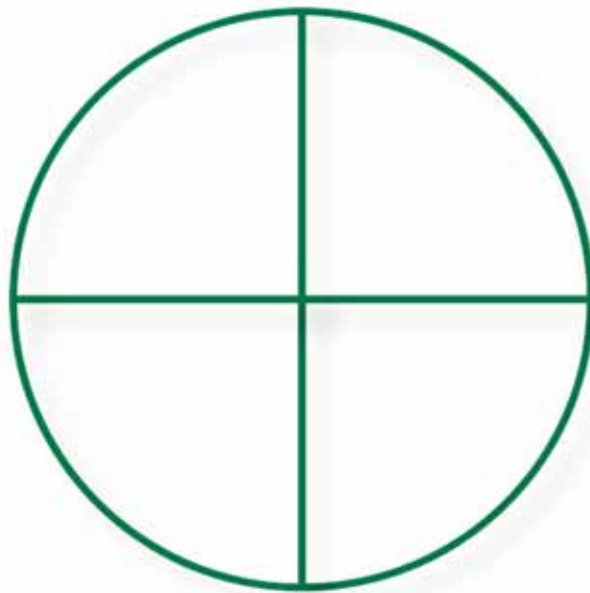# Function Points Analysis
# Training Course

Instructor: David Longstreet
David@SoftwareMetrics.Com
www.SoftwareMetrics.Com
816.739.4058

SoftwareMetrics.Com

# Table of Contents

# INTRODUCTION

## *Objective of Section:*

Introduce the basic concepts of Function Point Analysis and to introduce and reinforce unit cost estimating. The exercises at the end of the section help the student demonstrate they have gained the basic knowledge required.

## *Introduction:*

Systems continue to grow in size and complexity, becoming increasingly difficult to understand. As improvements in coding tools allow software developers to produce larger amounts of software to meet ever-expanding user requirements, a method to understand and communicate size must be used. A structured technique of problem solving, function point analysis is a method to break systems into smaller components, so they can be better understood and analyzed. This book describes function point analysis and industry trends using function points.

Human beings solve problems by breaking them into smaller, understandable pieces. Problems that may initially appear to be difficult are found to be simple when dissected into their components, or classes. When the objects to be classified are the contents of software systems, a set of definitions and rules, or a scheme of classification, must be used to place these objects into their appropriate categories. Function point analysis is one such technique: **FPA is a method to break systems into smaller components, so they can be better understood and analyzed**. It also provides a structured technique for problem solving. Function Point Analysis is a structured method to perform functional decomposition of a software application.

Function points are a unit measure for software much like an hour is to measuring time, miles are to measuring distance or Celsius is to measuring temperature. Function Points are interval measures much like other measures such as kilometers, Fahrenheit, hours, so on and so forth.

Function Points measure software by quantifying its functionality provided to the user based primarily on the logical design. Frequently the term end user or user is used without specifying what is meant. In this case, the user is a sophisticated user. Someone that would understand the system from a functional perspective --- more than likely someone that would provide requirements or does acceptance testing.

There are a variety of different methods used to count function point, but this book is based upon those rules developed by the Alan Albrecht and later revised by the International Function Point User Group (IFPUG). The IFPUG rules have much to be desired, so this book attempts to fill in gaps not defined by IFPUG.

**What is on the surface?**

The image to the right represents the tip of an iceberg. The real issue is not the tip, but what is under the surface of the water and can not be seen. The same is true when you design a software application.

One of the largest misconceptions of function points is understanding what functionality is being exposed to an end user versus the delivered functionality. One trend happening in software development today is self service applications like most major airlines are using.

If you visit American Airlines Website and/or Expedia, you will see a relatively simple screen exposed to the end user. The end user simply puts in their departure and destinations and the dates of travel. This appears on the surface to be a simple inquiry, but this is extremely complex. The process actually includes 1,000's of elementary processes, but the end user is only exposed to a very simple process. All possible routes are calculated, city names are converted to their international three characters, interfaces are sent to all the airline carriers (each one being unique), this is an extremely complex and robust process! When we size software applications we want to understand what is exposed and what is under the surface.

## *Elementary Process:*

A software application is in essence a defined set of elementary processes. When these elementary processes are combined they interact to form what we call a software system or software application. An elementary process is not totally independent existing alone, but the elementary processes are woven together becoming interdependent. There are two basic types of elementary processes (data in motion and data at rest) in a software application. Data in motion has the characteristic of moving data inside to outside the application boundary or outside to inside the application boundary. An elementary process is similar to an acceptance test case.

## *Definition:*

On a conceptual level, function point analysis helps define two abstract levels of data - data at rest and data in motion.

### Data in motion

Data in motion is handled via transactional function types or simple transactions. All software applications will have numerous elementary processes or independent processes to move data. Transactions (or elementary processes) that bring data from outside the application domain (or application boundary) to inside that application boundary are referred to as *external inputs*.

Transactions (or elementary processes) that take data from a resting position (normally on a file) to outside the application domain (or application boundary) are referred as either an *external outputs* or *external inquiries* (these will be defined later in this book).

### Data at rest

Data at rest that is maintained by the application in question is classified as *internal logical files*. Data at rest that is maintained by another application in question is classified as *external interface files*.

## *Benefits and Uses:*

A function point count has many uses. There are three types of function point counts. In the section *How are Function Point Useful* the benefits of function point counting is discussed in great detail. The article can be found on www.SoftwareMetrics.Com.

- Function Points can be used to communicate more effectively with business user groups.

- Function Points can be used to reduce overtime.

- Function points can be used to establish an inventory of all transactions and files of a current project or application. This inventory can be used as a means of financial evaluation of an application. If an inventory is conducted for a development project or enhancement project, then this same inventory could be used to help maintain scope creep and to help control project growth. Even more important this inventory helps understand the magnitude of the problem.

- Function Points can be used to size software applications. Sizing is an important component in determining productivity (outputs/inputs), predicting effort, understanding unit cost, so on and so forth.

- Unlike some other software metrics, different people can count function points at different times, to obtain the same measure within a reasonable margin of error. That is, the same conclusion will be drawn from the results.

- FPA can help organizations understand the unit cost of a software application or project. Once unit cost is understood tools, languages, platforms can be compared quantitatively instead of subjectively. This type of analysis is much easier to understand than technical information. That is, a non-technical user can easily understand Function Points.

There are several other uses of function points. The following list are some practical applications of Function Points and FPA. The article *Using Function Points* on the Website www.SoftwareMetrics.Com, in the article section of the Website, provides more detail regarding each of these items. Function Points can be used for:

- Defining When and What to Re-Engineer

- Estimating Test Cases
- Understanding Wide Productivity Ranges
- Understanding Scope Creep
- Calculating the True Cost of Software
- Estimating Overall Project Costs, Schedule and Effort
- Understanding Maintenance Costs
- Help with contract negotiations
- Understanding the appropriate set of metrics

## *When Not to Use Function Points*

Function points are not a very good measure when sizing maintenance efforts (fixing problems) or when trying to understand performance issues.   Much of the effort associated with fixing problems (production fixes) is due to trying to resolve and understand the problem (detective work).   Another inherent problem with measuring maintenance work is that much of maintenance programming is done by one or two individuals.   Individual skill sets become a major factor when measuring this type of work.  The productivity of individual maintenance programmers can vary as much as 1,000 percent.

Performance tuning may or may not have anything to do with functionality.  Performance tuning is more a result of trying to understand application throughput and processing time.  There are better metrics to utilize when measuring this type of work.

## *Types of Function Point Counts:*

Function point counts can be associated with either projects or applications. There are three major types of software projects (Development, Enhancements and Maintenance). In accordance with these types of function points there are three different types of function point counts (Development, Enhancement and Application).

### Development Project Function Point Count

Function Points can be counted at all phases of a development project from requirements up to and including implementation.  This type of count is associated with new development work. Scope creep can be tracked and monitored by understanding the functional size at all phase of a project.  Frequently, this type of count is called a baseline function point count.

### Enhancement Project Function Point Count

It is common to enhance software after it has been placed into production.  This type of function point count tries to size enhancement projects.  All production applications evolve over time. By tracking enhancement size and associated costs a historical database for your organization can be built.  Additionally, it is important to understand how a development project has changed over time.

### Application Function Point Count

Application counts are done on existing production applications.  This "baseline count" can be used with overall application metrics like total maintenance hours.  This metric can be used to

track maintenance hours per function point. This is an example of a normalized metric. It is not enough to examine only maintenance, but one must examine the ratio of maintenance hours to size of the application to get a true picture.

Additionally, application counts can assist organizations in understanding the size of the entire corporate portfolio (or inventory). This type of count is analogous to taking an inventory for a store. Like inventory, a dollar value can be associated with any application function point count and for the entire organization portfolio.

## What about Lines of Code (LOC)

There are several problems with using LOC as a unit of measure for software. Imagine two applications that provide the same exact functionality (screens, reports, databases). One of the applications is written in C++ and the other application written a language like Clarion (a very visual language). The number of function points would be exactly the same, but aspects of the application would be different. The lines of code needed to develop the application would not be the same. The amount of effort required to develop the application would be different (hours per function point). We are able to compare the productivity of the languages. Unlike Lines of Code, the number of function points will remain constant (should remain constant).

With this in mind;

1. The number of lines of code delivered is dependent upon the skill level of the programmer. In fact, the higher skill level of the programmer the fewer lines of code they will develop to perform the same function.

2. Higher-level languages such as Delphi, Progress 4GL, Forte, Dynasty, VB, Java Script, or other visual languages require far fewer lines of code than Assembler, COBOL, or C to perform the same functionality. That is, there is an inverse relationship between level of language and work output (when work output is LOC).

3. The actual number of LOC is not known until the project is almost completed. Therefore, LOC cannot be used to estimate the effort or schedule of a project. Function Points can be derived from requirements and analysis documents that are available early in a project life cycle.

4. There is no agreed upon method to count lines of code. The statement and type of statements used in Visual C++, Assembler, COBOL, SQL are completely different. It is common for applications to have a combination of different languages being utilized.

## Understanding Productivity:

The standard economic definition of productivity is "Goods or services per unit of labor or expenses" until 1979, when A.J. Albrecht of IBM published a paper about Function Points, there

was no definition of what "goods or services" were the output of software project. The good or service of software is the business functionality provided.

While software productivity is a relatively new subject "industrial productivity" has been a subject of interest for many years. One of the first individuals to study productivity was Frederick Taylor (1856-1912). Taylor's major concern throughout most of his life was to increase efficiency in production. Taylor decided that the problem of productivity was a matter of ignorance on the part of management. Taylor believed that application of scientific methods, instead of customs and rules of thumb could yield higher productivity. A century after Frederick Taylor most software managers use rules of thumb instead of systematic study.

### Hawthorne Studies
**Frederick W. Taylor**

Several scientists undertook the famous experiments at the Hawthorne plant of the Western Electric Company in 1927 and 1932. They began a study to determine the effect of illumination on workers and their productivity. They found that productivity improved when illumination was either increased or decreased for a test group. They found that when people felt they were being noticed then their productivity increased. They also found that the improvement in productivity was due to such social factors as morale, satisfactory interrelationships and effective management. They also found that the best managers were those that managed via counseling, leading, and communicating. The phenomenon, arising basically from people being "noticed," has been known as the **Hawthorne effect.**

*Productivity:*

The definition of productivity is the output-input ratio within a time period with due consideration for quality.

**Productivity = outputs/inputs (within a time period, quality considered)**

The formula indicates that productivity can be improved by (1) by increasing outputs with the same inputs, (2) by decreasing inputs but maintaining the same outputs, or (3) by increasing outputs and decreasing inputs change the ratio favorably.

*Software Productivity = Function Points / Inputs*

Effectiveness v. Efficiency:

Productivity implies effectiveness and efficiency in individual and organizational performance. Effectiveness is the achievement of objectives. Efficiency is the achievement of the ends with least amount of resources.

## *Understanding Software Productivity:*

Software productivity is defined as hours/function points or function points/hours. This is the average cost to develop software or the unit cost of software. One thing to keep in mind is the unit cost of software is not fixed with size. What industry data shows is the unit cost of software goes up with size.

How does size impact productivity

As the size of software development project becomes larger the cost per function point actually rises. As can be seen from the graph and data, the effort per unit does not remain constant as the size of the software project increases. This is self-evident because the tasks are not the same for software projects as the size increases.

*What is Marginal Cost?*

As some of you remember Marginal Cost is an economic term and is different from average cost.

**Average cost** is the total cost of producing a particular quantity of output divided by that quantity.   In this case to Total Cost/Function Points.

**Marginal cost** is the change in total cost attributable to a one-unit change in output. In our case, how does per unit cost change as the software project size change?  How does the cost of software change as the product becomes larger and larger?

Imagine the average cost per square foot of a one-story building compared to the cost per square foot of a 100-story building. No doubt the construction costs (per unit cost) for the 100-story building are much higher than a one-story building. This same concept is true for a software project.

