Software Size Units (SSU)

My major concern about all the software size measures described in Chapter 6, is that they use “Points” to signify software size. A “point” is something that does not occupy space in geometry. “Points” are used in games – points are “scored” or counted. A “point” where they are used (in games) – complexity does not affect them. A point never becomes more than one point whatever be the complex maneuvers that have to be achieved in scoring it. No factor, like wind factor, sun factor, humidity factor affect the points.

Only in software industry do we have points being affected by complexity and other factors.

Where there are no universally accepted units of measure, the un-written convention has been to use “Units” as the unit of measure.

For example, the unit of measure for heat was BTU (British Thermal unit).

Enzymes in medicine are measured in IU (International Units) or simply “Units”. Best example would be Insulin - administered to diabetics in “so many” units per day – say 15 Units of Insulin per day.

“Unit” is used in this context to mean a “Unit of Measure” for something.

Software size has come to be measured in “Points” – from the time that Function Points are introduced for software size estimation, all others have followed this. Perhaps, it is that “Points are scored” – so functions are counted (scored).

In enzymes that are measured in units, they catalyze some action. In heat too, BTU catalyzes some action (raise the temperature).

In software too, a Unit catalyzes development action.

Hence I propose **Software Size Units (SSU)** for measuring software size – taking inspiration from the above!

Definition of SSU (Software Size Unit)

What does a software system comprise of? It basically consist of two types of elements, namely,

a. **Data Elements** – the data that enters the system, is processed upon and is either stored or given out as outputs
b. **Process Elements** – the software routines that process data elements and achieve the desired functionality

The terms are explained below.

Data Element – it is an input to the system. It may enter the system thru user input; or read from a master data file/table; or received from another system. It could be a constant (used as parameters for the software system) or a variable. It is classified into three types, namely, Numeric, Alphanumeric and Control.

a. **Numeric Data Element consists** of digits 0 (zero) to 9, one decimal point and a positive or negative sign. Numeric Data Element transforms and is transformed in the system. The whole system revolves around this data element.

b. **Alphanumeric Data Element** consists of all humanly readable characters, including space character. The Alphanumeric Data Element simply passes thru the system. When it enters, validation checks are performed to ensure their type and length.

c. **Control Data Element** - Control Data Element triggers some process within the system. Links on a web site, command buttons on a screen etc. are examples of Control Data Elements.

Process Elements – Process Elements act on the Data elements and transform input to desired outputs. **Process Elements** are classified into three types, namely, Input Process, Output Process, Associate Process,

a. **Input Process Elements (IPE)** get Data Elements from external environment into the system – may be thru keyboard, or from a file/table, or from a device like scanner, or from a network.

b. **Output Process Element (OPE)** sends Data Elements from the system to the external environment – may be to a screen, or a report, or to a device like printer or to a network.

c. **Associate Process Element (APE)** – This is a process element that helps either IPE or OPE or assist in maintaining certain system functions the results of which are sometimes not seen by the users. Some of the examples of APE are –
   i) A software routine that helps generate data for a report where the report is generated by a reporting tool like Crystal Reports
   ii) A software routine that manages the session in a web application
   iii) A software routine that manages / interfaces with any of the network layers
   iv) A software routine that manages / interfaces with any of the application tiers

System – is a software that fulfils some need and performs a set of defined functions

Enters the system – it is a necessary input given either from the key board by the user, or is retrieved from a master file/table which was prepared thru some other system, or is received over network from another system.

The size of a software system is derived from the data elements and process elements that comprise the system. A SSU (Software Size Unit) is a Process Element that has five Numeric Data Elements.

Procedure for Software Size Estimation

There is a free software tool – **SSUPal** – that can be used to estimate software size in SSU on the web site http://www.effortestimator.com - feel free to download and use it.

Delivered Software Size (DSS) – this is measured in Software Size Units (SSU) and is used for agreement between the customer and the vendor. This would be used for estimating the effort required for developing the software. The effort required for delivering the software would be the sum of effort required for Requirements Analysis, Software Design, Construction and Testing.

