Myths of Correctness

Jade Parker

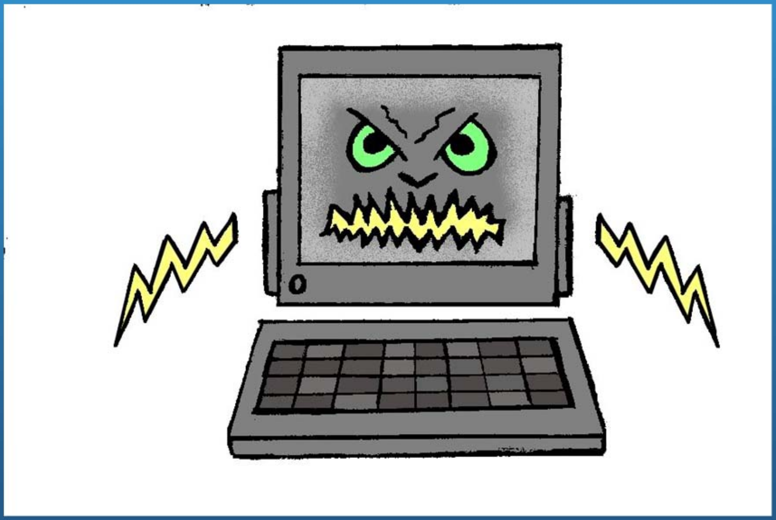
Summary

The Software Crisis came into existence because it is impossible to write bug-free programs, as testing is not thorough enough to catch everything. The main problem is working with and preventing complexities that can lead to error.

Is the Computer to Blame?

- Often blamed for problems when it may just be an “innocent bystander”
- Either hardware stopped working or a person messed up
  - Hardware failure infrequent
  - Crashes usually caused by programmer or user
- Unfriendly user interface can be just as much to blame as software

Software Crisis



- Coined 10-15 years ago
- Led to software engineering field
- Today, software makes up 80% of cost
- Inability to write error-free software
- Examples:
  - Election Night fiasco
  - Space Shuttle Orbiter

The Job is Never Done

- Programs fail not from being worn out, but from encountering problematic circumstances after running on erroneous code
- Maintenance = continued development
- Most software products are lemons

Bugs Your Exterminator Can't Find

- Inadequacy vs. incompetence
- Not possible to find ALL bugs by testing
- Testing corroborative, not definitive

Flexibility: It's a CURSE!!!

- Behavior can be changed radically quite easily
- More adjustments can make software:
  - More complex
  - Harder to read & Understand
  - More likely to contain errors
  - More likely to require future modification

This Relationship Isn’t Working Out…

- Interdependent program interface problems
- Conflicting assumptions
- Subtle dependencies nearly impossible to detect by studying program
- Problems grow rapidly with size and complexity increases

The Main Job:

Manage unavoidable complexity and avoid unmanageable complexity