COSC 310: SOFTWARE ENGINEERING

Dr. Gema Rodriguez-Perez
University of British Columbia
gema.rodriguezperez@ubc.ca

WHO I AM

- My name is Gema Rodriguez-Perez. I'm an Assistant Professor in computer science and I will be your instructor in COSC 310.
- This is my first time teaching the course, but I'm super motivated. If you have some feedback, please don't hesitate and let me know.
- My research interests is Software Engineering. I do research on Open Source Software, Software maintenance and evolution, EDI in Software Engineering.
- I'm always excited to talk about my research ☺. Don't hesitate to reach out to me.
- I'm originally from Spain. My pronouns are she/her/hers

AGENDA

- What is Software Engineering?
- Course Motivation
- Course Overview
- Cybersecurity Workshop

Medicine and Global Health

Education

Energy and Sustainability

Scientific Discovery

Security and Privacy

Transportation

Technology for Development

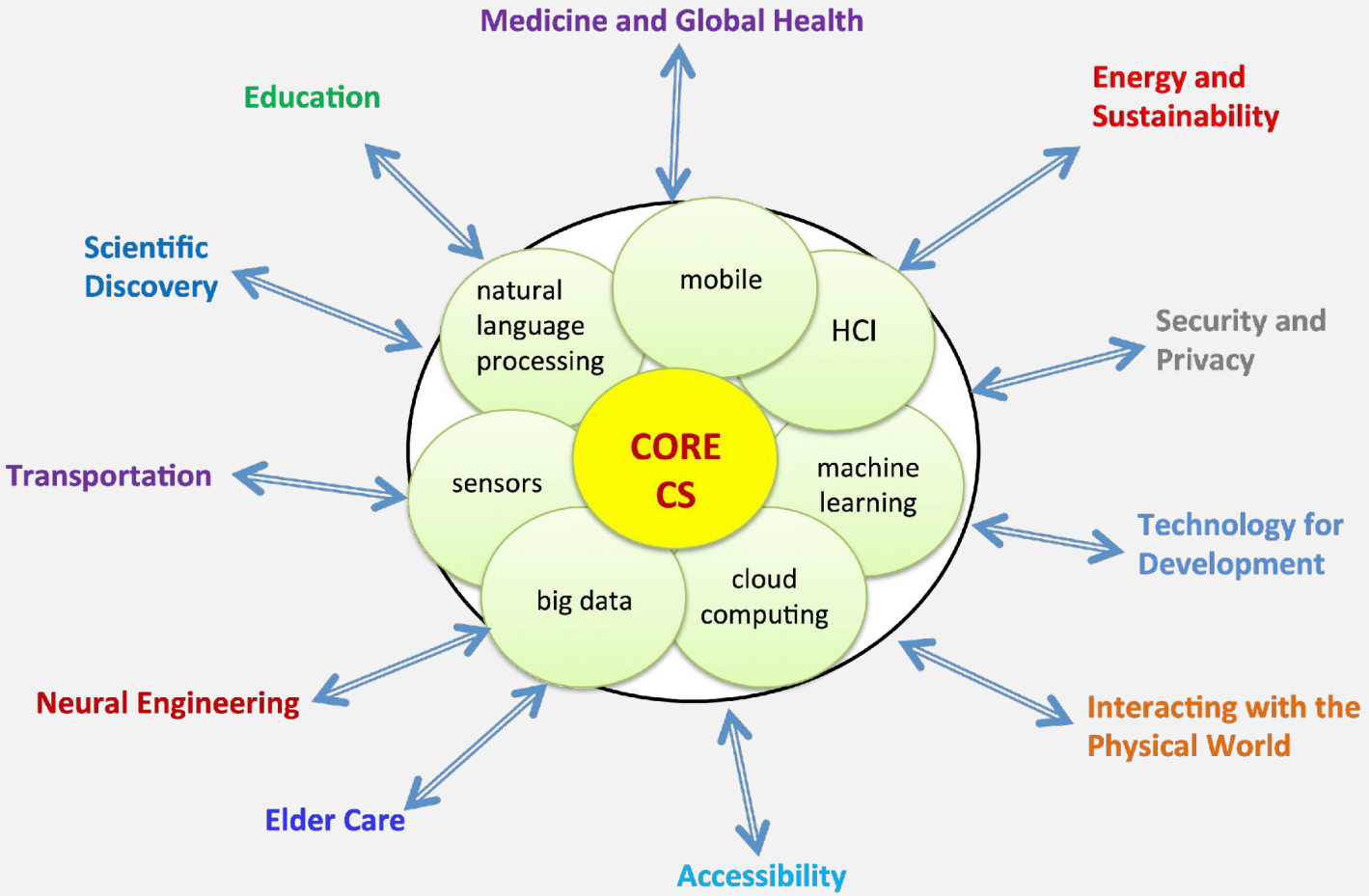
Neural Engineering

Interacting with the Physical World

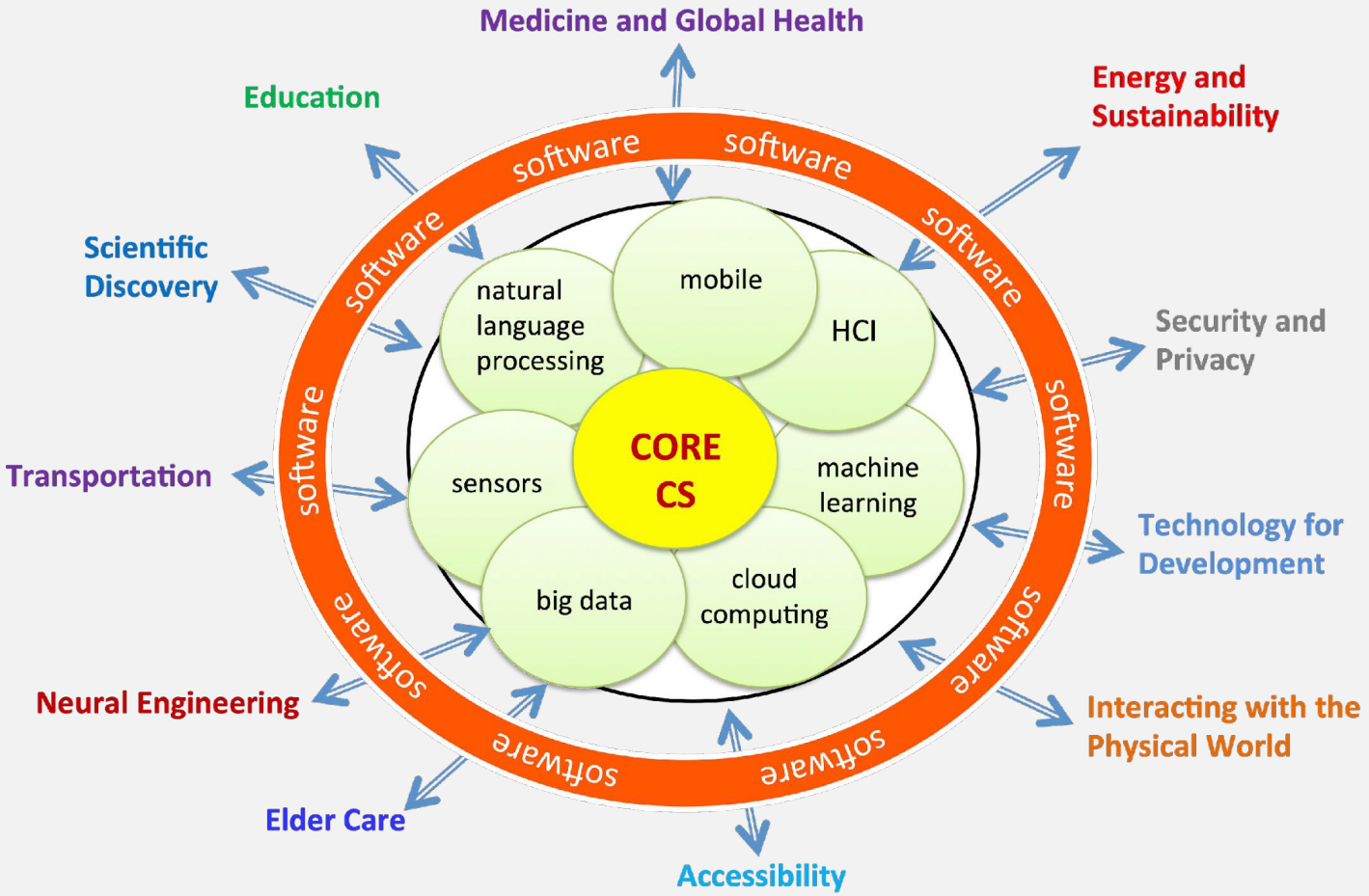
Elder Care

Accessibility

[Ed Lazowska]


