

Michael J. Sepcot (10294350)

CS 537 – Software Metrics

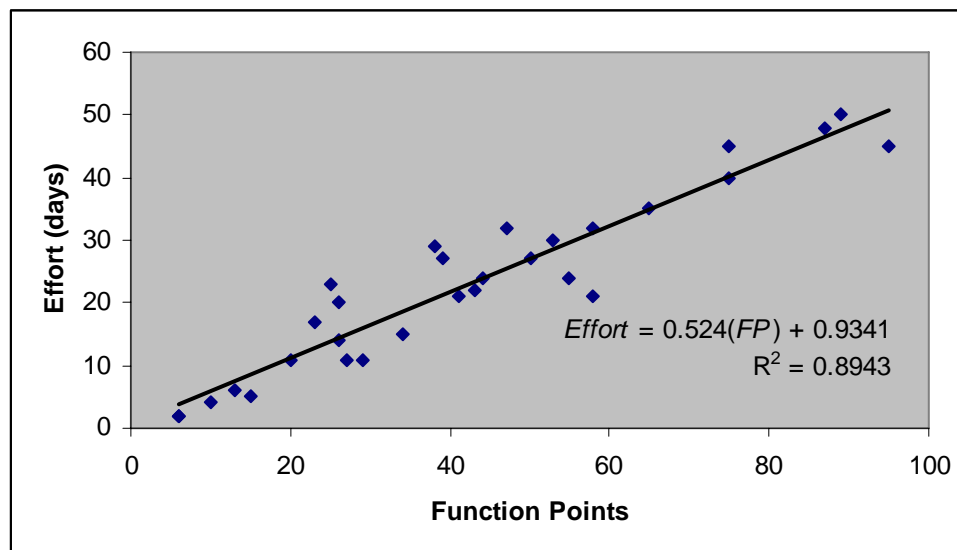
November 24, 2005

PROBLEM #1 (20 points)

Given the following data collected in a hypothetical software company:

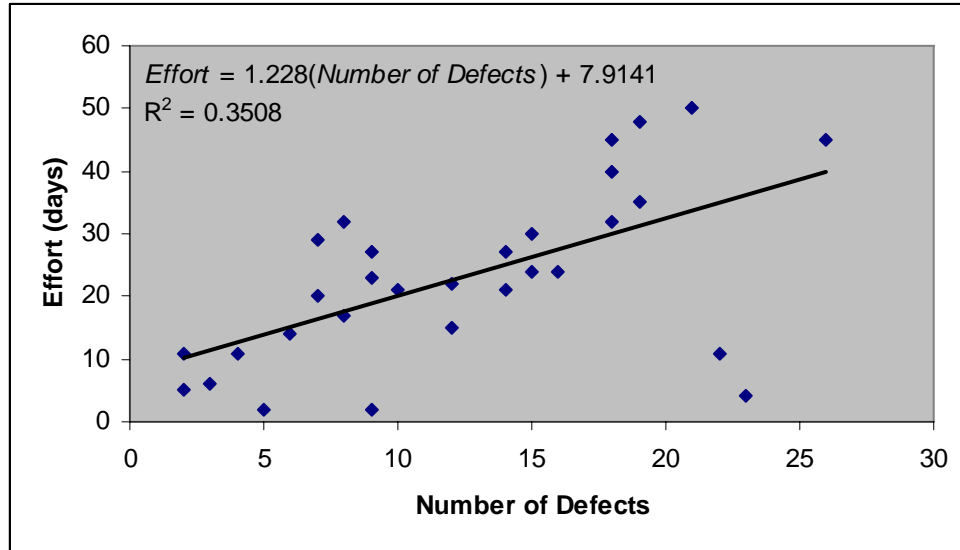
Module Number	Function Points	Effort (days)	Number of Defects	Module Number	Function Points	Effort (days)	Number of Defects
1	39	27	9	16	75	40	18
2	27	11	22	17	50	27	14
3	13	6	3	18	55	24	16
4	26	14	6	19	25	23	9
5	75	45	18	20	58	21	14
6	43	22	12	21	6	2	5
7	20	11	2	22	29	11	4
8	26	20	7	23	58	32	18
9	53	30	15	24	95	45	26
10	6	2	9	25	47	32	8
11	89	50	21	26	10	4	23
12	44	24	15	27	15	5	2
13	23	17	8	28	87	48	19
14	65	35	19	29	38	29	7
15	34	15	12	30	41	21	10

