



The University of Calgary
Department of Electrical and Computer Engineering

SENG 521 - Software Reliability and Software Quality
Project Assignments

Behrouz Far
Fall 2011
(Revision 1.00)

Assignment no. 1: Software Project Size Workshop

Delivery Date : October 3rd, 2011 (Monday), 4:30 PM
Grade : %20 of the total labs mark.

This assignment is a group assignment. The team is composed of 2 to 3 members.

1. Purpose: Measuring Function Point

In this assignment you are asked to measure the function point for a typical software system. This will help you reinforce the concepts studied during the course.

2. Background:

The overall objective is to determine adjusted function point count for a software system. There are several steps necessary to accomplish this. The actual sequence or order of steps is not necessary. Many counters will complete step 5 throughout the entire count – gathering information as they go;

1. Determine type of function point count
2. Determine the application boundary
3. Identify and rate transactional function types to determine their contribution to the unadjusted function point count.
4. Identify and rate data function types to determine their contribution to the unadjusted function point count.
5. Determine the value adjustment factor (VAF)
6. Calculate the adjusted function point count.

The unadjusted function point (UFP) count is determined in steps 3 & 4. It is not important if step 3 or step 4 is completed first. In GUI and OO type applications it is easy to begin with step 3.

The final function point count (adjusted function point count) is a combination of both unadjusted function point count (UFP) and the general system characteristics (GSC's).

3. Scenario:

Suppose that your group is a software development team assigned to build one of the projects listed on suggested project list (Appendix Page). You are supposed to measure the Function Point for this project.

You may proceed as follows:

1. Start with defining the requirements for your project. Note that you should proceed to the extent that the detailed requirements are sufficient to measure the function points.
2. Read relevant sections of the “Function Points Analysis Training Course” (110 Pages) (by David Longstreet, David@SoftwareMetrics.com, www.SoftwareMetrics.com). This document is downloadable from course web page. The document is a step-by-step guide to measure FP. Follow the steps mentioned there. Many of the concepts can only be comprehended and decided through the discussion among the team members.

4. Deliverables:

Your deliverable (report) should consist of:

1. Title page (project title, delivery date, team members’ name, student ID and email).
2. One page executive summary.
3. Project specification (1-2 pages) clearly mentioning what is included (system functions, modules) and support systems (DB server, Web server, etc., if needed)
4. FP measurement results for the different modules and overall project (no page limit).
5. One page of conclusions and comments including
 - a. Conclusions related to the project (summarizing the results)
 - b. Your comments related to this assignment (Your own viewpoint related to the assignment. What do you think about this assignment? Was it useful? How can it be improved? etc.)

5. Evaluation:

Evaluation of the reports is based on the effort the team has put into the project and is measured by “originality”, “correctness” and “completeness” of the project.

Notes:

1. Submit one report per team.
2. Write your own work only. Quotes from articles, textbooks and online materials must be properly referenced. Reports are evaluated comparatively with the current and previous years’ reports and other online materials. In case of proven plagiarism the assignment will be marked zero and suspected cases will be reported for further investigation. Please read the section in the University Calendar on policies described in the Schulich School of Engineering Advising Syllabus available at: <http://schulich.ucalgary.ca/undergraduate/advising>

Assignment no. 2: Software Project Time and Effort Workshop

Deadline: October 24th, 2011 (Monday), 4:30 PM

Grade: %20 of the total lab marks

This assignment is a group assignment. The team is composed of 2 to 3 members.

1. Purpose: Measuring Effort Using Software Metrics Tools

In this assignment you are asked to measure software cost and effort for a realistic project using COCOMO II tool. This will help you reinforce the concepts studied during the course.

2. Tools:

The tool based on Constructive Cost Model (COCOMO II).

3. Background:

COCOMO II includes three-stage series of models:

1. The earliest phases will generally involve prototyping, using the *Application Composition model* capabilities.
2. The next phases will generally involve exploration of architectural alternatives or incremental development strategies. To support these activities, COCOMO II provides an early estimation model called the *Early Design model*.
3. Once the project is ready to develop, it should have a life-cycle architecture, which provides more accurate information on cost driver inputs, and enables more accurate cost estimates. To support this stage, COCOMO II provides the *Post-Architecture model*.

