CS22510 Assignment - Part 2

Tom Leaman (thl5)

May 2, 2013

1 Language selection for part 1

1.1 Checkpoint Manager

I decided to write the Checkpoint Manager program in Java as this allowed me to make use of the Swing libraries to write the GUI. The Swing framework provides an easy-to-use set of widgets which (at least in theory) is not platform specific. This means that a single compiled version of the program can be run on a whole host of different hardware during the event. This would mean that the event organisers (who will probably want a computer at each monitored checkpoint) can make use of any and all machines that can be begged and borrowed for the event. The only requirement is that they are able to run a JVM.

1.2 Event Manager

I wrote the Event Manager program in C. This program is, in essence, a refactoring of the CS127 assignment and to write this again in another language (especially one which I am less familiar with) I felt was simply not sensible in the time alloted.

That said, I feel that this is the most important part of the system, especially in terms of race safety (it may be critical to know whether competitors are still running or not) and I think it would be easier to have confidence in this program (after a proper software audit) as every part of the system can be examined. That is, the program makes only very minimal use of external libraries and those that are used are well understood and trusted. If this program were to be written in Java, for example, it would be easier to make use of a library which may not do exactly what it claims.

1.3 Event Creator

My decision to write the Event Creator in C++ was essentially forced by the specification to write each program in a different language. I have made no use of C++'s ability to provide an Object Oriented class structure. Of the three programs, this was the most simple and C++ is the language I am least familiar with.

2 My Experiences

2.1 Java

Having programmed in Java for nearly 2 years before arriving at University, I feel I have a reasonably good grasp of what is and is not achievable, in addition to a good understanding of the Swing framework. This allowed me to write the Checkpoint Manager program reasonably quickly.

I feel strongly that Java's primary benefit is the wealth of libraries available (for free, no less!) in addition to the inline documentation provided by Javadoc. This makes it incredibly easy to write clean, reusable code with accurate, helpful documentation.

My only frustration with Java is that it is quite easy (even likely) that one will end up producing several files with little to no actual code within them. For example, a simple interface with only a single function will add an entire file, plus boiler-plate for the interface declaration, when in essence we only care about the single line defining the method in question. This can lead to a quite complicated directory structure before any useful code is actually written. On the other hand, this does allow the programmer to make good use of the Object Oriented-ness of Java and all that that brings with it (e.g. type safety).

2.2 C

I was new to C when starting the CS127 assignment but I have found it to be a very pleasant language to program in. I find the degree of control (lack of implicit *this* pointer etc.) actually works in my favour. That said, it's lack of error checking at compile time can sometimes be a hassle and working in C without tools like GDB would be a much less pleasant experience.

To begin with, I found it hard to know how to layout my programs in C. Having come from an Object Oriented background, it was sometimes hard to know when to put sections of code into separate files. I think by the time I got to this assignment, I felt more confident in knowing when separating the code out would be beneficial; and overall, I feel that I have produced a reasonably clean program.

2.3 C++

I had not written any C++ before this assignment and I think it would be fair to say that I did not make full use of its capabilities in this assignment. I think, given more time, I would have liked to make use of classes to model the data taken from the user and do some error checking (there is currently absolutely none in my implementation.

I feel that the code I have written is reasonably clean and tidy and understandable to anyone with any grasp of C++. However, I personally find C++'s syntax (especially with regard to namespaces and string output) to be quite clumsy and cluttered looking. I much prefer printf style output where one knows that all the variables will appear at the end of the line, as opposed to C++ with its angle brackets and EOL characters.

3 Reflection

My greatest frustration with this assignment was the specification that each program be developed in a different language. I appreciate that it is a requirement for us to demonstrate that we have some knowledge of each language but I feel that in a real-world scenario, the Event Manager and the Event Creator programs would, most likely, be written in the same language. If I were asked to do this again, I would make sure they were.

I still think Java is the best language to write the Checkpoint Manager program in, as stated above, it can be run on any system that is capable of running a JVM and developing GUIs in Swing is relatively easy.

If I were working with others, I would consult with them to determine a consensus as to whether the Event Manager and Event Creator programs be written in C++ or C. Personally, my preference would be to work in C. This is partly due to the same points Linus Torvalds raised in a posting to the gmane.comp.version-control.git newsgroup. He points out that

> "...the only way to do good, efficient, and system-level and portable C++ ends up to limit yourself to all the things that are basically available in C".

While this is not system-level programming (we're not writing an operating system), it is a crucial system that people's safety may rely on and it's just as important to get it right! Using libraries may be more efficient and produce cleaner source code but overall adds complexity and code-bloat to the overall system.

In addition, Torvalds points out that there are a great many programmers who know just enough C++ to be dangerous and, typically, C programmers have a greater understanding of how the code gets compiled and run which tends to lead to better code. I think this point could be debated at great length, but I have certainly felt it to be true in my limited experience.

The choice to write both Event programs in the same language would also allow for reuse of structures and functions relating to nodes, tracks etc. which should lead to cleaner code and shorter development cycles.

4 References

- Linus Torvalds newgroup post - http://harmful.cat-v.org/software/c++/linus