SOFTWARE DESIGN DOCUMENT (SDD)

FOR THE

C3 (Crafting Cost Calculator)

Prepared for:
Dr Nur Atiqah Sia Binti Abdullah

Prepared by:

Name	Student Id
1. Alif Muqri Bin Hazmi 2. Nurul Farhanah Binti Fazid 3. Syakir Zufayri Bin Azhar 4. Mohammad Amirul Luqman Bin Shamshi	2018413956 2018439772 2018223854 2018298154

AUTHORIZED BY:	Mohammad Amirul Luqman Bin Shamshi	APPROVED BY:	Dr Nur Atiqah Sia Binti Abdullah
Date:	14/6/2020	Date:	14/6/2020

DOCUMENT APPROVAL

REVISION	NAME	DATE
Verified By: *AlifMuqri* **PROJECT LEADER**	Alif Muqri Bin Hazmi	14/6/2020
Authenticated By: *AmirulLuqman* **PROJECT MANAGER**	Mohammad Amirul Luqman Bin Shamshi Nurul Farhanah Binti Fazid Syakir Zufayri Bin Azhar	14/6/2020
Approved By: **Client**	Dr Nur Atiqah Sia Binti Abdullah	14/6/2020

Software : Google Document
Archiving Place: /C^3/Software Engineering/SDD
Copies Available: DOC and PDF

TABLE OF CONTENT

DOCUMENT APPROVAL **1**

1 SCOPE **4**

1.1 Identification 4

1.2 System Overview 4

1.3 Document Overview 5

2 REFERENCED MATERIAL **5**

3 PRELIMINARY DESIGN **6**

3.0 C3 Overview 6

3.1 Sequence Diagram 6

3.2 Class diagram 11

4 DETAILED DESIGN **12**

4.1 C3 Structural Diagram 12

4.1.1 5userInput() 12

4.1.2 verifyInputs() 12

4.1.3 displayError() 13

4.1.4 rateGSC() 13

4.1.5 calcUFP() 14

4.1.6 verifyInputs() 14

4.1.7 displayError() 15

4.1.8 calcAFP() 15

4.1.9 calcFP(AFP, UFP) 15

4.1.10 verify() 16

4.1.11 displayError() 16

4.1.12 getLanguage() 16

4.1.13 calcKLOC(FP) 17

4.1.14 verify() 17

4.1.15 displayError() 18

4.1.16 getType() 18

4.1.17 verifyInputs() 18

4.1.18 displayError() 19

4.1.19 getSalary() 19

4.1.20 calcEffort(appType, kloc) 19

4.1.21 calcTimeDev(effort,AppType) 20

4.1.22 calcNumStaffMonth(effort,timedev) 20

4.1.23 calcNumStaffWhole(staffmonth,timedev) 20

4.1.24 calcCostEstimation(salary,staffwhole,timedev) 21
4.1.25 displayResults(fp,effort,timedev,staffmonth,staffwhole,cost) 21
4.1.26 saveCalculations(result) 21

5 DATA 22

5.1 Data input 22
5.1.1 userInputs[3] for UnadjustedFunctionPoint 22
5.1.2 userOutputs[3] for UnadjustedFunctionPoint 22
5.1.3 userInquiries[3] for UnadjustedFunctionPoint 22
5.1.4 internalFiles[3] for UnadjustedFunctionPoint 22
5.1.5 externalFiles[3] for UnadjustedFunctionPoint 22
5.1.6 GSC[14] for AdjustedFunctionPoint 22
5.1.7 language for selectLanguage 22
5.1.8 appType for appType 22
5.2 Data Output 23
5.2.1 Function point 23
5.2.2 Effort 23
5.2.3 Time development 23
5.2.4 Number of staff for a month 23
5.2.5 Number of staff for whole project 23
5.2.6 Cost estimation 23

6 REQUIREMENT TRACEABILITY 24

1 SCOPE

This section shall be divided into the following paragraphs:

1.1 Identification

System Number :
System Name : Crafting Cost Calculator
System Abbreviation : C^3

C^3 Number : N/A
C^3 Name : N/A
C^3 Abbreviation : N/A

1.2 System Overview

Purpose of the system:

The system name is “Crafting Cost Calculator(C^3).C^3 was developed for everyone who is working on a project that requires a number of months and cost estimations to deliver a complete project. It can calculate the cost of the production required to make the project and the duration of the project to be done. Other than that, it also will estimate the number of people that are required to develop the project.

The system shall:

- Accept the number of user input and outputs,user inquiries number of internal files and number of external files.
- Accept the rate entered by user for 14 General System Characteristic(GSC)
- Calculate the Unadjusted Function Points (UFP) , Adjusted Function Point (AFP) and Function Points(FP) based on the user rating for the General System Characteristic(GSC).
- Accept the language chosen by the user.
- Calculate KLOC based on language chosen and effort.
- Accept the application type chosen by the user.
- Calculate the effort and time development required based on the chosen type of application.
- Calculate the number of staff required for a month and the number of staff required for the whole project based on the effort and time development of the project.
- Calculate the cost estimation for the project.
- Display the result of all the calculations for the user.
- Save the result of the calculation in a file

1.3 Document Overview

This document describes the interaction between the actors with the C^3 .Analysis of the requirement applying to C^3 relies on ODAD UML notation using the WhiteStar UML tool and Star UML tool. Certain elements resulting from this analysis are presented in this document.

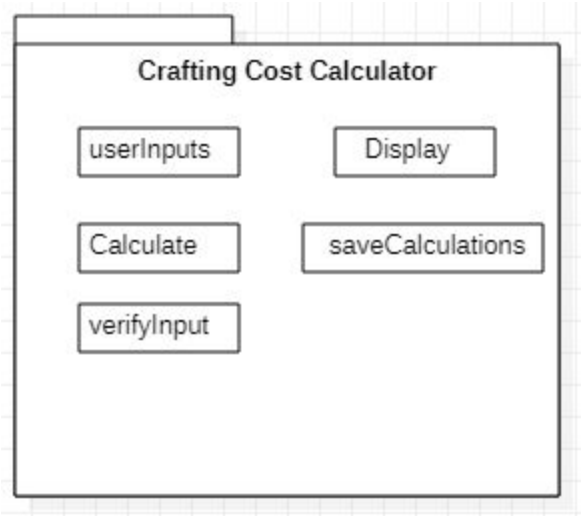
Chapter 1 Describe the Scope, identification, system overview, document overview.
Chapter 2 Describe the Referenced documents, government documents and non-government documents
Chapter 3 Describe the Preliminary Design
Chapter 4 Describe the Detail Design
Chapter 5 Describe the Data
Chapter 6 Describe Requirement Traceability