The *Application Composition model* is used in estimating early stage issues, where source code is not available. It uses counts entities such as user interfaces, software/system interaction to measure Object Points (OP) and then use the following relation to derive the effort (E):

$$E = OP / PROD$$

where

OP is the object point; *PROD* is the productivity rate defined below

Developers' experience and capability	Very Low	Low	Nominal	High	Very High
PROD	4	7	13	25	50

The *Early Design model* is used to evaluate alternative software/system architectures and concepts of operation. An unadjusted function point count (UFC) is used for sizing. This value is converted to KLOC. The Early Design model equation is:

$$E = 2.45 \times KLOC \times EAF$$

The KLOC may be computed using the COCOMO II tool or by converting the object point to KLOC.

The effort adjustment factor (*EAF*) is calculated as using 7 cost drivers.

	Cost Driver	Description	Counterpart Combined Post-Architecture Cost Driver
1	RCPX	Product reliability and complexity	RELY, DATA, CPLX, DOCU
2	RUSE	Required reuse	RUSE
3	PDIF	Platform difficulty	TIME, STOR, PVOL
4	PERS	Personnel capability	ACAP, PCAP, PCON
5	PREX	Personnel experience	AEXP, PEXP, LTEX
6	FCIL	Facilities	TOOL, SITE
7	SCED	Schedule	SCED

The *Post-Architecture model* is used during the actual development and maintenance of a product. The Post-Architecture model includes a set of 17 cost drivers and a set of 5 scale factors determining the projects scaling component. The Post-Architecture model equation is:

$$E = 2.45 \times (KLOC)^b \times EAF \quad b = 0.91 + 0.01 \sum_{j=1}^5 SF_j$$

The *SF* and *EAF* can be calculated using the default values given by the COCOMO II tool.

Additional documents related to COCOMO II tool and user manual can be downloaded from the following URLs:

1. Original manuals and documents for COCOMO Project (document repository)

<http://sunset.usc.edu/research/COCOMOII/index.html>

2. COCOMO II Model Manual (local copy)

<http://www.enel.ucalgary.ca/People/far/Lectures/SENG421/PDF/COCOMO/modelman.pdf>

3. COCOMO II User Manual (local copy)

<http://www.enel.ucalgary.ca/People/far/Lectures/SENG421/PDF/COCOMO/userman.pdf>

4. Scenario:

Suppose that your group is a software development team assigned to build one of the projects listed on

suggested project list (Appendix Page). You are supposed to measure the total effort required to build the project.

You may proceed as follows:

1. Start with defining the requirements for your project. Note that you should proceed to the extent that the detailed requirements are sufficient to measure the object points (or function points).
2. Read the COCOMO II user manual (available as a help file on the Windows system and also as a separate PDF file) and run the tool. Start with defining a new project and define modules, that you defined in Step 1, and proceed through the 3 phases to refine your estimation of effort. Note that sometimes you may need to revise earlier stage decisions and repeat estimation to avoid conflicts.
3. Try to change a few parameters (like project duration, risk, etc.) and how they may affect your original estimations.
4. Use the reporting tool of COCOMO II to generate report pages.

5. Deliverables:

Your deliverable (report) should consist of:

1. Title page (project title, delivery date, team members' name, student ID and email).
2. One page executive summary.
3. Project specification (1-2 pages) clearly mentioning what is included (system functions, modules) and support systems (DB server, Web server, etc., if needed). This can be the same as Assignment 1.
4. Estimation results for the 3 phases of the model (no page limit).
5. One page of conclusions and comments including summarizing the results

6. Evaluation:

Evaluation of the reports is based on the effort the team has put into the project and is measured by "originality", "correctness" and "completeness" of the project.

