Software Composition Technologies

Helping People Gain Control of Software Development

Looking for Function Point Keywords

Raymond Boehm

19 Homer Place, Metuchen, NJ 08840-2006

Voice: 732.906.3671 Fax: 732.906.5728

rayboehm@softcomptech.com

Agenda

- *Introduction*
- *Relevance of Research in the Context of Other Work*
- *Methodology*
- *Results*
- *Conclusions, Implications and Recommendations*

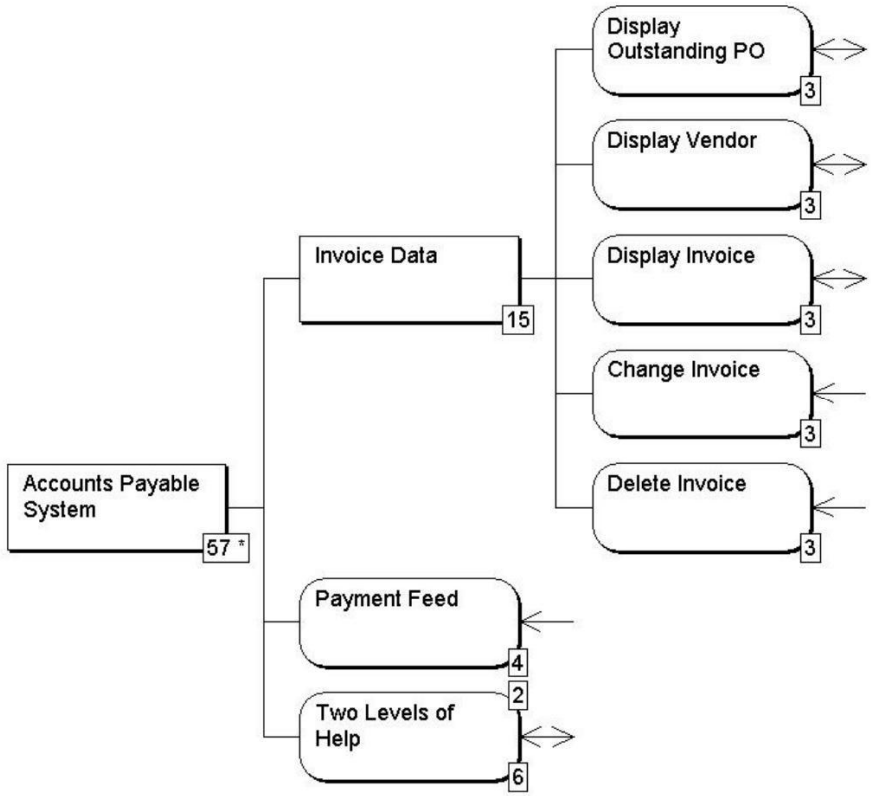
Introduction

- *There is a need for the ability to estimate project size early in the life cycle.*
- *Clues can be found in feasibility studies, use cases and user stories.*
- *Identifying keywords that can be associated with functionality of a fixed size would help.*

Relevance of Research in the Context of Other Work

- *Charismatek has keyword recognition built into Function Point Workbook.*
- *Jacque Jones used "active verbs" in FP-221 Estimating Project Size Early in the Life Cycle*
- *Ray Boehm began research into Agile Estimating.*
  - *Early Lifecycle Functional Estimating (ELFE) is part of Agile Estimating.*
  - *Empirical study of keywords was done for ELFE.*
  - *ELFE is now part of FP-221.*
- *Tom Cagley used FPW keywords in "Quick and Early Function Points."*

Methodology – 50 Counts Stored in Function Point Workbench



Methodology – 170 Possible Keywords Identified

- *Perl Program Generates Almost 50,000 Keyword Occurrences – For example, Display Outstanding PO generates 3 occurrences*
- *Counts for Almost 5700 Unique Keywords – Add Occurred 832 Times*
- *Keyword Types Identified for 170 Keywords Occurring 50 Times or More*

