
Subject: U++ development: Two philosophies
Posted by [koldo](#) on Thu, 13 Jan 2011 08:17:03 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello all

After MediaPlayer misunderstanding (see here) there has been a conversation in the background that has explained it all.

The conclusion is that in U++ there are at least two development philosophies:

- Lone genius does the best job
Sometimes there are duplicated works, but finally the best remains like in Natural Selection.

- Team does the best job
Coordinated job avoids duplication and improves focus to let arriving to the goal with less effort.

My point of view is the second: Uncoordinated jobs like doing StreamCypher after AESStream, or Honza effort after MediaPlayer, without any previous explanation are, from my point of view, an heresy.

Because of it, Web based project management systems would be super important.

Subject: Re: U++ development: Two philosophies
Posted by [tojocky](#) on Thu, 13 Jan 2011 11:45:12 GMT
[View Forum Message](#) <> [Reply to Message](#)

koldo wrote on Thu, 13 January 2011 10:17My point of view is the second: Uncoordinated jobs like doing StreamCypher after AESStream, or Honza effort after MediaPlayer, without any previous explanation are, from my point of view, an heresy.

Hello Koldo,

I do not agree with you to highlight someone bad.
If someone is proud to publish some ideas that may be repeated or come into contradiction with each other, our responsibility to analyze and expose our opinion where is repeated and where is contradiction.
I believe it will be the fairest way.

Only together we can do more!

I propose to implement voting messages with "like" and "unlike". Every message will have cont of like and unlike votes.

Moreover, to improve better development, I propose to add a new section where we can add new future requests and bugs with comments, voting and life cycle statuses.

This seems already to be:
<http://code.google.com/p/upp-mirror/>
or
<https://sourceforge.net/projects/upp/>

We need to decide, where we can do this?

Subject: Re: U++ development: Two philosophies
Posted by [koldo](#) on Thu, 13 Jan 2011 16:57:54 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello Ion

Imagine I want to do a translation package based in Google. I would have two options:

1. I can take code from here and there and prepare a new package.

Why not, is not this BSD?

2. I can talk to you and agree a new and better Google Translation version.

I would always choose the second one. For me U++ people are colleagues. In fact I have talked with more than one about personal subjects. I count on you and I want you count on me. . I think this way $1+1 > 2$.

Of course this is my personal view. Key, nice U++ people do not think the same.

If somebody thinks that $1+1 > 2$, s/he can count on me.

Subject: Re: U++ development: Two philosophies
Posted by [tojocky](#) on Thu, 13 Jan 2011 21:28:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

koldo wrote on Thu, 13 January 2011 18:57Hello Ion

Imagine I want to do a translation package based in Google. I would have two options:

1. I can take code from here and there and prepare a new package.

Why not, is not this BSD?

2. I can talk to you and agree a new and better Google Translation version.

I would always choose the second one. For me U++ people are colleagues. In fact I have talked with more than one about personal subjects. I count on you and I want you count on me. . I think this way $1+1 > 2$.

Of course this is my personal view. Key, nice U++ people do not think the same.

If somebody thinks that $1+1 > 2$, s/he can count on me.

I agree with you. You can only say where his work is repeated and propose to implement his new ideas in base of existing project for analysing and to improve the existing package.
Last example I tried to to propose for improving the Thread class, or scatter control.

Subject: Re: U++ development: Two philosophies
Posted by [281264](#) on Sat, 15 Jan 2011 14:03:02 GMT
[View Forum Message](#) <> [Reply to Message](#)

I fully agree with Koldo; actually it has been a bit confusing to see brilliant tools so similar each example Docking and DockCtrl-now solved).

So, I reckon that some coordination or management is essential. There is a word to describe $1+1>2$ = Synergy. How to make it?
Cheers,

Javier

Subject: Re: U++ development: Two philosophies
Posted by [mdelfede](#) on Sun, 16 Jan 2011 11:52:29 GMT
[View Forum Message](#) <> [Reply to Message](#)

koldo wrote on Thu, 13 January 2011 09:17Hello all

.....
My point of view is the second: Uncoordinated jobs like doing StreamCypher after AESStream, or Honza effort after MediaPlayer, without any previous explanation are, from my point of view, an heresy.