Besides size there are several other factors, which impact the cost of construction

- Where the building is located
- Who is the general contractor?
- Who does the actual labor?

Why increasing Marginal Costs for Software Development?

There are a variety of reasons why marginal costs for software increase as size increases. The following is a list of some of the reasons

- As size becomes larger complexity increases.

- As size becomes larger a greater number of tasks need to be completed.

- As size becomes larger there is a greater number of staff members and they become more difficult to manage.

- A the numbers of individuals in a project increases the number of communication paths increase also.  Communication in large projects can be very difficult.

- Since fixed costs for software projects is minimal. There are little if any economies of scale for software projects.

Function Points are the output of the software development process. Function points are the unit of software. It is very important to understand that Function Points remain constant regardless who develops the software or what language the software is develop in. Unit costs need to be examined very closely. To calculate average unit cost all items (units) are combined and divided by the total cost. On the other hand, to estimate the total cost each item is examined.

For example, assume you are going to manufacture a computer mousepad. The total Cost to manufacture 1,000 mousepad is $2,660. The unit cost is $2.66 (per pad). The cost break down is:

- Artwork is a fixed cost at $500 (or .50 per unit)
- Set Up costs are $250 (or .25 per unit)
- Shipping costs are $10 (or .01 per unit)
- Papers for production will cost $1.50 per unit.
- Rubber Pads are $ .15 per unit.
- Application of paper to pad cost is $.25 per unit

Notice the variation in the unit cost for each item. One of the biggest problems with estimating software projects is understanding unit cost. Software managers fail to break down items into similar components or like areas. They assume all units cost the same.

There are different costs for each of the function point components. The unit cost for external inputs is not the same as the unit cost of external outputs for example. The online external inputs and the batch external inputs do not have the same unit cost (or cost per function point). The cost per unit to build and implement internal logical files is not the same per unit cost as the building and implementing of online reports.

To accurately estimate the cost of an application each component cost needs to be estimated. The same is true for the mousepad problem above.

## Questions:

### Problem 1
How would you estimate the number of hot chocolates being sold at the AFC Championship game in Kansas City (use your imagination, the Chiefs could be there)?

What are the keys factors to consider?

Who would you benchmark against and why?

### Problem 2
What is the average cost per mousepad if you produce 1,000 units at the following costs?

Artwork is a fixed cost at $500
Sets Up costs are $250
Shipping costs are $10
Papers for production will cost $2.50 per unit.
Pads are $ .25 per unit.
Application of paper to pad cost is $.35 per unit

Are the unit costs the same for all items?

Is it correct to assume that unit costs are fixed for software? (Intuitively, do you expect the per unit cost to create reports the same as the per unit cost to build a data base.)

Notes:

# FUNCTION POINT COUNTING PROCESS

## 2

## *Objective of Section:*

The objective of this chapter is to introduce the student to the high level steps necessary to count function points and to perform function point analysis. Details of each step are discussed later in this book. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## *Introduction:*

Even though there have been attempts by the National Bureau of Standards (NBS) and IEEE to standardize terms and definitions, there are no industry wide practiced terms and definitions related to software development. IFPUG has developed some standard terms and definitions related to function points, but these terms and definitions need to be applied to a variety of different software environments.

Clients who have standardized their terminology within their own environments have seen significant jumps in productivity. That is, they have reduced the number of verbs used to describe transactions and other events.

Imagine if we compared a blue print document used for construction purposes with a typical software design document. While the blue print uses standard terminology the software design document uses a variety of different terminology to describe the same exact thing.

## *Definition:*

The overall objective is to determine adjusted function point count. There are several steps necessary to accomplish this. While you may not understand the mechanics of the following steps, they will be discussed in great detail in the remainder of the book. The actual sequence or order of steps is not necessary. Many counters will complete step 5 through out 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. Steps 3 & 4 are discussed later in this chapter and discussed in detail later in the book. 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).

## *Types of Function Point Counts:*

Function point counts can be associated with either projects or applications. There are three types of function point counts.

- Development project function point count
- Enhancement project function point count
- Application function point count
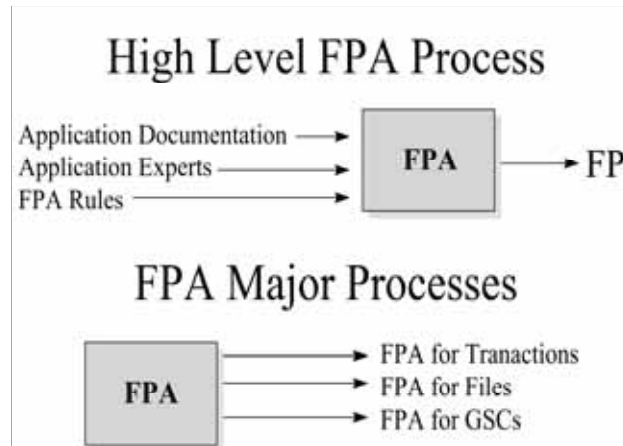


## *High Level Steps:*

- To complete a function point count knowledge of function point rules and application documentation is needed. Access to an application expert can improve the quality of the count also.

- Once the application boundary has been established, FPA can be broken into three major parts (FPA for transactional function types, FPA for data function types and FPA for GSCs).

## *Independence and Dependence:*

Since the rating of transactions is dependent on both information contained in the transactions and the number of files referenced, it is recommended that transactions are counted first. At the same time the transactions are counted a tally should be kept of all FTR's (file types referenced) that the transactions reference. It will be made clear in later chapters that every FTR must have at least one or more transactions.

## *FPA Steps for Transactional Function Types:*

Later in this document external inputs, external outputs and external inquiries are discussed in detail. Each transaction must be an elementary process. An elementary process is the smallest unit of activity that is meaningful to the end user in the business. It must be self-contained and leave the business in consistent state.

T1. Application documentation and transaction rules are used to identify transactions.

T2. The application documentation and transaction rules are used to determine type of transaction (external input, external output, or external inquiry).

T3. With the help of application documentation (data model and transaction model) and transaction rules the number data elements and file type referenced are determined.

FPA for Transactions

| Application Documentation | |
| --- | --- |
| Transaction Model | → T1. Identify Transactions |
| Data Model | → T2. Type of Transaction (EO, EI, EQ) |

| FPA Rules | |
| --- | --- |
| Transaction Rules | → T3. Number of DETs and FTRs |
| Functional Complexity | → T4. Determine Low, Ave, High |
| Tables of Weight | → T5. Values Determined |

T6. All Transactions are summed to obtain UFP for Transactions.

T4. Each identified transaction is assigned a value of low, average or high based upon type, data elements, and files referenced.

T5. A distinct numerical value is assigned based upon type and value (low, average, or high).

T6. All transactions are summed to create a transaction unadjusted function point count.
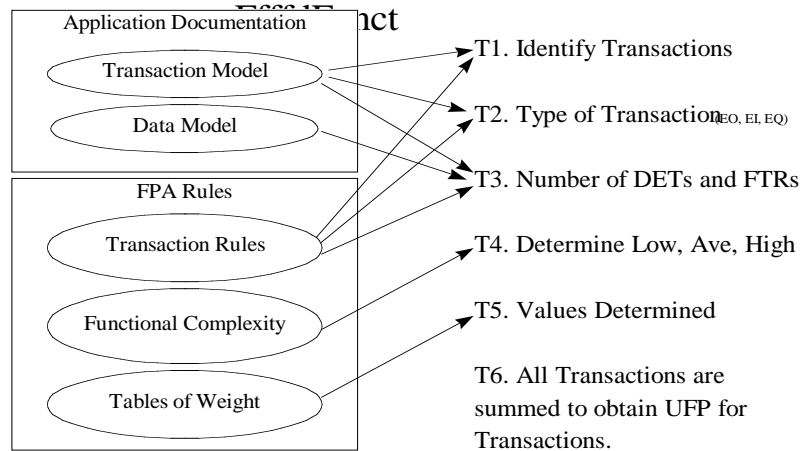
## *FPA Steps for Files:*

Later in this document internal logical files and external interface files are discussed in detail.

# FPA for Files

F1.  Application documentation and file rules are used to identify files.

F2. The application documentation (transaction model and data model) is used to determine type of file (either external interface file or internal logical file).

**Application Documentation**

Transaction Model

Data Model

**FPA Rules**

File Rules

Functional Complexity

Tables of Weight

F1. Identify Files

F2. Type of File(ILF or EIF)

F3. Number of DETs and RETs

F4. Determine Low, Ave, High

F5. Values Determined

F6. All Files are summed to obtain UFP for Files.

F3.  With the help of application documentation (data model) and file rules the number data elements and record element types are determined.

F4. Each identified file is assigned a value of low, average or high based upon type, data elements and record types.

F5. A distinct numerical value is assigned based upon type and value (low, average, or high).

F6.  All files are summed to create a file unadjusted function point count.

## *Questions:*

Is there any benefit to the sequence or order of counting function points?  That is, is there a benefit to counting transactions prior to FTR's?

Are transactions independent or dependent on FTR's?

What about FTR's?  Are they counted independent or dependent of Transactions?

# ESTABLISHING THE BOUNDARY

3

## Objective of Section:

Describe and define the concepts necessary to establish a boundary between applications.

## Definition:

Since it is common for computer systems to interact with other computer systems and/or human beings, a boundary must be drawn around each system to be measured prior to classifying components. This boundary must be drawn according to the sophisticated user's point of view. In short, the boundary indicates the border between the project or application being measured and the external applications or user domain. Once the border has been established, components can be classified, ranked and tallied.

One of the benefits of function point is analysis is creating ratios with other metrics such hours, cost, headcount, duration, and other application metrics. It is important the function point boundary be consistent with other metrics that are being gathered for the application and project.

## Identify the Boundary:

*   Review the purpose of the function point count.
*   Look at how and which applications maintain data.
*   Identify the business areas that support the applications.

The boundary may need to be adjusted once components have been identified. In practice the boundary may need to be revisited, as the overall application is better understood. Function point counts may need to be adjusted as you learn about the application.

## Standard Documentation:

*   General Specification Documents
*   Interface Documents
*   Other metric reports
*   Interviews with the users
*   User Documentation
*   Design Documentation
*   Requirements
*   Data flow diagrams

## Establishing the Boundary early in the Life cycle:

Boundaries can be established early in the software life cycle. If the application is a replacement project, then the project boundary should be similar (perhaps identical) to the previous

application. If the application is a new application, other applications boundaries should be reviewed to establish the correct boundary.

## *Technology Issues:*

### Internet/Intranet Applications

The boundary for an Internet/Intranet application is defined in a similar way for traditional applications. For traditional applications the boundary is not drawn just around the user interface or a group of screens but around the entire application. Frequently, Internet/Intranet applications are just extensions to current and existing applications. There is a tendency to create a "new" application for the Internet/Intranet extension, but this approach is incorrect.

### Client/Server

The boundaries for client/server applications need to be drawn around both the client and server. The reason is that neither the client nor server supports a users (or sophisticated) view. That is, one alone does not represent a total application. As mentioned early, any complete application needs both data at rest (server) and data in motion (client).

## *Tabulating:*

There is no special tabulating that needs to take place for establishing the boundary, but the boundary can dramatically impact the number of external inputs and external outputs.

## *Questions:*

In theory, how does making the boundary too large impact a function point count?

What if the boundary is too small?

# IDENTIFYING RET'S, DET'S, FTR'S

4

## Objective of Section:

Learn the necessary techniques to identify a RET, a DET and a FTR.  Understanding how to identify DET's and FTR's is critical to distinguish one transaction from another.  While in practice understanding the exact number of DET's and FTR's may not impact a function point count, understanding DET's and FTR's can help understand how to count function points for enhancement function point counts. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Definition:

*Record Element Type (RET)*: A RET is user recognizable sub group of data elements within an ILF or an EIF.  It is best to look at logical groupings of data to help identify them.  The concept of RET will be discussed in detail in the chapters that discuss internal logical file and external interface files.

*File Type Referenced (FTR)*: A FTR is a file type referenced by a transaction.  An FTR must also be an internal logical file or external interface file.

*Data Element Type (DET)*: A DET is a unique user recognizable, non-recursive (non-repetitive) field.   A DET is information that is dynamic and not static.   A dynamic field is read from a file or created from DET's contained in a FTR.  Additionally, a DET can invoke transactions or can be additional information regarding transactions.  If a DET is recursive then only the first occurrence of the DET is considered not every occurrence.

A data element can be either quantitative or qualitative.  A quantitative data element is data in numerical form.  A qualitative data element is data not in numerical form, but is in the form of text, photographs, sound bytes and so on.

Understanding the FTR's and DET's helped distinguish one transaction from another transactions.   This concept will be discussed in detail later in this book.