2 REFERENCED MATERIAL

- https://eusmartcities.eu/sites/default/files/201709/EIP_RequirementsSpecificationGLA_%20V2-5.pdf
- http://groups.umd.umich.edu/cis/course.des/cis375/projects/fp99/main.html
- https://www.javatpoint.com/cocomo-model
- https://ufuture.uitm.edu.my

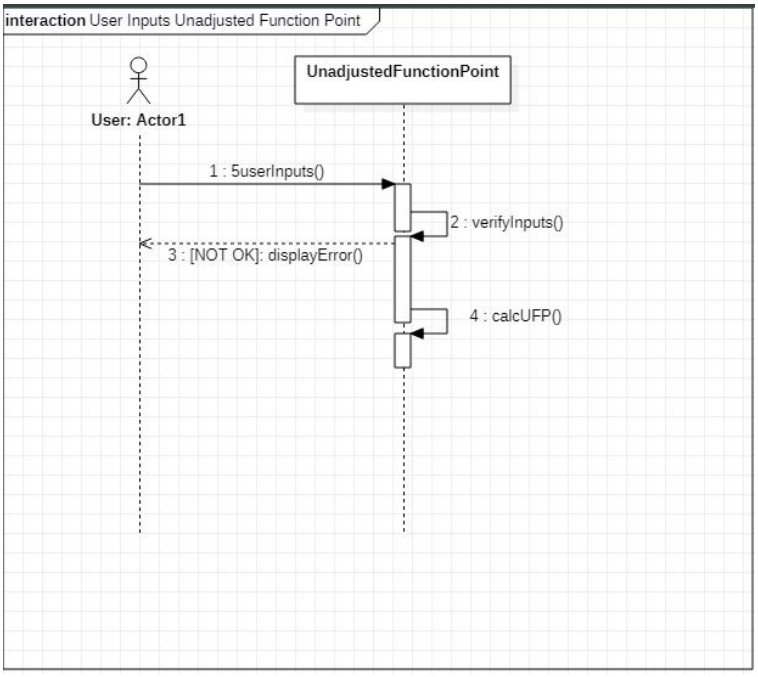
3 PRELIMINARY DESIGN

3.0 C^3 Overview

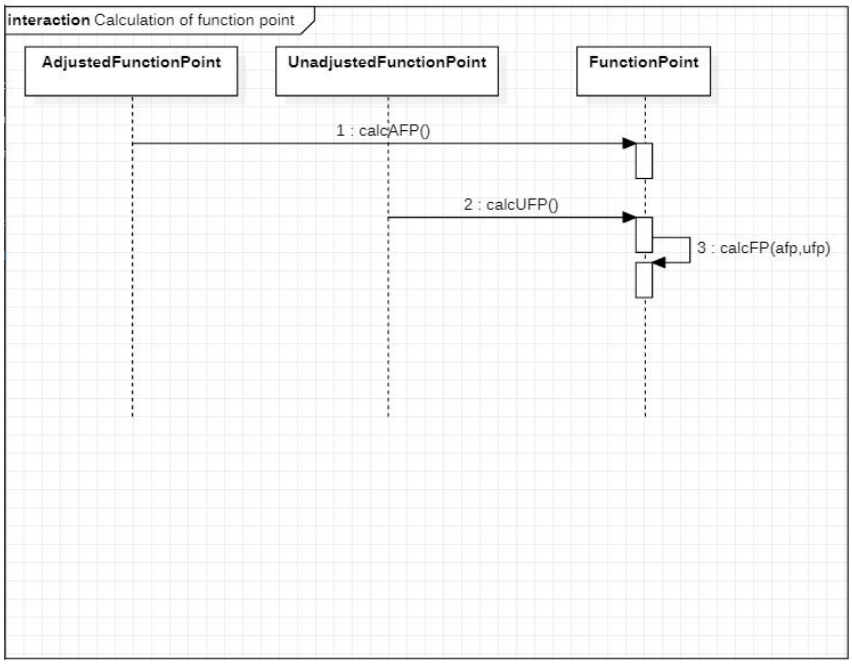


3.1 Sequence Diagram

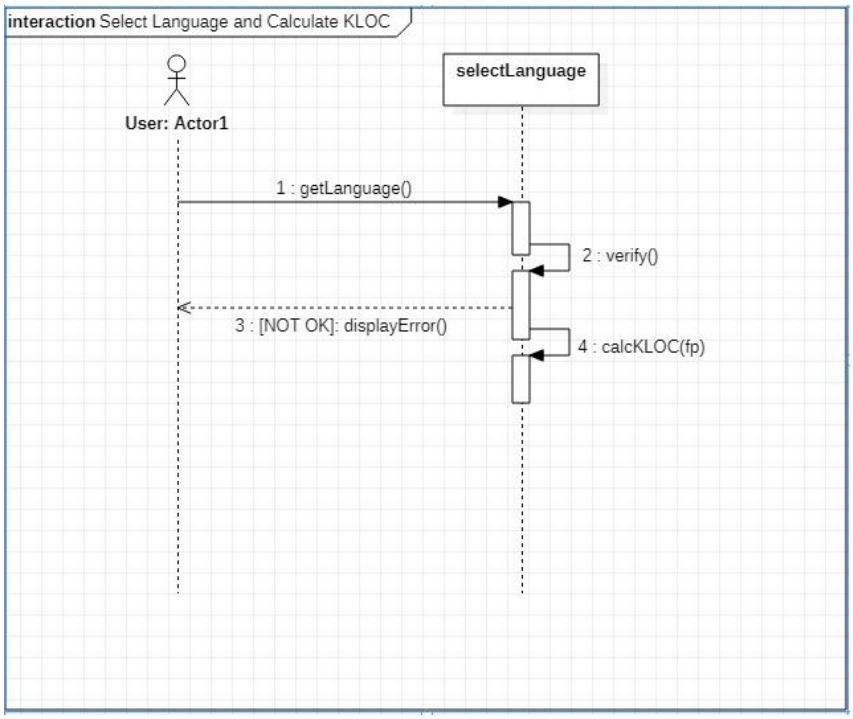
3.1.1 User Inputs Unadjusted Function Point