Use linear regression analysis to develop two prediction systems that can be used with a **significant level of confidence**. In the first prediction system, module Function Points should be used to estimate an effort: $Effort = F(FP)$. In the second prediction system, module Function Points should be used to predict the number of defects in the module: $NumOfDefects = F(FP)$.



Using linear regression with the set of data we have available to us, we get the following prediction system for estimating the project effort given function points:

$$Effort = 0.524(FP) + 0.9341$$



Using linear regression with the set of data we have available to us, we get the following prediction system for estimating the project effort given the number of defects:

$$Effort = 1.228(\text{NumberOfDefects}) + 7.9141$$

PROBLEM #2 (20 points)

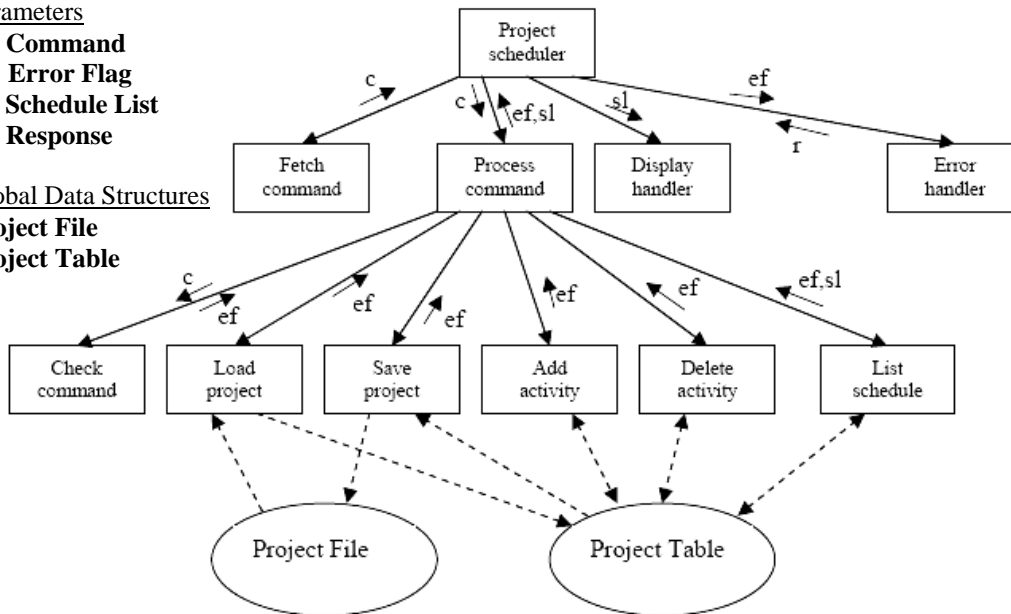
Given the following design (a structure chart) of a project scheduler system:

Parameters

- c:** Command
- ef:** Error Flag
- sl:** Schedule List
- r:** Response

Global Data Structures

- Project File**
- Project Table**



Use the Shepperd complexity measure (IF4 information flow measure) to analyze the design. Identify any weak points and suggest improvements. Redesign the system so that the Shepperd complexity measure will be improved. **Hint:** Do not forget about global data structures.