## *Rating:*

All of the components are rated based upon DET's, and either RET's or FTR's.

| Component | RET's | FTR's | DET's |
|---|---|---|---|
| External Inputs (EI) | | ✓ | ✓ |
| External Outputs (EO) | | ✓ | ✓ |
| External Inquiries (EQ) | | ✓ | ✓ |
| External Interface Files (EIF) | ✓ | | ✓ |
| Internal Logical Files (ILF) | ✓ | | ✓ |

## *Transaction DET's:*

- *External Inputs:* Data Input Fields, Error Messages, Calculated Values, Buttons
- *External Outputs:* Data Fields on a Report, Calculated Values, Error Messages, and Column Headings that are read from an ILF. Like an EQ and EO can have an input side and output sides.
- *External Inquiries:* Input Side - field used to search by, the click of the mouse. Output side - displayed fields on a screen.

## *Record Element Types (RET's):*

Record element types are one of the most difficult concepts in function point analysis. Most record element types are dependent on a parent - child relationship. In this case, the child information is a subset of the parent information. In a parent child relationship there is a one to many relationship. That is, each child piece of information is linked directly to a field on the parent file. More will be discussed about RET's in the internal logical file and external interface file sections.

## *Tips to Identify RET's and DET's early in the life cycle:*

RET's and DET's may be difficult to evaluate early in the software life cycle. Since RET's and DET's are essential to rating components, several techniques can be used to rate components.

- Rate all transactional function types and data function types as Average.
- Determine how are transactional function type and data function types rated in similar type applications. Are the majority of data function types rated as low in similar type applications?

## *DET's for GUI*

Using the strict definition of a data element provided by IFPUG's Counting Practices Manual. "A data element is a user recognizable, non recursive field." Unfortunately this does not provide much guidance when counting GUI applications. In fact, the IFPUG Counting Practices manual does not provide much detail on counting, radio buttons, check boxes, pick list, drop downs, look ups, combo boxes, so on and so forth. In GUI applications, a data element is information that is stored on an internal logical file or that is used to invoke a transaction.

## Radio Buttons

Radio Buttons are treated as data element types. Within a group of, a frame, radio buttons the user has the option of selecting only one radio button; so only one data element type is counted for all the radio buttons contained in the frame.

## Check Boxes

Check Boxes differ from radio buttons in that more than one check box can be selected at a time. Each check box, within a frame, that can be selected should be treated as a data element.

## Command Buttons

Command buttons may specify an add, change, delete or inquire action. A button, like OK, may invoke several different types of transactions.

According to IFPUG counting rules each command button would be counted as a data element for the action it invokes. In practice this data element will not impact the rating of the transaction, but it does help understand and dissect a screen full of transactions.

A button like next may actually be the input side of an inquiry or another transaction.

For example, a simple application to track distributors could have fields for Distributor Name, Address, City, State, Zip, Phone Number, and Fax Number. This would represent seven fields or (seven data elements) and the add command button would represent the eighth data element. In short, the "add" external input represent a one external input with eight data elements, the "change" external input represents another external input with eight (seven data elements plus the "change" command button), and the "delete" external input represents the last external input with eight data elements (seven fields plus the "delete" command button).

## Display of Graphical Images or Icons

A display of a graphical image is simply another data element. An inventory application, for example, may contain data about parts. It may contain part name, supplier, size, and weight and include a schematic image of the part. This schematic is treated as a single data element.

## Sound Bytes

Many GUI applications have a sound byte attached. This represents one data element. The number of notes played is simply recursive information. If the length of the sound byte increases, then the data element remains one. For example, you can play the "Star Spangled Banner" for two seconds or four seconds, but you'll still count the sound bytes as one data element. The longer it is played the more recursive information it has.

## Photographic Images

A photographic image is another data element, and is counted as one. A human resource application may display employee name, start date, etc. and a photograph of the employee. The photograph is treated the same as employee name or employee start date. The photograph is stored and maintained like any other piece of information about the employee.

Messages

There are three types of messages that are generated in a GUI application: **error messages**, **confirmation messages** and **notification messages**. Error messages and confirmation messages indicate that an error has occurred or that a process will be or have been completed. They are not an elementary or independent process alone, but they are part of another elementary process. A message that would state, "zip code is required" would be an example of an error message. A message that would state, "are you sure you want to delete customer" is an example of a confirmation message. Neither type of message is treated as a unique external output, but each is treated as a data element for the appropriate transaction.

On the other hand, a **notification messages** is a business type message. A notification is an elementary process, has some meaning to the business user and is independent of other elementary processes. It is the basis of processing and a conclusion being drawn. For example, you may try to withdraw from an ATM machine more money than you have in your account and you receive the dreaded message, "You have insufficient funds to cover this transaction." This is the result of information being read from a file regarding your current balance and a conclusion being drawn. A notification message is treated as an External Output.

## *DET's For Real Time Systems*

Using the strict definition of a data element provided by IFPUG's Counting Practices Manual. "A data element is a user recognizable, non recursive field." Unfortunately this does not provide much guidance when counting real time or embedded systems. In fact, the IFPUG Counting Practices manual does not provide any detail on counting these types of systems.

Some traditional definitions can be applied directly to real time and embedded systems. The fields on a diagnostics file: time of diagnostics, hardware state during diagnostics, temperature, voltage, so on and so forth would all be examples of data elements.

Real Time Systems may not have any "traditional user interface." That is, the stimulus for the Real Time System may be it's own output – or state. A real time or embedded systems can signal to determine current Hardware State (or location) and determine the appropriate adjustment (input) based on the current state.

## *Navigation*

Navigation is moving from one transaction to another.

## *Skill Builder:*

1. The following information is heard in the Rome Train Station. How many data elements are heard?   That is, what information varies from one train arrival to the next?

The train arriving from Florence will arrive on Track 46 at 8:30 a.m.
The train arriving from Naples will arrive on Track 43 at 11:00 a.m.

2. The totals on a particular report change ==colors== depending if the amount is ==above or below== ==$ 500.==

3. For example if the amount is -$250 it appears as **$250,** but if the amount is over 0 then the value appears blue. For example if the amount is **$1,000.** How many data elements are represented by the number and by the color?

# EXTERNAL INPUTS

<div style="text-align:right">5</div>

## Objective of Section:

Describe and define the concepts necessary to identify and rate External Inputs. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Definition:

*External Inputs* (EI) - is an elementary process in which data crosses the boundary from outside to inside. This data is coming external to the application. The data may come from a data input screen or another application. The data may be used to maintain one or more internal logical files. The data can be either control information or business information. If the data is control information it does not have to maintain an internal logical file.

If an external input adds, changes and deletes (maintains) information on an internal logical file, then this represents three external inputs. External inputs (especially change & delete) may be preceded by an external inquiry (see the section on external inquiries). Hence a full function screen is add, change, delete and inquiry (more will be discussed about inquiries later in the book).

## Rating:

Like all components, EI's are rated and scored. The rating is based upon the number of data element types (DET's) and the file types referenced (FTR's). DET's and FTR's are discussed earlier. The table below lists both the level (low, average or high) and appropriate score (3, 4 or 6).

| Files Type Referenced (FTR) | Data Elements (DET's) | | |
|---|---|---|---|
| | 1-4 | 5-15 | Greater than 15 |
| Less than 2 | Low (3) | Low (3) | Average (4) |
| 2 | Low (3) | Average (4) | High (6) |
| Greater than 2 | Average (4) | High (6) | High (6) |

## Counting Tips:

*Try to ask the question, do external inputs need more or less than 2 files to be processed?* For all the EI's that reference more than 2 FTR's, all that is needed to know is if the EI has more or less than 4 data element types referenced. If the EI has more than 4 DET's the EI will be rated as

high; less than 4 DET's the EI will be rated as average. Any EI's that reference less than 2 FTR's should be singled out and counted separately.

## *Examples:*

EI's can be business data, control data and rules based data.

*Business Data: Customer Name, Address, Phone, and so on and so forth.*

*Control Data:*

The data elements are those that invoke the transaction or change the behavior of the application. Each check box represents a data element. Additionally, the sort employee list radio buttons represents one data element as well as the time format radio buttons.
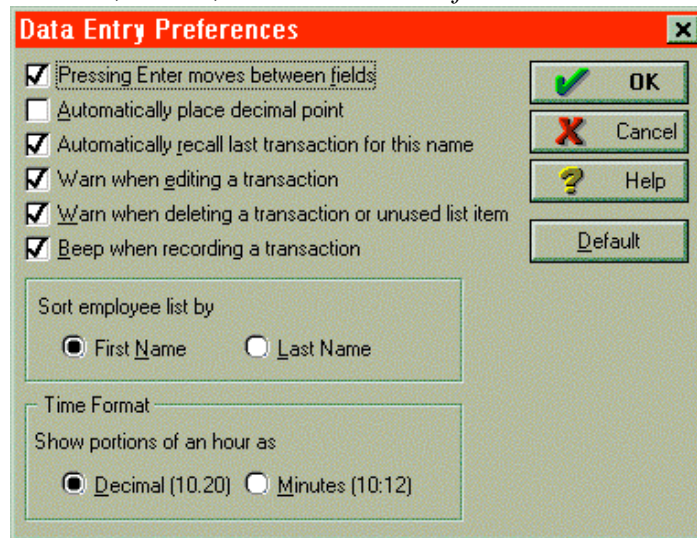
Control information changes or alters the state (or behavior) of the application. Control information specifies how, what, and when data will be processed.

## *Data Elements:*

Unique sets of data elements help distinguish external input from other external input.

- Data Input Fields
- Calculated Values or Derived Data that are stored
- Error Messages
- Confirmation Messages
- Recursive fields are only counted as one DET.
- Action keys (command buttons such as OK, Next, so on and so forth)
- Multiple Action Keys that perform the same function are counted only as one DET.

## *File Types Referenced (FTR's):*

Unique FTR's helps distinguish external input from other external input. An FTR must be either an Internal Logical File and/or External Interface File. Each internal logical file that an external input maintains is counted as an FTR. Any internal logical file or external interface file that is referenced by an external input as part of the elementary process of maintaining an internal logical file would be considered an FTR also. For example, an External Input may update an

internal logical file, but must also reference a "security file" to make sure that the user has appropriate security levels.   This would be an example of two FTR's.

# *Uniqueness:*

A unique set of data elements, and/or a different set of FTR's, and/or a unique set of calculations make one external input unique or different from other external inputs.  That is, one of the following must be true:

- Unique or different set of data elements
- Unique or different set of FTR's
- Unique or different calculations

Calculations alone are not an elementary process but part of the elementary process of the external input. A calculation (or derived data) does not make the transaction an external output. External outputs and derived data will be discussed in detail in the external output section of this document.

# *Understanding Enhancement Function Points:*

Modification of any of the items, which make an External Input unique from other external inputs, causes the EI to be "enhanced." If any of the following are true:

- DET's added to an EI
- DET's modified on an EI. The DET was included in the last FP Count.
- A New FTR
- Modifications to a calculation that appears on an EI.

# *Technology Issues:*

**GUI**
*Radio Buttons* - each set of radio buttons is counted as one DET.  Only one radio button can be selected at a time.

*Pick Lists*- The actual pick list (also known as drop downs, lookups) could be an external inquiry, but the result of the inquiry may be a DET for an external input.

*Check Box* - Each check box that can be simultaneously checked is a unique DET.

*Buttons* - Buttons can be DET's.  The OK button above would be a data element.  If there was a series of buttons Add, Change and Delete.  Each button would be counted as a DET for the associated transaction.

A single GUI "screen" may represent several transactional function types.  For example, it is common for a GUI "screen" to have a series of external inquiries followed by an external input.

*Other*

*Error Messages -* error messages are counted as data elements (DET's), not unique external inquiries. Count one DET for the entire input screen.  Multiple Error Messages are similar to recursive values.  An error message is part of another elementary process.

The number of error messages on a GUI screen is less than the number of error messages associated with traditional applications.  If used correctly, radio buttons and pick lists can force users to select correct information; therefore, eliminating the need to do editing behind the scenes.

In practice the number of DET's do not make much of a difference in evaluating an EI, understanding error or confirmation messages help in the understanding of uniqueness.

*Real Time and Embedded Systems*

In real time and embedded systems communication between hardware and software is common and should not be overlooked when counting these types of systems.  Other types of inputs for real time and embedded systems are: Operator Controls, Volume Controls, Sensor Readings, Radio Frequencies, Standards and Limit Settings (Alarms Settings, so on and so forth.

## *Standard Documentation:*

A good source of information to determine external inputs is Screen Layouts, Screen Formats & dialogs, and layouts of any input forms.  Additional inputs from other applications should be inventoried here.  Inputs from other applications must update internal logical files of the application being counted.
*   Screen Layouts
*   Screen Dialogs
*   Design Documentation
*   Functional Specifications
*   User Requirements
*   Any Input Forms
*   Context Diagrams
*   Data Flow Diagrams

## *Tips to Identify External Inputs early in the life cycle:*

The following types of documentation can be used to assist in counting EI's prior to system implementation.

*   Any refined objectives and constraints for the proposed system.
*   Collected documentation regarding the current system, if such a system (either automated or manual) exits.
*   Documentation of the users' perceived objectives, problems and needs.
*   Preliminary Data Flow Diagram.
*   Requirements Documentation.

