CSE 303:
Concepts and Tools for Software Development

Hal Perkins

Autumn 2008

Lecture S4— Societal Implications: Software Quality, Licensing, Defect Disclosure, ...

The Big Questions

Is software any good? Could society make it better?

Who should be allowed to write software? Is "real" software different?

What responsiblity do software writers, users, and sellers have?

Should you be able to restrict software's use? How?

When a critical defect is discovered, who bears responsibility for revealing or fixing it?

Quality Issues

"Kinds" of software??

- Mission-critical: nuclear-missile, hospital equipment, air-traffic control, ...
- Business-critical: Online retailer, stock market, your database, ...
- Computer-critical: operating system, browser, ...
- Get what you pay for: freeware, CSE homework, ...

How do we know what is what?

Bug Issues

- How often is it triggered?
- Can an adversary make it trigger?
- What damage does it do?
- What is a complex piece of software *supposed* to do?

Contrast with cars, buildings, etc.?

Software Release Cycle

Standard industry practice for large projects

- Prioritize bugs (P1 (blockers), P2, P3, . . .)
- Freeze features and non-essential changes as release approaches
- Release when code is "ok" (no more P1 bugs, or no more than n P1 or P2 bugs, or . . .); release might or might not be tied to the calendar

Used by many open-source projects as well as commercial and in-house.

When is a software release "done"? Is it even meaningful to talk about whether software is "finished" or "ready"?

- Many "agile" projects use frequent, incremental releases. Better?

Who is to blame?

- $\boldsymbol{A}$ writes some C code that has an array-bounds error in it that can be triggered if a function is called with certain arguments.
- $\boldsymbol{B}$ uses $\boldsymbol{A}$'s code to develop an application such as a web browser.
- $\boldsymbol{C}$ uses the web browser $\boldsymbol{B}$ develops.
- $\boldsymbol{D}$ sets up a website that $\boldsymbol{C}$ visits. The contents of the website trigger the array-bounds error.
- As a result of the error, $\boldsymbol{C}$'s computer connects to $\boldsymbol{E}$'s computer and deletes all the files there.
- $\boldsymbol{F}$ knew about the error but didn't tell anybody, in fact had nothing to do with writing the code.

Programmers

Would software be better if "public" code required licensed programmers?

Is a "software engineer" a real engineer?

Who would do the licensing?

What would you test?

Who would you blame?

Would you still allow "as is" code?

Would anyone use software that cost more?

Software Licenses

What can a software provider require a user to do/not-do/allow?

Can a software provider declaim liability in a shrink-wrap license? Does the user have any recourse if something does go wrong?

What about software-library writers?

Is open-source software more secure? Less secure? A lost-in-the-noise feature?

Business Concerns

If you're a business, how high-quality do you want your software?

Worth delaying the product?

Worth slowing down the product?

Worth having fewer features?

Worth charging more?

How do you feel as a customer? Can you determine quality?

When Something Goes Wrong

If a security-flaw is discovered:

- Should we have laws forbidding publicity?
- Should we have laws mandating publicity?
- Should we require patching? Penalties for violation? What about old/ancient software (Windows 95/98, MS-DOS, Classic Mac OS)?
- Is actively finding flaws good/bad/depends-what-you-do-with-it?
- Viruses that fix viruses?

Relevant issues: obscurity vs. security, malice vs. negligence, ...

The Plan

Choose 1 of 5 groups (bug pragmatics, people-licensing, software-licensing, business-customer perspective, revealing/fixing security bugs).

Choose 1 or 2 *theses*.

Choose 2-4 arguments for *each side*.

Report on the group's conclusion from weighing the arguments.

Participate!