$$IF4 = \sum_{i=1}^n (FI_i \times FO_i)^2$$

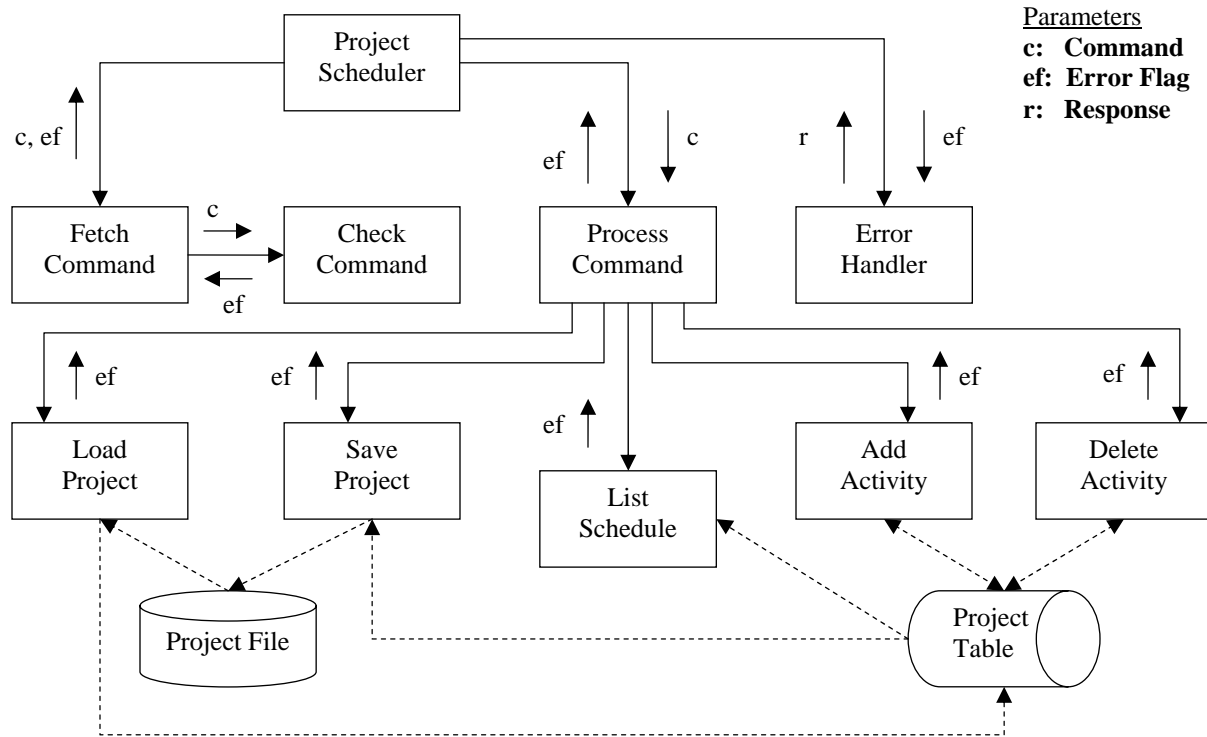
Module	Flow In	Flow Out	IF4 _i	Module	Flow In	Flow Out	IF4 _i
Project Scheduler	3	2	36	Load Project	1	5	25
Fetch Command	0	1	0	Save Project	4	2	64
Process Command	7	2	196	Add Activity	3	4	144
Display Handler	1	0	0	Delete Activity	3	4	144
Error Handler	1	1	1	List Schedule	3	4	144
Check Command	1	1	1	IF4 Total			755

Possible weak points in the system:

- Process Command (196)
- Add Activity (144)
- Delete Activity (144)
- List Schedule (144)

The Check Command module should be moved up a level. The accuracy of a command should be checked before being processed. This will reduce the IF4 value of the Process Command module. We should check the command before passing the command off to the Project

Scheduler to reduce the IF4 impact on that module. Since we only display via the List Schedule, the Display Handler's function can be accomplished inside the List Schedule module rather than passing the data through three modules. List Schedule should not write anything to the Project Table, just read data, so the module can be simplified. The redesigned system would look something like this:



The new Shepperd complexity measures are:

Module	Flow In	Flow Out	IF4 _i	Module	Flow In	Flow Out	IF4 _i
Project Scheduler	2	2	16	Load Project	1	5	25
Fetch Command	2	1	4	Save Project	3	2	36
Process Command	6	1	36	Add Activity	2	4	64
Display Handler				Delete Activity	2	4	64
Error Handler	1	1	1	List Schedule	3	1	9
Check Command	1	1	1	IF4 Total			256

The redesigned system indeed has a less Shepperd complexity measure than the original design (755 originally, 256 redesigned). All of our identified problem areas were all reduced in IF4 complexity and we were also able to remove a module that was performing a task that should have been wrapped in the calling module.

