

Computing and Software Department, McMaster University

# The Software Engineering Profession (adapted from Dr. Farmer's notes)

Dr. Spencer Smith

January 6, 2009

# Software Engineering Profession

- ▶ Administrative details
- ▶ What is software engineering?
- ▶ The PEO
- ▶ Software engineering in system design
- ▶ Therac-25
- ▶ The great gulf
- ▶ Challenges and opportunities for engineering
- ▶ Attributes of a good software engineer
- ▶ Software development process

# Administrative Details

- ▶ Assignment 1 deadlines
  - Files due by midnight January 19
  - E-mail partner files by January 20
  - Lab report due by the beginning of class on January 26
- ▶ Assignment 1 is posted to <http://www.cas.mcmaster.ca/~smiths/SFWRENG2AA4/Assig1.pdf>

# What is Software Engineering?

- ▶ An area of engineering that deals with the development of software systems that
  - Are large or complex
  - Exist in multiple versions
  - Exist for large period of time
  - Are continuously being modified
  - Are built by teams
- ▶ Software engineering is “application of a systematic, disciplined, quantifiable approach to the development, operation and maintenance of software” (IEEE 1990)
- ▶ D. Parnas (1978) defines software engineering as “multi-person construction of multi-version software”
- ▶ Like other areas of engineering, software engineering relies heavily on mathematical techniques, especially logic and discrete mathematics

# The PEO

- ▶ Degree from an accredited program
- ▶ Experience requirement
- ▶ Law and ethics exam
- ▶ Still debating what constitutes software engineering

# Software Engineering in System Design

- ▶ A physical system is often controlled by a software system called an embedded system
- ▶ As a result, software engineering is often a crucial part of system design
- ▶ Examples of embedded systems
  - Cell phones
  - Nuclear power plants
  - Automobiles
  - Aircraft
  - Pacemakers
  - mp3 players
  - Programmable household devices
- ▶ Embedded systems are rapidly appearing everywhere
- ▶ The developers of software for an embedded system needs to understand both the software and the physical device.

# Therac-25

- ▶ The Therac-25 was a radiation therapy machine for treating cancer
  - Produced by the Atomic Energy of Canada Limited (AECL)
  - Controlled by software
- ▶ How it worked
  - Provided both electron beam and X-ray treatment
  - The machine produced low to high energy electron beams
  - X-rays were produced by rotating a target into the path of a high energy electron beam
- ▶ Used in several clinics across North America

# Therac-25 Continued

- ▶ In six separate incidents in the 1980s, Therac-25 machines delivered overdoses of radiation causing severe physical damage or even death to the patients being treated
  - The second incident, which took place in Hamilton, resulted in administration of 13 000 – 17 000 rads of radiation (200 rads is regular treatment and 1000 rads can be fatal)
  - Three patients ultimately died from radiation poisoning
- ▶ What went wrong
  - Software failed to detect that the target was not in place
  - Software failed to detect that the patient was receiving radiation
  - Software failed to prevent the patient from receiving an overdose of radiation



# Therac-25 Continued

## Causes of failure

- ▶ Inadequate software design
- ▶ Inadequate software development process
  - Coding and testing done by only one person
  - No independent review of the computer code
  - Inadequate documentation of error codes
  - Poor testing procedures
  - Poor user interface design
- ▶ Software was ignored during reliability modelling
- ▶ No hardware interlocks to prevent the delivery of high-energy electron beams when the target was not in place

# The Great Gulf

- ▶ Engineers do not sufficiently understand or care about software
  - Many of the basic principles of software design and development are largely unknown to engineers
  - Engineers often do not appreciate the challenges and dangers inherent in software for embedded systems
- ▶ Software developers lack engineering training and professionalism
  - There is an entrenched culture of producing software without any guarantee whatsoever
  - There is no system for certifying either software or software developers
  - Most software developers lack the engineering background needed to produce software for embedded systems
- ▶ There is a gulf between software engineering and scientific computing as well

# Challenges and Opportunities for Engineering

## ► Challenges

- Engineers need to design systems that have safe, correct, high-quality software
- Software engineers need to produce software that they can guarantee
- No silver bullets
- Still an immature field

## ► Opportunities

- Software tools can greatly enhance the capabilities of engineers
- Software can greatly increase the effectiveness of the devices engineers design

# Attributes of a Good Software Engineer

- ▶ Is a good engineer!
- ▶ Can program in the large as well as in the small
- ▶ Has a solid understanding of computing and software
- ▶ Is comfortable with working with models at different levels of abstraction
- ▶ Can communicate and work effectively with other team members

# Software Development Process

- ▶ A rational development process is needed to produce quality software
- ▶ Any proposed rational process is necessarily an idealization
  - Humans inevitably make errors
  - Communication between humans is imperfect
  - Many things are not understood at the start
  - Supporting technology always has limitations
  - Requirements change over time