**Relentless Pursuit of Software Engineering Excellence (#3)**

Every time I am creating a new software, flashes of new insights on how softwares can be better engineered come to me. Usually, I take time to think about these insights. Very often, the results of these actions are that I come out with new understandings and pearls of wisdom on how softwares can be better engineered.

For the past few years, there has been this particular insight that has been reoccurring to me almost every time I am engineering a new software. I have always created time to think about it, just to realize what the insight is trying to teach me, but I have always failed.

About 4 days ago, I decided to dedicate my whole time to discovering the knowledge behind the insight, and I have just succeeded.

What I have just come to understand is a fact about driving down the cost of software engineering (code reusability). I call this newly discovered fact "The Fact of Fundamentalism". I will like to share it with you.

[ELABORATION]

The Fact of Fundamentalism is a fact of life that goes thus, “The more fundamental a thing is, the more ingrediential (capable of being used in the creation of new things) it becomes.”. From a software perspective, the Fact of Fundamentalism will go along this line, “The more fundamental (singly responsible and abstract) a code is, the more reusable it becomes.”.

Mathematically, we can describe it like this:

Ingredientialness (I) ∝ Fundamentalness (F)

I = kF

[SUMMARY]

In summary, what this fact is saying is that, if you want to be great at creating softwares cheaply (creating highly-reusable codes), learn to make your codes singly responsible and abstract to the extreme.

HOST