Student vs Software Engineer

As a Student
- New classes each semester
- Work alone
- Your work must be your own
- Plagiarism is forbidden, quotes and citations are good
- Some collaboration

As a Software Engineer
- Projects last for years
- Work in teams, in projects, in departments, in . . .
- Some work your own, often with or by someone else
- Reuse is good
  - Saves time
  - Usually tested and debugged
  - Citation usually not expected but giving credit is good
  - So what are the limits? When does it become unethical/illegal?
- Virtually all collaborative
Professional Codes of Ethics

є A context and discipline specific set of concrete guidelines about the use of specialized skills for the benefit of both individuals, companies and society

є Limitations:
- Difficult to enforce
- Often a minimal standard
- Multiplicity of standards and codes, eg
  - One for the discipline, university, funding agencies, etc

є Basic issues (ACM/IEEE SE Code of Ethics)
- Act in the public interest
- Act in the best interests of client and employer
- Ensure products meet the highest professional standards
- Maintain integrity and independence in judgment
- Subscribe to and promote ethical management
- Advance integrity and reputation of the profession
- Be fair to and supportive of your colleagues
- Participate in lifelong learning, and promote ethical behavior, in practice
General Issues

Resources
- Company resources
- Conflicts of interest

Intellectual Property
- Privacy
- Ownership
- Patents
- Licenses
- Plagiarism
- Reverse Engineering

Risk
- Reliability
- Safety
- Security
Company Resources

Typical policy:
- Only use company resources for company work
- UT has such a policy for example

Examples
- Time, phone, xerox, printers, internet, etc

Purist response
- Only use company resources for company business

Pragmatist response
- Some personal use ok as long as do not abuse it
- Often justified by:
  - I do company work at home using my own resources
  - Hence, comes out about even in the end

Abuse response
- Hey, I don’t get paid enough
  - so just augmenting my salary to where it ought to be, etc
- And as long as I don’t get caught . . .
Conflicts of Interest

Example: Reviewing papers

- **Strict guidelines**
  - Always in conflict MS or PHD students
  - In conflict with members of same department
  - In conflict for 5 years with a co-author

- **Downsides**
  - Those in conflict may be the best able to review
  - Leaves those with negative conflicts (dislike work, person etc)

- **Gray area**
  - Good friends
  - Larger organizational structures of company – eg,
    - Labs with departments, colleges/schools, divisions, etc

Examples in workplace

- What good is nepotism if it cant help your relatives 😊
- Merit reviews
- Judgments about project viability
- Hiring, especially when slots are scarce
IP - Privacy

- Fundamental: Do employees have a right to electronic privacy?

- Public versus private availability (e.g., computer files)
  - Aggressive: whatever I can get to, even without permission
    - Typical among many students and hackers
  - Conservative: only what is explicitly public is allowed
    - Metaphor:
      - What is on the bookshelf, on the desk is accessible
      - What is behind doors, in drawers in the desk is not, even if not locked

- Email
  - Company resources/assets, hence company rights to look at employees email
  - How private is your email anyway from snoopers, ISP providers, company email systems, etc?
  - Email privacy protection: PGP encryption

- Project state (your part of a project)
  - Anytime access vs explicit reporting
  - What are the pros and cons?
IP - Ownership

- Your company will own all your work
  - Typically part of your agreement to work for the company

- Problem areas
  - IP created prior to working for a company
    - Fairly safe – but companies tend to think all your time is theirs
  - IP unrelated to your company’s domain
    - Tricky –
      - Clear it with the company’s management and legal team
      - Get everything in writing to protect yourself later
    - Example –
      - Y obtained a patent while working for company X but totally unrelated to the X’s products
      - Y told his management and checked with the company lawyers, but did not get it in writing
      - Now part of a patent suit – problems establishing who owns the patent
**IP - Patents**

- Patent: confers the owner the sole right to exclude others from making, using or selling the patented invention for a specific number of years.
- Patent system there to “promote the advance of science” by granting inventors exclusive rights for a limited time.
- Often used (eg, IBM) to reach mutually beneficial partnerships.
- Software patents:
  - Debated topic - some want to get rid of software patents
  - Have to provide enough information so that one of ordinary skill in the art would be able to build the invention.
- Basic issues:
  - Prior Art – must go beyond what already exists
  - Obviousness – to one of ordinary skill in the art
    - Eg, automate an existing manual process
    - Combining two existing patented ideas
    - Hard problem: was it obvious before; often (always?) obvious after.
- Problems:
  - Unimplemented patents - no attempt to create a viable product
  - Patent trolls - buy patents for the sole purpose of suing
  - Broadening the patent claims to include more than originally allowed.
- Ethical/Legal problem: infringement.
IP - Licenses

