Alex Groce (agroce@gmail.com), Northern Arizona University

Confucius'* *Analects* is the oldest guide to applied software engineering research and practice in existence. You don't believe me? Fine. We'll use random sampling to demonstrate. I have Ezra Pound's peculiar, but engaging, translation to hand, and can write a small program to generate five random samples easily, since the Analects has twenty books, and each book has approximately twenty "items." I've sorted the resulting samples for our convenience.

(1, 16): *"He said: Not worried that men do not know me, but that I do not understand men."*

Confucius is emphasizing the importance of rigorous, high-quality research (his own topic was human nature and proper order of life, which by analogy here is understanding of how to engineer software) over publicized, buzz-wordy, popular, research. It is better to be obscure but correct than famous and wrong. If more researchers heeded this sage advice, the replication crisis would go away.

(3, 9): *"He said: I can speak of the Hsia ceremonial, but you can't prove if by Chi (data); I can speak of the Yin ceremonies but Sung (data) won't prove it. The inscribed offerings are insufficient to argue from, were they adequate they could bear me out."*

Again, Confucius is talking about responsible use of data. Yes, he has an instinct and an intuitive understanding of the value of certain ritual practices (development methods), but he will not claim the data supports his beliefs beyond the degree to which this is so. I'll violate random sampling for a moment to say that this is essentially an application of the principle of (2, 1): "... Have no twisty thoughts."

(4, 14): *"He said: Not worried at being out of a job, but about being fit for one; not worried about being unknown but about doing something knowable."*

Ok, the Analects sort of harp on this matter of integrity above success (and are not, overall, averse to success), but it's a good guideline in both software engineering research and in actual software engineering.

(12, 1): *"1. Yen Yuan asked about full manhood. He said: Support oneself and return to the rites, that makes a man. [The "support oneself" is fairly literal. It cannot be limited to superficial idea of making a living, but certainly need not be taken ascetically. 'Determine the character' might render one side of the phrase.] If a man can be adequate to himself for one day and return to the rites, the empire would come home to its manhood. This business of manhood sprouts from oneself, how can it sprout from others? 2. Yen Yuan said: Wish I had the eye to see it, may I ask? If something is contrary to the rites, don't look at it; don't listen to it, don't discuss it, if it is contrary to the rites don't spend energy on it. Yen Yuan said: I am not clever but I would like to act on that advice."*

Integrity in the analects is not a purely individualistic concept, defined by self-consistency or the satisfaction of conscience. Rather, it is also defined by accordance with ritual practices, which may be something the reader (who is not always so clever as the reader thinks) does not fully understand, but applies in all the same. What is a ritual practice in software research? The current standards and best practices of the research community, based partly on absolute truths such as statistical arguments (e.g., little can be inferred from a too-small sample size, or a too-noisy measure), but also on common practices that could be otherwise, but are essential for communication with others, and harmony of the community. Writing a paper (even if it conveys good concepts and empirical or mathematical support for these concepts) that is extremely unlike other papers is not only likely to get your paper rejected, it is also, if done with sufficient irreverence, a threat to the comity and mutual understanding of the community, even if the rituals are not, themselves, necessarily the best ways. This goes beyond the parochial concerns of researchers: even if you have a better way to format code, for example, or a clearer set of calling conventions for an API, or a better workflow for using git and the bug tracking system, first you may not be right; but more importantly observing the "rites" of a project may be essential to your own acceptance by the community, and to the continued flourishing of the community. We drive on the left or right, and all must drive on the left or right, even if the superior way is not known.

Abandoning random sampling, we see there is advice about joining a startup:

(4, 16): *"The proper man understands equity, the small man, profits."*

And, critically, one essential way to have no twisty thoughts:

(13, 3): *"1. Tze-Lu said: The Lord of Wei is waiting for you to form a government, what are you going to do first? 2. He said: Settle the names (determine a precise terminology). 3. Tze-Lu said: How's this, you're divagating, why fix 'em? 4. He said: You bumpkin! Sprout! When a proper man don't know a thing, he shows some reserve. 5. If words (terminology) are not (is not) precise, they cannot be followed out, or completed in action according to specifications… 7. Therefore the proper man must have terms that can be spoken, and when uttered be carried into effect; the proper man's words must cohere to things, correspond to them (exactly) and no more fuss about it."*

Form a government, start a research project, start a requirements document, start designing an API or class hierarchy...

Follow this one rule, set the terms in order, and much good research, and good software, will follow.

[Q: Is this column a joke of some kind?

A: It is a joke and it is not a joke; if you read the Analects, and adapt the approach taken therein

to these domains, you will profit; it's a short book, what do you have to lose?]

* Did Confucius himself write down the Analects? No. But I think for the purposes of this column, we don't really care.