

Re: From the LuxAsm list.

Source: <http://coding.derkeiler.com/Archive/Assembler/alt.lang.asm/2004-08/0468.html>

From: Beth (*BethStone21_at_hotmail.NOSPICEDHAM.com*)

Date: 08/10/04

Date: Tue, 10 Aug 2004 01:45:51 GMT

T.M. Sommers wrote:

- > *If it were my project (and obviously it is not), I would not even*
- > *create an IDE, but would create a LuxAsm mode for emacs. I do*
- > *not want to waste time and effort building an editor when one*
- > *already exists much better than anything I could build, or*
- > *building a make tool, or a versioning system.*

But what if the editor IS better?

The one proposed has "feedback" incorporated...it eradicates "assemble time / speed" by adopting a different style of approach to the problem...things that other editors simply do NOT typically provide...

- > *You actually missed the main point of the Unix philosophy, which*
- > *is that each program should do one thing, and do it well.*
- > *Integrated tools generally do many things, and do them, at best,*
- > *adequately.*

No, you've missed the point with LuxAsm...the method of "integration" is not as you believe...each of the tools is a quite separate model that does one thing and attempts to do it as well as possible...the "integration" is only a user interface thing...

If you like, it's a "visual shell script" with multiple "command-lines" hidden behind the press of a toolbar button...in the sense that the IDE provides buttons which perform the operation of calling all these tools in the required sequence to complete a more complex task...the actual toolset itself is a series of separate "modules" that simply "link" into the main IDE interface...

Note that even the "editor" is a separate "module" that links into the IDE...the "integration" is purely logical...NOT physical...that is something that we're doing differently to Rene...and, also unlike Rene, there will be an intention that every single one of these tools can be accessed via a command-line interface separately and independently to do one and only one task as best as it possibly can...

alt.lang.asm: Re: From the LuxAsm list.

Indeed, is GCC violating this "UNIX philosophy"? Because it automatically calls tools like "ld" or "ar" or whatever to perform its work of compiling the C / C++ (or one of the other supported languages) source code into a working executable (though you can ask only for compiling but no linking and so forth too)...

LuxAsm will operate in the same exact method _except_ simply that the "user interface" style is graphical, not textual...these are _separate_ tools in the "toolset"...the "integration" is simply about _visual presentation_ on the graphical screen...and the "interface" (as in "connecting" to the tool, as opposed to the UI) will be _programmatic_...

The reason for the _programmatic_ nature is an attempt to _detach_ the tool's work from the tool's "user interface" (which is _entirely cosmetic_...does English "stdout" text actually have anything to do with "doing one job and doing it well"? Nope...arguably, I'm pursuing this "philosophy" far MORE than most)...this permits the ability of providing more than one "method" for accessing the tool, if we consider it as an "object" in OOP terms (an encapsulated "object" for fulfilling one set of related tasks: "Code re-use" before you try to attack that too)...this allows for a level of "tight integration" – the assembler to call the linker, the editor to call the assembler, the project manager to call the assembler and linker and so forth in all manner of "combinations" – to do various tasks for each other...the tools, though, remain completely "encapsulated"...

The "mutual exclusion" you're alluding to is purely imaginary...

The PURPOSE of the "UNIX philosophy" is the notion of creating a set of simple, reliable, single-task "building blocks" that can then be used together ("pipes") or in sequence in order to create larger scale functionality...this is exactly what is intended 100%...don't let the fact that it doesn't use a command-line and that most GUI programs tend to be "monolithic" confuse you into thinking that "GUI = monolithic, command-line = UNIX philosophy" ...it's not that simple...the entire design I've proposed to the LuxAsm team – which I believe they could confirm – is _based_ on the idea of NOT meaning that just because we've got a GUI, we have to abandon "UNIX philosophy"...

The IDE simply serves as a "server"...the graphical equivalent of the "shell"...there are buttons and menu options and text boxes instead but these actions are converted into the equivalent of, say, "editor | LuxAsm | linker << file"...except this is a false representation because rather than use a command-line interface (OR a GUI one either), the tools actually "interface" is entirely _programmatic_...that is, the IDE loads up the editor "module" and a click on a button does the "pipe" programmatically by calling a function in the assembler module...and so forth...user interface components deal with the input / output aspects of the tool (converting an "error code" return into an English language string for the command-line or a French language dialogue box...the I/O is _unrelated_ to the task in hand...I'm being MORE "UNIX philosophy" than you've ever seen before here,

Re: From the LuxAsm list.

alt.lang.asm: Re: From the LuxAsm list.

in fact...the I/O is NOT strictly a part of that "one task" which needs to be "done well"...the tool, therefore, needs no I/O directly...this is a function of the "user interface" so the "filters" which connect them to the IDE "server" (or, for the command line, same idea but different "user interface" style of text I/O) provide this functionality, not the tools themselves)...

Indeed, this might be hard to see because I don't believe I've ever seen any program that takes such an approach to demonstrate as an example of the "principle" in full...you can merely get a grasp from how GCC operates to get a rough idea...the tools are independent and the "unifying layer" is the IDE itself and its "UI filters" to the tools...it, therefore, will have no bias to the GUI or command-line interface (beyond any natural "bias" that team members have to like the GUI aspects and, well, spend more time there than elsewhere...the structure, though, is unbiased: The tools have NO user interface at all)...the tools are independent –with "one task and do it well" – but may be "integrated" logically by the combinations in which they are used (no different – but the UI style – to notions of "pipes" and "redirections" and such...works entirely programmatically – because, again, there is no "bias" towards "command-line" or "GUI" mentality – but the logical structure of the method is the same)...

Although, what's a touch hilarious is condemning these ideas and then proposing it be made into a "mode" for "emacs"? The beast of a thousand faces that "emacs" is...yeah, right...and Rene makes equal sense attacking Microsoft with a "PE / Win32 specific" assembler as well...

Plus, you did not logically back up this bizarre assertion that "re-invention" somehow means it'll be "twice the cost" and "twice the effort"? Ummm, surely, if one is entirely duplicating what already exists bit for bit then the cost is the same as the thing duplicated and the effort is the same as the thing duplicated? You're just copying what they did before...hence, it'll be the same cost and effort, roughly speaking, as was originally taken...on the other hand, I personally did not make that original effort...so what effort someone else might have made has no bearing on that...or is the logic here that because someone else somewhere in the human race once used an ATM machine to get money out of the bank then that's it, no-one else is allowed to "duplicate" and do anything approaching the same thing thereafter? Nice logic...that means creating ATM machines was a complete waste of time...and, indeed, if "progress" is now halted because you're not feeling too happy about it all, then all the progress that lead up to this point is instantly rendered in vain...that logic seems flawed to me...pardon if I don't take the suggestion until I can see that the logic actually seems to add up...feel free to join up the dots, though, if I really am being dense and haven't seen something you feel is crucial that prohibits or invalidates what I'm talking about here...

Beth :)

Re: From the LuxAsm list.