Queen Mary
University of London

ECS505U
SOFTWARE ENGINEERING

MUSTAFA BOZKURT

LECTURER IN SOFTWARE ENGINEERING

Queen Mary
University of London

WEEK 12

LEGAL, SOCIAL AND ETHICAL ISSUES IN SOFTWARE ENGINEERING

REVISION MATERIAL

- Computing Curricula: Social, Ethical, and Professional Issues by DODIG-CRNKOVIC
- What is software ethics? some of the material in lecture slides are from this chapter
- Software Engineering Code of Ethics and Professional Practice (ACM Guidelines)
- Code of Conduct (BCS guidlelines)
- Code of Practice (BCS guidlelines)
- ACM Software Engineering Code of Ethics and Professional Practice
- ACM ethics list
- Computer Malpractice by Kaner
- Software Product Liability by Lvey and Bell

Revision material for this week's lecture

ENGINEERING

Engineering is a multi-faceted, real-world endeavor with ethical dimensions:

Engineering is an inherently risky activity. In order to underscore this fact and help in exploring its *ethical implications*, we suggest that engineering should be viewed as an experimental process. It is not, of course, an experiment conducted solely in a laboratory under controlled conditions. Rather, *it is an experiment on a social scale involving human subjects*.

[Gotterbarn]

WHAT IS ETHICS?

Definition:

> Moral principles that govern a person's behaviour or the conducting of an activity.
>
> Oxford Dictionary

WHAT IS ETHICS

"*There are few things wholly evil or wholly good. Almost everything...is an inseparable compound of the two, so that our best judgment of the preponderance between them is continually demanded*."

Abraham Lincoln

(President, not the vampire hunter)

WHAT IS ETHICS

Ethics in Software Engineering deals with

- What is good?
  - What makes actions/practices good?
- What is right?
  - What makes actions/practices right?

and also bad and wrong using the definition of good and right.

WHY STUDY ETHICS?

What is the point in studying ethics?

The goal is to *foster moral autonomy, i.e. the skill and habit of thinking rationally about ethical issues, as well as to improve the ability to think critically about moral matters*.

VIEWS ON ETHICS

“Ethical problems aggravated, transformed or created by computer technology.”

Walter Manner

VIEWS ON ETHICS

"Computers pose new versions of standard moral problems and moral dilemmas, exacerbating the old problems, and forcing us to apply ordinary moral norms in uncharted realms."

Deborah Johnson

VIEWS ON ETHICS

"A typical problem in Computer Ethics arises because there is a policy vacuum about how computer technology should be used. Computers provide us with new capabilities and these in turn give us new choices for action. ***Often, either no policies for conduct in these situations exist or existing policies seem inadequate****. A central task of Computer Ethics is to determine what we should do in such cases, that is, formulate policies to guide our actions. . . . One difficulty is that along with a policy vacuum there is often a conceptual vacuum. Although a problem in Computer Ethics may seem clear initially, a little reflection reveals a conceptual muddle. What is needed in such cases is an analysis that provides a coherent conceptual framework within which to formulate a policy for action."*

James Moor

(not Gordon E. Moore of Moore's Law)

VIEWS ON ETHICS

"There is little attention paid to the domain of professional ethics – the values that guide the day-to-day activities of computing professionals in their role as professionals. By computing professional I mean anyone involved in the design and development of computer artifacts. . . . ***The ethical decisions made during the development of these artifacts have a direct relationship to many of the issues discussed under the broader concept of computer ethics****.*