## *Typical Vocabulary:*

The following words are associated with external input or "inputs." While reading textual document or application description look for these type of words, they may indicate an add, change or delete aspect of an external input.

| | |
|---|---|
| Add | Modify (change) |
| Activate | Override (change) |
| Amend (change and delete) | Post (add, change and delete) |
| Cancel | Remove (delete) |
| Change | Reactivate (change) |
| Convert (change) | Remit |
| Create (add) | Replace (change) |
| Delete | Revise (change and delete) |
| Deassign | Save (add, change or delete) |
| Disable | Store (add) |
| Disconnect (change or delete) | Suspend (change or delete) |
| Enable | Submit (add, change or delete) |
| Edit (change) | Update (add, change or delete) |
| Insert ( add and change) | Voids (change and delete) |
| Maintain (add, change, or delete) | |
| Memorize (add) | |

## *Skill Builder:*
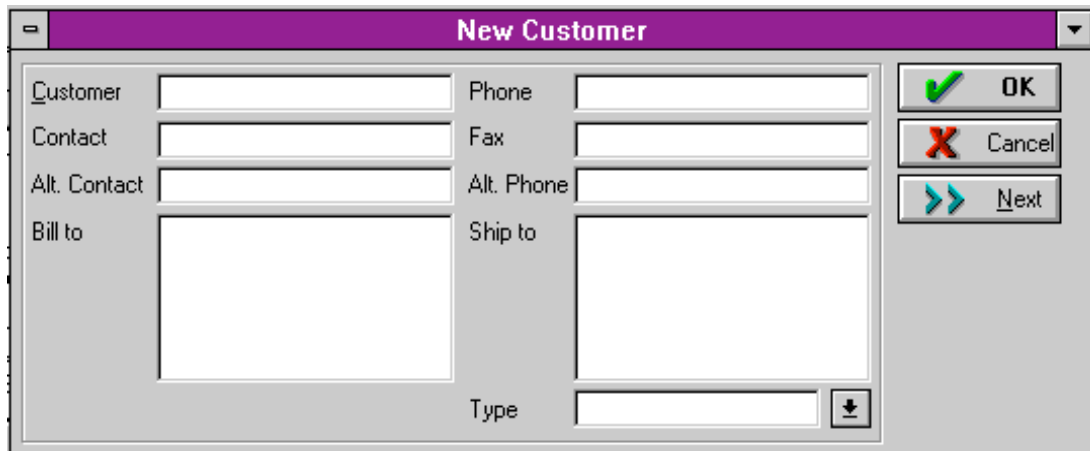
The following questions are used to help build on the concepts discussed in this section.  They are designed to encourage thought and discussion.

1.  If an EI has one file type referenced and 5 data elements is it rated, low average or high? What about 7 data elements?      Or 25 Data elements?

2.  How many data elements are there on the control input in the body of the chapter (page 29)?

3.  Does every EI have to update an ILF?       Why?

4.  What are the criteria for an EI to be rated high?

5.  Fill in the "value" of a low _____ average _____ and high _____ EI?

The following screen is used to add a new customer to an application.  The OK command button and the Next command button both add the new customer to the database.

6. How many data elements are there in this input screen?

| □ | **New Customer** | ▼ |

Customer [          ]     Phone [          ]     ✔ **OK**
Contact [          ]      Fax [          ]       ✘ Cancel
Alt. Contact [          ]  Alt. Phone [          ]  ≫ Next
Bill to [          ]      Ship to [          ]

Type [          ] ↓

7. If this screen updates one internal logical file how many unadjusted function points does this screen represent?

8. How many data elements does the phone number represent?

9. Is the Cancel command button counted as a data element?

Application A has a batch input file. The batch file is one physical file, but contains many different types of records. The first field is a record identifier number. The record identifier number can range from 1-75. The second field describes if the record is new and adds to the file, changes a previous batch input or a deletes a previous batch input (add, change and delete). Depending on the record identifier number there are a unique set of data elements, a different set of files are updated and referenced, and different processing logic is followed. Every single record identifier number updates more than 3 files (has more than 3 FTR's) and contains more than 5 data elements. How many function points does this one batch input represent?

**The answers to chapter questions are part of the online training course**
http://www.MetricsTraining.Com

# EXTERNAL OUTPUTS

6

## Objective of Section:

Describe and define the concepts necessary to identify and rate External Outputs. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Definition:

*External Outputs* (EO) - an elementary process in which **derived data** passes across the boundary from inside to outside.   Additionally, an EO may update an ILF.  The data creates reports or output files sent to other applications.  These reports and files are created from information contained in one or more internal logical files and external interface files.



External Output
(1 EO, 2 FTR's)
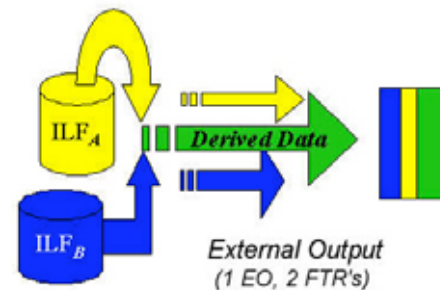
Derived Data is data that is processed beyond direct retrieval and editing of information from internal logical files or external interface files.  Derived data is usually the result of algorithms, or calculations.  *Derived data occurs when one or more data elements are combined with a formula to generate or derive an additional data element(s).* This derived data does not appear in any FTR (internal logical file or external interface file).

An algorithm is defined as a mechanical procedure for performing a given calculation or solving a problem in a series of steps.

A calculation is defined as an equation that has one or more operators.  An operator is a mathematical function such as addition, subtraction, multiplication, and division (+, -,x, /).

Transactions between applications should be referred to as interfaces.  You can only have an external output or external inquiry of data external to your application.  If you get data from another application and add it to a file in your application, this is a combination get and add (external inquiry and external input).

## Rating:

Like all components, EO's are rated and scored.   The rating is based upon the number of data elements (DET's) and the file types referenced (FTR's).  The rating is based upon the **total number of unique** (combined unique input and out sides) data elements (DET's) and the file types referenced (FTR's) (combined unique input and output sides).  DET's and FTR's were discussed earlier in this section.    The table below lists both the level (low, average or high) and appropriate score (4, 5 or 7).

| File Types Referenced (FTR) | Data Elements | | |
|---|---|---|---|
| | 1-5 | 6-19 | Greater than 19 |
| less than 2 | Low (4) | Low (4) | Average (5) |
| 2 or 3 | Low (4) | Average (5) | High (7) |
| Greater than 3 | Average (5) | High (7) | High (7) |

## Counting Tips:

*You may ask the question, Do external outputs need more or less than 3 files to be processed?* For all the EO's that reference more than 3 files, all that is needed to know is if the EO has more or less than 5 data element types. If the EO has more than 5 data element types then the EO will be rated as high, less than 5 the EO will be rated as average. Any EO's that reference less than 3 files should be singled out and counted separately.

## Terminology:

The definition states an EO contains information, which ***derived data*** passes across the boundary from inside to outside. Some confusion may arise because an EO has an input side. The confusion is the definition reads data passes across the boundary from inside to outside. The input side of an EO is search criteria, parameters, etc does not maintain an ILF. The information that a cross from outside to inside (input side) is not permanent data, but it is transient data. The intent of the information coming from outside the application (input side) is not to maintain an ILF.

## Examples:

Unlike other components EO's almost always contain business data. Rule base data and control based "outputs" are almost always considered External Inquiries. This is true due to the fact that rule data and control type data is not derived (or derivable).

*Notification Messages* are considered EO's. A notification message differs from an error message. A notification message is an elementary process, while an error message (or confirmation message) is part of an elementary process. A notification message is the result of some business logic processing. For example, a trading application may notify a broker that the customer trying to place an order does not have adequate funds in their account.

*Derived Data* displayed in textual fashion (rows and columns) and graphical format is an example of two external outputs.

## Data Elements:

Unique sets of data elements help distinguish one external output from another. Keep in mind that a DET is something that is dynamic.
(A DET is a unique user recognizable, non-recursive (non-repetitive) field)
• Error Messages

- Confirmation Messages
- Calculated Values (derived data)
- Values on reports that are read from an internal logical file or external interface file.
- Recursive values or fields (count only once)
- Generally, do not count report headings (literals) as data elements unless they are dynamic. That is, if the report headings are read from files that are maintained they may be DET's also.
- System generated dates that are on the tops or reports or are displayed are normally not counted as DET's. If system generated dates is part of business information of the external output they should be counted as DET's. For example, the date an invoice is printed or the date a check is printed.

## File Types Referenced (FTR):

Unique FTR's help distinguish one external output from another. An FTR must be either an Internal Logical File and/or External Interface File.

The elementary process associated with an external output may update an internal logical file or external interface file. For example, the elementary process that produces as payroll check may include an update to a file to set a flag to indicate that the payroll check was produced. This is not the same as maintaining the file. Maintained is the process of modifying data (adding, changed and deleting) via an elementary process (via an External Input). The primary intent of an EO is not to maintain an ILF.

## Uniqueness:

A unique set of data elements, and/or a different set of FTR's, and/or a unique set of calculations makes one external output unique or different from other external outputs. That is, one of the following must be true:

- Unique or different set of data elements
- Unique or different set of FTR's
- Unique or different calculations
- Unique processing logic

## Understanding Enhancement Function Points:

Modification of any of the items, which make an External Output unique from other external outputs, causes the EO to be "enhanced." If any of the following are true:

- DET's added to an EO
- DET's modified on an EO. The DET was included in the last FP Count.
- A New FTR
- Modifications to a calculation that appears on an EO.

## *Technology Issues:*

Each media that a report is sent to is counted as a unique EO. If a report were available on line, paper and electronic it would be counted as three EO's. Now some organizations choose to count this as only one EO. Whatever decision is made, the organization needs to stick with it.

*Disk Cache*: Information that is prepared, processed, and derived and put on cache files for another application to utilize should not be overlooked. These cached files may be external outputs or external inquiries.

## *Standard Documentation:*

- Report Layouts
- Design Documentation
- Functional Specifications
- User Requirements
- Database descriptions
- Field Sizes and Formats
- Graphical Report Layouts

## *Tips to Identify External Outputs early in the life cycle:*

The following types of documentation can be used to assist in counting External Outputs prior to system implementation.

- Any refined objectives and constraints for the proposed system.
- Collected documentation regarding the current system, if such a system (either automated or manual) exits.
- Documentation of the users' perceived objectives, problems and needs.
- Preliminary Data Flow Diagrams.

## *Typical Vocabulary:*

The following words are associated with an "external outputs." While reading textual documents or application descriptions look for these types of words. They may indicate an external output. Notice these words are very similar to those words used for an External Inquiry (discussed in the next chapter).

| | |
|---|---|
| Browse | Reports |
| Display | Request |
| Get | Retrieve |
| On-lines | Seek |
| Output | Select |
| Print | View |
| Query | |

## *Special Issues and Concerns:*

### When to count DET's for Report Headings:

Report headings are counted when they are dynamic. That is, if report headings are being read from an internal logical file they should be counted as DET's.

### Can an External Output have an input side?

Since the input side is not stand-alone (independent or an elementary process) it should be considered as part of the entire external output. The FTR's and DET's used to search should be combined with unique outside DET's and FTR's for at grand total FTR's and DET's for the entire EO. In short, an external output can have an input side.

### Can an External Output Update an Internal Logical File?

An external output can update an internal logical file, but it is incorrect to say that an external output can maintain an internal logical file. The update is part of the elementary process of the external output. An external input maintains data on and ILF file. The maintain process is an elementary process alone. The definition for maintaining is discussed in the internal logical file and external input sections of this book.

### Graphs

Graphs are counted the same way as the textual EO's. That is, the graph is rated and scored based on the number of DET's and the number of FTR's. In fact, recursive information is easily seen in a graph, but can be more difficult to visualize in a text report.

There are 10 data elements in the following table
1. Days
2. Hits
3. % of Total Hits
4. User Sessions
5. Total Hits (weekday)
6. Total % (weekday)
7. Total User Sessions (weekday)
8. Total Hits (weekend)
9. Total % (weekend)
10. Total User Sessions (weekend)

| Activity Level by Day of the Week | | | |
|---|---|---|---|
| Day | Hits | % of Total Hits | User Sessions |
| 1 Sun | 1004 | 8.73% | 111 |
| 2 Mon | 1887 | 16.41% | 201 |
| 3 Tue | 1547 | 13.45% | 177 |
| 4 Wed | 1975 | 17.17% | 195 |
| 5 Thu | 1591 | 13.83% | 191 |
| 6 Fri | 2209 | 19.21% | 200 |
| 7 Sat | 1286 | 11.18% | 121 |
| Total Weekdays | 9209 | 80.08% | 964 |
| Total Weekend | 2290 | 19.91% | 232 |

Days, Hits, % of Total Hits and User Sessions all have recursive data.