Methodology – 32 Transaction Keywords Identified

		Type				
Keyword	Data	EI	EQ	EO	(blank)	Grand Total
Add	Count of Keyword	773	29	18	12	832
	Min of Points	3	3	4	9	3
	Average of Points	5	4	6	24	5
	StdDevp of Points	2	2	1	13	3
	Max of Points	18	12	7	48	48

- *Add is Usually an EI*
- *On Average, It Has Average Complexity*
- *Some Keywords Need Combining, Like Rpt and Report*

Methodology – 32 Component Keywords Identified

		Type				
Keyword	Data	EI	EQ	EO	(blank)	Grand Total
Access	Count of Keyword	20	26	7	51	104
	Min of Points	3	3	4	4	3
	Average of Points	5	4	6	20	12
	StdDevp of Points	4	1	1	18	15
	Max of Points	18	6	7	80	80

- *Usually a Component*
- *On Average, 20 Unadjusted Function Points*
- *Most of the Time, Less Than 40 UFPs*

Methodology – Over 100 Possible Keywords Rejected

		Type				
Keyword	Data	EI	EQ	EO	(blank)	Grand Total
account	Count of Keyword	93	119	47	55	314
	Min of Points	3	3	4	3	3
	Average of Points	5	4	6	24	8
	StdDevp of Points	4	1	1	35	16
	Max of Points	18	15	7	200	200

- *Account Occurs Many Times*
- *Not Clear Whether EI, EQ or Component*
- *High Standard Deviation of Component*
- *Other Rejects: to, by, Data, Service, EQ, EI, EO*

Results – EI Transaction Keywords

- *Add – Average*
- *Change – High*
- *Create – High*
- *Delete – Average*
- *Link – High*
- *Modify – High*
- *Remove – Low*
- *Request – High*
- *Response – High*
- *Update - High*

Results – EO Transaction Keywords

- *Adjust – Average*
- *Audit – High*
- *Detail – High*
- *Event – Average*
- *Get – Average*
- *History – Average*
- *Report – Average*
- *Status – Average*
- *Sum – Average*
- *Summary - High*

Results – EQ Transaction Keywords

- *Display – High*
- *Log – Average*
- *Name – Average*
- *Query – Average*
- *Record – High*
- *Search – High*
- *Send – High*
- *Transfer – Average*
- *View - Average*

Results – Small Component Keywords (with Range of UFPs)

- *Address – 15-20*
- *Assign – 15-20*
- *Charges – 10-20*
- *Code – 15-20*
- *Edit – 15-20*
- *Error – 10-20*
- *Master – 15-20*
- *Message – 10-20*
- *Number – 10*
- *Rate – 15-20*
- *Recurring – 15-20*
- *State – 10-20*
- *Strategy – 10*
- *Type – 15-20*

Results – Medium Component Keywords (with Range of UFPs)

- *Access – 20-40*
- *Activity – 20-30*
- *Contact – 15-30*
- *Group – 15-30*
- *Information – 15-30*
- *Inventory – 20-40*
- *Location – 15-30*
- *Tax – 20-30*
- *Usage – 20-30*

Results – Large Component Keywords (with Range of UFPs)

- *Contract – 25-50*
- *Discount – 35-60*
- *Feature – 45-50*
- *Level – 25-60*
- *Map – 65-130*
- *New – 25-50*
- *Payment – 25-40*
- *Price/Pricing – 40-90*
- *Sub – 25-40*

Conclusions, Implications and Recommendations

- *Conclusions – Keywords Can Be Identified and Categorized*
- *Implications*
  - *Can be Used for Early Estimating*
  - *Can be Used for FP Auditing*
- *Recommendations*
  - *Explore Impact of Syntax*
  - *Use in Conjunction with Patterns*

Looking for Function Point Keywords

by
Raymond E. Boehm

Abstract – Function point counters have long used keywords like add, change and delete as indicators that a transaction is an EI. Research has been done to generalize this practice and identify a larger set of keywords and an indication of the size of the software that might be required to implement them. This work is useful for function point counting and software development estimating.