Donald Gotterbarn

VIEWS ON ETHICS

"Computers bring new ethical problems" Maner

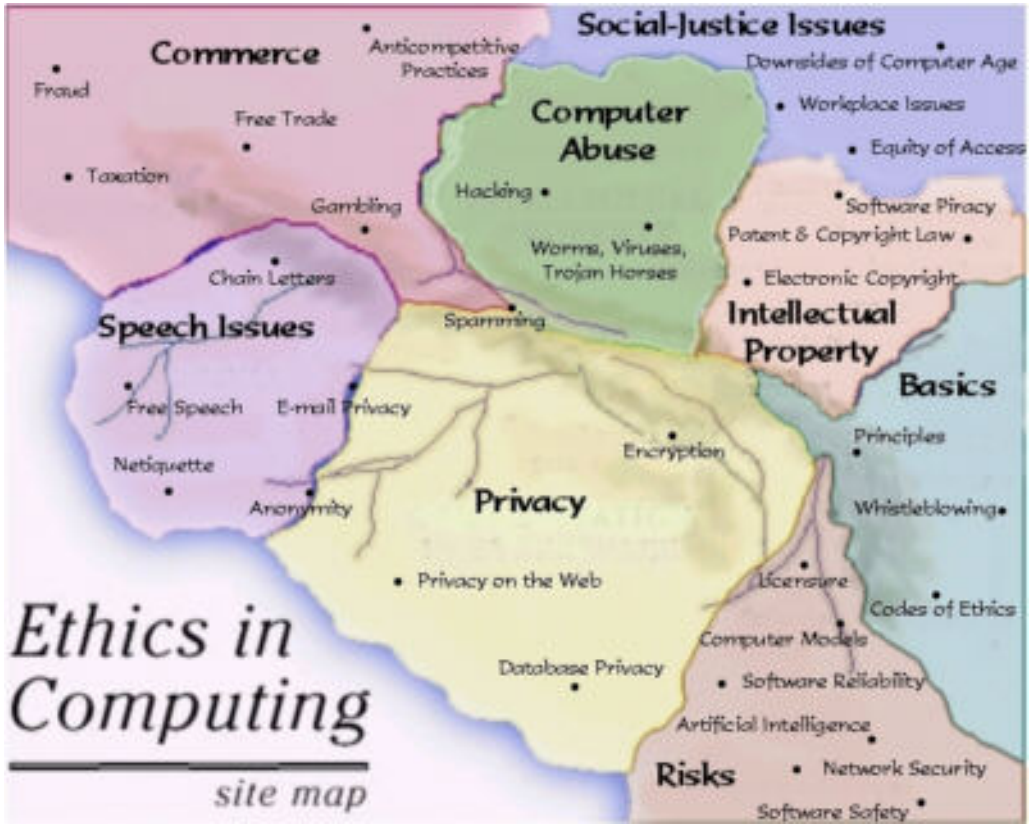
"Computers bring a new dimensions to old problems" Johnson

"No or insufficient policies regarding computer ethics" Moor

"Applied ethics for computer professionals" Gotterbarn

Discuss!

COMPUTING ETHICS



https://ethics.csc.ncsu.edu/

TOPICS IN COMPUTING ETHICS

Queen Mary University of London

Computing Curricula 2001, ACM/IEEE

- Social context of Computing
- Methods and tools of analysis of ethical argument
- Professional and ethical responsibilities
- Risks and liabilities of safety-critical systems
- Intellectual property
- Privacy and civil liberties
- Social implications of the Internet
- Computer crime
- Philosophical foundations of Ethics

SOCIAL CONTEXT OF COMPUTING

Queen Mary University of London

- *Refers to anything that deals with the impact of computing technology on people and groups of people.*
- *Technologies such as blogs, email, instant messaging, social network services, wikis, social bookmarking and etc…*
- *Facebook, twitter and Google+?*
  - *Picture tagging without permission*
  - *Data kept forever*
  - *Cyber crimes like bullying*
  - *Is creating a fake account crime?*

PRIVACY AND CIVIL LIBERTIES

Queen Mary University of London

- Privacy protection
  - Privacy is a fundamental human right recognized in all major international agreements regarding human rights such as Article 12 of the Universal Declaration of Human Rights (United Nations, 1948).
- Surveillance
  - Many civil rights and privacy groups, have expressed concern that by allowing continual increases in government surveillance of citizens we will end up in a mass surveillance society, with extremely limited, or non-existent political and/or personal freedoms. [American Civil Liberties Union. January 15, 2003]

COMPUTER CRIME

Computer crime or cybercrime

- Hacking
- Cracking
- Malware (Viruses, worms and etc ..)
- Phishing
- Fraud and identity theft
- Spam
- Copyright infringement
- Child pornography

and recently ***cyber bullying.***

COMPUTER CRIME

Kent police website:

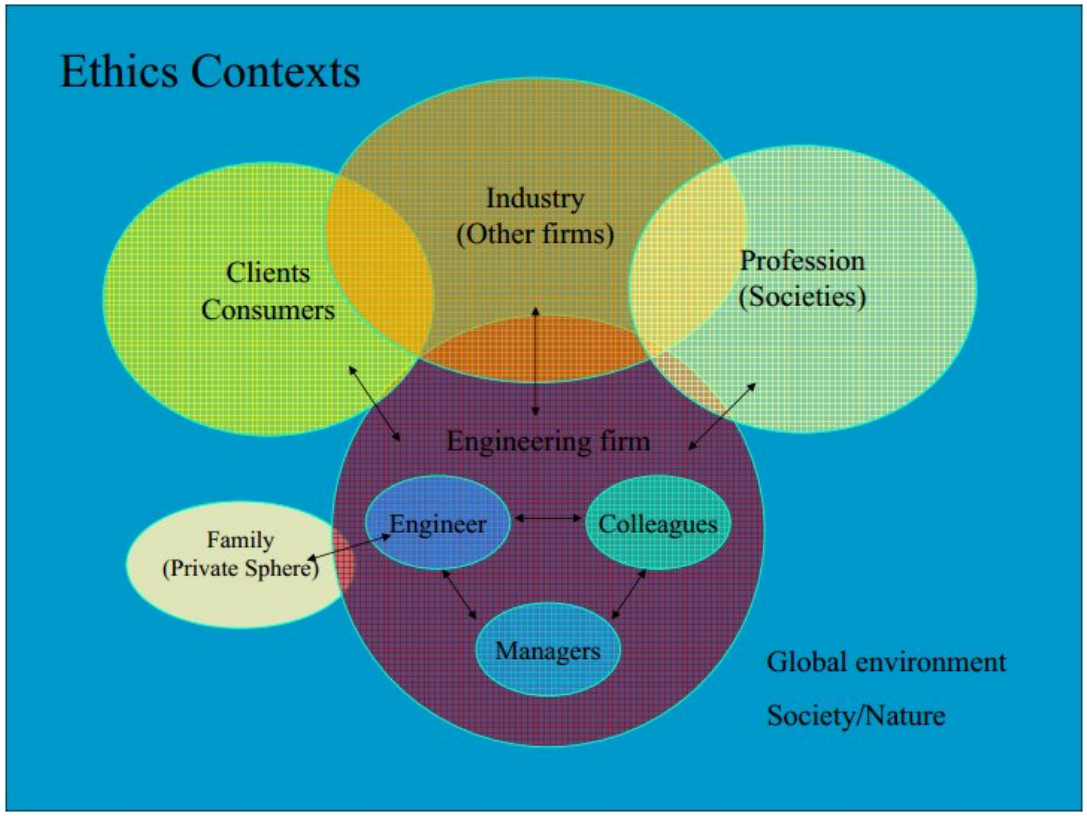
There is not a specific law which makes cyberbullying illegal but it can be considered a criminal offence under several different acts including Protection from Harassment Act (1997), Malicious Communications Act (1988), Communications Act (2003) Obscene Publications Act (1959) and Computer Misuse Act (1990).

INTELLECTUAL PROPERTY

- Patent infringement
  - Apple vs Google or Apple vs Samsung
- Copyright infringement
  - Torrents and illegal movie websites
- Trademark infringement
  - Google Adwords
- Trade secrets
  - Major company employees

ENGINEERING ETHICS: PROFESSIONALISM

Queen Mary University of London



http://www.idt.mdh.se/kurser/cd5590/Archives/06_11/lectures/Professional-issues.pdf

WHAT IS A PROFESSION?

A profession is a calling that requires

- Specialized knowledge and trained judgment
- Long and intensive intellectual preparation

IT PROFESSIONALS

Partial list of specialists in an IT project

- programmers
- systems analysts
- software engineers
- database administrators
- network administrators
- testers

WHY LIABILITY IMPORTANT

"As society increasingly relies on software to perform critical functions in everything from manufacturing to life support systems, ***the risk that an error in a software program will lead to economic loss, property damage, or personal injury increases***. Prudent software developers will be cognizant of these risks and will take steps to minimize their exposure to this type of liability. Determining how far a software developer should go in this effort requires balancing the degree of exposure to software product liability against the adverse impact, if any, of the steps required to limit this exposure on the software vendor's ability to market and sell its software."