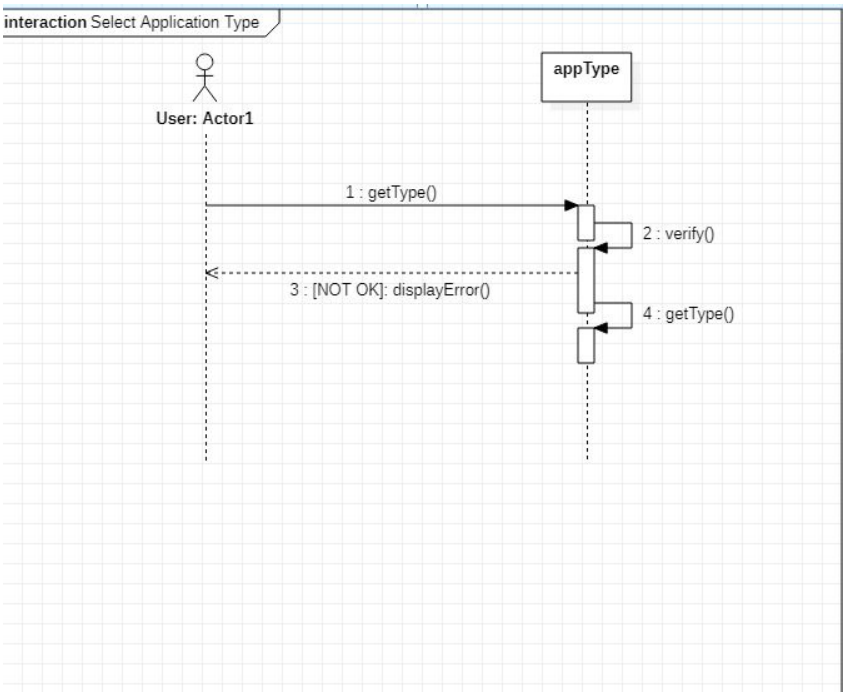
3.1.2 Calculation of Function Point



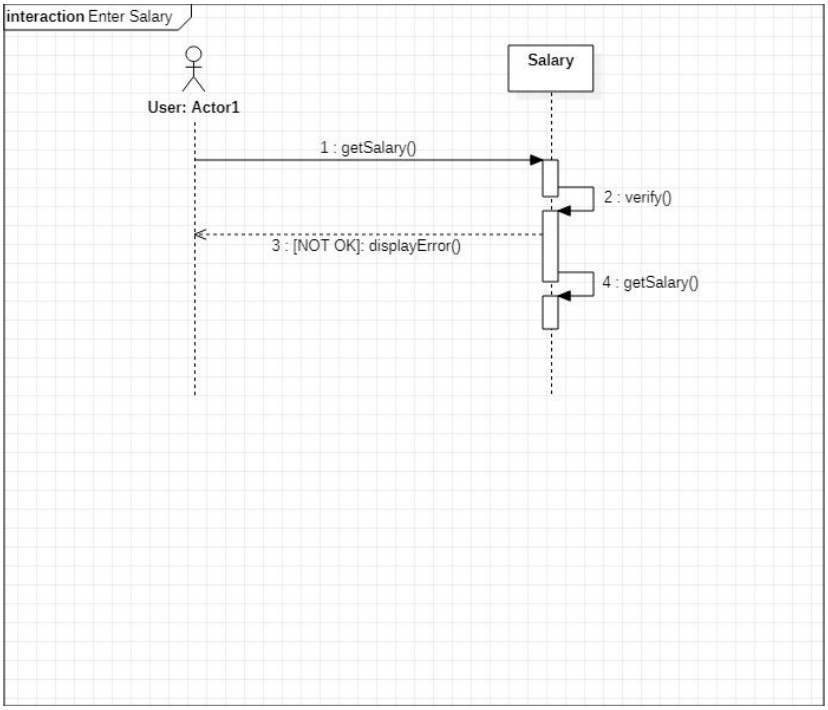
3.1.3 Select Language and Calculate KLOC