Notes:

1. Submit one report per team.
2. Write your own work only. Quotes from articles, textbooks and online materials must be properly referenced. Reports are evaluated comparatively with the current and previous years' reports and other online materials. In case of proven plagiarism the assignment will be marked zero and suspected cases will be reported for further investigation. Please read the section in the University Calendar on policies described in the Schulich School of Engineering Advising Syllabus available at: <http://schulich.ucalgary.ca/undergraduate/advising>

Assignment no. 3: Defining Necessary Reliability Workshop

Delivery Date : November 6th, 2011 (Monday), 4:30 PM

Grade : %20 of the total labs mark.

This assignment is a group assignment. The team is composed of 2 to 3 members.

Similar to many other projects, you start with the requirements analysis and then design. However, a big difference is that you must first finalize the requirements by defining what functions the system offers (functional requirements) together with what reliability criteria must be satisfied (non-functional requirements). In this assignment the focus is on the latter, which is usually ignored in conventional requirements analysis. To do so you must concentrate on defining the necessary reliability for the product. You can have a better understanding of the reliability centered requirements by answering the following questions:

1. How will you define failure for the product (by severity class, as product-specific as possible)?

Answer in detail what the failures that you expect the system handle are and how severe they may be.

2. Choose the natural or time unit you will use for the product.

Explain why you select such unit(s) and how to convert one to the other if you have more than one measurement unit. Time unit is useful when the operations of the system can be measured with respect to time, e.g., a server must be up 98% of running time. Other units, such as number of transactions, jobs, etc., may be used otherwise.

3. Set the product failure intensity objective (FIO).

Define FIO target for your product. If impossible to define it now leave it open for a later stage.

4. Find the expected product acquired failure intensity, based on the failure intensities of the hardware and acquired software components (if applicable).

Check whether your software should run on a specific platform (IBM PC + Windows xx, or SUN Sparc + Solaris 2.x, Linux, etc.), find the failure intensity for that platform if you can. Check if the maker gives the FI for its product. If not, try to guess or use the values given in the textbook.

5. Determine the product-developed software failure intensity objective (if applicable).

Straightforward by subtracting platform's FIO from target FIO. However, if you start by dividing

your project into a number of independent modules, you should define FIO for each module and then add them up.

6. How will you balance fault prevention, fault-removal, and fault tolerance strategies?

Consider your specific requirements for reliability, timely delivery, and cost and allocate your resource percentages among the following six activities. You may add other activities to the list if they are significant, but the percentages must total 100.

- Fault prevention

Requirements reviews; Design reviews

- Fault removal

Code inspection; Unit test; System test

- Fault tolerance (design for fault tolerance)

If you select fault prevention as the main strategy, you should demonstrate how you could adopt the ISO 9000-3 guidelines in your project. If you select fault tolerance, you should start your design to account for fault tolerance, i.e. redundant modules, etc.

Write a summary of your selected strategies and how you are going to implement them.

Deliverables:

- 1) Project specification (1-2 pages) clearly mentioning what is included (system functions, modules) and support systems (DB server, Web server, etc., if needed). This can be the same as Assignment 1.
- 2) Use-cases and use-case specification document (for object-oriented software) or alternatively, design documents (for non-object oriented software).
- 3) Reliability requirements document containing answers to the above questions.

Notes:

1. Submit one report per team.
2. Write your own work only. Quotes from articles, textbooks and online materials must be properly referenced. Reports are evaluated comparatively with the current and previous years' reports and other online materials. In case of proven plagiarism the assignment will be marked zero and suspected cases will be reported for further investigation. Please read the section in the University Calendar on policies described in the Schulich School of Engineering Advising Syllabus available at: <http://schulich.ucalgary.ca/undergraduate/advising>

Assignment no. 4: Developing Operational Profiles

Delivery Date: November 21st, 2011 (Monday), 4:30 PM

Grade: %20 of the total labs mark.

This assignment is a group assignment. The team is composed of 2 to 3 members.

Here we go one step ahead and define the modes and profile of operation for the developed software. You should also finalize the analysis and design and start developing your code at this stage. The code will be used later during the running test phase (Assignment 5). You should answer the following questions and complete the following tasks.

1. What factors are likely to yield different operational modes for your system?

Construct a list of possible operational modes. Reduce the list to a set of operational modes that are significantly different from each other and frequently executing. Limiting the number of operational modes is important; otherwise, you may create excessive work for yourself later in the software reliability engineering process.

