Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by tvanriper on Tue, 22 Apr 2008 12:09:59 GMT
View Forum Message <> Reply to Message

First and foremost, the documentation holds Ultimate++ back from mainstream adoption. It took me almost a month to get mostly comfortable with the toolkit, not because it isn't easy to use, but because I had to literally study the code in order to figure out how to work with it. This acts as a barrier to entry, when you have other toolkits with far better documentation (e.g. MFC *choke*).

Next, you need to get the attention of some heavy hitters. Most of these guys, from what I can tell, hang out at boost.org. In the United States, I've worked for about five different large-ish companies, and in all of them, they held a tremendous respect for the efforts at boost.org, because of the whole peer-review system, and because often the code at boost.org makes its way into the standard library over time.

You mentioned an interest in getting Bjarne Stroustrup's attention. You can find him participating at boost.org. If you seriously want to cause Ultimate++ to be put into the standard C++ library (which would be an amazing feat for a library like this), put the library up for review.

If you do this, I can tell you right now what will happen, at least initially.

Ultimate++ will be shot down.

 Poor documentation; they're picky about having helpful documentation.
 The use of Ultimate++-centric interfaces instead of standard interfaces. More on this later.
 Possible portability issues.
 Possible size of system.
 namespace isn't in boost.
 Duplication of technologies already available in boost (e.g. boost::thread already exists for multi-threading, and boost already has a system for handling signals and slots).
 Difficulty in using Ultimate++ without using TheIDE. I know you're already working (or maybe finished working) on addressing this issue.

The NTL system you developed will most likely be an issue for the boost.org team, not because the underlying technology is terrible, but because the interfaces do not match the kind of interfaces used with the standard library. It doesn't 'look' like something you'd see in the standard library.

I think that's a small issue. You can accomplish both your technical goals (fast, agile code) and still have the library 'feel' like it belongs as part of the standard template library.

Although I'm confident you'd get shot down, there are several significant benefits in submitting the library for review, regardless.

 You'll get serious, useful information for improving Ultimate++.

You'll get serious, useful exposure to a very large community of developers, some of whom are surely interested in the problems solved by Ultimate++.

Note that almost all first attempts at a library get shot down at boost.org, so it isn't necessarily a bad thing. Personally, of all the toolkits out there, this is the only one I think could come close to being put up for review at boost.org without being outright mocked. You already have the right kind of license. You tend to follow ideals that seem in line with what I've seen at boost.org. You solve a problem domain that a lot of people need solved.

However, maybe you do not have a goal to cause Ultimate++ to become part of the standard library. In that case, don't bother submitting it to boost. You'd only be wasting each other's time. Personally, I think that'd be a pity, as I think both of you (boost and Ultimate++) could benefit from each other.