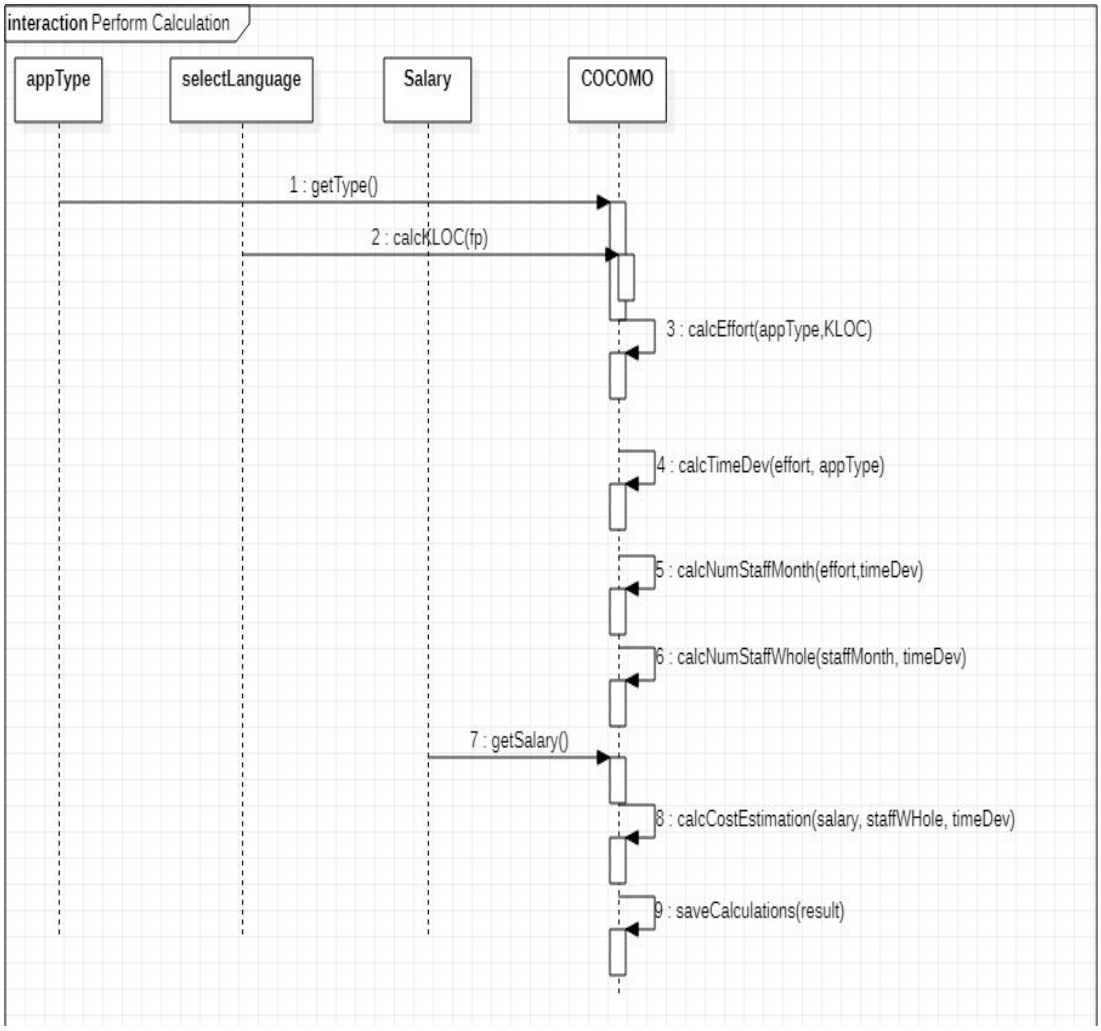
3.1.4 Select Application Type



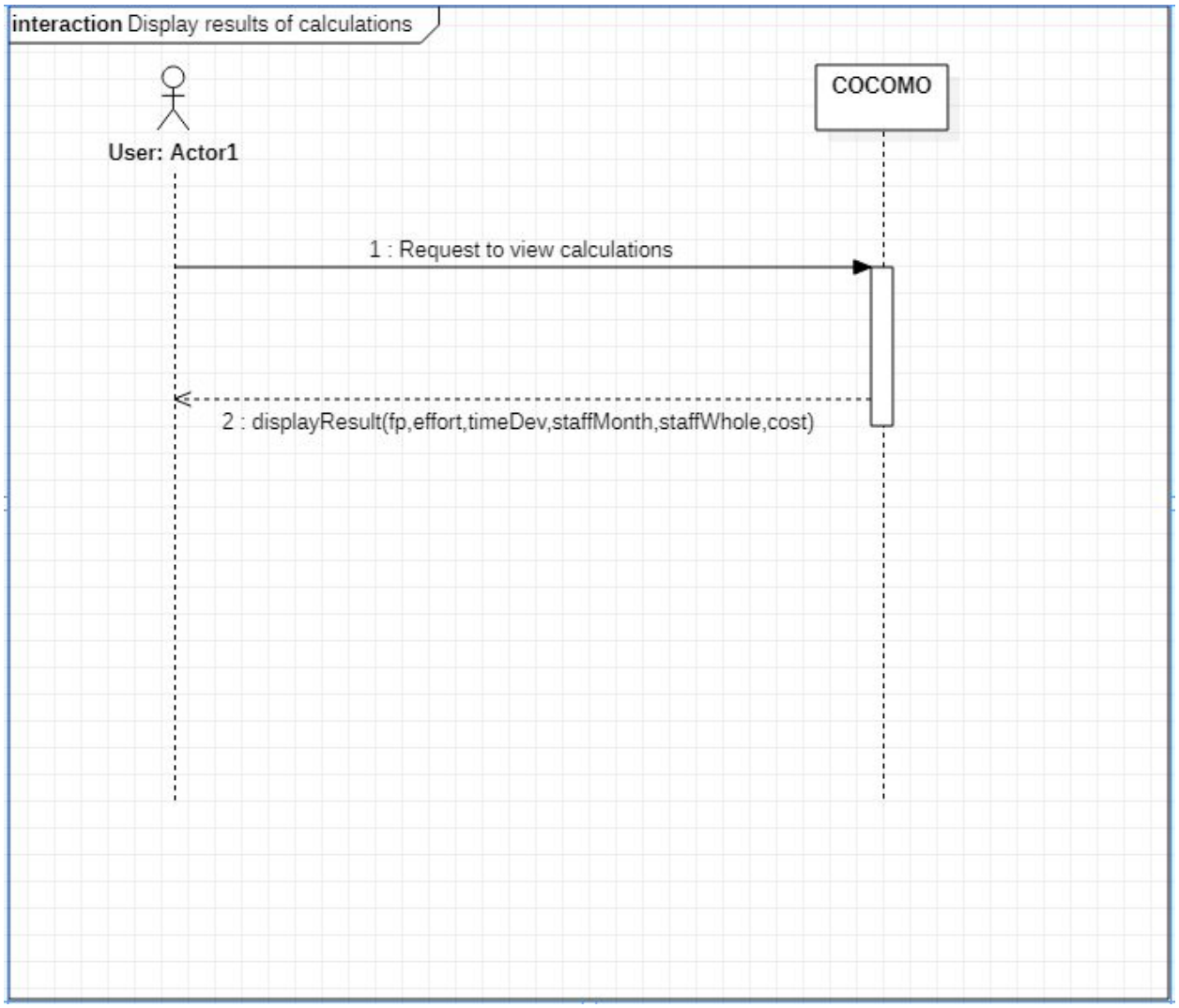
3.1.5 Enter Salary



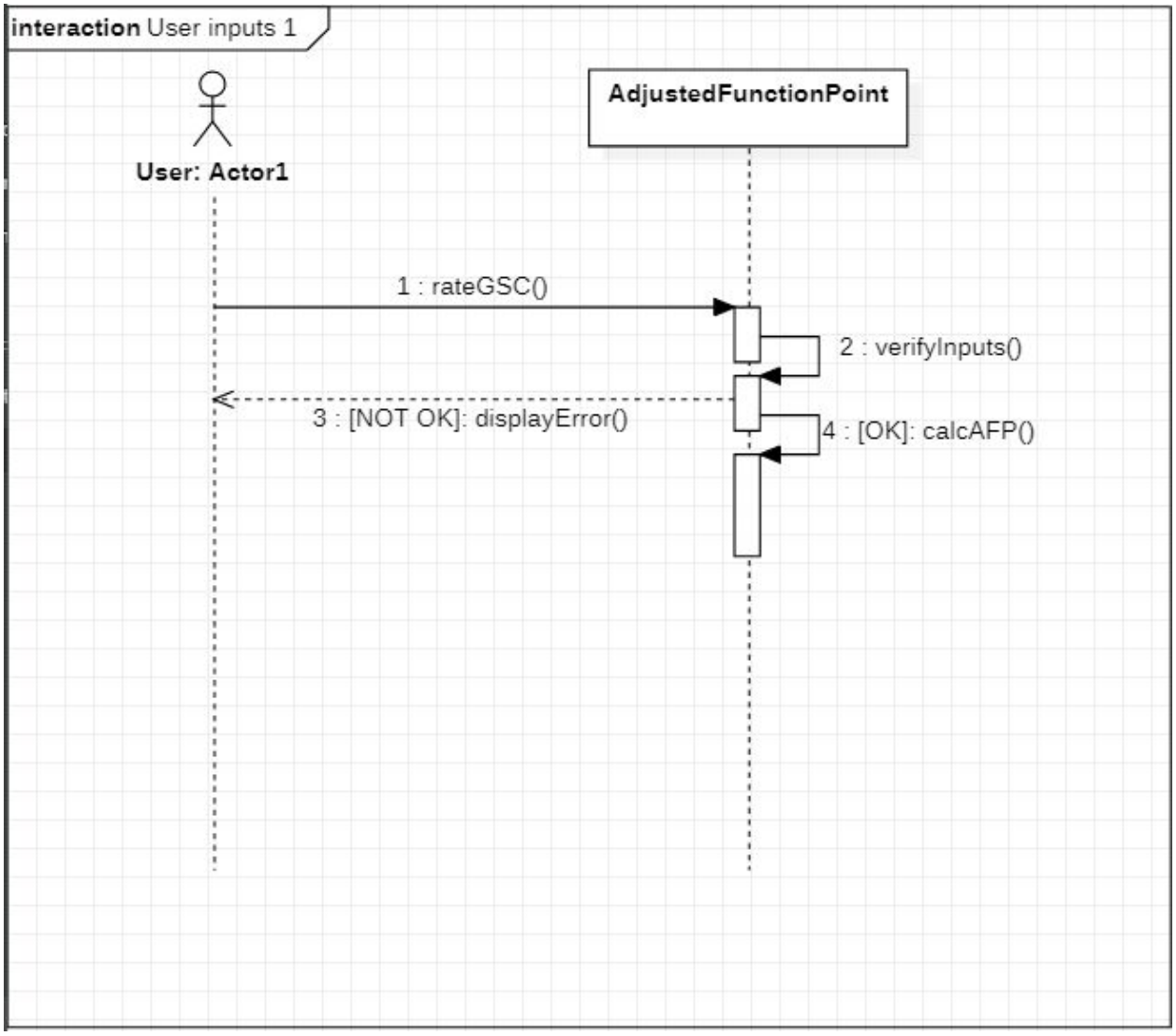
3.1.6 Perform Calculation



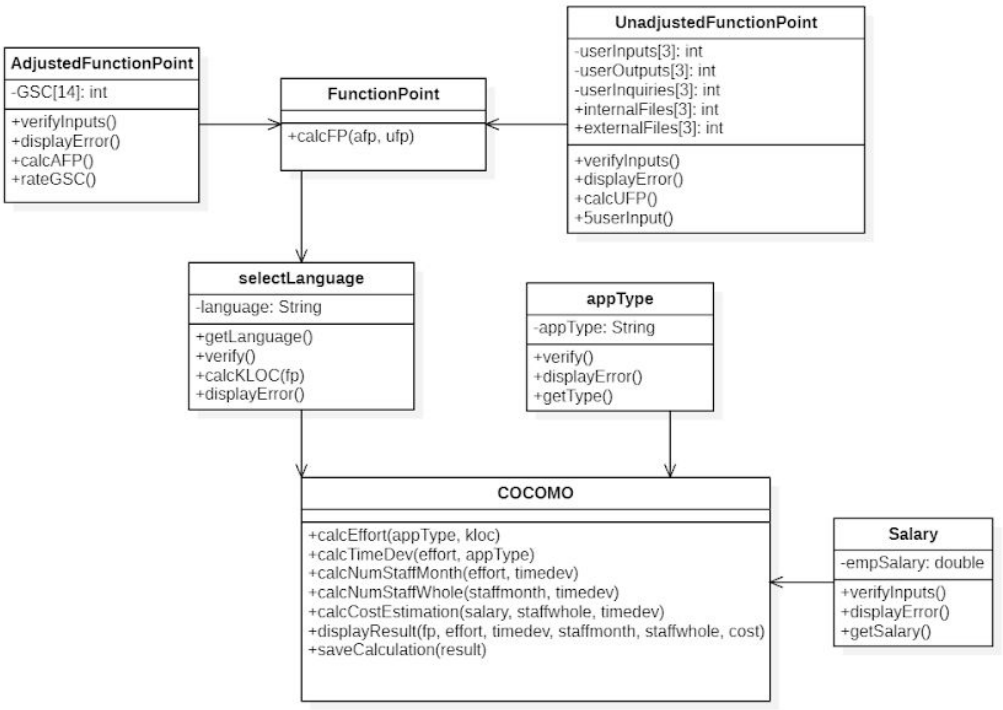
3.1.7 Display Results of Calculations



3.1.8 User inputs Adjusted Function Point



3.2 Class diagram



4 DETAILED DESIGN

4.1 C^3 Structural Diagram

4.1.1 5userInput()

Methods purpose: To receive value in unadjusted function point input by user
Argument: None
Return value: None
Algorithm:

```
BEGIN
        INPUT user inputs number for userInputs
        STORE the user input in userInputs[] variable
        INPUT user inputs userOutputs
        STORE the user input in userOutputs[] variable
        INPUT user inputs userInquiries
        STORE the user input in userInquiries[] variable
        INPUT user inputs internalFiles
        STORE the user input in internalFiles[] variable
        INPUT user inputs externalFiles
        STORE the user input in externalFiles[] variable
END
```

4.1.2 verifyInputs()

Methods purpose: To check existence value in unadjusted function point input by user
Argument: None
Return value: None
Algorithm:

```
BEGIN
        Set check to true
        FOR A IN 0 to 2
                IF userInputs[A] IS NULL
                        Set check to false
                END IF
        END FOR
        FOR A IN 0 to 2
                IF userOutputs[A] IS NULL
                        Set check to false
                END IF
        END FOR
        FOR A IN 0 to 2
                IF userInquiries[A] IS NULL
                        Set check to false
                END IF
        END FOR
```

```
        FOR A IN 0 to 2
            IF internalFiles[A] IS NULL
                Set check to false
            END IF
        END FOR
        FOR A IN 0 to 2
            IF externalFiles[A] IS NULL
                Set check to false
            END IF
        END FOR
        IF check IS false
            Call function displayError()
        END IF
    END
```

4.1.3 displayError()

