2024-04-04 DI - Design Intent

DI - Design Intent

The goal is to communicate design - quickly and easily.

The goal is to allow push back on the design and, if someone likes the design ideas, to steal the ideas and to reuse them elsewhere.

A notation is measured by how easily it communicates ideas between humans.

If you want to communicate scripts to CPUs, you *must* use assembler. Period. By definition.

Haskell, for example, ultimately compiles down to assembler. The CPU doesn't care that you used Haskell. You, and other developers, are the only ones who care that you used Haskell.

Can we come up with alternate notations from Haskell for communicating between humans?

Can we use more than one notation for communicating between humans? For example, using Haskell for expressing complex calculations (military ballistics, crypto) plus using StateCharts for expressing scripting recipes plus PEG for expressing text parsers. Maybe more notations? A 2-type language for playing around with ideas and refining them incrementally vs. a zillion-type, static language for expressing optimization and Production Engineering.

It seems that the first choice of just about everyone is to use sketches drawn on napkins at diners, or, whiteboards in meeting rooms.

Can we make it such that those sketches contain enough "high level detail" to be refined into scripts for running machines?

Can we preserve provenance from the original airy-fairy sketches down to the actual scripts that are fed into machines?

Code is cheap, thinking is hard.

Appendix - See Also

See Also

References *https://guitarvydas.github.io/2024/01/06/References.html*
Blog *https://guitarvydas.github.io/*
Blog *https://publish.obsidian.md/programmingsimplicity*
Videos *https://www.youtube.com/@programmingsimplicity2980*
[see playlist “programming simplicity”]
Discord *https://discord.gg/Jjx62ypR (Everyone welcome to join)*
X *(Twitter) @paul_tarvydas*
More writing *(WIP): https://leanpub.com/u/paul-tarvydas*