PROBLEM #3 (25 points)

The following formula is used to estimate an effort in the COCOMO model:

$$Effort = a \cdot size^b \cdot c$$

The goal of this problem is to calibrate the COCOMO prediction model for semidetached projects (for which $a=3.0$ and $b=1.12$), in particular to calibrate coefficients a and b . Assume that the following data were collected for a set of semi-detached projects:

Project Number	Actual Size	Actual Effort	c (cost driver's multiplier)
1	6	13	0.75
2	10	32	1.00
3	15	45	0.80
4	30	110	1.00
5	38	140	1.10
6	32	120	0.90
7	35	135	1.05
8	12	55	1.15
9	31	100	1.04
10	34	145	1.11
11	31	110	0.95
12	36	133	1.15
13	14	57	1.12
14	37	145	1.05
15	11	54	1.15

I. Part 1

Step 1: Calibrate the prediction model with respect to parameter a . Assume that the value of $b = 1.12$. Determine the prediction improvement of the calibrated model (an improvement in the cumulative error of prediction) for the collected data as opposed to the un-calibrated model.

Project Number	Actual Effort	$Effort = 3.0 \cdot size^{1.12} \cdot c$	Un-calibrated Difference	$Effort = 2.5 \cdot size^{1.12} \cdot c$	Calibrated Difference
1	13	16.73832	3.738322	13.9486	0.948602
2	32	39.5477	7.547702	32.95642	0.956418
3	45	49.82341	4.82341	41.51951	-3.48049
4	110	135.3621	25.36209	112.8017	2.801744
5	140	194.0312	54.0312	161.6927	21.69267
6	120	130.9579	10.95791	109.1316	-10.8684
7	135	168.9144	33.91443	140.762	5.762029
8	55	55.78303	0.783029	46.48586	-8.51414
9	100	146.0426	46.04265	121.7022	21.7022
10	145	172.8624	27.86243	144.052	-0.94797
11	110	133.4043	23.40434	111.1703	1.170283
12	133	190.9316	57.93164	159.1097	26.1097
13	57	64.56582	7.565821	53.80485	-3.19515
14	145	179.7614	34.76142	149.8012	4.801181
15	54	50.60331	-3.39669	42.16942	-11.8306

The calibrated value of a was determined by evaluating the following equation:

$$a_{Average} = \frac{\sum_{Project=1}^{15} \frac{ActualEffort_{Project}}{Size_{Project}^{1.12} \cdot c_{Project}}}{15}$$

By estimating the value of $a = 2.5$, we obtain an average calibrated difference of 3.14 man-months per project compared to the un-calibrated difference of 22.34 man-months per project. Not only is the calibrated value of a better over average, but a better estimate on 13 of the 15 projects we have information about.

II. Part 2

Step 1: Calibrate the prediction model with respect to parameter b . Assume that the value of $a = 3.0$. Determine whether the calibration in Part 2 is “better” than in Part 1.

Project Number	Actual Effort	$Effort = 3.0 \cdot size^{1.12} \cdot c$	Un-calibrated Difference	$Effort = 3.0 \cdot size^{1.09} \cdot c$	Calibrated Difference
1	13	16.73832	3.738322	15.86234	2.862345
2	32	39.5477	7.547702	36.90806	4.908063
3	45	49.82341	4.82341	45.93574	0.93574
4	110	135.3621	25.36209	122.2316	12.23158
5	140	194.0312	54.0312	173.9715	33.97148
6	120	130.9579	10.95791	118.0259	-1.97412
7	135	168.9144	33.91443	151.8255	16.8255
8	55	55.78303	0.783029	51.77578	-3.22422
9	100	146.0426	46.04265	131.7464	31.74643
10	145	172.8624	27.86243	155.5093	10.50926
11	110	133.4043	23.40434	120.3453	10.34529
12	133	190.9316	57.93164	171.4703	38.47027
13	57	64.56582	7.565821	59.65115	2.651152
14	145	179.7614	34.76142	161.306	16.30597
15	54	50.60331	-3.39669	47.09091	-6.90909