This document is part of a dissertation in Agile Estimating that is currently under development. It describes a process in the Early Lifecycle Functionality Estimating (ELFE) process that involves the use of keyword recognition. The research that was done to identify and categorize the keywords is described under rationale. Finally, the source of a Perl program used in this analysis is included.

Identify Transactions

Identify all of the active verbs used in requirements as they stand now. These requirements might be user stories or the written result of some type of feasibility study. Use the verbs to classify each sentence into a probable input (EI), output (EO) or inquiry (EQ). The keyword analysis that is about to be described may help in this respect. However, an understanding of the requirement will usually be sufficient to make this determination. For example, if “Create a Customer Entry” existed as a user story or as part of a feasibility study, the estimator would realize that some information was being introduced to the application. It is an external input (EI) or part of an EI. Sometimes

several of these entries together will make a single transaction. Use the standard IFPUG definition of an elementary process to make this determination.

In ELFE, a set of keywords having significance to the estimating process have been identified. The keywords may be verbs, nouns or modifiers. Some keywords are ambiguous. For example, the keyword **record** could be any of these. For example, in record patient weight, it is a verb; in enter a customer record, it is a noun; and in expecting record crowds, it is a modifier. Any of the sentences in the requirements may contain one or more keywords. Like the small mannerisms that tell a poker player's hand, subtle differences in keywords often indicate significant differences in an estimate. The difference between "create a customer entry" and "create a new customer entry" illustrate this. As will be described below, the former will probably be a high complexity external input. The latter will probably be a group of transactions with an unadjusted function point count of between 25 and 50.

In ELFE, there are two different types of keywords, transactions and components. (Table 1) shows the transaction keywords. When one of these keywords is found in a requirement, the sentence is assumed to be a transaction of the type and complexity shown in the table. For example, the user story "Add a customer" would be estimated as an EI with high complexity. If more than one keyword is in the same sentence, then the transaction with the higher complexity should be chosen. Basically, the same rule would apply if a component keyword is discovered. In all of these cases, the estimator must use judgment to determine the type and complexity of the transaction, or whether the requirement is part of a group of transactions. The decisions made at this point are preliminary. They may be modified in the Refine Complexity step. On the other hand, it

might be decided that the complexity assigned at this point is adequate for the required estimate. Having a more precise estimate might not lead to any better decision making. This is one of the essences of agile thinking. Work to the easiest solution that satisfies the problem.

Table 1. ELFE Transaction Keywords

Keyword	Type	Complexity		Keyword	Type	Complexity
Add	EI	Average		Name	EQ	Average
Adjust	EO	Average		Query	EQ	Average
Audit	EO	High		Record	EQ	High
Browse	EQ	Average		Remove	EI	Low
Change	EI	High		Report	EO	Average
Create	EI	High		Request	EI	High
Delete	EI	Average		Response	EI	High
Detail	EO	High		Revenue	EO	High
Display	EQ	High		Search	EQ	High
Event	EO	Average		Send	EQ	High
File	EO	High		Status	EO	Average
Get	EO	Average		Sum	EO	Average
History	EO	Average		Summary	EO	High
Link	EI	High		Transfer	EQ	Average
Log	EQ	Average		Update	EI	High
Modify	EI	High		View	EQ	Average

Components are usually composed of several transactions. (Table 2) shows the component keywords. When one of these keywords is found in a requirement, both the expected and the maximum values in unadjusted function points are given. The expected value is just that. The maximum value is a reasonable upper bound. For example, the sentence “assign salesperson to customer,” would be expected to be 15 unadjusted function points in size. However, the size can often go as high as 20 unadjusted function

point counts. The sentence may be implemented with any mixture of external inputs, external outputs and external inquiries. ELFE does not predict what this mixture might be. The minimum would probably be a single transaction. In the case of the example, it would most likely be an external input. However, the ELFE methodology does not utilize this minimum value.