Levy and Bell

LEGAL POSSIBILITIES

Legal liabilities based on the following

- Misrepresentation
- Negligence
- Professional malpractice

IT PROFESSIONALS

Legal perspective

- IT workers are not recognized as professionals, because they are not licensed.
- Consequently, IT workers are not liable for malpractice

HISTORY OF COMPUTER MALPRACTICE

Queen Mary University of London

The story is all too common. A governmental or corporate entity hires a major software developer or consulting company to analyze requirements and then design and construct a software-based “system” to support some major activity. The system might support a major corporate function (e.g., pension management) or some governmental function (e.g., health care administration or homeland security).

Work begins with the best of intentions on both sides, but by the time the system is delivered, things have gone bad. The system is late, fails to deliver desired features and functions, is error-prone, and does not meet with customer approval. **Litigation ensues**.

In most cases, the customer claims that the developer has been negligent (in the manner in which it has applied software practices) and is therefore not entitled to payment. The developer often claims that the customer has repeatedly changed its requirements and has subverted the development partnership in other ways. In every case, the quality of the delivered system comes into question.

HISTORY OF COMPUTER MALPRACTICE

Queen Mary University of London

The case of Chatlos Systems v. National Cash Register Corp. (1979) is the first important computer malpractice case.

An NCR salesman did a detailed analysis of Chatlos' business operations and computer needs, and advised Chatlos to buy NCR equipment. Relying on NCR's advice, Chatlos bought a system that never provided several promised functions.

NCR was held liable for breach of contract.

HISTORY OF COMPUTER MALPRACTICE

Queen Mary University of London

Wang Laboratories sold a computer and a service contract to Orthopedic & Sports Injury Clinic.

While attempting to fix the computer, Wang's employee used, and corrupted, the Clinic's last backup disk, thereby losing five years of the clinic's medical and accounting data. The contract limited the amount of damages that Orthopedic could collect from Wang, but Louisiana law allows the plaintiff to recover all damages if the defendant committed gross negligence.

The Court ruled that Orthopedic hadn't proved that this use of the backup disk was gross negligence. However, ***it did allow the lawsuit to go forward as a suit for ordinary negligence***.

BCS GOOD PRACTICE SOFTWARE DEVELOPMENT

Queen Mary University of London

What is good practice in software engineering?

BCS GOOD PRACTICE SOFTWARE DEVELOPMENT

Queen Mary University of London

Designing New Systems:

- Recommend the organisation to adopt new technology, but only when it is sufficiently well proven for the organisation, offers a cost-effective solution and is compatible with the organisation's IT strategy.
- Strive to understand the corporate needs of the organisation and aim to design systems that benefit those needs.
- Consider the needs for scalability, connectivity, capacity, performance, resilience, recovery, access, security and create cost-effective solutions that meet those needs.
- Produce design specifications that clearly state the objectives, scope, features, facilities, reliability, resilience, constraints, environment, system functions, information flows and traffic volumes as well as identifying requirements not met and scope for improvement.
- Resist the pressure to build in-house when there may be more cost effective solutions available externally and vice versa.

Queen Mary University of London

SOFTWARE DEVELOPMENT

Designing Software:

- Strive to achieve well-engineered products that demonstrate fitness for purpose, reliability, efficiency, security, safety, maintainability and cost effectiveness
- Take responsibility for ensuring the design balances requirements for functionality, service quality and systems management.
- Encourage re-usability; consider the broader applications of your designs and, likewise, before designing from new seek out any existing designs that could be re-used.
- Ensure your designs facilitate later stages in the development lifecycle, particularly testing.
- Check the products of your designs can be used by both experienced and inexperienced users; in particular check that they can be used for training purposes (e.g., on-line help, training databases).

