This course supports the assessment for Software Engineering. The course covers 5 competencies and represents 4 competency units.

Introduction

Overview
Software engineering is a complex discipline. Over the years it has evolved from an informal process to more formal processes. There are many methodologies that have been developed and are no longer used; however, there are still some in use, or have been developed in recent years. IT professionals are continually seeking ways to make it more efficient and effective. The process of software engineering starts with scope and project definition, and then detailed requirements. The requirement-gathering process relies on stakeholders who may not understand what they require; therefore communicating this to software professionals is difficult at best. Business analysts almost need to be able to read a stakeholder's mind when formulating requirements, and they certainly need to understand the subject matter. Stakeholders also do not always understand software and its constraints, and in some cases, they may ask for something that is simply not possible to implement, or may be too costly. There needs to be clear, concise communication between the many stakeholders, developers, project managers, business owners, and others. This requires a solid understanding of the software engineering process, including where its flexibility may be, how to interpret user needs and wants, as well as understanding the technical side.

This course will start with the basics of software—what it is, why it is complex, how it has evolved, and why it’s so important. Legacy software will also be covered. This is software that was developed at a time when discipline was scarce. This course also provides a generic process framework, since all methodologies are based on common phases, supporting processes, and activities. It provides the groundwork for formal process models. There is an in-depth view of the Waterfall model, a prescriptive process model, which is also called the classic model or the Software Development Life Cycle (SDLC). The SDLC is a generic term used to describe the overall process. There are many resources available on the SDLC, but many of them are slightly different.

Agile development is more commonly used. Simply put, it's a different way to organize the phases, processes, and involved parties of the SDLC. There are many different agile methodologies. The most common is extreme programming, which has also evolved over time. After the introduction and review of the formal prescriptive process models, the course will visit the main phases and processes in more detail (e.g., requirements engineering). This course is meant to provide a general understanding of software engineering, and also to provide some insight into the different phases. It is not intended to make you an expert, but rather to provide a solid framework to build upon when entering the IT industry.

Watch the following video for an introduction to this course:

Note: To download this video, right-click the following link and choose "Save as...": download

video.

Watch the following video for information about getting started in this course:

Note: To download this video, right-click the following link and choose "Save as...": download video.

Competencies
This course provides guidance to help you demonstrate the following 5 competencies:

- **Competency 4020.1.1: Introduction to Software Engineering**
  The graduate describes software and legacy software; identifies the Core Principles of software engineering, the Generic Process Framework, introductory software engineering concepts and terms (e.g., SDLC, Prescriptive Process model); and selects appropriate model activities when given project descriptions.
- **Competency 4020.1.2: Software Engineering and Process Models**
  The graduate describes software engineering process models (e.g., waterfall, agile methodologies) and selects appropriate models when given project descriptions.
- **Competency 4020.1.3: Requirements Engineering**
  The graduate defines requirements engineering concepts, describes requirements engineering processes, and selects appropriate graphical models for specific projects.
- **Competency 4020.1.4: Software Design Concepts, Including Architecture**
  The graduate describes software design, design concepts and classes, and selects appropriate uses of design concepts for specific projects.
- **Competency 4020.1.5: Quality Concepts, Software Quality Assurance, and Software Testing**
  The graduate describes quality assurance processes and explains how to implement quality and software testing concepts in specific cases.

Course Instructor Assistance
As you prepare to successfully demonstrate competency in this subject, remember that course instructors stand ready to help you reach your educational goals. As subject matter experts, mentors enjoy and take pride in helping students become reflective learners, problem solvers, and critical thinkers. Course instructors are excited to hear from you and eager to work with you.

Successful students report that working with a course instructor is the key to their success. Course instructors are able to share tips on approaches, tools, and skills that can help you apply the content you're studying. They also provide guidance in assessment preparation strategies and troubleshoot areas of deficiency. Even if things don’t work out on your first try, course instructors act as a support system to guide you through the revision process. You should expect to work with course instructors for the duration of your coursework, and you are encouraged to contact them as soon as you begin. Course instructors are fully committed to your success!

Preparing for Success

The information in this section is provided to detail the resources available for you to use as you complete this course.

Learning Resources

The learning resources listed in this section are required to complete the activities in this course. For many resources, WGU has provided automatic access through the course. However, you may need to manually enroll in or independently acquire other resources. Read the full instructions provided to ensure that you have access to all of your resources in a timely manner.

Automatically Enrolled Resources

You can access the learning resources listed in this section by clicking on the links provided throughout the course. You may be prompted to log in to the WGU student portal to access the resources.

Connect

The Connect resource is available to you within this course. It contains adaptive reading and lessons, as well as quizzes. You will be directly linked to the specific sections required within the activities that follow.