Medicine and Global Health
Education
Energy and Sustainability
Scientific Discovery
natural language processing
mobile
HCI
Security and Privacy
Transportation
sensors
CORE CS
machine learning
Technology for Development
big data
cloud computing
Neural Engineering
Interacting with the Physical World
Elder Care
Accessibility

[Ed Lazowska]


Medicine and Global Health
Energy and Sustainability
Security and Privacy
Technology for Development
Interacting with the Physical World
Accessibility
Elder Care
Neural Engineering
Transportation
Scientific Discovery
Education
software
natural language processing
mobile
HCI
sensors
CORE CS
machine learning
big data
cloud computing

your
life depends on
code

Transportation Industry

(Electric and Autonomous Cars)

Banking Industry

(Robo Advisors and Risk Management)

Law Enforcement

(Facial Recognition)

Home Appliances

(Smart Refrigerators)

Social Media

(Facebook User Recommendations)

Healthcare

(Prioritizing Medical Care in Emergencies)

What's

SE?

WHAT'S SE?

- "An engineering discipline whose aim is the production of fault-free software that is delivered on time, within budget, and satisfies the user's needs." (Software engineering, Ian Sommerville)
- "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software."
- All aspects of software production (Not just programming) - Software engineering involves all phases of software development including specification, design, implementation, testing, and maintenance. This includes project management, **personnel issues**, and the development of tools and techniques to support software production.

SPECTACULAR SOFTWARE FAILURES

- There are numerous examples of software development efforts that failed **MISERABLY**.
- These failures are not simply related to programmer ability but to higher-level factors such as:
  - **Project management and team collaboration**
  - **Unrealistic expectations** or **poor system specification**/design
  - Absence of formal development procedures

- Examples of major failures:
  - Software for Canadian gun registry supposed to cost $1 million has cost over $750 million.
  - FBI's virtual case system totally discarded after $170 million development costs.

WHY DO WE NEED SE?

- Software engineering provides techniques and processes that makes software development less costly and more reliable.
- For small software projects involving one or two people, software engineering techniques are less important because most of the effort is spent programming.
- For large software projects, the collaboration and cooperation of developers is more important. We need procedures to manage and document the ***process*** as systems are too large for one person to understand, and each person works on only their small part. The project scale introduces significant overhead that must be ***managed***.
- Software must deliver the required functionality and performance to the user and must be maintainable, dependable, usable, and secure (etc….)

SOME BASIC DEFINITIONS

- ***Software*** is the program code ***and*** associated documentation and configuration data.
- A ***software system*** consists of a set of programs, data sets, and documentation.
- A ***software process*** is a set of activities (specification, design, implementation, testing, maintenance) whose goal is the development of software.
- A ***software process model*** is a simplified representation of a software process. (e.g. waterfall model)
- ***Computer-Aided Software Engineering*** (***CASE***) tools are programs that help design and build software.
- ***Unified Modeling Language*** (***UML***) is a standardized graphical language used in object-oriented development to specify various views of a system.

TYPES OF SOFTWARE

There are many ways to categorize software. By its intended user base.

- *Generic or mass-market software* is designed for multiple users in different organizations.
  - E.g. Microsoft Office, Linux, Oracle
- *Customized software* is designed and built for a particular user.
  - E.g. air traffic control systems, military applications, control systems

PROJECT DIVERSITY

While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system.

- Aerospace systems
- Automotive
- Health
- Banking
- Entertainment / Consumer

PROJECT DIVERSITY

While all software projects have to be professionally managed and developed, different techniques are appropriate for different types of system.

- Aerospace systems
- Automotive
- Health
- Banking
- Entertainment / Consumer

Safety critical control systems require a complete and analyzable specification to be developed

Privacy and security concerns; legal

Risk of failure?