- **Landscape:** open source, free SW, proprietary SW
- Some argue that its an ethical issue that software should be free, not owned by anyone (eg, Gnu*)
- **Basic problem:** use of SW in commercial systems
  - The use of proprietary SW in building proprietary SW
  - The use of open source SW in proprietary SW
- Is this issue akin to quotations and citations?
- **Shrink-wrap Licenses/End-User License Agreements**
  - Often come with digital rights management mechanisms
  - Problems: multiple usage, copying, piracy, etc
- **Ethical issues**
  - Vendor has legal rights to his IP
  - Average consumer not a legal expert - usually don't see the license until after purchase
  - Company licenses, however, usually well understood
IP - Plagiarism

♫ Plagiarism vs Reuse - the one bad, the other good
♫ Textbook algorithms and data structures
♀ Eg, Knuth’s series, standard algorithms and data structures
♀ Useful source for reuse
♀ Good manners to provide citation
♫ Libraries and frameworks
♀ Often need licenses or purchase agreements
♀ Use them typically, not copy them
♫ Suppose you bring software source from another company?
♀ Your own - is that plagiarism?
♀ Someone else’s software - plagiarism?
♫ Downloading from the web?
IP – Reverse Engineering

Reverse Engineering (RE):
- discovering the design of a SW system by a variety of means on the basis of its function and operation – usually with the intent of recreating the product

Independent design vs Using someone else’s design
- Fundamental questions:
  - how many different ways are there to design a system?
  - Does the process matter how you design the system?
  - Are there good uses of reverse engineering?

Example:
- Product licensed to company X – created via hard work by Y
  - RE prohibited in the license
- Licensed to be used solely in a production context
- Using the licensed system, created their own via RE
  - Used licensed system as the perfect testing oracle
- Result: theft of IP and the effort to produce it
- How could this have been done properly?
Risk - Reliability

Basic Fact: no fault-free software system exists

Basic questions:
- how do you make software as reliable as possible
- Under normal circumstances; under abnormal circumstances

Ethical issues:
- Negligence in design
  - Poor processes – often lack of experience
  - Inadequate software engineers – often cost related
  - Deliberate – often due to management decisions
- Deliberate misrepresentation

Example
- Company X represents software system ready for primetime
- Company Y has throughput demands far above current usage
  - Y held to throughput deadline constraints – fines if not met
- X’s quality assurance team internally forewarned failure months ahead
  - Inadequate load testing, no beta testing
- X deliberately with-held this information from Y
- Complete disaster when Y went live with X’s system
- Why did X deliberately proceed without warning Y?
- What should X have done?
Risk - Safety

Growing class of safety critical systems
- Lives depend on the proper functioning of the systems
  - Eg, medical devices, computerized automotive systems, etc
- Significant accident risks if not done extremely well
  - Software driven airplanes (eg, Boeing 757)
  - Air traffic control system

Three kinds of issues:
- The software NOT doing something
- The software DOING something it should NOT do
- Instability in the software system or environment

Cause: not knowing all the normal and/or abnormal conditions

Solutions:
- Increased depth of domain specific knowledge
- Increased depth of software engineering fundamentals
  - Requirements, architecture, design and implementation
- Resources, process and staff beyond ordinary
- Appropriate reasoning, analysis and testing tools
Risk - Security

- Physical security
  - relatively straightforward and well understood

- Information Security
  - Well-understood classification schemes
    - Unclassified vs classified
    - Levels of classification and need to know
  - Vulnerable to software system security issues

- Software system security - Primary causes:
  - Interconnectivity - network security
    - Various network threats
    - Wireless is open broadcasting
  - Software faults that can be exploited
    - Enable spybots, viruses, worms etc
    - Enable unauthorized access
  - User irresponsibility
    - Allowing ready access: not locking the system when unused, etc
    - Poor security practices: easily breakable passwords, etc
    - Unauthorized sharing
Conclusions

- A lot of pro-active large impact unethical behavior
  - IP theft in various forms

- A lot of small impact unethical behavior
  - Misuse of resources, licenses, etc

- A lot of passive/unintended behaviors that enable the unethical behavior of others
  - Poor development practices
    - Methods, techniques, processes, technologies, use of tools, etc
  - Poor management decisions
  - Poor personal decisions

- A significant lack of professionalism
  - Lack of proper training and education
  - Lack of integrity and caring
  - Lack of commitment to the best we can do