Table 2. ELFE Component Keywords

Keyword	Expected	Maximum		Keyword	Expected	Maximum
Access	20	40		Location	15	30
Activity	20	30		Map	65	130
Address	15	20		Master	15	20
Assign	15	20		Message	10	20
Charges	10	20		New	25	50
Code	15	20		Number	10	10
Contact	15	30		Payment	25	40
Contract	25	50		Price/Pricing	40	90
Discount	35	60		Rate	15	20
Edit	15	20		Recurring	15	20
Error	10	20		State	10	20
Feature	45	50		Strategy	10	10
Group	15	30		Sub	25	40
Information	15	30		Tax	20	30
Inventory	20	40		Type	15	20
Level	25	60		Usage	20	30

Rationale – Using active verbs to identify transactions is far from being a new idea. Function point counters tend to do this intuitively. Jacque Jones has taught an IFPUG course on estimating function points early in the lifecycle for several years.[3] This course explicitly describes a process of using active verbs to identify transactions. Function Point Workbench (FPW) is a software program used by many function point

counters.[1] When a transaction is inserted in FPW, the tool uses a set of built-in keywords to set the transaction type. The type must often be changed by the counter. Most recently, Tom Cagley appears to have blended the active verb approach and the FPW keywords into a technique called “Quick and Early Function Points.”[2] The current research used empirical analysis to move this keyword-based approach to the next level.

A set of about 50 function point counts were chosen for the study. The counts are studied to identify keywords and discover their associated function type and size. There were several steps taken to insure that the results are as general as possible. The counts were done by several different counters. More than one client’s work was involved. However, each client considers their counts to be proprietary. Therefore, details of the counts cannot be published. However, summary results can be shared. They were all maintained in the FPW tool. This allowed for the development of automated tools that read the counts and facilitated the keyword analysis.

(Figure 1) shows an example function point count. It is not an actual count. It is a small example used to illustrate some points about the keyword analysis. In the figure, the rounded shapes are transactions. They are external inputs, external outputs or external inquiries. Each is a certain number of unadjusted function points. The square shapes are components. They are a way to logically collect transactions. As such, they indicate what function the counter thinks the group of transactions is accomplishing. They do not have any size of their own, but can be considered as being the total size of the transactions that they are composed of. For example, the Invoice Data component can be thought of as being 15 unadjusted function points because it is composed of five

transactions that are each three unadjusted function points in size. All of the words were used in the keyword analysis. For example, the word **invoice** occurred four times. Twice it was a three function point external input. Once, it was a three function point external inquiry. It was also associated with a 15 function point component. There is only one occurrence of **data**, and it is associated with a 15 function point component.

