**BBE3105 Software Engineering**

Coursedescription

Software engineering is the discipline concerned with the application of theory, knowledge, and practice to build effectively and efficiently software systems that satisfy the requirements of users and customers. Software engineering is applicable to small, medium, and large-scale systems. It encompasses all phases of the life cycle of a software system. The life cycle includes requirement analysis and specification, design, construction, testing, and operation and maintenance. The development of programs benefits from the concepts and practices derived from software engineering. Software engineering employs engineering methods, processes, techniques, and measurement. It benefits from the use of tools form an aging software development; analyzing and modeling software artifacts; assessing and controlling quality; and for ensuring a disciplined, controlled approach to software evolution and reuse. Software development, which can involve an individual developer or a team of developers, requires choosing the tools, methods,and approaches that are most applicable for a given development environment.

CourseObjectives

 To introduce thestudenttotheevolution andscopeofSoftware Engineeringasadisciplineand asaProfession

 To expose the student to the practical imperatives underpinning software development – requirementsanalysis,design,implementation,testing,deployment,usertrainingandsupport, andmaintenance

CourseContent

**HistoryandOverview**

Indicatesomereasonsforstudyingsoftwareengineering

Highlightsomepeoplethatinfluencedorcontributedtotheareaofsoftwareengineering

Indicatesomeimportanttopicareassuchasthesoftwareprocess,requirements, specifications,design,testing,validation,evolution,andprojectmanagement

Contrastsoftwareengineeringwithcomputerengineering

Mentionsomeexamplesthatwouldusethesoftwareengineeringapproach

Indicatetheexistenceofformalizedsoftwareprocessessuchasthesoftwarelifecycle

Explainthatrequirementsandspecificationsmaychangeslightlyasasoftwareproject evolves

Indicatetheimportanceoflanguageselectionwhendoingsoftwaredesign

Highlighttheimportanceoftestingandvalidationinasoftwareprojects

Exploresomeadditionalresourcesassociatedwithsoftwareengineering

Explainthepurposeandroleofsoftwareengineeringincomputerengineering

**SoftwareProcesses**

Softwarelifecycleandprocessmodels(Waterfalls,Spiral,Extremeprogramming,Agile, RAD,RUP)

Processassessmentmodels

Softwareprocessmetrics

**SoftwareRequirementsandSpecifications**

Requirementselicitation

Requirementsanalysismodelingtechniques

Functionalandnonfunctionalrequirements

Prototyping

Basicconceptsofformalspecificationtechniques

SoftwareRequirementsSpecification

**SoftwareDesign**

Fundamentaldesignconceptsandprinciples

Software architecture (Software quality attributes–Run time, Business, and Engineering; Architecturalviewsandviewpoints;Thetechnicalarchitectingprocess;Organisational architecting; Architectural styles)

ModelingwithUML

Structureddesign

Object-orientedanalysisanddesign

Domain-drivendevelopment

Designforreuse

**SoftwareVerificationandValidation**

Verification(StaticandDynamic)

Validationplanning

Testingfundamentals,includingtestplancreationandtestcasegeneration

Black-boxandwhite-boxtestingtechniques

Unit,integration,validation,andsystemtesting

Object-orientedtesting

Inspections

**SoftwareEvolution**

Softwaremaintenance:thedifferentformsof maintenance;theassociateddisciplesandthe roleandthenatureofconfigurationmanagementandversioncontrol

Impactanalysis

 Regressiontesting

Associatedsoftwaresupport

Characteristicsof maintainablesoftware

Softwarere-useinitsdifferentforms–theirstrengthsandweaknesses

Reengineering

Legacysystems

Softwareretirement

**SoftwareToolsandEnvironments**

Programmingenvironments

Requirementsanalysisanddesignmodelingtools

Testingtools

Configurationmanagementtools

Toolsbasedondatabases–theirdesignanddevelopment

AdditionalpossibilitiesincludingCASEtools

Toolintegrationmechanisms

**LanguageTranslation**

Therangeoftoolsthatsupportsoftwaredevelopmentforthecomputerengineer;the role of a formal semantic sofa language

Differentpossibilitiesregardinglanguagetranslation:comparisonofinterpretersandcompilers forhigh-levellanguages,andsiliconcompilersforhardwaredescriptionlanguages,additionalpossibilities

Languagetranslationphases(lexicalanalysis,parsing,generationphase,optimization); separatecompilationortranslation-thebenefitsandthemechanisms;machine-dependent andmachine-independentaspectsoftranslation

**SoftwareProjectManagement**

Teammanagement:teamprocesses;teamorganizationanddecision-making,rolesand responsibilitiesinasoftwareteam;roleidentificationandassignment;projecttracking;team problemresolution

Projectscheduling

Softwaremeasurementandestimationtechniques

Riskanalysis

Softwarequalityassurance

Softwareconfigurationmanagement

Projectmanagementtools

Softwareprojectdocumentation

**SoftwareFaultTolerance**

Softwarereliabilitymodels

Softwarefault-tolerancemethods:N-versionprogramming,recoveryblocks,rollbackand recovery

Faulttoleranceinoperatingsystemsanddatastructures

Faulttoleranceindatabaseanddistributedsystems

LearningOutcomes

Oncompletionofthiscoursethestudentwillbeableto:

 Select, with justification, thesoftware development models most appropriate for the development andmaintenance ofdiversesoftwareproducts;andexplaintheroleofprocess maturitymodels.

