Subject: Re: U++ development: Two philosophies
Posted by harmac on Mon, 17 Jan 2011 21:46:08 GMT
View Forum Message <> Reply to Message

I'm new to C++ and U++. People seem to use both for their projects to solve specific problems of theirs or to get some work done, and while both tools are of some help, their main concern is not necessarily the improvement of the tools but to solve their problem. Considering this, the notion of "development philosophies" might not be the best approach to describe U++ packages development, for although two different package developers might face similar problems, they might still have different requirements for their specific problems, possibly at different times, and this might be a reason for packages with similar functionality, which could then be seen as uncoordinated efforts.

It would indeed be very desirable to avoid such parallel efforts, but U++ itself doesn't seem to be a coordinated effort. Does it seek to be one? Things would sure be somewhat easier, if people knew their future demands in advance, or if U++'s developers weren't distributed people with different circumstances as regards their jobs or tool requirements. This is rather an inherent problem that the main development effort of these people is not U++ itself.

Apart from that, I'm with Koldo in that cooperation looks like a better approach than competition, or, as he phrases it, "1+1 > 2". I'd say, that is, that a team has properties not present in either of its members (I think, Wikipedia would call that "emergence"), and that might be in addition to any synergies.

So, if it can be done, work could be much more productive and enjoyable, if people would engage in joint efforts to create something great. Because of that, it is all the more dissatisfying, that there seems to be some lack of actually great tools for project management. Also, while the discipline of software engineering knows some abstract methodologies to organize development and possibly people involved (you may, e.g., have heard of the Rational Unified Process, the Microsoft Solution Framework, or Extreme Programming), I'm not aware of an abundance of software to support such development processes. And as regards distributed development, there might be even less in existence. Or is there any and it is not just well advertised, so that I've never become aware of it?

After all, software development with multiple people is also a social activity, and I think, Koldo makes a good point when he mentions having also spoken to people here on a personal level. It is probably an aspect underestimated more often than not, that people should be first class objects in any development process. This applies to both users and developers and whoever else. The development community of U++ is geographically distributed and there are diverse interests, which can be a challenge to coordinated efforts and may sometimes lead to misunderstandings (see MediaPlayer). I'm not aware of a definite solution, which is somewhat unfortunate, but such is life. There's probably still room for some innovative invention, though.

The welcoming and responsive and willing-to-help nature of some members of the U++ community seems to be something really worthwhile for each side. Think of it: For one, explaining something to others helps the own understanding, and on the other hand, being new to something should be really much less of a pain, if there are helpful people around. This should also remind people of writing good documentation, which might be something even more important, as poor to no documentation may be the root of all evil in the first place.

So with much digression, what did I want to say?

1. Working together should outperform working concurrently without communication. So the former should be done whenever possible.

But with people facing time constraints, having diverse interests or requirements, being emotionally unstable, possibly maintaining different cultural mentality, etc., it just might not always be possible to do it right, and conflict solving may be necessary. In doubt, being friendly and assuming no bad intention from others is likely to be the most relaxed way to tackle misunderstandings and should be taken as a starting point.

2. Web based project management systems would be something really nice to have. Do you have something in particular in mind?

3. I consider awesome documentation as something at least as much important as the software itself. Surprisingly, most software's documentation nowadays sucks. Documentation might even be a prerequisite for coordinated project management. Interfaces should be specified and documented before they're implemented. This would also facilitate coordinated implementation by multiple people.