Subject: Re: Collaborative U++ Projects
Posted by cbpporter on Fri, 19 Sep 2008 15:06:40 GMT
View Forum Message <> Reply to Message

Sure, getting some open-source projects out here could help popularity.

But I see some problems with that:
1. Who should do that? I don't think there are a lot of people (using U++ or not) who actually have time to sit down and just start coding on some random project. I for one only code new projects in three distinct cases: it brings me money, it is really fun (happens rarely, but it does) or I need a small tool for something that the normal tools can't quite hack (and in such cases I'm usually done in some days; I have a lot of small apps that do data processing mostly or organize stuff, and for some strange reason I never wrote one in a language more suited for such tasks). And I think something similar to this applies to all working U++ developers.
2. What exactly to code. It should be useful, yet should not reinvent the well. I think that hundreds of programs that do the same stuff, and also forking have in general a quite negative effect on software as a hole, and especially on open-source. Please install Linux, and install all the calculators you can find in your distro. Then install all the Paint like programs, then the text editors. And the list goes on and on. If all the people who ever contributed to a text editor would have pulled their effort, we would have 2-3 extraordinary editors (which hopefully would not have turn out a monstrosity like emacs) and not 10 GUI ones installed almost by default, out which 5 are from Qt, 3 Gtk+ and 2 are just plain X.
If someone wants to write a new text editor for example, having a revolutionary design or at least one that caters for special needs of a subgroup of people and which is feature-wise quite different from the ones that exist, than that someone just might have a good reason to do that. If instead the reason is that some toolkit is better than the rest (in reality or just perceived), and there is not a program to solve the same task written in that toolkit, that that someone's reason is not that good and IMO will just harm open source by causing more redundancy.
3. Rewriting something in U++ would give the best results when that certain something is written in C. Unfortunately, most software out there is written in plain C, and most of the time not even in a civilized subset of it. Rewriting something like that would show of our capabilities quite well and lead to a better piece of software. But rewriting something which uses STL, Qt, Gtkmm or wxWidgets would not bring any significant benefit. Sure, you could probably pinpoint some code sections, but in the end, the whole effort would not be too justified.

I had the pleasure of working a little with two open-source C code bases, which I'm not going to name. One of them is one of the most horrible pieces of C software in regard to it's design and the way C is used to make it unreadable (but it's functionality is exceptionally good, making it required software). This project, quite used and probably installed on you favorite distro would in principle benefit hugely from a rewrite, and would get at least 80% of that benefit from using a string class. Any string class, but something smart like String would reap the best rewards. The second one is very well written, and I don't think that using anything, not even the absolutely perfect language and library that don't exist would have a major benefit on it, neither on functionality, nor on code clearness. Sure, being C, there are tons of stuff you could clean up, like using modules and namespaces, getting rid of macros, a little const correctness, etc., but this could be done in about a week or two and the function implementations would be virtually unaltered.