- Pressman, R. & Maxim, B. (2014). *Software engineering: a practitioner's approach* (8th ed.). New York, NY: McGraw-Hill Education. ISBN: 978-0-07-802212-8

Note: The e-text is available to you as part of your program tuition and fees, but you may purchase hard copies at your own expense through a retailer of your choice. If you choose to do so, please use the ISBN listed to ensure that you receive the correct edition.

Note: If you require assistance or approval from the course instructor for a second attempt on the objective assessment, the course instructor will rely on the gradebook reporting feature.

Complete the Preassessment

If you believe you have previous knowledge of some or all topics covered in this course, start by taking the preassessment before you begin and use its results to focus your studies.

- Complete the preassessment located in the Assessment tab.

Course instructors can help you develop a study plan based on your preassessment results.

Getting Started Video

Overview of Software Engineering

Please watch the video for an overview of software engineering:

Note: To download this video, right-click the following link and choose "Save as...": download video.

Pacing Guide

The pacing guide suggests a weekly structure to pace your completion of learning activities. It is provided as a suggestion and does not represent a mandatory schedule. Follow the pacing guide carefully to complete the course in the suggested timeframe.

- Pacing Guide: Software Engineering

Note: This pacing guide does not replace the course. Please continue to refer to the course for a comprehensive list of the resources and activities.

Introduction to Software Engineering

In this section, you will learn the fundamental concepts of software engineering, including the definition of software and legacy software. You will also learn about a generic process framework for software engineering, including process and umbrella activities. The generic process framework provides the basis for understanding other process models (e.g., SDLC, Waterfall, Agile) since they contain the same activities, phases, and supporting processes, just organized differently. This lesson will then introduce prescriptive process models and approaches to software process assessment and improvement.

Lesson 1: Software and General Concepts of Software Engineering

This topic will introduce you to the fundamentals of software engineering and the Software Development Life Cycle (SDLC).

This topic addresses the following competency:

- **Competency 4020.1.1: Introduction to Software Engineering**
  The graduate describes software and legacy software, the Core Principles of software engineering, the Generic Process Framework, and introductory software engineering concepts and terms. The graduate also selects appropriate model activities when given project descriptions.

This topic highlights the following objectives:

- Describe software characteristics and categories.
- Describe legacy software.
- Describe the core principles of software engineering.
- Identify the appropriate approach for software engineering for a given project description.

Complete: Lesson 1

Complete all reading and activities included in the following module:

- Software and General Concepts of Software Engineering

Lesson 2: Software Engineering Process Categories and Prescriptive Process Models

This topic will introduce you to process categories and prescriptive process models. Process

categories are generic groupings of the types of processes you'll learn about in the following activities. These include linear, iterative, and evolutionary. The prescriptive process models are the formal models you'll learn about such as waterfall and agile. Process categories and prescriptive process models are the foundation for great software engineering in practice.

This topic addresses the following competencies:

- **Competency 4020.1.1: Introduction to Software Engineering**
  The graduate describes software and legacy software, the Core Principles of software engineering, the Generic Process Framework, and introductory software engineering concepts and terms (e.g., SDLC, Prescriptive Process model). The graduate also selects appropriate model activities when given project descriptions.
- Competency 4020.1.2: Software Engineering and Process Models
  The graduate describes software engineering process models (e.g., waterfall, agile methodologies) and selects appropriate models when given project descriptions.

This topic highlights the following objectives:

- Identify software engineering process flows and approaches to software process assessment and improvement.
- Describe the prescriptive process models.
- Identify when to use the waterfall model for software development in given scenarios.
- Describe situations in which the incremental process model is useful.
- Describe prototyping processes.
- Describe the spiral model.
- Define the concurrent model and the Unified Process (UP) model.
- Identify an appropriate model to use with a given project description.
- Describe process technology.

Complete: Lesson 2

Complete all reading and activities included in the following module:

- Process Categories and Prescriptive Process Models

Software Engineering and Process Models

In this section, you will learn more about software engineering process models.

The activities in this section will prompt critical thinking about real-world problems businesses face in relation to software engineering.

Lesson 3: Agile Development and Agile Process Models

This topic will introduce you to the agile development and associated process models.

This topic addresses the following competency:

- **Competency 4020.1.2: Software Engineering and Process Models**

The graduate describes software engineering process models (e.g., waterfall, agile methodologies) and selects appropriate models when given project descriptions.

This topic highlights the following objectives:

- Describe the principles and key assumptions associated with agile development.
- Describe extreme programming.
- Describe additional examples of agile processes.
- Identify the appropriate agile approach when given a project description.

