Subject: Re: Which is the biggest drawback of  U++ "unpopuliarity"?
Posted by tvanriper on Sat, 26 Apr 2008 00:36:48 GMT
View Forum Message <> Reply to Message

luzr wrote on Fri, 25 April 2008 07:52tvanriper wrote on Fri, 25 April 2008 07:14luzr wrote on Tue, 22 April 2008 08:44tvanriper wrote on Tue, 22 April 2008 08:09
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.

I am not quite sure about it being a "small issue".

IMO it all starts with reasons for String vs std::string...

Mirek
This is where the peer review system is very helpful, I think.

How do you think it could help to resolve this problem?

The real issue IMO is that boost is std:: extension, while U++ is std:: replacement... The basic design fundamentals and requirements are quite alien to std:: / boost concepts. Therefore I just fail to see how std:: replacement could do anything with boost.

Well, the only advantage would be to "get the attention", but I am not quite sure that is a good idea either....

(OTOH, peer review process itself is a very good idea, I wish we had something like that for U++ too...).

Mirek

If I have it right, your primary concern with std:: involves its relatively terrible performance, and I think the technical issue had to do with the way you manipulate memory.  If that's the concern, someone could potentially help you find a way to achieve the same performance you currently get with NTL, while using a more std::-like interface.

I could, of course, be mistaken.  I'm not completely clear on why you feel these are so incompatible... as perhaps I'm not 100% clear on your design goals, or I'm ignorant of the fundamental problem you see in std::.  I've read and re-read this page, but I still can't quite see how U++ and std:: can be so incompatible that there's no hope of improving the std::-style system to the point of matching U++ performance.  I've certainly seen quite a few tricks used at boost:: that involved improving performance in a variety of ways (sometimes by providing a replacement for an existing library in std::, I think)... I should think they'd take quite a bit of interest in your techniques, and could possibly figure out a way to keep the same performance while making fewer changes to the existing interfaces.

I only pose this idea because it feels to me like you and boost have similar goals.  I could, of course, be wrong.  I know, for example, that boost has less of an emphasis on performance and more of an emphasis on their idea of 'correctness', so you may differ significantly there.  (This is certainly not to say you have no concern for 'correctness', but that you may have a slightly different idea of what is 'correct' from boost).

You can best decide for yourself whether or not to approach them by reading some of the reviews people have left, and examine the decisions used to arrive at their conclusions.

Regardless, you could probably have your own peer-review system... it'd be interesting to see how people here would work with that.  At the very least, such a system might help you gain some continuity between various objects in the U++ system.

As for gaining the attention of some of those heavy-hitters... well, there can be positive and negative results, to be sure.  I thought it was possibly of interest to you only because you had mentioned attracting the attention of Stroustrup.

Perhaps someone could submit an article to Dr. Dobb's Journal showcasing the use of Ultimate++; that's a fairly popular magazine, at least here in the United States (the CUJ folded to Dr. Dobb's a few years ago, sadly, or I would have recommended it instead).