Subject: Re: Which is the biggest drawback of U++ "unpopuliarity"?
Posted by Novo on Sun, 27 Apr 2008 16:26:45 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 26 April 2008 02:11

BTW: I mostly care about "optimality" with U++. If I would care about "popularity" more, I would certainly use another path and boost would be the part of it.

I still do not get the strategy behind U++. You are saying you do not care about "popularity", but you have posted almost five thousand messages on this forum. It looks to me like you care very much about users and "popularity".

What "popularity" means to you? And how do you understand "optimality"?

As I understand, U++ is a self-contained application framework, which is planned with two goals in mind: performance and easiness to use.

Everything that doesn't match these goals gets ignored. That includes third-party libraries like boost, dll-based builds, makefiles for Unix and project files for MSVS.

1) Boost is not just containers and algorithms. It intersects with U++ at least in Boost.Thread, Boost.Spirit, Boost.Wave, and Boost.Function. But that is an external dependency.

2) Dll-based architecture seems to introduce some performance worsening. Automatic exporting of all classes and functions would impair optimization. Manual marking functions and classes for export would require some time, and there still be some performance worsening.

3) Project files / make files for other build systems would definitely improve popularity of U++. It could be included in popular Linux distributives. That doesn't seem to break either "popularity" or "optimality" principles. Without make files it is not possible to build U++ in regular way.

IMHO, before investing resources in making U++ more popular it is absolutely necessary to understand (and clearly explain to others) what actually U++ is, and what it is going to be. Otherwise you will just confuse new users.