The same data could be processed and presented as bar graph. But on the following bar graph there are only two data elements (user session and day of week). The bar graph is a separate

external output and is unique from the above table. In short, it provides different business slightly different information than the table.
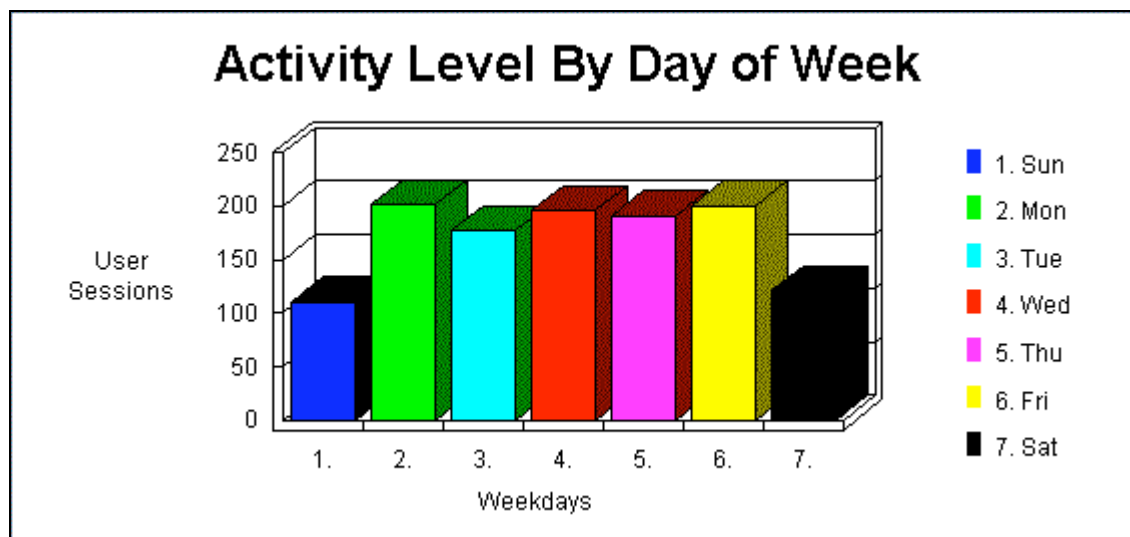


**The answers to chapter questions are part of the online training course**
http://www.MetricsTraining.Com

## Skill Builder:

The following questions are used to help build on the concepts discussed in this section. They are designed to encourage thought and discussion.

## Ice Cream Cone Sales by Month

| Flavor | Jan | Feb | Mar | Apr | May | Jun | Jul | Aug | Sep | Oct | Nov | Dec | Total |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Vanilla | 80 | 85 | 85 | 90 | 110 | 120 | 135 | 145 | 90 | 84 | 75 | 70 | 1169 |
| Chocolate | 75 | 80 | 70 | 83 | 100 | 105 | 109 | 120 | 80 | 70 | 69 | 65 | 1026 |
| Strawberry | 30 | 35 | 35 | 40 | 70 | 80 | 95 | 105 | 40 | 34 | 25 | 20 | 609 |
| Pistachio | 8 | 9 | 9 | 9 | 11 | 12 | 14 | 15 | 9 | 8 | 8 | 7 | 119 |
| Other | 12 | 13 | 13 | 13 | 15 | 17 | 19 | 20 | 14 | 13 | 13 | 12 | 174 |
| Total | 205 | 222 | 212 | 235 | 306 | 334 | 372 | 405 | 233 | 209 | 190 | 174 | |

1. How many data elements are there in the above chart?

2. Is there recursive (repetitive) information?        What is it?

3. How many data elements are there in the following line chart?    Can recursive information be seen easier in graphs?

**Ice Cone Sales by Month**
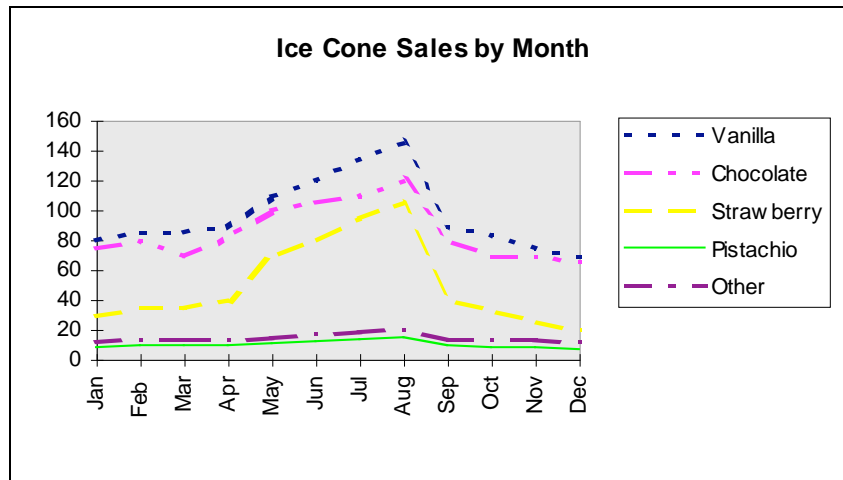


**Figure 1**

4. How many data elements are in the following chart with 2 y - axis?

**Max Average Daily Temperature in Kansas City
Data is from 1893 - Present**
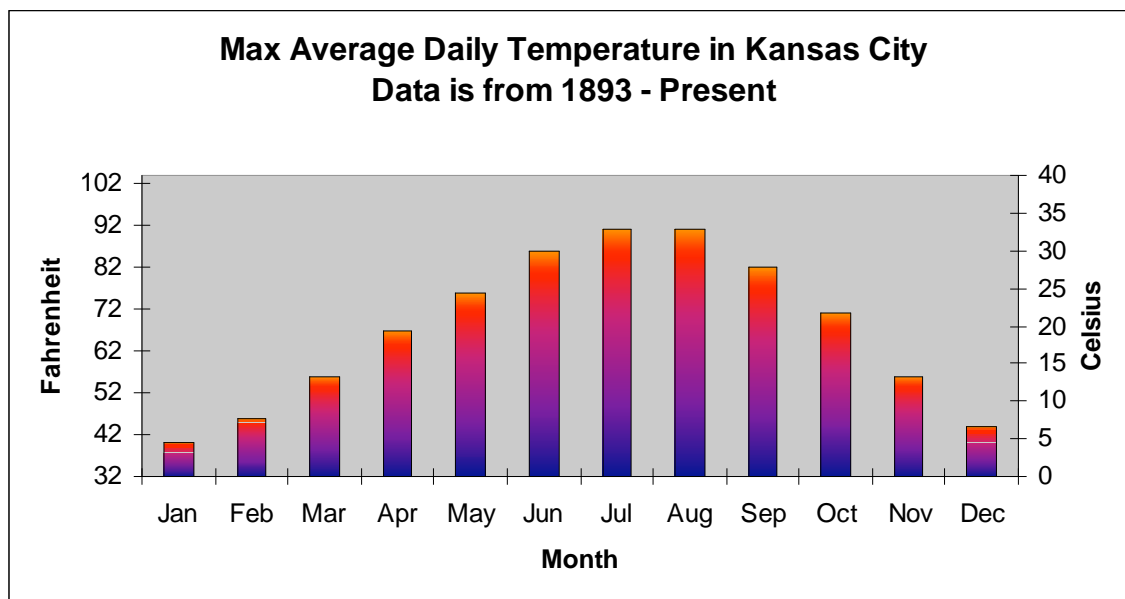
5. How many data elements are there in the following pie chart?

**Percent of Cones Sold by Flavor**
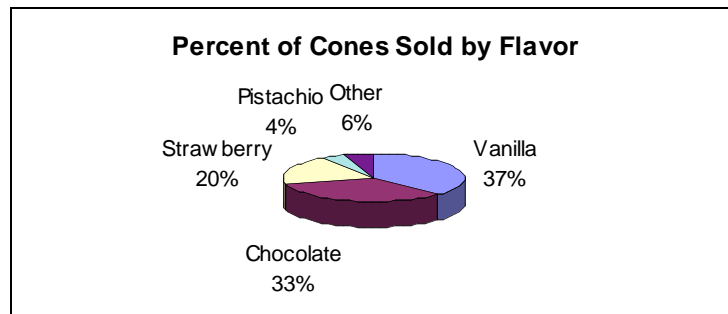


**Figure 2**

6. If an EO has 4 file types referenced and 15 data elements is it rated, low average or high?

7. What about 5 data elements with 4 FTR's?     Or 45 Data elements with 4 FTR's?

8. Is it possible to have an EO that does not reference any ILF's?     Why?

9. What is the criterion for an EO to be rated low?

10. Fill in the "value" of a low ____ average ____ and high ____ EO?  How does this compare to an EQ?     Why the difference?

11. You have a list of 25 reports and you can safely assume that each report is separate elementary processes, estimate the number of unadjusted function points.

12. You are given a list of the following 5 reports and the only information you have are the number of FTR's.
Report 1, 3 FTR's
Report 2, 5 FTR's
Report 3, 1 FTR
Report 4, 2 FTR's
Report 5, 1 FTR

Estimate the number of unadjusted function points. What method did you use?

13. How would estimate the unadjusted number of function points if you were provided the following information.

Report 1, 4 DET's
Report 2, 25 DET's
Report 3, 10 DET's
Report 4, 15 DET's
Report 5, 2 DET's

14. What method did you use?

15. Previously, the line graph of ice cream cone sales was counted as one unique External Output. If a graph were exactly the same except in Italian, would this be considered another unique external output?

16. Two separate checks are created an expense check and a payroll check. Both checks look identical and have the following fields, employee name, employee address, amount of check, date of check is printed. The expenses check uses the expenses reimbursement file and the employee file and the payroll check uses the payroll file and the employee file. The calculations for each check are different.    How many external outputs are there? Explain your answer?

**The answers to chapter questions are part of the online training course**
http://www.MetricsTraining.Com

# EXTERNAL INQUIRIES

<div style="text-align: right">**7**</div>

## Objective of Section:

Describe and define the concepts necessary to identify and rate External Inquiries. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Definition:

*External Inquiry* (EQ) - an elementary process with both input and output components that result in data retrieval from one or more internal logical files and external interface files.  The input process does not update or maintain any FTR's (Internal Logical Files or External Interface Files) and the output side does not contain derived data.



1 EQ, 1 FTR

Transactions between applications should be referred to as interfaces.  You can only have an external output or external inquiry of data external to your application.  If you get data from another application and add it to a file in your application, this is a combination get and add (external inquiry and external input).

## Rating:

Like all components, EQ's are rated and scored. Basically, an EQ is rated (Low, Average or High) like an EO, but assigned a value like and EI.   The rating is based upon the **total number of unique** (combined unique input and out sides) data elements (DET's) and the file types referenced (FTR's) (combined unique input and output sides).  DET's and FTR's were discussed in an earlier chapter.  If the same FTR is used on both the input and output side, then it is counted only one time.  If the same DET is used on both the input and output side, then it is only counted one time.

Functional Complexity Matrix  (shared table between EO and EQ)

| File Types Referenced (FTR) | Data Elements | | |
|---|---|---|---|
| | 1-5 | 6-19 | Greater than 19 |
| less than 2 | Low (3) | Low (3) | Average (4) |
| 2 or 3 | Low (3) | Average (4) | High (6) |
| Greater than 3 | Average (4) | High (6) | High (6) |

## *Examples:*

EQ's can contain business data, control data and rules based data.

*Business Applications:* An example of Business data is customer names, addresses, phone number, so on and so forth. An example of Rules Data is a table entry that tells how many days a customer can be late before they are turned over for collection.

Drop Down List (a listing of customers by name) would be an example of an EQ.

A screen full of customer address information would be an example of an EQ.

Reset (or restore) functionality where all the modified fields are reset to their saved values.  The key to understanding this an external query is the "reset to their saved values."  Clearly a table is being read.

## *Terminology:*

The definition states that an EO contains information, which ***derived data*** passes across the boundary from inside to outside.   Some confusion may arise because an EO has an input side. The confusion is the definition reads data passes across the boundary from inside to outside. The input side of an EO is search criteria, parameters, etc does not maintain an ILF.  The information that a cross from outside to inside (input side) is not permanent data, but it is transient data.  The intent of the information coming from outside the application (input side) is not to maintain an ILF.

## *Data Elements:*

Unique sets of data elements help to distinguish one external inquiry from another external inquiry.

- Input Side
    - ☆ Click of a the mouse
    - ☆ Search values
    - ☆ Action keys (command buttons)
    - ☆ Error Messages
    - ☆ Confirmation Messages (searching)
    - ☆ Clicking on the an action key
    - ☆ Scrolling
    - ☆ Recursive fields are counted only once.
- Outside
    - ☆ Values read from an internal logical file or external interface file
    - ☆ Color or Font changes on the screen
    - ☆ Error Messages
    - ☆ Confirmation Messages
    - ☆ Recursive fields are counted only once.
- The combined (unique) total input and outside DET's are used when rating EQ's.

Like an EI, action keys that perform the same function but appear multiple times are counted as only one DET.