 Applykeyelementsandcommonmethodsforelicitationandanalysistoproduceasetof softwarerequirementsforamedium-sizedsoftwaresystem;useacommon,non-formalmethod tomodeland specify(inthe formofarequirementsspecificationdocument)therequirementsfor amedium-sizesoftwaresystem(e.g.,structuredanalysisorobject-oriented-analysis);conducta reviewofasoftwarerequirements documentusingbestpracticestodeterminethequalityofthe document;andtranslateintonaturallanguageasoftwarerequirements specification writtenina commonlyusedformalspecificationlanguage.

 Evaluatethequalityofmultiplesoftwaredesignsbasedonkeydesignprinciplesandconcepts; usingasoftwarerequirementspecification andacommonprogramdesignmethodologyand notation,createandspecifythesoftwaredesignforamedium-size softwareproduct(e.g.,using structureddesign or object-orienteddesign);and using appropriateguidelines,conduct the reviewofasoftwaredesign.

 Demonstratethe application of the different types and levels of testing (unit, integration, systems,andacceptance) tosoftwareproductsofmedium size;undertake, aspartofateam activity,aninspectionofamedium-sizecodesegment;anddescribetherolethattoolscanplay inthevalidationofsoftware.

 Identifytheprincipalissuesassociatedwithsoftwareevolutionandexplaintheirimpactonthe softwarelifecycle;developaplanforre-engineeringamedium-sized productinresponsetoa changerequest;discusstheadvantagesanddisadvantagesofsoftwarereuse;and demonstratethe abilitytoexploitopportunitiesforsoftwarereuseinavarietyofcontexts.

 Select,withjustification,anappropriatesetoftoolstosupportthesoftwaredevelopmentofa rangeofsoftwareproducts; analyzeandevaluate asetoftoolsinagivenareaofsoftware development(e.g.,management,modeling,ortesting);anddemonstrate thecapabilitytousea rangeofsoftwaretoolsinsupportofthedevelopmentofasoftwareproductof mediumsize.

 Compareandcontrastcompiledandinterpretedexecutionmodels,outliningtherelativemeritsof each;describethephasesofprogramtranslationfromsourcecodetoexecutablecodeandthe files produced by these phases; explain the differences between machine-dependent and machine-independenttranslation;andshowthemannerinwhichthesedifferencesareevidentin thetranslationprocess.

 Demonstratethroughinvolvementinateamprojectthecentralelementsofteambuildingand teammanagement;prepareaprojectplanforasoftwareprojectthatincludesestimatesofsize and effort,a schedule,resourceallocation,configurationcontrol,changemanagement,and

Projectriskidentification andmanagement; andcompareandcontrastthedifferentmethodsand techniquesusedtoassurethequalityofasoftwareproduct.

 Understand the concept of software faults and reliability of software; understand various redundancy methodsusedtoallowsoftwaretodetectsoftware faultsandproduce correctresults inthepresenceofsoftwarefaults;andunderstand softwarefaulttoleranceapproaches usedin operatingsystems,databasesystems,anddistributedsystems.

ReferenceMaterials

1. RogerS.Pressman.*SoftwareEngineering:APractitioner’sApproach*,McGraw-Hill,2004ISBN

007301933X,97800783019338

2. IanSommerville*.SoftwareEngineering*,Pearson/Addison-Wesley,2004.ISBN0321210263,

9780321210265

3. IEEEStd1471-2000, IEEERecommendedPracticeforArchitecturalDescriptionofSoftware- IntensiveSystems

4. IEEEStd830-1998, IEEERecommendedPracticeforSoftwareRequirementsSpecifications

(RevisionofIEEEStd830-1993)

5. LenBass, PaulClements, RickKazman, April11, 2003.SoftwareArchitectureinPractice, SecondEdition.AddisonWesley.ISBN:0-321-15495-9

6. JeffGarland&RichardAnthony,*LargeScaleSoftwareArchitecture,*JohnWileyandSons,ISBN

0 470848499

7. StephenT.Albin, 2003.TheArtofSoftwareArchitecture: DesignMethodsandTechniques, JohnWiley&Sons, ISBN: 0471228869

8. Bennett, S, 2005.Object*-OrientedSystemsAnalysisandDesignUsingUML*,3rdEdition, McGraw-Hill,ISBN:0077110005,ISBN13:9780077110000

9. DonaldYatesandTonyWakefield, 2004.SystemsAnalysisandDesign, SecondEdition, Prentice

Hall,PearsonEducationLimited.ISBN0273655361

*10.* IEEEStd730– 2002.*IEEEStandardforSoftwareAssurancePlans*

11. IEEEStd829-1998.*IEEEStandardforSoftwareTestDocumentation*

Requirements

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| HoursperSemester | | | | Weighted  Total  Mark | Weighted  ExamMark | Weighted  Continuous  Assessment Mark | Credit  Units |
| LH | PH | TH | CH | WTM | WEM | WCM | CU |
| 45 | 00 | 30 | 60 | 100 | 40 | 100 | 4 |