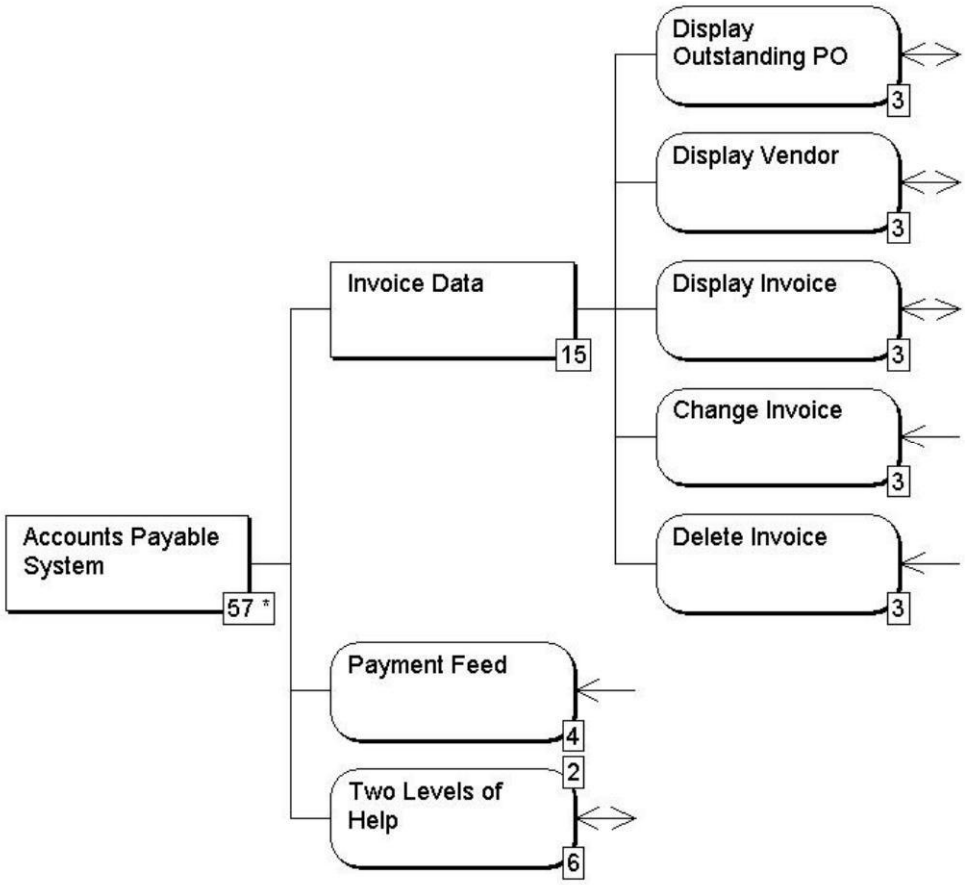


Figure 1. Keyword Example

FPW allows comma separated value files to be created with both a hierarchy outline list and a transaction list. The former lists the names of all of the components and transactions, as well as the associations between them. The latter shows details of each transaction, such as the type and number of unadjusted function points. A pair of files was created for each of the function point counts in the sample. The PERL program in Appendix D processed the set of these files to identify all of the keywords along with their types and sizes. From this, the number of times that a keyword occurred was calculated. (Table 3) shows the keywords that occurred 50 times or more in the sample. The data was divided by types. (Table 4) shows the first entries of the keyword types. It is included to explain how this data is transformed into the keywords of ELFE.

Table 3. Keyword Counts

Keyword	Total	Keyword	Total	Keyword	Total	Keyword	Total
Add	843	New	151	Map	88	Output	62
View	592	Reports	150	Circuit	87	Remove	62
Update	544	ASR	147	Inventory	87	toll	62
Rpt	534	Loc	146	Print	87	Transaction	62
Delete	521	Table	143	MCN	86	Monthly	61
Report	519	Type	143	SVC	86	other	61
File	445	port	135	Component	85	CLLI	60
Change	441	Feed	132	Address	84	Discount	60
Order	359	proc	131	DACS	84	Price	59
Detail	331	Status	131	Page	84	Product	59
to	324	Trf	126	Rates	84	Search	59
usage	318	LD	125	Invoice	83	wats	59
Account	314	local	125	revenue	83	Contact	58
Data	307	Tax	122	Orders	82	master	58
Service	239	Details	119	Rsp	82	vtns	58
by	232	Ckt	118	pricing	81	Charges	57
cust	231	C	115	Record	81	Client	57
EQ	226	Group	114	Id	80	Feature	57
Request	219	Chg	113	State	80	Link	57
Create	218	sdn	113	msg	78	daily	56
Customer	218	Error	111	non	78	NPA	56
Summary	218	Info	111	PROFILE	78	Bu	55
Display	215	LEC	111	recurring	78	LOG	55
Get	213	Tariff	108	Req	78	Select	55
Send	212	Access	104	S	78	Interfaces	54
Inquiry	211	for	104	Input	77	Java	54
Process	203	TS	103	ABS	76	Maintain	54
List	200	Modify	102	Disconnect	76	Out	54
EI	198	Screen	100	Up	75	Edit	53
Billing	192	Information	99	Audit	74	Wtn	53
User	192	Contract	97	OneNet	73	Maintenance	52
Rate	187	Admin	95	All	71	Message	52
EO	184	query	95	AT	70	Sub	52
Plan	174	Snd	94	line	70	event	51
Lvl	171	of	93	and	68	from	51
Sum	171	Acct	92	Del	66	Maint	51
Interface	169	PVC	92	NE	65	Rspn	51
Location	166	Activity	91	T1	65	Strategy	51
bill	165	Browse	90	Date	64	adj	50
Code	163	Control	89	Response	64	AIO	50
call	161	Number	89	History	62	Attributes	50
Level	160	T	89	name	62	Payment	50
A	157	det	88				

