Groovy, or Ruby.
- C++ not supported either – well not really.
- C is the language of choice:
  - It's C, but not as we know it.
  - 8051 features.
  - ARM features.

Compare Some Languages

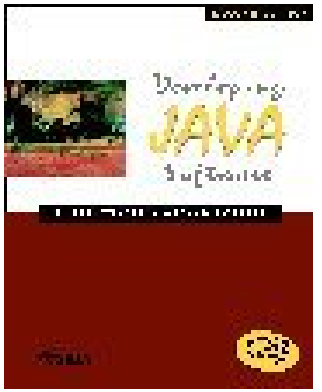
- Python, Groovy, Ruby, Java, C#, C++:
  - Factorial
  - Mail merge with LaTeX
  - Email mailshots

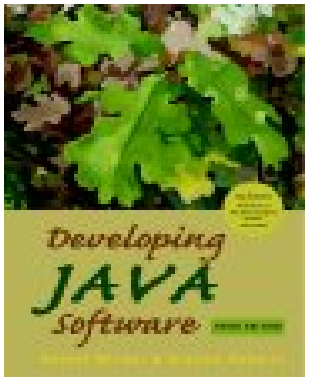
Factorial

- A truly trivial application.
- Implementing it raises many questions:
  - Native data types.
  - Exceptions for error handling.

LaTeXMerge

- Mail merge facility for LaTeX.
- Originally in C.
- Ported to (bad) C++.
- Ported to Java (see *Developing Java Software*, first and second editions, not third).
- Ported to Python, Groovy and Ruby.
- Ported to better C++.





Emailshots

- Not a spamming tool.
- Email version of mail merge nonetheless.
- Originally in Perl.
- Rewritten in Python, Ruby, Groovy, Java, C++.

Computers, computers everywhere . . .

- . . . and not a single one is a sequential uni-processor.
- Parallelism is (finally) here.
- How to program them? Threads perhaps?

Threads

- Threads
- Locks
- Semaphores
- Monitors

- Lock-free programming.

Most programmers cannot handle multi-threaded programming. Doesn't matter whether it is C, C++ Java, etc.

Alternative Paradigms

- Functional programming:
  - Graph rewriting.
  - Graph reduction.
- Different techniques for parallelizing software.

The Functional Imperative

- Functional systems separate program and execution engine.
- Imperative: programmer manages parallelism.
- Functional: engine manages parallelism.

Functional Failure

- SML/NJ, OCaml, Haskell have not taken off.
- Imperative too ingrained:
  - C, C++
  - Fortran
  - Java
- Hegemony of the object-oriented paradigm.

Declarative Success

- Fortran: Whole array operations.
- C: object-based applicative programming.
- C++: STL.
- Python: List comprehensions.
- Ruby, Groovy: Closures.

Declarative styles of programming are the future.

Domain Specific Languages

- Development is about creating the right language to describe the solution.
- Dynamic languages (with MOPs) make this easy.
  - Python
  - Groovy
  - Ruby
- Glimmer of hope for C++?

Ludwig Wittgenstein:
Logisch-Philosophische Abhandlung
(Tractus Logico-Philosphicus)
Philosophische Untersuchungen
(Philosophical Investigations)

Preferred Development Strategies

- Prototype in Groovy
- Refactor into Java as needed.
- Refactor to C if needed.

- Prototype in Python.
- Refactor into C as needed.

C++? Why bother?

Conclusions

- C++ templates are a “feature to far”.
- C++ is too complicated.
- C, Python and Groovy are the languages of choice.