Error Messages and confirmation messages can and do occur on either the input side and/or output side.   If a user initiates a search and a message is displayed, "*please wait searching*" is an example of a confirmation message on the input side.   The message "*all fields must be populated*" is another example of an error message on the input side.   On the other hand, if the message is "*customer not found*" is an example of an error message on the output side.  That is, the input side contained no problems.  The database was searched and the "error" has occurred on the output side of the transaction.

## File Type Referenced (FTR's):

Unique FTR's help distinguish one external inquiry from another external inquiry.

Both the input side and output side must be considered when evaluating the FTR's used by an external inquiry.  Normally they are the same but there are instances where they may not be the same.   The combined total should be used when evaluating an EQ.  For example, a security check may be done on the input side of an external inquiry.  The security check is done to make sure the user of the application has the appropriate level of authority to view the data.

## Uniqueness:

A unique set of data elements, and/or a different set of FTR's make one external inquiry unique or different from other external inquiry.  That is, one of the following must be true:



1 EQ, 3 FTR's

- Unique or different set of data elements
- Unique or different set of FTR's
- Unique processing logic

Sorting does not make on external inquiry unique from another since the data elements and FTR's are the same.
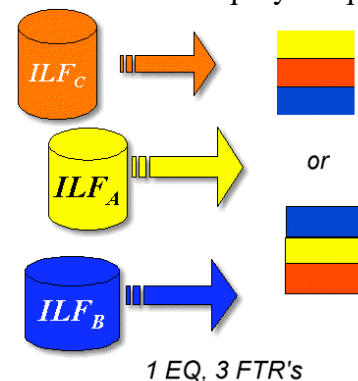
An external inquiry *cannot have calculated values or derived data*.  This characteristic distinguishes an external inquiry from an external output.

## Understanding Enhancement Function Points:

Modification of any of the items, which make an External Inquiry unique from other external inquiries, causes the EQ to be "enhanced." If any of the following are true:

- DET's added to an EQ
- DET's modified on an EQ. The DET was included in the last FP Count.
- A New FTR

**Example of Graphical Data**

Imagine the following map. There are two different ways to get the same exact data. One you can click on the specific state or you can use the drop down list. Once you choose a state data is generated and presented to the screen. These two EQ are repetitive and do the same exact thing. We would not consider this as two EQ's but only one. You can view this map by visiting http://quickfacts.census.gov/qfd/index.html

**State & County QuickFacts**

Quick, easy access to facts about people, business, and geography

To begin, select a state from this list or use the map to the right.

Alabama ▾ Go

The population of the U.S. is 281,421,906 people (April 1, 2000). The population has grown 13.1% since 1990.

View more USA QuickFacts.

See our thematic maps

Source: U.S. Census Bureau

## Technology Issues:

GUI applications are usually rich with EQ's (and EO's).
A dynamic pick list that reads from a file is an example an External Inquiry.
GUI screens my have a series of EQ's prior to an EI.

## Standard Documentation:

- Screen Layouts
- Design Documentation
- Functional Specifications
- Table Layouts
- User Requirements
- Database descriptions
- Pick lists
- Field sizes and formats

## *Tips to Identify EQ's early in the life cycle:*

The following types of documentation can be used to assist in counting internal logical files prior to system implementation.

- Any refined objectives and constraints for the proposed system.
- Collected documentation regarding the current system, if such a system (either automated or manual) exits.
- Documentation of the users' perceived objectives, problems and needs.
- Preliminary Data Flow Diagrams.

## *Typical Vocabulary:*

The following words are associated with an "external inquiry."   While reading textual document or application description look for these type of words.  They may indicate an external inquiry.  Notice the words are very similar to those related to external outputs.

| | |
|---|---|
| Browse | Query |
| Display | Scan |
| Extract | Seek |
| Fetch | Select |
| Find | Show |
| Gather | View |
| Get | Reports |
| Drop Down | |
| Lists | |
| Look Ups | |
| On-lines | |
| Output | |
| Pick Lists | |
| Print | |

## *Special Issues and Concerns:*

Can an External Inquiry not have an input side?
Even though it may not be visible all external inquiries have an input side.  In cases where the input side is not readily visible is referred to as an implied inquiry.

Can an External Inquiry Update an Internal Logical File?

Like an external output, an external inquiry may update an internal logical file, but it is incorrect to say that an external inquiry can maintains an internal logical file.  The update is part of the elementary process of the external inquiry.  The definition for maintaining is discussed in the internal logical file and external input sections of this book.  The only component that maintains an internal logical file is an external input.

## Menus (Dynamic Menus)

The menu displayed to the right is a
dynamic menu.  Word displays the last
several files that have been opened.  We
can easily conclude that this information is
being read from some type of internal file.
Hence, the information is dynamic.  The
menu would be counted as an external
inquiry.

Even though the IFPUG Manual explicitly
states that menus are not counted, in this
case it is clear that the menu is dynamic
and changes.

The real distinction is if a menu is
dynamic or static.  That is, are the contents
of the screen or report dynamic (read from
some file) or are they static (hard coded).
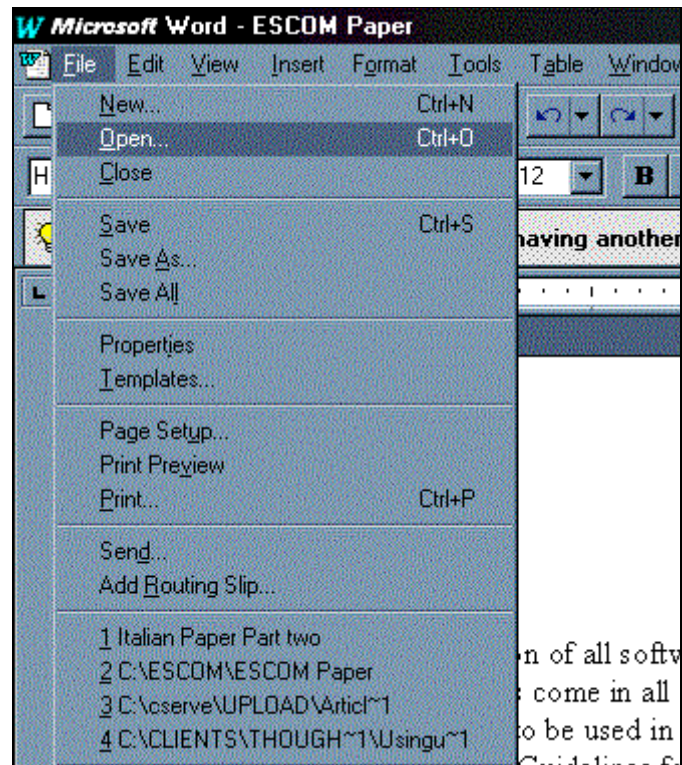
## *Skill Builder:*

The following questions are used to help build on the concepts discussed in this section.  They are designed to encourage thought and discussion.

The following customer list is displayed by clicking on the title bar "Customer."  The following list is displayed (and is read from a file).

If a particular customer is double clicked additional information is displayed.

1.  How many EQ's does the **Customer Button**, **Customer: Job List** and **Edit Customer** represent?

If **Customer:Job** is clicked then the following menu is displayed.  If new is selected a blank (empty screen appears – same fields as **Edit Customer**).  If delete is selected on delete confirmation is displayed.
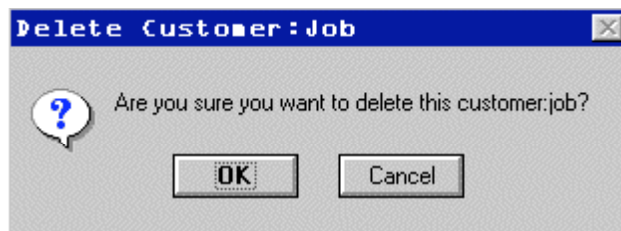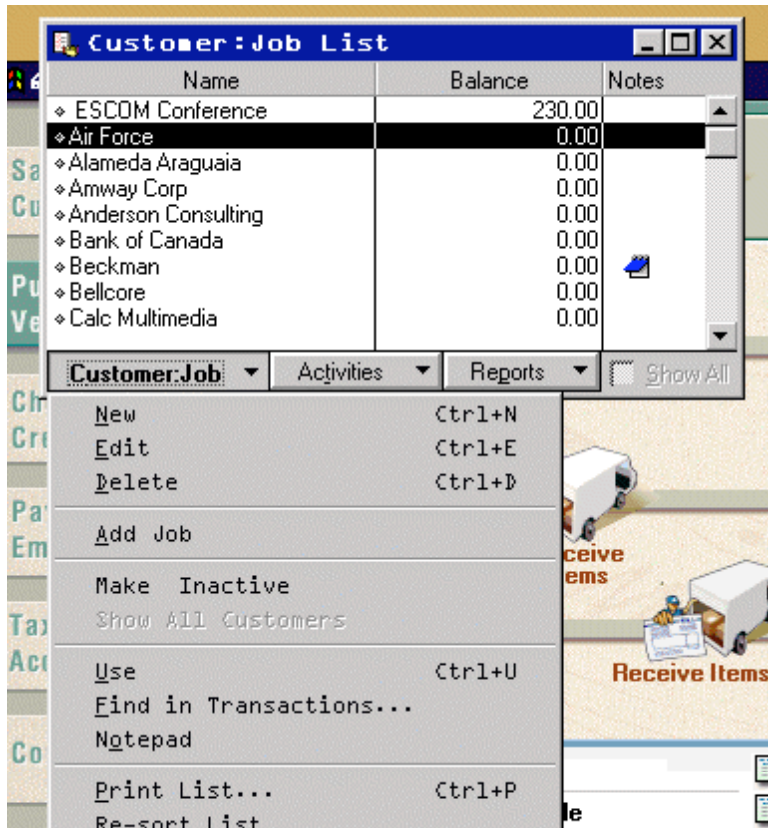
2. How many EI's does this series of screens (Edit, New and Delete) represent?


3. If an EQ references one file type and has 25 data elements is it rated, low average or high?  What about 5 data elements?     Or 45 Data elements?


4. Does every EQ have to have at least one FTR?         Why?
   How does this differ from an EO?


5. What is the criterion for an EQ to be rated high?


**The answers to chapter questions are part of the online training course**
http://www.MetricsTraining.Com

# TRANSACTION REVIEW

8

## Objective of Section:

To review the three types of transactional function type (external input, external output and external inquiry).

If the transaction can perform the "activity" then place a check in the appropriate column.

| Description or Activity | Transactions | | |
| --- | --- | --- | --- |
| | *External Input* | *External Output* | *External Inquiry* |
| DET's retrieved from FTR's | | | |
| Sorting of Data | | | |
| Updates an ILF | | | |
| Maintains an ILF | | | |
| Contains Derived Data | | | |
| Information from outside the boundary to inside | | | |
| Shares complexity matrix table | | | |
| Are valued the same for Low, Ave, and High | | | |
| Never Contains Derived Data | | | |
| At least on FTR is referenced | | | |
| Information from inside the boundary to outside | | | |

**The answers to chapter questions are part of the online training course**
http://www.MetricsTraining.Com

## Multiple Languages

Consider an application that is a single language. More than likely report headings, text descriptions are all "hard coded." That is the user cannot dynamically change the headings or the text. Now consider an application that has been developed with multiple languages in mind. The report headings, text descriptions are all read from files. Compare the following chart in

Spanish to the English chart presented earlier. Is this chart a unique external output or the same external output?

| Actividad por día de la semana | | | | |
|---|---|---|---|---|
| | Día | Aciertos | % del total | Sesiones |
| 1 | Dom | 1004 | 8.58% | 111 |
| 2 | Lun | 1887 | 16.13% | 201 |
| 3 | Mar | 1547 | 13.22% | 177 |
| 4 | Mié | 1975 | 16.88% | 195 |
| 5 | Jue | 1591 | 13.6% | 191 |
| 6 | Vie | 2271 | 19.41% | 212 |
| 7 | Sáb | 1423 | 12.16% | 148 |
| | Total para los días hábiles | 9271 | 79.25% | 976 |
| | Total para los fines de semana | 2427 | 20.74% | 259 |

The Spanish chart is not a unique external output. If external outputs are available in multiple languages then several things need to be considered. First there is probably some control input that allows the user to dynamically select the language. Second, there is an additional FTR referenced that contains the language text. Third, this language internal logical file is maintained by an external input. Fourth, there are more data elements in the report. If an external output is available in more than one language then it is not considered an unique external output, but the external output is more complex (more DET's and more FTR's).

## *Display of Graphical Images or Icons*

A display of a graphical image is simply another data element. An inventory application may contain data about parts. It may contain part name, supplier, size, and weight and include a schematic image of the part. This schematic is treated as another data element.

Another example would be a map. The map may be "hot." As the mouse pointer is moved over the map different city names are displayed. If the user clicks on a particular hot point details about that city is displayed. The details about each city are contained in an internal logical file or external interface file then the details could be an external inquiry.

The following map of the United States is "hot." If you click on Kansas City,

then you get the following information.