The calibrated value of b was determined using the following equation:

$$b = \frac{\sum_{Project=1}^{15} Size_{Project} \sqrt{\frac{ActualEffort_{Project}}{3.0 \cdot c_{Project}}}}{15}$$

By estimating the value of $b = 1.09$, we obtain an average calibrated difference of 11.31 man-months per project. While this is better than the un-calibrated difference of 22.34 man-months per project, and better at 13 of the 15 un-calibrated predictions, estimating the value of b leaves

us worse of than the estimated value of a by an average of 8.17 man-months per project. The calibration in Part 1 is better than the calibration in Part 2.

PROBLEM #4 (20 points)

Given a brief specification of a system:

The system enables instructors to enter student grades for a set of courses that he/she teaches in a given semester. Thus, grades can be updated, but the instructors cannot change the basic course information and student information, as the course lists are the responsibility of the system administrator. The system is menu-driven, with the instructor selecting from a choice of courses and then a choice of operations. The operations include:

- a. enter coursework scores for a course
- b. enter final grades for a course
- c. update final grades in a course
- d. give a permission to a student to take a course
- e. display a list of students with grades in a course
- f. compute averages in the course
- g. send an email with grades to students in a course
- h. change PIN to access the system
- i. get information about a student registered in a course

Compute the number of function points in this system, stating carefully the assumptions you are making. Assume that all technical complexity factors are average. In addition, assume that all function point complexity weights are also average. Notice that the system accesses at least two databases: a student database and a course database.

Using the Albrecht's approach to compute the unadjusted function point count, we use the following equation: $UFC = 4A + 5B + 4C + 10D + 10E$.

- **External Inputs (A):** those items provided by the user that describe distinct application-oriented data (such as file names and menu selections). These items do not include inquiries, which are counted separately.
- **External Outputs (B):** those items provided to the user that generate distinct application-oriented data (such as reports and messages, rather than the individual components of these).
- **External Inquiries (C):** interactive inputs requiring a response.
- **External Files (D):** machine-readable interfaces to other systems.
- **Internal Files (E):** logical master files in the system.

Since we are assuming that all of the technical complexity factors and all of the function point complexity weights are average, the unadjusted function point count will be suitable for our total function point count.

The first thing a professor needs to do to access the system is select a course and enter their PIN. This is a type C function point because the system needs to load a list of course from the database and a type B function point because the system needs to check the PIN before loading the menu system. There are two internal files (E) in this system a Student and a Course database. So before getting to the menu system we have the following statistics: 0A, 1B, 1C, 0D, 2E.

- a. enter coursework scores for a course
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to enter grades for a specific student for specific coursework
- b. enter final grades for a course
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to enter the final grade for a specific student
- c. update final grades in a course
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to enter the updated final grade for a specific student
- d. give a permission to a student to take a course
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to set the permission on a student to take a course
- e. display a list of students with grades in a course
 - We have an external input (A) to select the menu option
 - There needs to be an external output (B) to display the list of students and grades
- f. compute averages in the course
 - We have an external input (A) to select the menu option
 - There needs to be an external output (B) to display the computed averages
- g. send an email with grades to students in a course
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to select a student from a course list
 - There needs to be an external output (B) to send the email out to the student
- h. change PIN to access the system
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to take in and change the professor's PIN
- i. get information about a student registered in a course
 - We have an external input (A) to select the menu option
 - There needs to be an external inquiry (C) to select a student from a course list
 - There needs to be an external output (B) to display the student information

After processing all nine of the menu items, we get the following statistics: 9A, 5B, 8C, 0D, 2E. Plugging these numbers into the unadjusted function point count formula, we get:

$$UFC = 4(9) + 5(5) + 4(8) + 10(0) + 10(2) = 113$$

The number of function points in our system is 113.