2. Pick one operational mode of the product of broad scope and identify its operation initiators.

The operational mode that you select may be usually the one executed the most in the field. And operation initiators may be among user types, external systems, own system.

3. Decide between tabular or graphical representation for the operational profile.

4. If you are using the tabular representation, list the operations for the operational mode you have selected.

Consider the different initiators to generate the list for each operational mode.

5. If you are using the graphical representation, draw the network.

For large network, depending on the size of operations and attributes, you may limit the size to not more than 10 operations or attributes or draw part of the network.

6. How will you determine occurrence rates and occurrence probabilities for the operational profile of the operational mode you have selected?

Is it possible to estimate them? Can you obtain any data from already available resources, log files, etc? Can you measure or collect data?

In either case represent the occurrence probabilities using simple numbers.

Deliverables:

- 1) The documented operational mode(s) and operational profile(s). This document provides answer to the above mentioned questions and completion of the above mentioned tasks.
- 2) Detailed design documents (collaboration or sequence diagrams, class diagrams for the analysis and design phases for object-oriented software and equivalent documents for non-object oriented software.)

Notes:

1. Submit one report per team.
2. Write your own work only. Quotes from articles, textbooks and online materials must be properly referenced. Reports are evaluated comparatively with the current and previous years' reports and other online materials. In case of proven plagiarism the assignment will be marked zero and suspected cases will be reported for further investigation. Please read the section in the University Calendar on policies described in the Schulich School of Engineering Advising Syllabus available at:
<http://schulich.ucalgary.ca/undergraduate/advising>

Assignment no. 5: Prepare, Execute Test and Release Workshop

Delivery Date: December 5th, 2011 (Monday), 4:30 PM.

Grade: %20 of the total labs mark.

This assignment is a group assignment. The team is composed of 2 to 3 members.

At this stage you identify how many test cases you need; how much time you want to spend on testing the product; and finally define your test-cases and develop your test suite. You must perform the following tasks:

1) Estimate the number of test cases you need to prepare

If this is the first release, estimate the total number of test cases required. If this is not the first release, estimate the number of new test cases for this release. Your estimation can be based on time and/or cost criteria.

2) Allocate the number of test cases among the associated and sub-systems to be tested.

How many new test cases should be created if this is not the first release? How many test cases will go to the other infra-structure systems? How many to the OS? How many will be assigned to the acquired and developed products?

3) Allocate the number of test cases for the developed product to its operations

Make sure you identify critical operations properly and assign enough test cases to them. New test cases will naturally be assigned to new operations in the first and subsequent releases.

4) Specify the test profile for one of the operational modes

If you have more than one operational mode (e.g., prime time, peak, off time, etc.) select the one most frequently used and specify its test profile.

5) Document your test cases

You need to document them all for your own use. Include a few (2-5) cases in your report.

6) Determine how you will divide hours of test among the associated systems you have defined

7) Determine the number of hours you will devote to feature, regression (if needed), and load test for the product

8) Allocate the hours of load test among the operational modes.

9) Run your program and execute (run) the test for one operational mode

You may need to write some scripts to select cases at random, run them and record the output. To do so you must have your code be ready and stable.

10) Use a reliability growth tool, such as CASRE to verify the actual reliability growth

11) Determine whether you have achieved your target and may/may not release your software

If the answer is yes, explain why. If the answer is no what do you suggest? Reduce functions? Postpone release? Add additional test cases? Do more testing? Drop the project?

Deliverables:

- 1) The documented test profile(s). This document provides answer to the above mentioned questions and completion of the above mentioned tasks.
- 2) Report on the failures detected for the operational mode the test was running.
- 3) CASRE reliability growth report.
- 4) Release recommendation document.