Kansas City, Missouri: Population 435,146: Location: 39.1 N, 94.5 W
Houston, Texas: Populations 2,231,130: Location: 29.8 N, 95.4 W
Chicago, Illinois: Population 2,783,726: Location: 41.8 N, 87.6 W

This would be an example of another inquiry.

## *Messages*

There are three types of messages that are generated in a GUI application: Error messages, Confirmation Messages and Notification Messages. An error message and a confirmation message indicate that an error has occurred or that a process will be or have been completed. A message that would state, "Zip code is required" would be an example of an error message. A message that would state, "Are you sure you want to delete the customer?" is an example of a confirmation message. Neither of these types of messages is treated as a unique External Output, but they are treated as data elements for the appropriate transaction.

On the other hand, a notification messages is a business type message. It is the basis of processing and a conclusion being drawn. For example, you may try to withdraw from an ATM machine more money than you have in your account and you receive the dreaded message, "You have insufficient funds to cover this transaction." This is the result of information being read from a file regarding your current balance and a conclusion being drawn. A notification message is treated as an External Output.

Notification Messages may be the result of processing and the actual processing or derived data my not be seen. If a message is created to be sent to a pager (beeper) at a given time. This is much like an alarm. That is current time is compared to set time and they are equal the message is sent. The pager message has one data element the text message.
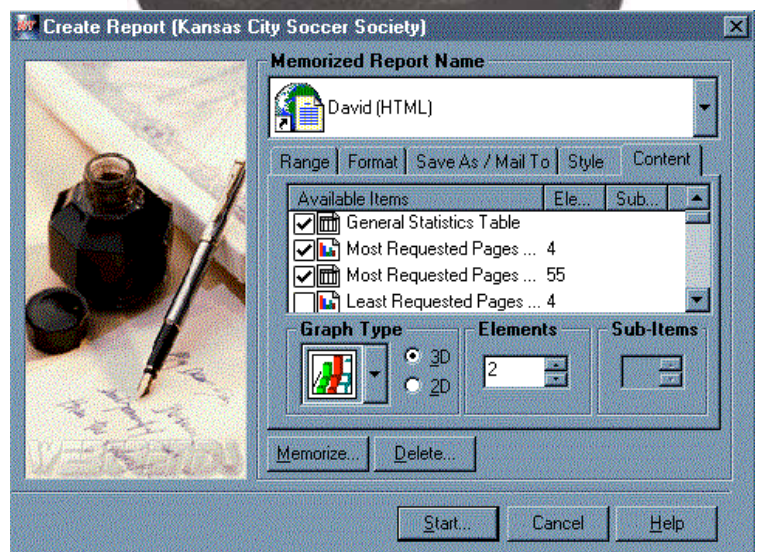
## *Complex Control Inputs*

Control inputs change the behavior of an application or the content of a report. In the "Create Report" control screen, the user has the ability to select which reports are going to be produced. This particular screen has several data element types. The check box, graph type, dimensions elements, sub-items and the action keys.

Note that the users can choose each report individually. In fact each report is as an object. The generated report is a combination of several reports (or objects). Each object has several attributes.

## *Hyperlinks on WebPages*

Many hyperlinks are nothing more than menus. In this case the meomix.com, dogchow.com are nothing more then links to other pages. In this case, they are not treated as an EI, EO or EQ.

According to the rules for an external inquiry a request must come from outside the application boundary and information must be displayed from inside to outside the application boundary. A hyperlink is just that – a hyperlink. A hyperlink is navigation to another part of the application or another Internet/Intranet site. No information crosses the boundary.

An external inquiry must reference at least one internal logical file and/or one external interface file. Both an internal logical file and an external interface file must be a logical group of related information. Imagine hyperlinking to another Website -- all the information displayed is not a logical group of information.

On the other hand, a hyperlink that sends a parameter that is used to search could be an example of an external inquiry. That is, the hyperlink follows the rules required for an external inquiry. There is an input side (the parameter) and there is an output side the results of the search. In this case the output side is dynamic and changes. This is in sharp contrast to a static hyperlink that navigates to another part of the Website.
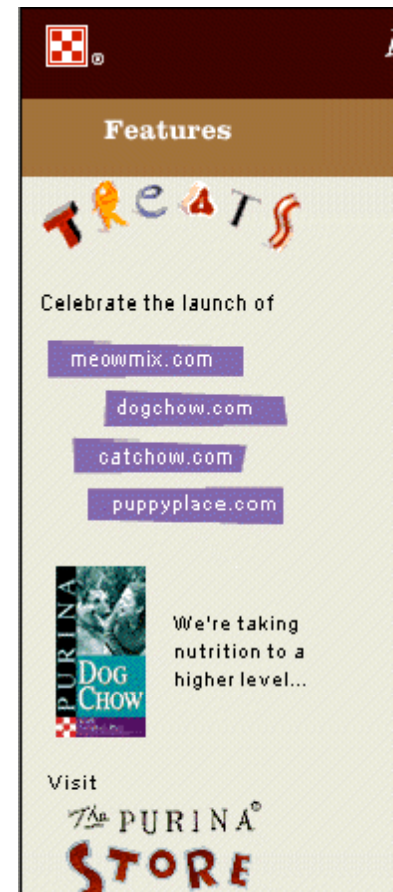
# INTERNAL LOGICAL FILES

<div style="text-align: right;">9</div>

## Objective of Section:

Describe and define the concepts necessary to identify and rate Internal Logical Files. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Definition:

*Internal Logical Files* (ILF) - a user identifiable group of logically related data that resides entirely within the application boundary and is maintained through External Inputs. An internal logical file has the inherent meaning it is internally maintained, it has some logical structure and it is stored in a file.

Even though it is not a rule, an ILF should have at least one external output and/or external inquiry. That is, at least one external output and/or external inquiry should include the ILF as an FTR. Simply put, information is stored in an ILF, so it can be used later. The EO or EQ could be from another application. It is worth noting that it is possible that a specific ILF is not referenced by EO or EQ, but it is used by an EI (other than the EI that maintains it).

Again, even though it is not a rule, an ILF should have at least one external input.

## Rating:

Like all components, ILF's are rated and scored. The rating is based upon the number of data elements (DET's) and the record types (RET's). DET's and RET's were discussed earlier. The table below lists both the level (low, average or high) and appropriate score (7, 10 or 15).

| Record Element Types (RET) | Data Elements | | |
|---|---|---|---|
| | 1 to 19 | 20 - 50 | 51 or More |
| 1 RET | Low (7) | Low(7) | Average (10) |
| 2 to 5 RET | Low (7) | Average (10) | High (15) |
| 6 or More RET | Average (10) | High (15) | High (15) |

## Counting Tips:

Determine the appropriate row first then the column. Ask the question, *do all files contain one record type or more than one record type?* If all or many of the files only contain one record type, then all that is needed to know if the file contains more or less than 50 data elements types (DET's). If the file contains more than 50 data elements the file will be rated as average, if less than 50 data element types the file will be considered low. Any files that contain more than one record type can be singled out and counted separately.

## *Examples:*

ILF's can contain business data, control data and rules based data. The type of data contained in an ILF is the same type of data an EI to contains and maintains.

It is common for control data to have only one occurrence within an ILF. For example control data file may only contain parameter settings, or a status setting. For example, part of the on board automobile system only contains current information, oil pressure, engine temperature, so on and so forth. This particular process of the on board system does not care about historical data – only the current instance. When the status changes the file is updated with current information and there is no historical information. The on board system may keep track of historical changes in diagnostics files, but this would be a totally separate process. This process is not used to keep the car running, but to help a mechanic understand what has been going on with the engine.

*Real Time and Embedded Systems:* For example, Telephone Switching is made of all three types, Business Data, Rule Data and Control Data. Business Data is the actual call, Rule Data is how the call should be routed through the network, and Control Data is how the switches communicate with each other. Like control files it is common real time systems will have only one occurrence in an internal logical file.

*Business Applications:* An example of Business data is customer names, addresses, phone number, so on and so forth. An example of Rules Data is a table entry that tells how many days a customer can be late before they are turned over for collection.

## *Record Element Types:*

The idea behind RET's is to quantify complex data relationships maintained in a single FTR.

Record element types are one of the most difficult concepts in function point analysis. Most record element types are dependent on a parent - child relationship. The child information is a subset of the parent information. In a parent child relationship there is a one to many relationship.
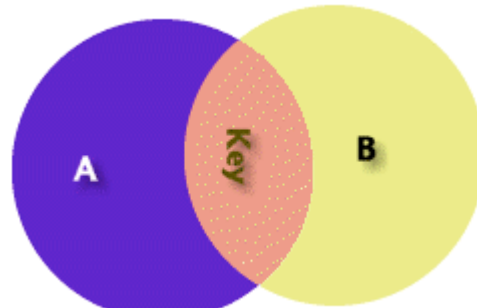


**Figure 3** (two ILF, one RET each)

Figure 3 represents two separate logical groups of data A and B. In this case *some A are B*.

Figure 4 represents one logical group of data A two record types. In this case *All B are A*.
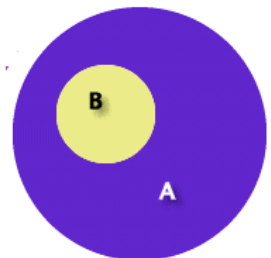


Imagine a customer file that contains Name, Address, so on and so forth. In addition all the credit cards and credit card numbers of the customer are contained in the file. This would be an example of 2 record types. There would be multiple occurrences of credit cards and numbers for each

**Figure 4** (two RET, one ILF)

customer.  The credit card and numbers are meaningless when not linked to the customer.

Additionally, a short article, *Understanding RET's* can be found at Website\Articles\ret.htm.

## Data Element Types:

Count a DET's for each unique user recognizable, nonrecursive field on the ILF or EIF.  Fields that are redundant and appear more than one time are only counted one time.   Fields that are redundant because of implementation concerns are counted only one time.

Count a DET's for each piece of data in an ILF or EIF that exists because the user requires a relationship with another ILF to be maintained (key information).  If an EIF has multiple key fields only the key fields that relate back to an ILF are counted as data element types.

## Technology Issues:

Lotus Notes refers to data stores as "forms." Powerbuilder Applications may store information on the host or client.  Count it only one time. COBOL Applications may use a variety of data stores such as IMS, DB2 etc.…  It is important to view data from the "logical model."

In Internet applications an html can be a data store if it is maintained.

## Standard Documentation:

- Table Layouts
- Database descriptions
- Logical data models
- Field sizes and formats
- Design Documentation
- Functional Specifications
- User Requirements

## Tips to Identify ILF's early in the life cycle:

The following types of documentation can be used to assist in counting internal logical files prior to system implementation.

- Any refined objectives and constraints for the proposed system.
- Collected documentation regarding the current system, if such a system (either automated or manual) exits.
- Documentation of the users' perceived objectives, problems and needs.
- Preliminary Data Models.

## Other comments:

Code maintenance may not be maintained by the application and they may not be maintained by any other application, but they exist.   The issue is that these same tables may be used by external

inquiries. A strict interpretation of the rules would not allow the inquiries to be counted. It is recommended that this type of tables be treated as external interface file.

## *Skill Builder:*

The following questions are used to help build on the concepts discussed in this section. They are designed to encourage thought and discussion.

1. If a single internal logical file is separated into 3 physical files because of implementation concerns, then how many internal logical files are counted?

2. A logical group of data is best described as?

3. If an ILF has one record type and 25 data elements is it rated, low average or high? What about 5 data elements? Or 45 Data elements?

4. Does every ILF have to have at least one EI? Why?

5. Should every ILF have at least one external output or external inquiry? Why?

6. What are the criteria for an ILF to be rated high?

7. Fill in the "value" of a low _____ average _____ and high _____ ILF? How does this compare to an EIF? Why the difference?

Examine the following tables.  The user requires detail information about customers and sales representatives.

1.  How many internal logical files?

2.  How many data elements?      Is there more than one record type?

3.  Can the tables be formed to combine one internal logical file?

**Customer Table**

| Customer Number | Name | Address | City | State | Zip Code | Balance | Credit Limit | Sales Rep Number |
|---|---|---|---|---|---|---|---|---|
| AN91 | Atwater Nelson | 215 Watkins | Oakdale | IN | 48101 | $347 | $700 | 04 |
| AW52 | Alliance West | 266 Ralston | Allanson | IN | 48102 | $49 | $400 | 07 |
| BD22 | Betodial | 542 Prairie | Oakdale | IN | 48101 | $57 | $400 | 07 |
| CE76 | Carson Enterprise | 96 Prospect | Bishop | IL | 61354 | $425 | $900 | 11 |

**Sales Representative Table**

| Sales Rep Number | Last Name | First Name | Address | City | State | Zip Code | Area Manager Number |
|---|---|---|---|---|---|---|---|
| 04 | Right | Mike | 95 Stockton | Oakdale | IN | 48101 | 14 |
| 05 | Perry | Tom | 198 Pearl | Oakdale | IN | 48101 | 17 |
| 07 | Sanchez | Rachel | 867 Bedford | Benson | MI | 49246 | 17 |
| 11 | Morris | Katie | 96 Prospect | Bishop | IL | 61354 | 21 |

Imagine a database that stores information about albums.