Hmmmm.... No, I don't agree with you. I did StreamCypher because I needed it, and I liked to share it, as most of my code.
I can assure you, it would have been *much* easier to code it just for my purpose and keep on my pc, but I think the good (and most important benefit) of open source is SHARING, not team coding.

BTW, about StreamCypher, as you've seen, was replaced by a more generic Cypher package which aimed to enclose all previous efforts on that subject, following suggestions got from forum. The changes were made to allow to merge AESStream on it; it supports also block cyphering backends, which former didn't.

I think having the same stuff, or almost the same stuff, coded in different ways, can be useful sometimes, and can help us to see the different approaches on the same subject.... some of them can lead to better implementations, or even some *parts* of them can do it. Merging codes later is always possible if people agree.

What I don't like, and that's what I call "stealing other work" is, IMHO (hope not launch a flame about....) what Apple did with OpenBSD. They took other's people work, added some fancy stuffs and sold it as a proprietary job, as if they did all the hard work. Worse than that, they don't allow people run it on other machines which are not sold by them.....

With this I don't say that people isn't free to take other's code and make it better, but just that credits should be given to original creators and **no** tell the world that the code was homebuilt completely. The best would be of course **share** at least some of the changes that made code better.

I'm developing (and I posted my latest efforts about...) an Updater package which is quickly becoming a web installer package. I know that there's an Updater in Upp (I must say that I didn't look at it in depth) but I wanted some features that I've not seen in original Updater, for example the ability of update itself without using external helper apps, and the ability of handling more versions.

I don't want absolutely to say that my updater is better or that it wants to show that I can do better code (I can't, indeed), but it's just a different approach to a similar problem. Both codes could benefit on other's one efforts, and I find it good.

Coordinated efforts **can** be a positive stuff too, but they're difficult to manage without a convenient organization. Upp **IS** a convenient organization that allow to merge efforts in most fields... people propose changes and if they're good they're accepted, and quite easily (as opposite, for example, as Wine project on which having your code merged is a nightmare sometimes....).

Coordinating Bazaar snippets would mean having one or more persons to follow it's development 8 hours a day.... something unfeasible, imho. I find much better some sort of auto-coordination when people like it or find it useful.

Ciao

Max

p.s.: I find Bazaar one of the **greatest** stuffs of Upp. It allow us to share (and sometimes FORCES us to polish....) some code that otherwise would be lost in our own hard disks. It's one of the best ideas that I've ever seen around, and I really think it should continue like this, a place for sharing code snippets.

Subject: Re: U++ development: Two philosophies
Posted by [Mindtraveller](#) on Mon, 17 Jan 2011 15:21:02 GMT
[View Forum Message](#) <> [Reply to Message](#)

Quote:doing StreamCypher after AESStream I must apologize for the lack of my effort on porting AESStream functionality to StreamCypher interface as I promised long time ago.
Frankly speaking, I did a number of attempts to start porting but couldn't really apply my functionality to the common interface proposed. From my point of view it is not good to ask author to change his interface personally for me, so for now I really don't know what to do.

There is another example: some years ago I proposed approach to make really good help articles on U++, here on forum. It was a kind of forgotten and now I meet the same ideas from another people who is thinking of it from the start, once again. I f we had good coordination, these people's time could be used more effectively, without the need of reinventing the wheel (U++ itself is an exception of this rule, of course).

IMO, Koldo is absolutely right: good coordination + team work is VERY effective. IMO we MUST have subforums for each U++ team project. Each project interface MUST be discussed and documented on forum BEFORE the code is written.