Now this needs some explanation.

a. Numeric Data Element has a weight of 1; an Alphanumeric Data Element has a weight of 0.35 and a Control Data Element has a weight of 0.75. These weights are used for normalizing the data elements while estimating the software size.

b. Input Process Element has a weight of 1, Output Process Element has a weight of 0.75 and Associated Process Element has a weight of 1.25. These weights are used for normalizing the process elements while estimating the software size.

Now we are ready to carry out Delivered Software Size.

a. Enumerate the process elements that comprise the system along with its nature – that is – Input / Output / Associated
b. Against each process element count data elements, namely, NDE, ADE and CDE
c. Against each process element compute equivalent Total Data Elements (TDE) for the Process Element using the formula –
    i. TDE = NDE + (ADE * 0.35) + (CDE * 0.75)
d. Compute the Software Size Units for each process element using the formula
    i. SSU = (TDE / 5) * Process Element Weight
e. Sum up the SSUs of each Process Element to obtain SSUs for the project before contingency allowance
f. Add a contingency allowance of 10% towards possible requirement changes to the Total SSU to obtain size estimate for the project

The below table would illustrate the procedure for a Warehouse Management System software development project.

Table 5.1 – Example of a software estimate using SSU

Sl. No	Process Element	Nature of PE	NDE	ADE	CDE	Total Data Elements (TDE)	Equivalent SSU
1	Material Master Definition	Input	4	3	0	5.05	1.01
2	Supplier Master Definition	Input	1	12	0	5.2	1.04
3	Material Categories Definition	Input	1	1	0	1.35	0.27
4	Units Definition	Input	1	1	0	1.35	0.27
5	Category - Supplier Mapping	Input	1	1	0	1.35	0.27
6	Administration - User Definition	Input	0	4	0	1.4	0.28
7	Material Availability Enquiry	Input	1	4	0	2.4	0.48
8	Material Enquiry search routine	Associated	1	4	0	2.4	0.48
9	Purchase Order Entry	Input	11	12	2	16.7	3.34
10	Material Receipts Entry	Input	9	12	1	13.95	2.79
11	Material Issues	Input	3	8	0	5.8	1.16
12	Material Returns	Input	1	8	0	3.8	0.76
13	Reports	Input	6	2	0	6.7	1.34
14	Issues Costing Routine	Associated	3	12	0	7.2	1.44
15	Balances Costing Routine	Associated	6	20	0	13	2.6
16	Below Re-order Level Report	Output	4	12	0	8.2	1.64
17	Category-wise Report	Output	6	15	0	11.25	2.25
18	Consumption Report for a period	Output	2	18	0	8.3	1.66
19	Goods Receipts Register for a period	Output	7	11	0	10.85	2.17
20	HML Report	Output	2	7	0	4.45	0.89

21	Material Category Report	Output	3	8	0	5.8	1.16
22	Material Category Summary	Output	2	8	0	4.8	0.96
23	Material Code Consumption for a period	Output	5	9	0	8.15	1.63
24	Material Issues Register	Output	4	14	0	8.9	1.78
25	Non-Moving Items Register	Output	5	9	0	8.15	1.63
26	Po Receipts for a period	Output	6	14	0	10.9	2.18
27	Priced Stores Ledger for a period	Output	18	15	0	23.25	4.65
28	Priced Stores Ledger Summary for a period	Output	18	15	0	23.25	4.65
29	Project Consumption	Output	4	8	0	6.8	1.36
30	Source-wise Report	Output	3	7	0	5.45	1.09
31	Stock Report as on a date	Output	4	8	0	6.8	1.36
32	VED Report	Output	4	8	0	6.8	1.36
33	Vendor-wise Receipts report	Output	8	15	0	13.25	2.65
34	Online Help	Output	0	40	0	14	2.8
35	Totals		154	345	3	277	55.4
36	Contingency Alloance @10%		15.4	34.5	0.3	27.7	5.54
37	Estimated SSUs for the project		169.4	379.5	3.3	304.7	60.94

Now, we may round up the software size to the next higher multiple of 5 units and say the estimated size of the software project for development of software for Warehouse Management System is 65 SSUs.

Software Development Effort Estimation from SSU

Now once having estimated the size of the software to be developed, we have to estimate the software development effort.