Complete: Lesson 3

Complete all reading and activities included in the following module:

- Agile Development and Agile Process Models

Requirements Engineering

In this section, you will learn the fundamental concepts of requirements engineering, which is a phase/process embedded in any software engineering approach. You will also learn more specific aspects of requirements engineering including the differences among types/levels of requirements and the process and activities associated with it. Usage scenarios will be introduced as well.

Lesson 4: Introduction to Requirements Engineering

This topic will introduce you to the fundamental concepts of requirements engineering including associated tasks and activities. It will also provide information on the different levels of requirements.

This topic addresses the following competency:

- **Competency 4020.1.3: Requirements Engineering**
  The graduate defines requirements engineering concepts, describes requirements engineering processes, and selects appropriate graphical models for specific projects.

This topic highlights the following objectives:

- Describe requirements engineering.
- Define key requirements engineering tasks and issues.
- Identify the main sections of a software requirements specification (SRS) template.
- Describe requirement engineering processes.
- Explain the basic principles of quality function deployment (QFD).
- Describe requirement analysis processes and activities.
- Identify appropriate analysis packages when given project descriptions.
- Describe a usage scenario (use case).
- Identify appropriate graphical models for given scenarios.

Read: Lesson 4

Complete all reading and activities included in the following module:

- Introduction to Requirements of Engineering

Lesson 5: Introduction to Design Concepts

In this section, you will learn the fundamental concepts of design, which is a phase/process embedded in any software engineering approach. You will also learn more specific aspects of design, including how a design traces to the requirements, and what should be considered and included. Object-oriented design will be introduced, including classes.

This topic addresses the following competency:

- **Competency 4020.1.4: Software Design Concepts, Including Architecture**
  The graduate describes software design, design concepts and classes, and selects appropriate uses of design concepts for specific projects.

This topic highlights the following objectives:

- Describe software design processes and characteristics.
- Identify appropriate uses of various design concepts.
- Describe design classes.

Complete: Lesson 5

Complete all reading and activities included in the following module:

- Introduction to Design Concepts

Quality Concepts, Software Quality Assurance, and Software Testing

In this section, you will learn the fundamental concepts of quality assurance and software testing in relation to software engineering, which are phases/processes/philosophies embedded in any software engineering approach. You will also learn more specific aspects of these concepts. Defects and bugs will be introduced, as well as basic concepts of quality metrics.

Lesson 6: Introduction to Quality Concepts and Software Quality Assurance

This topic will introduce you to the core concepts of quality assurance and software testing in relation to software engineering. The Requirements Trace Matrix will be covered as well as quality metrics and definitions of bugs/defects.

This topic addresses the following competency:

- **Competency 4020.1.5: Quality Concepts, Software Quality Assurance, and Software Testing**
  The graduate describes quality assurance processes and explains how to implement quality and software testing concepts in specific cases.

This topic highlights the following objectives:

- Define *quality* as it relates to software engineering.
- Describe core concepts of software quality.

Complete: Lesson 6

Complete all reading and activities included in the following module:

- Introduction to Quality Concepts & Software Quality Assurance

Lesson 7: Software Testing

This topic will introduce you to the core concepts of quality for software engineering.

This topic addresses the following competency:

- **Competency 4020.1.5: Quality Concepts, Software Quality Assurance, and Software Testing**
  The graduate describes quality assurance processes and explains how to implement quality and software testing concepts in specific cases.

This topic highlights the following objectives:

- Identify appropriate software testing methods for given scenarios.
- Identify criteria for testing different types of software applications.
- Describe system and unit testing.

Complete: Lesson 7

Complete all reading and activities included in the following module:

- Protection and Security

Final Steps

Congratulations on completing the activities in this course! This course has prepared you to complete the assessment associated with this course. If you have not already been directed to complete the assessment, schedule and complete your assessment now.

First Attempt Checklist

One of the many things that makes WGU unique is its competency-based education model. If you know the material, all you have to do is prove it by passing the exam. If you can do this, you can accelerate the receipt of your degree.

To make sure you have the best chance possible to pass the exam on your first attempt, the following steps should be completed successfully before you take it:

1. Complete all 7 lessons in the Connect platform.
2. Complete the assigned quizzes, striving for 80% or higher on each.
3. Take the preassessment. View the Coaching Report to see your results and review each

individual topic with a score of less than 80%.
4. Schedule the final exam if you have scored 80% or higher on the preassessment.

If you have completed the steps above and you feel comfortable with all of the concepts presented, you are most likely ready to attempt the exam.

If you fail your first attempt, you will be required to contact the course instructor to see what went wrong and how you can prepare to ensure a successful second attempt. After determining you are ready, your course instructor will approve your request once to make another exam attempt.