Subject: Re: U++ development: Two philosophies
Posted by [fudadmin](#) on Mon, 17 Jan 2011 15:47:14 GMT
[View Forum Message](#) <> [Reply to Message](#)

Mindtraveller wrote on Mon, 17 January 2011 15:21 IMO we MUST have subforums for each U++ team project.

+1. Long time ago that was the idea behind " U++ users applications in progress..." where I expected 1 thread per project. But then came the name bazaar which now reflects the nature it... (russian word "bardak" would reflect better)

Maybe we could use some stages similar to Archlinux repos like AUR, trusted users-community etc. Ideas?

Subject: Re: U++ development: Two philosophies
Posted by [mdelfede](#) on Mon, 17 Jan 2011 16:47:51 GMT
[View Forum Message](#) <> [Reply to Message](#)

Mindtraveller wrote on Mon, 17 January 2011 16:21Quote:doing StreamCypher after AESStream I must apologize for the lack of my effort on porting AESStream functionality to StreamCypher interface as I promised long time ago.

Frankly speaking, I did a number of attempts to start porting but couldn't really apply my functionality to the common interface proposed. From my point of view it is not good to ask author to change his interface personally for me, so for now I really don't know what to do.

.....

Don't worry, just ask

The interface can change, of course.

BTW, the "updated" package is Cypher, StreamCypher was the former and it's superseded by latter one.

Ciao

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.

Subject: Re: U++ development: Two philosophies
Posted by [fudadmin](#) on Mon, 17 Jan 2011 23:05:09 GMT
[View Forum Message](#) <> [Reply to Message](#)

Welcome to the forum, harmac!
I hope you will enjoy our cosy, steadily growing community!
May I ask you what breeze took you here?
Your post betrays you as a good documentation writer...

Subject: Re: U++ development: Two philosophies
Posted by [koldo](#) on Tue, 18 Jan 2011 08:26:11 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello all

I think maybe I have been somewhat excessive in the presentation.

Honza is a great person and, after two years and a half of contact, I have no doubt that absolutely

everything he does, he does it selflessly and with the best intentions. And this is not rhetorical. I say it sincerely.

After the initial anger came the reflection: why this can happen.
After that comes learning, what we can learn.
And after that the action has to come.

Subject: Re: U++ development: Two philosophies
Posted by [dolik.rce](#) on Tue, 18 Jan 2011 19:44:07 GMT
[View Forum Message](#) <> [Reply to Message](#)

koldo wrote on Tue, 18 January 2011 09:26Hello all

I think maybe I have been somewhat excessive in the presentation.

Honza is a great person and, after two years and a half of contact, I have no doubt that absolutely everything he does, he does it selflessly and with the best intentions. And this is not rhetorical. I say it sincerely.

After the initial anger came the reflection: why this can happen.
After that comes learning, what we can learn.
And after that the action has to come.
Hi Koldo

I took no offense, don't worry. Since the beginning I believed it was just a mere misunderstanding, because I know you are such a good person. I am really glad to hear you are not angry with me anymore.

And maybe, in some way, it was all a good thing - it started the debate about issue tracking and better coordination which can only help the U++ project and community. In Czech there is a proverb that says "everything evil is good for something". I hope that this is the case

Best regards,
Honza

Subject: Re: U++ development: Two philosophies
Posted by [harmac](#) on Wed, 19 Jan 2011 19:59:40 GMT
[View Forum Message](#) <> [Reply to Message](#)

Thanks, fudadmin! I haven't actually written any documentation until now but for a lack of experience in software development. So now I need to learn about some tools. After a first exposure to Delphi many years ago, which by the way actually felt somewhat more productive than what came after, I got to learn Java, and that's basically where I am halfway. A want to use

OpenGL took me to reading about Win32 GUI programming. Reading the first example code, I thought, oh my god, how ugly! I didn't do anything with it, read about how X does it (not much better).

Now, having grasped the notion of APIs in existence, there are fortunately some cross-platform approaches, of which I hope U++ will be interesting. Now, sorry for off-topic, I hope I can learn something.
