### **CPS714 Course Overview:**

* **Assignments:**
  + Five group-based assignments (up to 5 members per group).
  + Each assignment contributes to a group project.
  + **Project Management Software:**
    - Groups must use project management tools like **Microsoft Project** or free alternatives such as **Jira**, **ClickUp**, or **Zoho**.
    - **Late submissions:**
      * Penalized 1% per day.
      * After 3 days, the submission receives zero marks.

### **Project Management Tools:**

* **Categories:**
  1. **Low-end tools:**
     + Ideal for small projects.
     + Cost: under $200 per user.
  2. **Midrange tools:**
     + Suitable for multiple projects.
     + Cost: $200-$1,000 per user.
     + **Microsoft Project** is the most popular option.
  3. **High-end tools:**
     + Also known as **Enterprise Project Management (EPM)** software.
     + Licensed per user.
* **Free or open-source tools:** Jira, ClickUp, Zoho.

### **Project Management Techniques:**

* **Tools and Techniques for Project Management** include:
  + **Scope management:**
    - Project charter, scope statement, Work Breakdown Structure (WBS).
  + **Time management:**
    - Gantt charts, network diagrams, critical path analysis, critical chain scheduling.
  + **Cost management:**
    - Cost estimates, earned value management.

### **Marking Scheme:**

* **Assignment-based group project** (including final presentation and peer evaluation): **30%**
* **Term Tests** (1-2): **60%**
* **Quizzes/Assignments:** **10%**

### **Instructor Information:**

* **Experience:**
  + Over **15 years** of teaching, professional, and research experience in Software Engineering.
  + **7 years** at Toronto Metropolitan University (TMU).
  + Former positions include:
    - **Adjunct Assistant Professor** in Software Engineering at Queen's University.
    - **Lecturer** at HU (3 years).

### **Instructor’s Research Interests:**

1. **Software Security Analysis:**
   * Internet of Things (IoT)
   * Blockchain
   * Mobile and Web applications
   * Machine learning for cybersecurity
2. **Software Design Recovery & Evolution:**
   * Migrating web applications to SOA.
   * SQL to NoSQL migration.
   * Detection of feature interactions in dynamic scripting languages.
3. **Model Driven Software Engineering (MDE):**
   * Model pattern engineering.
   * Variability identification in automotive systems.

### **Select Publications:**

* **Automotive Systems & IoT Security:**
  1. "Security Analysis for SmartThings IoT Applications" (ICSE 2019)
  2. "Security Smells in Smart Contracts" (QRS 2019)
  3. "Modeling AUTOSAR Implementations in Simulink" (ECMFA 2018)
* **Software Evolution:**
  1. "Framework for Migrating Web Applications from SQL to NoSQL" (CASCON 2019)
  2. "Detection of Feature Interaction in Dynamic Scripting Languages" (CASCON 2019)

### **Why Software Project Management?**

Software project management is crucial for ensuring the successful delivery of a product on time, within budget, and with the agreed-upon quality characteristics. Whether it’s a software project or any other type of project, having a structured approach is essential.

To achieve this, one needs:

* A **process** to define schedules, budgets, and quality characteristics.
* A set of **techniques** to define, plan, execute, and monitor key aspects like goals, time, quality, and costs.

### **Skills and Goals of the Course:**

By the end of this software project management course, you will be able to address the following critical questions:

* **Task Estimation:** How long will it take to complete a task?
* **Costing:** How much should I charge for a project?
* **Team Management:** How do I keep the team motivated and ensure projects are an opportunity for growth?
* **Risk Management:** How do I handle project risks?
* **Progress Assessment:** Is the project on time and on budget?
* **Quality Control:** How can I control the quality of the final product?

These skills are essential for managing software development projects, which require specific competencies, techniques, and management abilities.

### **Why is Software Project Management Unique?**

Software project management stands out because of the following factors:

1. **Intangibility:** Unlike physical products, software is not something you can touch.
2. **Flexibility:** Software products can be developed in various forms, with different sizes, constraints, and levels of complexity.
3. **One-off Nature:** Many software projects are custom-built and not repetitive.
4. **Flexible Development Process:** The development of software is highly adaptable, which allows for changes and iterations.
5. **Increasing Complexity:** The complexity of software systems is growing exponentially, especially as more is demanded of them.
6. **Safety-Critical Systems:** In some cases, like in aviation or healthcare, human lives may depend on the software functioning correctly, making quality and precision paramount.

### **Complexity in Software Projects:**

To highlight the complexity, consider this fact: the entire **Saturn V rocket** (which carried astronauts to the moon) had less computing power than a modern smartphone. This illustrates how much technology has advanced, making software projects ever more challenging to manage.

