
Abstract
XH: Experience and/or Heuristic
Studies reported here of several iterations of the OS/360 implementation, supported by source code control data, generate a number of findings concerning the degradation of source code, including three “laws” of code change. They indicate that keeping code from degrading is met by periodically investing in cleaning up the code’s structure.

Question - [Generalization / characterization]
<issues>
How does code degrade?

<requirement>
What can we do to keep code from degrading?

Results - [Report]
<list-of-findings>
The authors report the following rules of thumb, which they call “laws”:
• A system that is used undergoes continuing change until it is judged more cost effective to freeze and recreate it.
• The entropy of a system (its un-structuredness) increases with time, unless specific work is executed to maintain or reduce it.
• Growth trend measures of global system attributes may appear to be stochastic locally in time and space, but, statistically, they are cyclically self-regulating, with well-defined long-range trends.

<design-heuristic>
Obviously, based on the laws above, it makes sense to plan for the need to periodically invest in cleaning up the code’s structure.

Validation – [Experience]
_Application_
The authors examined data related to code changes in the development of several versions of OS/360. They plotted, for example, size and numbers of modules as a function of time. Their support for the first and second laws is relatively strong, since the numerous little plots clearly trends consistent with the authors’ claims. The third law is less strongly supported, mainly because the authors appear to take the liberty of boxcar smoothing the data, thereby ensuring that their third law would be supported; nonetheless, the claimed long-term trends are readily apparent in the plots, yielding some confirmation for their claims.

<supporting technology>
Note that not all the data came from the same type of code control systems: the authors apparently had to aggregate data that was manually tracked by programmers, as well as data from traditional source code / configuration management software that became available by the time that later OS/360 versions underwent development.