Table 4. Keyword Types

		Type				
Keyword	Data	EI	EQ	EO	(blank)	Grand Total
a	Count of Keyword	115	16	9	13	153
	Min of Points	3	3	5	10	3
	Average of Points	7	4	6	137	18
	StdDevp of Points	4	1	1	220	74
	Max of Points	24	6	7	880	880
ABS	Count of Keyword	10	17	40	9	76
	Min of Points	4	3	4	3	3
	Average of Points	6	4	5.7	18	7
	StdDevp of Points	2	1	1.4	11	6
	Max of Points	12	6	7	35	35
Access	Count of Keyword	20	26	7	51	104
	Min of Points	3	3	4	4	3
	Average of Points	5	4	6	20	12
	StdDevp of Points	4	1	1	18	15
	Max of Points	18	6	7	80	80
account	Count of Keyword	93	119	47	55	314
	Min of Points	3	3	4	3	3
	Average of Points	5	4	6	24	8
	StdDevp of Points	4	1	1	35	16
	Max of Points	18	15	7	200	200
Acct	Count of Keyword	32	39	13	8	92
	Min of Points	3	3	4	4	3
	Average of Points	4	4	5	13	5
	StdDevp of Points	1	1	1	5	3
	Max of Points	9	6	7	21	21
Activity	Count of Keyword	27	26	27	11	91
	Min of Points	3	3	4	6	3
	Average of Points	5	5	6	21	7
	StdDevp of Points	2	1	1	9	6
	Max of Points	12	6	7	42	42
Add	Count of Keyword	773	29	18	12	832
	Min of Points	3	3	4	9	3
	Average of Points	5	4	6	24	5
	StdDevp of Points	2	2	1	13	3
	Max of Points	18	12	7	48	48
Address	Count of Keyword	21	47	4	12	84
	Min of Points	3	3	5	4	3
	Average of Points	5	4	5	14	6
	StdDevp of Points	2	1	0	6	4
	Max of Points	12	6	5	22	22

Obviously, the number of occurrences is important in identifying a keyword. If there are too few occurrences, then there is not enough data to assign expected values, types, complexities or any other attributes. Furthermore, if it is not a common keyword, having these values would be of little value to an estimator. No keyword with less than 50 occurrences is included.

However, the number of occurrences is not enough to make a keyword significant. For example, the keyword **Rpt** has many occurrences. However, FPW has a limited number of characters for a description. **Rpt** is an obvious abbreviation for report. Its data is combined with that of **report** to establish the type and complexity of the **report** keyword. **To** is common as a keyword but too ambiguous to be of value to an estimator.

The type information is used to establish type, complexity, expected and maximum size. For example, (Table 4) shows that the **add** keyword is almost always an EI. Because the average number of points is 5, the complexity is closest to average. The **access** keyword was usually a component. On average, it was 20 unadjusted function points. Most of the time, the value should be less than one standard deviation away from the average. This keyword rounds to a maximum of 40 unadjusted function point counts.