Method purpose: To display error message for unadjusted function point existence if user does not enter required inputs
Argument: None
Return value: None
Algorithm:

```
    BEGIN
        Display “Please input value for unadjusted function point”
    END
```

4.1.4 rateGSC()

Methods purpose: To receive value in adjusted function point input by user
Argument: None
Return value: None
Algorithm:

```
    BEGIN
        INPUT user rate number for GSC
        STORE the user input in GSC[] variable
    END
```

4.1.5 calcUFP()

Method purpose: To calculate unadjusted function point
Argument: None
Return value: double ufp
Algorithm:

```
BEGIN
    EI=(userInputs[0]*3)+(userInputs[1]*4)+(userInputs[2]*6)
    EO=(userOutputs[0]*4)+(userOutputs[1]*5)+(userOutputs[2]*7)
    EQ=(userInquiries[0]*3)+(userInquiries[1]*4)+(userInquiries[2]*6)
    ILF=(internalFiles[0]*7)+(internalFiles[1]*10)+(internalFiles[2]*15)
    EIF=(externalFiles[0]*5)+(externalFiles[1]*7)+(externalFiles[2]*10)
    ufp=EI+EO+EQ+ILF+EIF
    RETURN ufp
END
```

4.1.6 verifyInputs()

Methods purpose: To check existence value in adjusted function point input by user
Argument: None
Return value: None
Algorithm:

```
BEGIN
    Set check to true
    FOR A IN 0 to 13
        IF GSC[A] IS NULL AND GSC[A] >=0 AND GSC[A] <=5
            Set check to false
        END IF
    END FOR
    IF check IS false
        Call function displayError()
    END IF
END
```

4.1.7 displayError()

Method purpose: To display error message for adjusted function point existence if user does not enter required inputs
Argument: None
Return value: None
Algorithm:
```
BEGIN
    Display “Please rate General System Characteristic(GSC)”
END
```

4.1.8 calcAFP()

Method purpose : To calculate Adjusted Function Point
Argument : None
Return value : double afp
Algorithm :
```
BEGIN
    Set total to zero
    FOR A IN 0 to 13
        total+=GSC[A]
    END FOR
    afp=0.65+(0.01*total)
    RETURN afp
END
```

4.1.9 calcFP(AFP, UFP)

Method purpose: To calculate Function Point
Argument: afp:double from calcAFP() while ufp:double from calcUFP()
Return value: double fp
Algorithm:
```
BEGIN
    fp= AFP * UFP
RETURN fp
END
```

4.1.10 verify()

Method purpose: To verify existence choice of language input by user.
Argument: None
Return value: None
Algorithm:

```
BEGIN
    Set check to true
    IF language IS NULL
        Set check to false
    END IF
    IF check IS false
        Call function displayError()
    END IF
END
```

4.1.11 displayError()

Method purpose: To display error message for language existence if user does not enter required inputs
Argument: None
Return value: None
Algorithm:

```
BEGIN
    Display “Choose Language”
END
```

4.1.12 getLanguage()

Method purpose: To get language from user input
Argument: None
Return value: String language
Algorithm:

```
BEGIN
    ENTER language
    RETURN language
END
```

4.1.13 calcKLOC(FP)

Method purpose: To calculate KLOC based on selected language
Argument: FP:int from calcFP(AFP,UFP)
Return value: int kloc
Algorithm:

```
BEGIN
    CASE  language  OF
        Access                  : loc = 38
        Basic                   : loc = 107
        C                       : loc = 128
        C++                     : loc = 53
        COBOL                   : loc = 107
        Delphi                  : loc = 29
        Java                    : loc = 53
        Machine Language        : loc = 640
        Visual Basic 5                  : loc = 29
    ENDCASE
    kloc= FP*loc/1000
    RETURN kloc
END
```

4.1.14 verify()

Method purpose: To verify existence choice of application type by user.
Argument: None
Return value: None
Algorithm:

```
BEGIN
    Set check to true
    IF appType IS NULL
        Set check to false
    END IF
    IF check IS false
        Call function displayError()
    END IF
END
```

4.1.15 displayError()

Method purpose: To display error message for language existence if user does not enter required inputs
Argument: None
Return value: None
Algorithm:
```
BEGIN
    Display “Choose Application Type”
END
```

4.1.16 getType()

Method purpose: To get application type from user input
Argument: None
Return value: String appType
Algorithm:
```
BEGIN
    ENTER appType
    RETURN appType
END
```

4.1.17 verifyInputs()

Methods purpose: To check existence value for salary input by user
Argument: None
Return value: None
Algorithm:
```
BEGIN
    Set check to true
    IF empSalary IS NULL
        Set check to false
    END IF
    IF check IS false
        Call function displayError()
    END IF
END
```

4.1.18 displayError()

Method purpose: To display error message for empSalary existence if user does not enter required inputs
Argument: None
Return value: None
Algorithm:
```
BEGIN
    Display “Please enter the salary for each employee”
END
```

4.1.19 getSalary()

Method purpose: To get language from user input
Argument: None
Return value: double empSalary
Algorithm:
```
BEGIN
    ENTER empSalary
    RETURN empSalary
END
```

4.1.20 calcEffort(appType, kloc)

Method purpose: To calculate the effort for the project based on the type of application.
Argument: appType:String from getType() and kloc:int from calcKLOC(fp)
Return value: double effort
Algorithm:
```
BEGIN
    CASE  appType  OF
        Organic             : a = 2.4
        Semidetached        : a = 3.0
        Embedded            : a = 3.6
    ENDCASE
    CASE  appType  OF
        Organic             : b = 1.05
        Semidetached        : b = 1.12
        Embedded            : b = 1.20
    ENDCASE
    effort= a(kloc) ^ b
RETURN effort
END
```

4.1.21 calcTimeDev(effort,AppType)

Method purpose: To calculate the time development based on the type of application
Argument: effort:double from calcEffort(appType,kloc)
Return value: double timedev
Algorithm: BEGIN

```
CASE  appType  OF
      Organic          : c = 2.5
      Semidetached     : c = 2.5
      Embedded         : c = 2.5
ENDCASE
CASE  appType  OF
      Organic          : d = 0.38
      Semidetached     : d = 0.35
      Embedded         : d = 0.32
ENDCASE
timedev= c(effort) ^d
RETURN timedev
END
```

4.1.22 calcNumStaffMonth(effort,timedev)