Notes:

1. Submit one report per team.
2. Write your own work only. Quotes from articles, textbooks and online materials must be properly referenced. Reports are evaluated comparatively with the current and previous years' reports and other online materials. In case of proven plagiarism the assignment will be marked zero and suspected cases will be reported for further investigation. Please read the section in the University Calendar on policies described in the Schulich School of Engineering Advising Syllabus available at: <http://schulich.ucalgary.ca/undergraduate/advising>

Appendix: Suggested Project List (SENG 521)

Below is a list of suggested projects for SENG 521. This list may be used as a reference or as the last choice. **You are encouraged to define your own project (students may receive additional 2% bonus points for their originality of the project).**

1) Transit tracking system

Description: Calgary's Transit System can help a lot of student's from freezing out in the winter. Currently the website has a scheduling system that probably works out of a database backend and shows the timings of a bus at different stops. Your job is to write an application that will give "real time data" as to where the bus is at any given time. It would be nice if the program could be intelligent enough to know how long it takes the bus to reach from one stop to another given past experiences.

2) Gas price tracker

Description: With the high gas prices a lot of websites are posting gas prices across Calgary. These prices are basically entered by individual users. When a user goes to such a website the site lists the highest and lowest 10 prices of the day. However this does not say much. Users want to know how much they are saving. If the user has to drive 30 km to get the lowest price it may or may not be worth it. It would be very helpful if the user could get a more intelligent answer as to how much will he/she will save going to that gas station from the user's origin.

3) Document management system

Description: Executives in companies share a lot of documents. Documents that were written and viewed years ago sometimes need to come alive when someone needs to view it. Generally the practice is to keep these documents on a share drive on the network. However when documents are saved in the same place and the share drive grows, the task of browsing through all the documents one by one is a very hard one. It would save executives a lot of agony if they could somehow search for documents. Keep in mind the possibility of overwriting of files by different users.

4) Course management system for students

Description: Some students are not sure whether or not they can handle a certain course load given their schedule and other commitments. Generally when they sign up for courses all it says is when the course is and how long it lasts and whether or not it conflicts with other courses that he/she is signed up for. It would help if the application is intelligent enough to make suggestion as to whether a certain course load will be difficult, easy, impossible etc. A lot of factors can be considered. It is up to the creative programmer to decide which

factors should be used and what kind of information should be presented to the user.

5) Hotel reservation system

Description: Write a specification for a hotel booking system which keeps track of current guests, available rooms, future bookings and payment of accounts. Include a part for room service orders which keeps track of the orders, bills them to the appropriate room, and will show the total sales for a given period.

6) Hospital

Description: Write a specification to manage a hospital system. There will be a waiting list of patients needing different treatment e.g. surgical, medical. The bed state should be determined and if beds are available, the next appropriate patients on the list notified. Nurses should be allocated to wards depending on ward sizes, what type of nursing is needed, operating schedules etc.

7) Library

Description: Write a specification to organize the lending of library books. Only so many may be borrowed, and if a user has the maximum one must be returned before another is borrowed. Books are subject to recall. The librarian will have special borrowing privileges. There may be several copies of a particular book on the shelf, or they may all be out on loan.

8) Online registration system

Description: Write a specification for an online registration system in which students could add courses, drop courses, view their courses, view all courses, look up their grades for courses, etc. Teachers could view the students in their courses, and assign grades online. Administrators could create, read, update and delete course info, including assigning teachers to courses, assigning courses to classrooms, etc. The system could have some intelligence, such as denying registration to students with unpaid fees, doing prerequisite checks, etc.

9) Realtor system

Description: Write a specification for a system to manage a real estate catalogue. It should keep track of properties with their amenities, selling price or rental etc, and be able to be queried by potential customers with different priorities. Take into account constraints on the property such as offers made, terms and conditions, etc.

10) Electronic shop

Description: Write a specification for an electronic shop that allows several vendors set it up and sell their

goods to several customers on the Web.

11) Software Design documents management system

Description: Write a specification for a software design documentation repository system for saving several forms of documents from various phases of a software development project and indexing them for future use, using data-centered structure. The documents may be either unstructured (pure text) or structured (Word, PDF, Excel, etc.). The system should answer to queries from users via intranet.

12) Other

Define your own project. Consult with course instructor and/or TAs regarding the depth and scope of the project and obtain their approval.