SOFTWARE PROCESS ACTIVITIES

- Software engineering involves the following general processes:
  - **Software Specification**: where customers and engineers define the software that is to be produced and the constraints on its operation.
  - **Software Development**: where the software is designed and programmed.
  - **Software Validation**: where the software is checked to ensure that it is what the customer requires.
  - **Software Evolution**: where the software is modified to reflect changing customer and market requirements.
- Keep in mind that these process activities are frequently done iteratively (more of this later…)

HISTORY OF SE

- SE naturally evolved from programming.
  - Initially most programs were written by one or two people. As software got more complex, more programmers were needed.
- SE was first proposed in 1968 at a conference to discuss the 'software crisis'.
  - The 'crisis' was the challenge of creating significantly larger software systems to run on the ever more powerful hardware.
  - People saw that current solutions did not scale. Engineering approaches were required to build large software systems.
- Various tools, methodologies, and techniques were proposed to improve the software development process. However, there is no 'silver bullet' (Brooks 1987) to software development as the task of building these systems is fundamentally complex.

PROFESSION OF SE

- SE is a **profession**. SE follow ethical standards in the development of their systems.
- Numerous professional societies such as the Association of Computing Machinery (ACM) and the Institute of Electrical and Electronic Engineers (IEEE) provide resources, guidelines, and opportunities to their members.
  - Both organizations have defined a code of ethics and professional practice.
- Unlike other engineering professions, SE are not regulated or licensed, although that has been proposed.

What's

COSC 310?

COURSE LEARNING OBJECTIVES

1. Evaluate software engineering processes used to build modern industrial-calibre systems by justifying their benefits and tradeoffs.
2. Elicit, deconstruct, and refine functional requirements and quality attributes such that they are described succinctly, completely, and precisely.
3. Devise and justify high- and low-level designs to support a given set of requirements and in support of future evolutionary needs.
4. Carry out the implementation of a design incorporating ethical and security implications of code-level choices and software process and methodological approaches.
5. Independently acquire and apply modern and unfamiliar technology and language stacks.
6. Validate systems using both black-box and white-box approaches to reason about, and improve the quality of a software system.

ASSESSMENT

- **Project 60%:**
- Weighted by teamwork grade
  - Project Grade **= Teamwork Grade** * ((M1 grade + M2 grade + M3 grade + M4 grade+ M5 grade))
  - **Teamwork Grade** = % meetings attended * % Surveys submitted (This is an INDIVIDUAL GRADEmultiplier on the Project Grade)
- **Quizzes: 20%**
  - 2 quizzes 10%
- **Individual work: 20%**
  - TA weekly feedback * surveys evaluation (average of your peer's review)

THE DEVELOPMENT PROJECT

- Software is built in teams, this project is no different.
- You get to choose your teammates. This is your decision. Make your choice carefully as teams cannot be changed. We tried to accommodate your decisions
- You will work in teams of 5.
- Your team members don't have to be in your lab section, but it's highly recommendable as you will have the same TA
- The lab's meeting will be used for weekly meetings with the TA. At least one team member should assist and they are the responsible for all the team

PROJECT DELIVERABLES

- **Milestone 1** (M1): Week 3 (details on Canvas) (team) 5%
- **Milestone 2** (M2): Week 6 (details on Canvas) (team) 10%
- **Milestone 3** (M3): Week 9 (details on Canvas) (team) 20%
- **Milestone 4** (M4): Week 12 (details on Canvas) (team) 10%
- **Milestone 5** (M5): Week 14 (details on Canvas) (team) 15%

PROJECT'S EXPECTATIONS

- Aiming for ~360 on-task hours.
- 6 hours per week * 5 people * 12 weeks.
- The project might take slightly more or slightly less, depending on your past experience.
- Treat all your teammates well. They are your lifeline.
- Typical failure cases: leaving to last minute; problems withing the teams.

ASSESSMENT

- **Project**:
  - If you score < 40%, your max course grade is 50%.
- **Quizzes**:
  - Must pass the quizzes to pass the course.

CONCLUSION

- ***SE*** is an engineering discipline concerned with the cost-effective production of high-quality software.
- **SE** is important as the size and complexity of software is increasing. This requires formal procedures and techniques for managing the software development process.
- **SE** is more than computer programming. It uses the fundamental theories of computer science and an engineering approach to produce software that is correct, dependable, usable, and efficient.
- **SE** behave ethically and morally and adhere to professional standards during their practice.