Method purpose: To calculate the number of staff required in a month.
Argument: effort:double from calcEffort(appType,kloc) and timedev:double from calcTimeDev(effort,appType)
Return value: double staffmonth
Algorithm: BEGIN

```
staffmonth= effort / timedev
RETURN staffmonth
END
```

4.1.23 calcNumStaffWhole(staffmonth,timedev)

Method purpose: To calculate the number of staff required for the whole project.
Argument: staffmonth:double from calcStaffMonth(effort,timedev) and timedev:double from calcTimeDev(effort,appType)
Return value: double staffwhole
Algorithm: BEGIN

```
staffwhole=  staffmonth / timedev
RETURN staffwhole
END
```

4.1.24 calcCostEstimation(salary,staffwhole,timedev)

Method purpose: To calculate the number of staff required for the whole project.
Argument:salary:double from getSalary(), staffwhole:double from calcStaffWhole(staffmonth,timedev) and timedev:double from calcTimeDev(effort,appType)
Return value: double cost
Algorithm: BEGIN
cost=salary*staffwhole*timedev
RETURN cost
END

4.1.25 displayResults(fp,effort,timedev,staffmonth,staffwhole,cost)

Method purpose: To display the results of the calculation
Argument: fp:double from calcFP(AFP,UFP) ,salary:double from getSalary(), staffwhole:double from calcStaffWhole(staffmonth,timedev) ,staffwhole:double from calcStaffWhole(staffmonth,timedev),timedev:double from calcTimeDev(effort,appType) and cost:double from calcCostEstimation(salary,staffwhole,timedev).
Return value: double cost
Return value: None
Algorithm: BEGIN
Display fp, effort ,salary, timedev,staffmonth,staffwhole ,cost
END

4.1.26 saveCalculations(result)

Method purpose: To save all the calculations done by the user.
Argument: result from displayResults(fp,effort,timedev,staffmonthstaffwhole,cost)
Return value: None
Algorithm: BEGIN
SAVE result in file
END

5 DATA

5.1 Data input

5.1.1 userInputs[3] for UnadjustedFunctionPoint

- Number start from 0
- Operator include multiplication and addition

5.1.2 userOutputs[3] for UnadjustedFunctionPoint

- Number start from 0
- Operator include multiplication and addition

5.1.3 userInquiries[3] for UnadjustedFunctionPoint

- Number start from 0
- Operator include multiplication and addition

5.1.4 internalFiles[3] for UnadjustedFunctionPoint

- Number start from 0
- Operator include multiplication and addition

5.1.5 externalFiles[3] for UnadjustedFunctionPoint

- Number start from 0
- Operator include multiplication and addition

5.1.6 GSC[14] for AdjustedFunctionPoint

- Number from 0 to 5
- Operator include multiplication and addition

5.1.7 language for selectLanguage

- Choose the language between Access, Basic, C, C++, COBOL, Delphi, Java, Machine Language and Visual Basic 5

5.1.8 appType for appType

- Choose the appType between Organic, Semidetached and Embedded

5.1.9 empSalary for Salary

- Number start from 0

5.2 Data Output

5.2.1 Function point

- The answer calculated after performing multiplication of ufp and afp

5.2.2 Effort

- The answer calculated after performing multiplication and exponent of KLOC, value of a and b based on appType

5.2.3 Time development

- The answer calculated after performing multiplication of effort, value of a and b based on appType

5.2.4 Number of staff for a month

- The answer calculated after performing division between effort and time development

5.2.5 Number of staff for whole project

- The answer calculated after performing division between number of staff in a month and time development

5.2.6 Cost estimation

- The answer calculated after performing multiplication of salary, number of staff for the whole project and time development.

6 REQUIREMENT TRACEABILITY

REQUIREMENT

Source	Allocated	Description
Lecture Notes of StarUML using UML	SRS_REQ_1 ● SRS_REQ_1_1 ● SRS_REQ_1_2 ● SRS_REQ_1_3 ● SRS_REQ_1_4 ● SRS_REQ_1_5	Enter inputs for unadjusted function point(ufp)
Lecture Notes of StarUML using UML	SRS_REQ_2 ● SRS_REQ_2_1 ● SRS_REQ_2_2 ● SRS_REQ_2_3 ● SRS_REQ_2_4 ● SRS_REQ_2_5 ● SRS_REQ_2_6 ● SRS_REQ_2_7 ● SRS_REQ_2_8 ● SRS_REQ_2_9 ● SRS_REQ_2_10 ● SRS_REQ_2_11 ● SRS_REQ_2_12	Enter inputs for adjusted function point(afp)
Lecture Notes of StarUML using UML	SRS_REQ_3 ● SRS_REQ_3_1 ● SRS_REQ_3_2 ● SRS_REQ_3_3 ● SRS_REQ_3_4 ● SRS_REQ_3_5 ● SRS_REQ_3_6 ● SRS_REQ_3_7 ● SRS_REQ_3_8 ● SRS_REQ_3_9 ● SRS_REQ_3_10 ● SRS_REQ_3_11 ● SRS_REQ_3_12 ● SRS_REQ_3_13 ● SRS_REQ_3_14 ● SRS_REQ_3_15	Calculate for unadjusted function point(ufp)

	• SRS_REQ_3_16	
Lecture Notes of StarUML using UML	SRS_REQ_4 • SRS_REQ_4_1	Calculate for adjusted function point(afp)
Lecture Notes of StarUML using UML	SRS_REQ_5 • SRS_REQ_5_1	Calculate for function point(fp)
Lecture Notes of StarUML using UML	SRS_REQ_6 • SRS_REQ_6_1	Select language
Lecture Notes of StarUML using UML	SRS_REQ_7 • SRS_REQ_7_1	Calculate KLOC
Lecture Notes of StarUML using UML	SRS_REQ_8 • SRS_REQ_8_1	Select application type
Lecture Notes of StarUML using UML	SRS_REQ_9 • SRS_REQ_9_1	Calculate effort based on type of application
Lecture Notes of StarUML using UML	SRS_REQ_10 • SRS_REQ_10_1	Calculate time development based on type of application
Lecture Notes of StarUML using UML	SRS_REQ_11 • SRS_REQ_11_1	Calculate number of staff required in a month
Lecture Notes of StarUML using UML	SRS_REQ_12 • SRS_REQ_12_1	Enter salary for a person per month
Lecture Notes of StarUML using UML	SRS_REQ_13 • SRS_REQ_13_1	Calculate number of staff required in the project
Lecture Notes of StarUML using UML	SRS_REQ_14 • SRS_REQ_14_1	Calculate cost estimation for the project
Lecture Notes of StarUML using UML	SRS_REQ_15 • SRS_REQ_15_1	Display result of calculation of the project for user
Lecture Notes of StarUML using UML	SRS_REQ_16 • SRS_REQ_16_1	Save result of calculations for user in file