As the work carried out at various stages requiring persons of different and varying skill sets in software development life cycle, it is necessary to have different productivity figures for each of the skill sets, namely

a. **Requirements Analysis Effort** – We use SSUs for estimating the effort required for carrying out Requirements Analysis, which includes eliciting user requirements, developing software requirements, preparation of necessary documentation to capture requirements and peer review of the documentation and fixing defects, if any. To convert SSUs into effort, we use productivity of Requirements Analysis (say 6 hours per SSU rounded off to the next higher multiple of 8)
b. **Software Design Effort** – We use SSUs for estimating the effort required for carrying out software design including database design, architecture design, user interface design, report design and program design, preparation of necessary documentation, and peer review of design and fixing defects, if any. To convert SSUs into effort, we use productivity of Software Design (say 8 hours per SSU rounded off to the next higher multiple of 8)
c. **Construction Effort** – We use SSUs for estimating the effort required for construction activity that includes coding, independent verification, independent unit testing and fixing defects, if any. To convert SSUs into effort, we use productivity of Construction (say 10 hours per SSU rounded off to the next higher multiple of 8)

d. **Testing Effort** – We use SSUs for estimating the effort required to carry out independent software testing. Software testing includes integration testing, system testing and acceptance testing. To convert SSUs into effort, we use productivity of Testing (say 4 hours per SSU rounded off to the next higher multiple of 8)

The rounding off to the next higher multiple of 8 is to ensure that we arrive at a round figure of person days. We may estimate in fragment of a person day, but in reality, any fragment of a day would be wasted during execution.

Thus for the above project, let us arrive at the effort required for software development, using the below table

Table 5.2 – Effort Estimation using SSU

Sl. No	Stage	SSU Size	Productivity in PH per SSU	Total Effort in PH	Total Effort rounded to next multiple of 8
1	Requirements Analysis	65	6	390	392
2	Software Design	65	8	520	520
3	Construction	65	10	650	656
4	Testing	65	4	260	264
5	Total Effort				1832

Thus the effort in person days is 1,832. Now add the person days required for administrative activities like Project Initiation, Project Planning, and Project Closure and Project Management overhead, if necessary.

Now we may use this value to schedule the project and arrive at the final effort for computing the cost of the project.

How to obtain Productivity figures?

The productivity is unique to an organization owing to the specific environment and unique processes that are used. Therefore, the best method is to conduct a productivity study using Work Sampling (it is a work measurement technique) method. This is best accomplished by a qualified Industrial Engineer. The local chapter of Industrial Engineers would be glad to oblige your request for carrying out the Work Sampling Study and develop the productivity norms for your organization. I suggest that you carry out this study whenever there are major changes in the development environment or methods (development processes) or organizational structure.

FAQ – (Frequently Asked Questions) about SSU computation

Q: Can a screen be equated to an Input Process Element?

A: If a screen just takes inputs only from the user – then it can be equated to an Input Process Element. But in many cases, a screen captures some info, derives and displays some info to assist the user in the entry process. In such cases, the screen

needs to be considered as more than one Input Process Element – it may contain input, out put and associated process elements too.

Q: Then does it mean, that one screen can have multiple Input Output Process and Associated Elements?
A: Yes

Q: How do we treat Enquiry Screens?
A: An enquiry screen comprises of one or more input data elements and a set of output data elements. Therefore, the input data capture portion may be treated as Input Process Element and the output portion of the screen may be treated as Output Process Element. One suggestion, treat all screens as “Screens” – do not classify them as “Input Screens” or “Enquiry Screens” or “Output Screens”.

Q: How do we treat Combo Box, List Box, List View etc?
A: SSU is derived and used before design / development of software. We derive SSUs from the user specification. The data elements are as perceived by the end-user. How these are implemented in the code come later when the application is designed and constructed. Remember, that SSU is pre-design. If however, you are working based on an existing application as your specification (as it happens in porting projects or some web site applications), there could be one APE to fill those controls and the number and type of data elements may be counted depending on the number selectable from that control.

Q: Are the weights for data elements and process elements to be used as they are or can we customize them?
A: If one thing that characterizes software development, it is diversity. If you feel that different weights are justified in view of the unique nature of your projects, you may certainly do so.