### **Characteristics of a Project:**

1. **Temporary:**
   * Projects have a **definite start and end**. The project ends when its goals are achieved or if it's closed because the goals cannot or will not be met.
   * However, a project’s **results are not temporary**. For example, a project might create a product that continues to exist long after the project ends (as seen in the project and product lifecycle).
2. **Unique Products, Services, or Results:**
   * A project results in a **unique output**, whether it's a tangible product, a new capability to perform a service, or knowledge shared through documentation and presentations.
3. **Progressive Elaboration:**
   * Projects develop **in steps and increments**, refining as the project progresses. This approach ensures that the project stays within its scope and adjusts as necessary.
4. **Resource Constrained:**
   * Like any real-world process, projects have **limited resources** (time, money, personnel, etc.).

### **Project Lifecycle:**

Projects typically follow a structured lifecycle:

1. **Initiate**
2. **Plan**
3. **Execute**
4. **Monitor**
5. **Close**

Each phase adds cumulative work over time, with **progressive elaboration** ensuring that the project evolves step-by-step.

### **Project Management Context:**

1. **Subprojects:**
   * Projects can be divided into smaller **subprojects**, which may also be referred to as "projects" and managed independently but within the larger project context.
2. **Program Management:**
   * A **program** is a set of related projects managed in a coordinated manner to achieve a collective benefit or goal.
   * Example: A program may involve multiple projects working toward improving a software product.
3. **Portfolio Management:**
   * A **portfolio** includes **unrelated projects or programs** grouped together for easier management and to meet strategic objectives.
   * Example: A company's portfolio may consist of software development projects, marketing campaigns, and business expansion plans.

### **Projects vs. Operational Work:**

* **Commonalities:**
  + Both projects and operational work are **performed by people**, involve **limited resources**, and are **planned, executed, and controlled**.
* **Differences:**
  + **Projects** are temporary and aimed at achieving specific goals, after which they are terminated.
  + **Operational work** is continuous and focuses on sustaining business operations.

### **Examples of Projects and Operational Work:**

* **Projects:**
  + Building a car
  + Designing a car
  + Writing a research paper
  + Developing a software system
* **Operational Work:**
  + Cooking dinner
  + Maintaining a software system

### **Software Development Framework**

A general software project management framework serves to:

* Establish a **shared vision** about project goals, expected outcomes, and the characteristics of the development process.
* Organize the work as a **progressive refinement**, beginning with specifications and moving toward the final goals.
* Minimize the **impact of uncertainties** and unknowns throughout the project.
* Identify **deviations from the plan** in terms of goals, costs, and quality.
* Ensure the **coherency and quality** of the project artifacts despite unforeseen changes or uncertainties.
* **Motivate the team**, fostering engagement and productivity.

### **Key Concerns in Software Project Management**

1. **Feasibility Assessment**
2. **Scope Management**
3. **Time Management**
4. **Cost Management**
5. **Change Control and Configuration Management**
6. **Quality Management**
7. **Risk Management**
8. **Human Resource Management**

These concerns map to the key project phases:

* **Initiate**: Kick off, formalize goals, and assess feasibility.
* **Plan**: Define schedule and costs.
* **Execute & Monitor**: Ensure project goals, cost, and schedule are on track; implement change control and configuration management.
* **Close**: Collect outputs, evaluate quality, and ensure project approval.

### **Project and Product Life Cycles**

#### **Project Life Cycle**

A **project life cycle** includes several phases that define:

* The work to be performed.
* The deliverables to be produced and when.
* The people involved in each phase.
* How management will control and approve deliverables.

Phases of the project life cycle:

1. **Early Phases**:
   * **Low resource needs** but **high uncertainty** and risk.
   * Stakeholders have the most influence here.
2. **Middle Phases**:
   * **Increased certainty** of project completion.
   * More resources are required.
3. **Final Phase**:
   * Focuses on ensuring that project requirements are met and the project is approved by the sponsor.

#### **Product Life Cycles**

Product life cycles can follow different models, including:

* **Predictive**: Example: Waterfall model.
* **Iterative**: Software is developed in repeated cycles, refining along the way.
* **Incremental**: Components of the product are developed in increments.
* **Adaptive**: Flexible, used when requirements evolve during the project (e.g., Agile).
* **Hybrid**: Combines elements of multiple models.

#### **Popular Life Cycle Models**

1. **Waterfall Model**: Linear and well-defined stages of development.
2. **Spiral Model**: Iterative approach focusing on risk assessment.
3. **Prototyping Model**: Helps clarify user requirements through prototype development.
4. **Rapid Application Development (RAD)**: Aims for fast development without sacrificing quality.

### **Scrum**

Scrum is a widely used framework for managing and completing complex projects, especially in software development. It emphasizes iterative progress, adaptability, and regular feedback loops.