DESIGN

Source	Allocated	Description
Lecture Notes of StarUML using UML	SRS_DES_1 ● SRS_DES_1_1 ● SRS_DES_1_2 ● SRS_DES_1_3 ● SRS_DES_1_4 ● SRS_DES_1_5	Enter inputs for unadjusted function point(ufp)
Lecture Notes of StarUML using UML	SRS_DES_2 ● SRS_DES_2_1 ● SRS_DES_2_2 ● SRS_DES_2_3 ● SRS_DES_2_4 ● SRS_DES_2_5 ● SRS_DES_2_6 ● SRS_DES_2_7 ● SRS_DES_2_8 ● SRS_DES_2_9 ● SRS_DES_2_10 ● SRS_DES_2_11 ● SRS_DES_2_12	Enter inputs for adjusted function point(afp)
Lecture Notes of StarUML using UML	SRS_DES_3 ● SRS_DES_3_1 ● SRS_DES_3_2 ● SRS_DES_3_3 ● SRS_DES_3_4 ● SRS_DES_3_5 ● SRS_DES_3_6 ● SRS_DES_3_7 ● SRS_DES_3_8 ● SRS_DES_3_9 ● SRS_DES_3_10 ● SRS_DES_3_11 ● SRS_DES_3_12 ● SRS_DES_3_13 ● SRS_DES_3_14 ● SRS_DES_3_15 ● SRS_DES_3_16	Calculate for unadjusted function point(ufp)
Lecture Notes of StarUML using UML	SRS_DES_4 ● SRS_DES_4_1	Calculate for adjusted function point(afp)

Lecture Notes of StarUML using UML	SRS_DES_5 ● SRS_DES_5_1	Calculate for function point(fp)
Lecture Notes of StarUML using UML	SRS_DES_6 ● SRS_DES_6_1	Select language
Lecture Notes of StarUML using UML	SRS_DES_7 ● SRS_DES_7_1	Calculate KLOC
Lecture Notes of StarUML using UML	SRS_DES_8 ● SRS_DES_8_1	Select application type
Lecture Notes of StarUML using UML	SRS_DES_9 ● SRS_DES_9_1	Calculate effort based on type of application
Lecture Notes of StarUML using UML	SRS_DES_10 ● SRS_DES_10_1	Calculate time development based on type of application
Lecture Notes of StarUML using UML	SRS_DES_11 ● SRS_DES_11_1	Calculate number of staff required in a month
Lecture Notes of StarUML using UML	SRS_DES_12 ● SRS_DES_12_1	Enter salary for a person per month
Lecture Notes of StarUML using UML	SRS_DES_13 ● SRS_DES_13_1	Calculate number of staff required in the project
Lecture Notes of StarUML using UML	SRS_DES_14 ● SRS_DES_14_1	Calculate cost estimation for the project
Lecture Notes of StarUML using UML	SRS_DES_15 ● SRS_DES_15_1	Display result of calculation of the project for user
Lecture Notes of StarUML using UML	SRS_DES_16 ● SRS_DES_16_1	Save result of calculations for user in file

TEST

Source	Allocated	Description
Lecture Notes of StarUML using UML	SRS_TEST_1 ● SRS_TEST_1_1 ● SRS_TEST_1_2 ● SRS_TEST_1_3 ● SRS_TEST_1_4 ● SRS_TEST_1_5	Enter inputs for unadjusted function point(ufp)
Lecture Notes of StarUML using UML	SRS_TEST_2 ● SRS_TEST_2_1 ● SRS_TEST_2_2 ● SRS_TEST_2_3 ● SRS_TEST_2_4 ● SRS_TEST_2_5 ● SRS_TEST_2_6 ● SRS_TEST_2_7 ● SRS_TEST_2_8 ● SRS_TEST_2_9 ● SRS_TEST_2_10 ● SRS_TEST_2_11 ● SRS_TEST_2_12	Enter inputs for adjusted function point(afp)
Lecture Notes of StarUML using UML	SRS_TEST_3 ● SRS_TEST_3_1 ● SRS_TEST_3_2 ● SRS_TEST_3_3 ● SRS_TEST_3_4 ● SRS_TEST_3_5 ● SRS_TEST_3_6 ● SRS_TEST_3_7 ● SRS_TEST_3_8 ● SRS_TEST_3_9 ● SRS_TEST_3_10 ● SRS_TEST_3_11 ● SRS_TEST_3_12 ● SRS_TEST_3_13 ● SRS_TEST_3_14 ● SRS_TEST_3_15 ● SRS_TEST_3_16	Calculate for unadjusted function point(ufp)
Lecture Notes of StarUML using UML	SRS_TEST_4 ● SRS_TEST_4_1	Calculate for adjusted function point(afp)

Lecture Notes of StarUML using UML	SRS_TEST_5 ● SRS_TEST_5_1	Calculate for function point(fp)
Lecture Notes of StarUML using UML	SRS_TEST_6 ● SRS_TEST_6_1	Select language
Lecture Notes of StarUML using UML	SRS_TEST_7 ● SRS_TEST_7_1	Calculate KLOC
Lecture Notes of StarUML using UML	SRS_TEST_8 ● SRS_TEST_8_1	Select application type
Lecture Notes of StarUML using UML	SRS_TEST_9 ● SRS_TEST_9_1	Calculate effort based on type of application
Lecture Notes of StarUML using UML	SRS_TEST_10 ● SRS_TEST_10_1	Calculate time development based on type of application
Lecture Notes of StarUML using UML	SRS_TEST_11 ● SRS_TEST_11_1	Calculate number of staff required in a month
Lecture Notes of StarUML using UML	SRS_TEST_12 ● SRS_TEST_12_1	Enter salary for a person per month
Lecture Notes of StarUML using UML	SRS_TEST_13 ● SRS_TEST_13_1	Calculate number of staff required in the project
Lecture Notes of StarUML using UML	SRS_TEST_14 ● SRS_TEST_14_1	Calculate cost estimation for the project
Lecture Notes of StarUML using UML	SRS_TEST_15 ● SRS_TEST_15_1	Display result of calculation of the project for user
Lecture Notes of StarUML using UML	SRS_TEST_16 ● SRS_TEST_16_1	Save result of calculations for user in file