

CSC 444 Software engineering I

Fall 2003

Instructor bio

- Post-doctoral fellow at UofT
- On leave of absence from IBM Toronto
- Areas of interests
 - Software engineering
 - Business intelligence
 - Process modeling

Why are you here?

- Required course?
 - What do you expect to gain?
 - How much effort are you willing to spend
- Optional course?
 - Why are you taking this course?
 - What are you looking for?
- Career aspirations?

What you can learn

- A broad survey of software engineering issues.
 - Requirement specification
 - Design, modeling, and architecture
 - Quality, testing, verifications
 - Software processes
- An experience in building an application software.
 - Team work
 - Differences between software and other engineering artifacts.
 - Good enough vs. perfection

What will you have to do

- Fair amount of readings and research
 - Most of it is light readings
 - Need to apply it
- Lots of planning, execution, control of your project phases
 - Phases add up
 - Continuous work not a last minute brave effort
- Programming?

Grading

- Course project
 - 5 phases
 - requires work during practical sessions
- Course presentation and report
 - based on your project
 - based on your experience
- Final exam

Books

- Hans van Vliet, “Software engineering: principles and practice”, 2nd edition, Wiley
- Booch, Rumbaugh, Jacobson, “The Unified Modeling Language User Guide”, Addison-Wesley
- PMI, “A guide to Project Management Body of Knowledge”
- Design patterns, Software architecture, The mythical man-month

Software engineering

- ... establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines ... NATO 68
- ... application of systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is the application of engineering to software ... IEEE

On the negative side

- There are many stories about software failures:
 - Challenger (NASA), Ariane (ESA)
 - Y2K
 - Common security issues with commercial products

On the positive side

- Compare the facilities of today with that of a decade ago, two, three
 - internet technologies
 - telecommunication
 - databases, search engines
 - graphics, entertainment

Conclusion

- It is difficult to build a software of large size.
 - Intangible, invisible, abstract
 - Doesn't wear or tear, no physical constraints
 - It is flexible and evolves
 - Software (humans) make mistakes

Conclusion

- No silver bullet
- Good engineering practices can help
 - Effective development process
 - Project management principles
 - Automated support
- Good programming does NOT equal good software engineering