There were other reasons to disallow a keyword. Some possible keywords, such as ABS or EQ, appear to be domain specific. The former might be a telecommunications abbreviation. The latter is clearly associated with function point counting, itself; it would not be something that a customer would usually be using to describe requirements. Other possible keywords have too large a range of probable values. For example, consider the

keyword **a**. Even if the keyword had some significance, its values would have an expected value of 20, but a maximum value of 90. This range is too wide to make the keyword valuable to an estimator.

Source for Keyword.pl

```
# This program creats a listing of keywords from a sample of function point counts.
# The samples were maintained in and generated from Function Point Workbench.
# Keywords from transactions have their associated function point information.
# It is speculated that ADD might be such a keyword, with the function point
# information of EI's of various complexity and their corresponding number of
# unadjusted function point.  The same is done for components.  However,
# components are collections of transactions, so their only information
# is number of unadusted function points.  It is speculated that a keyword
# like MAINTAIN probably exists, and its function point count is in the neighborhood
# of 16 (an average complexity add, change, delete and inquiry).
use Text::ParseWords;

# Sample.text contains a list of the names of the function point counts
# It is assumed that a Heirarchy Outline List exists with that name + hol.csv exists.
# In addition, a Transaction List with that name + tl.csv also exists.
open (SAMPLE, "Sample.txt");

# The Keywords.csv file has all of the keywords, their function point count and
# enough information to trace back to the original count if necessary.
open (KEYWORDS, ">Keywords.csv");

# This loops through all of the function point counts in the sample.
while ($count = <SAMPLE>) {
        chomp ($count);
        open (HOL,$count."hol.csv");
        open (TL, $count."tl.csv");

#       Read in all of the identifiers and transaction names for both components
#       and transactions.  The identifiers are stored in an array so that order
#       is preserved.  The names are in a hash for quick access.
        @id = ();
        %name = ();
        while (<HOL>) {last if /COMPONENT \/ TRANSACTION/;} # Skip past headings
        while ($line = <HOL>) {
                next if $line eq "\n";
                chomp ($line);
                @fields = quotewords (',', 0, $line);
                push @id, $fields[0];
                $name{$fields[0]} = $fields[1];
                }

#       Read in all of the transactions and their associated data from the transaction
#       list.  Add the function points to all components above it in the heirarchy.
        %type = ();
        %complexity = ();
        %multiplier = ();
        %points = ();
        while (<TL>) {last if /^"TRANSACTIONS:"/;} # Skip past headings
        while ($line = <TL>) {
                next if $line eq "\n";
                last if $line =~ /^"Number of Functions"/;
                chomp ($line);
                @fields = quotewords (',', 0, $line);
                $type{$fields[0]} = $fields[2];
                $complexity{$fields[0]} = $fields[3];
                $multiplier{$fields[0]} = $fields[4];
                $points{"0"} += $fields[5];
                @levels = split (/\./, $fields[0]);
                $id = '';
                foreach $level (@levels) {
                        $id .= $level;
                        $points{$id} += $fields[5];
                        $id .= '.';
                        }
```

```
                }

#       For each transaction or component, write out each keyword as a separate record.
        foreach $id (@id) {
                $name = $name{$id};
                $name =~ s/\W/ /g;   # Change all non-alphanumerics to spaces
                $name =~ tr/  / /s;  # Collapse multiple spaces to one
                @words = quotewords (' ', 0, $name);
                foreach $word (@words) {
                        print KEYWORDS "\"$count\"\,\"$id\"\,\"$word\"\,\"$name{$id}\"\,";
                        print KEYWORDS "\"$type{$id}\"\,\"$complexity{$id}\"\,";
                        print KEYWORDS "\"$multiplier{$id}\"\,\"$points{$id}\"\n";
                        }
                }
        }
```

References

[1] "Function Point Workbench," 5.52 ed: CHARISMATEK Software Metrics, 2004.

[2] T. Cagley, "Turing Perfectly Good Words into Numbers," presented at Functional Sizing Summit, Cambridge, MA, 2006.

[3] J. Jones, "FP-221 Estimating Project Size Early in the Life Cycle," International Function Point Users Group, 1998-2005.