SOFTWARE DEVELOPMENT

Programming:

- Strive to produce well-structured code that facilitates testing and maintenance.
- Follow programming guidelines appropriate to the language and encourage your colleagues to do likewise.
- Produce code that other programmers will find easy to maintain; use meaningful naming conventions and avoid overly complex programming techniques, where these are not strictly necessary.
- Make yourself aware of the limitations of the platform (operating system and hardware) and avoid programming techniques that will make inefficient use of the platform.
- Wherever possible, avoid platform-specific techniques that will limit the opportunities for subsequent upgrades.

SOFTWARE DEVELOPMENT

Testing:

- Plan the tests to cover as many paths through the software as possible, within the constraints of time and effort.
- Assure yourself that the coverage of the testing is sufficient; take appropriate actions to resolve any shortcomings in the tests planned by yourself or by your colleagues.
- Promote the use of test tools that will maximise the effectiveness of the testing.
- Create a test environment whereby tests can be re-run and the results are predictable.
- Do not rely solely on the direct outputs of tests, but check values are as expected in internals tables, databases and error logs.
- Recommend improvements that will improve the effectiveness of the software under test.
- Maintain a detailed testing log.
- Accurately document all anomalies arising during the testing and make sure they are investigated; but remain impartial, trying not to provoke whoever may be at fault.
- Design regression testing to identify any undesirable side effects of a software change.
- Resist any pressure to curtail testing; make anyone overruling or neglecting your professional advice formally aware of the consequent risks.

SOFTWARE DEVELOPMENT

Integrating:

- Check that all software components meet the defined criteria for test.
- Devise integration tests that build upon component tests already performed and demonstrate that the components interface correctly with each other.
- Check the documentation of the components and assure yourself that they are compatible with each other and with the target platform.
- Maintain a configuration management system that records the status of each component.
- Devise workarounds that will enable the software to be used correctly despite known shortcomings.

SOFTWARE DEVELOPMENT

Writing technical document:

- Document all work to a level of detail that others could take over your work if need be.
- Set a high standard of documentation and, by setting a good example, encourage your colleagues to do likewise.
- Follow the appropriate documentation standards: the organisation's house style and specific standards for the type of document.
- Identify omissions or shortcomings in the organisation's documentation standards and actively seek out ways to improve them.
- Strive to keep documentation up to date.
- Ensure documentation is sufficient to enable effective ongoing maintenance.

SOFTWARE DEVELOPMENT

Writing user documentation:

- Investigate the subject of the documentation, through hands-on use, talking to experts and reading related documents; do not assume it works in a particular way.
- Assure yourself that you understand the real purpose of the document and structure the document accordingly.
- Strive to understand the potential readership, their expectations and abilities; be aware that some readers may have difficulties with reading, language or comprehension.
- Write the document in a straightforward style appropriate to the readership; avoid jargon.
- Make intelligent use of diagrams that complement the text and aid overall understanding.
- Check with experts that the document is correct and with potential readers that it meets their expectations.

ACM/IEEE CODE OF ETHICS

An ACM/IEEE-CS Joint Task Force has produced a Software Engineering Code of Ethics and Professional Practices (Version 5.1). The code states:

PUBLIC: Software engineers shall act consistently with the public interest.

CLIENT AND EMPLOYER: Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.

PRODUCT: Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

JUDGMENT: Software engineers shall maintain integrity and independence in their professional judgment.

MANAGEMENT: Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

PROFESSION: Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

COLLEAGUES: Software engineers shall be fair to and supportive of their colleagues.

SELF: Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

ETHICAL DECISION MAKING

- Recognize a moral issue
- Get the facts
- Evaluate the alternative actions from various moral perspectives
- Make a decision
- Act
- Reflect on the results of the decision afterwards.

Queen Mary
University of London

DISCUSSION

Ethics of using

a public API,

borrowed code,

or hacked/stolen code

in your coursework/project

Queen Mary University of London

DISCUSSION

Ethics of collecting data

- NSA and GCHQ
- Google, Facebook and Apple
- QMUL

SUMMARY

- Ethics are crucial.
- You should learn the ethics regarding your job.
- Act according to the ethics set by professional organisations!
- Question actions whether they are ethical.