Vote Item: Is "coding error" a too-difficult concept for research?

Lutz Prechelt
prechelt@inf.fu-berlin.de
Freie Universität Berlin, Institut für Informatik
Berlin, Germany

ABSTRACT

Background: In programming, human error can lead to program defects, which can lead to program failure. Human error is an extremely important phenomenon in software development.
Information, Idea, Arguments: We have attempted research on programming error based on IDE-based programming process protocols, but gave up when we recognized we felt unable to pin down errors of ommission.
Vote: Is "coding error" a too-difficult concept for research?

KEYWORDS

human error, software development process

ACM Reference Format:
Lutz Prechelt. 2022. Vote Item: Is "coding error" a too-difficult concept for research?. In *pseudoCHASE 2022: Pseudo-submission to 15th International Conference on Cooperative and Human Aspects of Software Engineering, May 21-22, 2022, Pittsburgh, PA, USA*. ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/xxx.yyy

1 BACKGROUND

Human error is of interest wherever people perform work and several high-risk professions such as surgery and flying have successfully studied it and managed to greatly improve their security record. In programming, the type of human error that is of probably the most interest is error which leads to a faulty program logic (a defect), because such defects can lead to program failure, which can have many negative effects.

In software engineering, there has long been some research on defects, but relatively little on error. Yet from the point of view of root cause analysis, understanding error is valuable in order to learn to avoid the causes of defects rather than only removing the defects themselves by means of debugging, review, or automated program analysis.

2 INFORMATION, IDEA, ARGUMENTS

My group started research on error in 2004, gave it up again in 2007, and has not made a second attempt since.

2.1 Our research approach

Our research approach (which we tend to use in most of our research on human action in software development) was sensemaking: We wanted to collect enough observations of error episodes and their context that we would be able to conceptualize (using Grounded Theory Methodology) the relevant phenomena that led to error and the nature of those errors themselves. The goal (and again this holds for all our research) was to offer useful practical advice to practitioners.

We built an infrastructure for collecting a detailed protocol of programmer activities in the Eclipse IDE. We refined this infrastructure towards reporting the activity at a useful level of abstraction (*"move to second nested 'if' in method 'determine_case'"*) even when the raw protocol was at a too-low level of abstraction (*"PageUp, Up, Up, Up, Up"*).

We envisioned we might eventually be able to build a recording and analysis tool that would help a developer to understand their personal typical-error patterns.[1] By pointing out to it the precise location of a defect, the tool would search for those stretches of the activity record that pertained to creating this defective piece of code. Such stretches and their vincinity would then be treated as *candidate error episodes*.

To realize such a tool, two ingredients appeared needed:

(1) Find all typical-error meta-patterns: Understand what concepts (types of elements) might occur in an abstract typical-error pattern and what the grammar was for composing a concrete pattern from them.
(2) Building heuristic search logic that would be able to point out those parts in a candidate error episode that were common across many episodes and could therefore be considered to be instances of a typical-error pattern for this particular developer.

2.2 Theoretical basis

For number 1, we did not need to start from zero, because previous research had suggested, among other things, two fundamental discriminations for the type of an error at hand:

- **Mistake**: Working according to an unsuitable plan [2].
- **Slip**: Working according to a suitable plan, but executing that plan incorrectly [2]. And in doing so:
  - **Error of commission**: Doing something that is unsuitable [1].
  - **Error of ommission**: Failing to to something that is required [1].

The mistake/slip distinction is at the genotype level (*"where do errors come from?"*), whereas the commission/ommission distinction

[1] This would be a strictly local, private data collection, so we expected the obvious privacy concerns would, at least for many people, not get in the way.

pseudoCHASE 2022, May 21-22, 2022, Pittsburg, PA, USA

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/10.1145/xxx.yyy

is at the phenotype level (*"what do errors look like?"*). The latter should be easier to work with, because the observation data itself is also at a descriptive level. That seemed nice, because we had expected our work would only address slips (not mistakes), since in a mistake situation the developer would act confidently: the activity record would contain no clues that anything was not in order.[2]

2.3 The problem

During the research, we recognized three tall obstacles:

- In programming, the **mistake/slip distinction is vague**, because in the near-omnipresent absence of tactical-level specifications that is typical of most modern development work, it is often unclear what can and should be considered "the plan". This made it hard to determine the boundaries of our work.
- The **semantic gap is huge** from the strictly linear sequence of strictly syntactical events in the observed data to a meaningful pattern of behavior that a developer could easily learn from. We became more and more skeptical whether we would be able to produce anything that a real developer might want to use.
- At the heart of the analysis would be a set of rules for determining the moment (or possibly few moments) at which an error had happened. For **errors of omission**, this appeared extremely difficult. Most programmers do not work in a fashion that has a particular, canonically ordered sequence of steps. But if the steps to a succesful result can occur in many different orders, the missing step could have occured at many different points in time, so at which of those will we call its absence an error?

3 VOTE

Is the notion of "coding error" a concept that is so difficult to operationalize that effective research on it is not practical?

- Yes, because one cannot grasp errors of omission.
- Yes, because the semantic gap is too large.
- No, because there will be only few patterns when an omitted step would normally have happened. These can be found.
- No, because big data and crowdsourcing approaches will allow to overcome the semantic gap.
- No, for some other reason that I will explain in person.

Note on the nature of Vote Items

This is only an example Vote Item that serves to explain the article structure. In a real Vote Item …

- …the topic can be anything: a judgment of research difficulty like here or one of many other things such as a judgment of relevance of a question, of likeliness-to-succeed of some research approach, of best-way-to-proceed with some incomplete research, of interpretation of some finding, or what-have-you.
- …you need not use a yes/no question with different justifications as response options like here. Rather, you can use any type of closed question and any kind of response option structure instead, such as a five-point agree/disagree scale, a diverse list of candidate explanations, or whatever the audience can readily understand.
- …you should probably include important literature, starting with the most helpful in particular in the background section, until the two pages are full.
- …you should obey the instructions regarding double-blind reviewing.

[2] Today, one might consider a Big Data approach that might be able to provide the analysis apparatus with enough knowledge that it can spot some kinds of mistakes as well.

REFERENCES

[1] Erik Hollnagel. 1991. The phenotype of erroneous actions: Implications for HCI design. *Human-computer interaction and complex systems* (1991), 73–121.
[2] James Reason. 1990. *Human error.* Cambridge University Press.