The database is broken down as Artist, Album Name, Publication Date, and Songs.  The key to the database is both Artist and Album Name.  The field *songs* have three subset fields.  Song contains tack number, song name and length of playing time.

For example,

> Bruce Springsteen, Born to Run, Songs.   The first row of the song subset is #1, Born To Run, 4:30.

```
1. Born to Run (4:30)
2. Thunder Road (4:48)
3. Badlands (4:02)
4. River (5:00)
5. Hungry Heart (3:20)
6. Atlantic City (3:56)
7. Dancing in the Dark (4:03)
8. Born in the U.S.A. (4:41)
9. My Hometown (4:12)
10. Glory Days (3:49)
11. Brilliant Disguise (4:15)
12. Human Touch (5:10)
13. Better Days (3:44)
14. Streets of Philadelphia (3:16)
15. Murder Incorporated (4:27)
16. Secret Garden (3:57)
17. Blood Brothers (4:34)
18. This Hard Land (4:51)
```

**Figure 5 - Songs Field**

1. How many internal logical files are represented by this database?

2. How many total data elements?

3. How many total record types are there on the database?

4. What is the recursive information?

# EXTERNAL INTERFACE FILES

<div style="float:right">

# 10

</div>

## *Objective of Section:*

Describe and define the concepts necessary to identify and rate External Interface Files. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## *Definition:*

*External Interface Files* (EIF) - a user identifiable group of logically related data that is used for reference purposes only. The data resides entirely outside the application boundary and is maintained by another applications external inputs. The external interface file is an internal logical file for another application. An application may count a file as either a EIF or ILF not both. An external interface file has the inherent meaning it is externally maintained (probably by some other application), an interface has to be developed to get the data and it is stored in a file.

Each EIF included in a function point count must have at least one external output or external interface file against it. At least one transaction, external input, external output or external inquiry should include the EIF as a FTR.

Every application, which references the EIF, needs to include it in their FP Count. Some organizations have a pull theory and others have a push theory of data. The pull theory is an external application "reaching into" another applications and retrieving data. Those organizations which have push theory require applications to create interfaces (EO or EQ) which other applications read.

## *Rating:*

Like all components, EIF's are rated and scored. The rating is based upon the number of data elements (DET's) and the record types (RET's). DET's and RET's were discussed earlier in this section. The table below lists both the level (low, average or high) and appropriate score (5, 7 or 10).

| Record Element Types (RET) | Data Elements | | |
|---|---|---|---|
| | 1 to 19 | 20 - 50 | 51 or More |
| 1 RET | Low (5) | Low(5) | Average (7) |
| 2 to 5 RET | Low (5) | Average (7) | High (10) |
| 6 or More RET | Average (7) | High (10) | High (10) |

## Counting Tips:

Only count the part of the file that is used by the application being counted not the entire file. The internal logical file, of another application, that you access may have a large amount of data, but only consider the DET's and/or RET's that are used when rating an EIF.

Determine the appropriate row first then the column. Ask the question, *do all files contain one record type or more than one record type?* If all or many of the files only contain one record type, then all that is needed to know if the file contains more or less than 50 data elements types (DET's). If the file contains more than 50 data elements the file will be rated as average, if less than 50 data element types the file will be considered low. Any files that contain more than one record type can be singled out and counted separately.

## Examples:

EIF's can contain business data, control data and rules based data.

*Real Time and Embedded Systems:* For example, Telephone Switching is made of all three types, Business Data, Rule Data and Control Data. Business Data is the actual call, Rule Data is how the call should be routed through the network, and Control Data is how the switches communicate with each other.

*Business Applications:* An example of Business data is customer names, addresses, phone number, so on and so forth. An example of Rules Data is a table entry that tells how many days a customer can be late before they are turned over for collection.

## Technology Issues:

Lotus Notes refers to data stores as "forms." Client/Server Applications may store information on the host or client. Count it only one time. COBOL Applications may use a variety of data stores such as IMS, DB2 etc…. It is important to view data from the "logical model."

## Standard Documentation:

- Table Layouts
- Interface Diagrams
- Database descriptions
- Logical data models
- Field sizes and formats
- Design Documentation
- Functional Specifications
- User Requirements

## *Tips to Identify EIF's early in the life cycle:*

The following types of documentation can be used to assist in counting external interface files prior to system implementation.

- Any refined objectives and constraints for the proposed system.
- Collected documentation regarding the current system, if such a system (either automated or manual) exits.
- Documentation of the users' perceived objectives, problems and needs.
- Preliminary Data Models.

# GENERAL SYSTEM CHARACTERISTICS

## Objective of Section:

Describe and define the concepts necessary to rate the General System Characteristics (GSC's) to determine the overall Value Adjustment Factor. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Definition:

The value adjustment factor (VAF) is based on 14 general system characteristics (GSC's) that rate the general functionality of the application being counted. Each characteristic has associated descriptions to determine the degrees of influence.

## Rating:

The degrees of influence range on a scale of zero to five, from no influence to strong influence. Each characteristic is assigned the rating based upon detail descriptions provided by the IFPUG 4.1 Manual. They ratings are:

0   Not present, or no influence

1   Incidental influence

2   Moderate influence

3   Average influence

4   Significant influence

5   Strong influence throughout

## Standard Documentation:

- General Specification Documents
- Interviews with the users

## Rating GSC's early in the life cycle:

GSC's can be rated relative early in the software life cycle. In fact, if a user cannot answer these fourteen questions, then the entire project needs to be re-evaluated.

## *Tabulating:*

Once all the 14 GSC's have been answered, they should be tabulated using the IFPUG Value Adjustment Equation (VAF) --

$$VAF = 0.65 + \left[ \left( \sum_{i=1}^{14} C_i \right) / 100 \right]$$

where: $C_i$ = degree of influence for each General System Characteristic
$i$ = is from 1 to 14 representing each GSC.
$\sum$ = is summation of all 14 GSC's.

Another way to understand the formula is VAF = (65 + TDI)/100, where TDI is the sum of the results from each question. A Microsoft Excel formula would be: =0.65+SUM(A1:A14)/100 ; assuming that the values for the characteristics were in cells A1 – A14.

## *GSC's at a Glance:*

| General System Characteristic | Brief Description |
|---|---|
| 1. Data communications | How many communication facilities are there to aid in the transfer or exchange of information with the application or system? |
| 2. Distributed data processing | How are distributed data and processing functions handled? |
| 3. Performance | Did the user require response time or throughput? |
| 4. Heavily used configuration | How heavily used is the current hardware platform where the application will be executed? |
| 5. Transaction rate | How frequently are transactions executed daily, weekly, monthly, etc.? |
| 6. On-Line data entry | What percentage of the information is entered On-Line? |
| 7. End-user efficiency | Was the application designed for end-user efficiency? |
| 8. On-Line update | How many ILF's are updated by On-Line transaction? |
| 9. Complex processing | Does the application have extensive logical or mathematical processing? |
| 10. Reusability | Was the application developed to meet one or many user's needs? |
| 11. Installation ease | How difficult is conversion and installation? |
| 12. Operational ease | How effective and/or automated are start-up, back up, and recovery procedures? |
| 13. Multiple sites | Was the application specifically designed, developed, and supported to be installed at multiple sites for multiple organizations? |
| 14. Facilitate change | Was the application specifically designed, developed, and supported to facilitate change? |

## *Considerations for GUI Applications*

GSC items such as Transaction Rates, End User Efficiency, On Line Update, and Reusability usually score higher for GUI applications than on traditional applications. On the other hand, Performance, Heavily used configuration, multiple sites, will score lower for GUI applications than traditional applications.

## *Detail GSC's:*

## 1. Data Communications

The *data* and *control* information used in the application are sent or received over communication facilities. Terminals connected locally to the control unit are considered to use communication facilities. Protocol is a set of conventions, which permit the transfer, or exchange of information between two systems or devices. All data communication links require some type of protocol.

| Score As | Descriptions to Determine Degree of Influence |
|---|---|
| 0 | Application is pure batch processing or a standalone PC. |
| 1 | Application is batch but has remote data entry *or* remote printing. |
| 2 | Application is batch but has remote data entry *and* remote printing. |
| 3 | Application includes online data collection or TP (teleprocessing) front end to a batch process or query system. |
| 4 | Application is more than a front-end, but supports only one type of TP communications protocol. |
| 5 | Application is more than a front-end, and supports more than one type of TP communications protocol. |

Comments:

TCP/IP (Transmission Control Protocol/Internet Protocol). TCP/IP provides a common language for interoperation between networks that use a variety of local protocols (Ethernet, Netware, AppleTalk, DECnet and others) are examples of TP.

An application that allows query of application via a web based solution and local access would receive a value of 3.

An application that allows for the update of ILF's via the Internet and local update would receive a value of a 5.

## 2. Distributed Data Processing

Distributed data or processing functions are a characteristic of the application within the application boundary.

| Score As | Descriptions To Determine Degree of Influence |
|:---:|:---|
| 0 | Application does not aid the transfer of data or processing function between components of the system. |
| 1 | Application prepares data for end user processing on another component of the system such as PC spreadsheets and PC DBMS. |
| 2 | Data is prepared for transfer, then is transferred and processed on another component of the system (not for end-user processing). |
| 3 | Distributed processing and data transfer are online and in one direction only. |
| 4 | Distributed processing and data transfer are online and in both directions. |
| 5 | Processing functions are dynamically performed on the most appropriate component of the system. |

Comments:

Copying files from a mainframe to a local PC or copy files from an Internet or intranet would receive a value of 2.

Reading via a client or via Internet or intranet would receive a value of 3.

Reading and updating via Internet or intranet would receive a value of 4.

Depending on available resources, the application processes either local, on server, on intranet or Internet application would receive a value of 5.

## 3. Performance

Application performance objectives, stated or approved by the user, *in either* response or throughput, influence (or will influence) the design, development, installation, and support of the application.

| Score As | Descriptions To Determine Degree of Influence |
|:---:|:---|
| 0 | No special performance requirements were stated by the user. |
| 1 | Performance and design requirements were stated and reviewed but no special actions were required. |
| 2 | Response time or throughput is critical during peak hours. No special design for CPU utilization was required. Processing deadline is for the next business day. |
| 3 | Response time or throughput is critical during all business hours. No special design for CPU utilization was required. Processing deadline requirements with interfacing systems are constraining. |
| 4 | In addition, stated user performance requirements are stringent enough to require performance analysis tasks in the design phase. |
| 5 | In addition, performance analysis tools were used in the design, development, and/or implementation phases to meet the stated user performance requirements. |

Comments:
Again for a client/server or for internet/intranet application this remains the same.

## 4. Heavily Used Configuration

A heavily used operational configuration, requiring special design considerations, is a characteristic of the application.  For example, the user wants to run the application on existing or committed equipment that will be heavily used

| Score As | Descriptions To Determine Degree of Influence |
|:---:|:---|
| 0 | No explicit or implicit operational restrictions are included. |
| 1 | Operational restrictions do exist, but are less restrictive than a typical application. No special effort is needed to meet the restrictions. |
| 2 | Some security or timing considerations are included. |
| 3 | Specific processor requirements for a specific piece of the application are included. |
| 4 | Stated operation restrictions require special constraints on the application in the central processor or a dedicated processor. |
| 5 | In addition, there are special constraints on the application in the distributed components of the system. |

Comments

Does this application share hardware that is busy?.  For example, an application that shares a server with 5 other applications would need to be optimized because it shares resources with 4 other applications.

## 5. Transaction Rate

The transaction rate is high and it influenced the design, development, installation, and support of the application

| Score As | Descriptions To Determine Degree of Influence |
|---|---|
| 0 | No peak transaction period is anticipated. |
| 1 | Peak transaction period (e.g., monthly, quarterly, seasonally, annually) is anticipated. |
| 2 | Weekly peak transaction period is anticipated. |
| 3 | Daily peak transaction period is anticipated. |
| 4 | High transaction rate(s) stated by the user in the application requirements or service level agreements are high enough to require performance analysis tasks in the design phase. |
| 5 | High transaction rate(s) stated by the user in the application requirements or service level agreements are high enough to require performance analysis tasks and, in addition, require the use of performance analysis tools in the design, development, and/or installation phases. |

## 6. Online Data Entry

Online data entry and control functions are provided in the application.

| Score As | Descriptions To Determine Degree of Influence |
|---|---|
| 0 | All transactions are processed in batch mode. |
| 1 | 1% to 7% of transactions are interactive data entry. |
| 2 | 8% to 15% of transactions are interactive data entry. |
| 3 | 16% to 23% of transactions are interactive data entry. |
| 4 | 24% to 30% of transactions are interactive data entry. |
| 5 | More than 30% of transactions are interactive data entry. |

## 7. End-User Efficiency

The online functions provided emphasize a design for end-user efficiency.  The design includes:

- Navigational aids (for example, function keys, jumps, dynamically generated menus)
- Menus
- Online help and documents
- Automated cursor movement
- Scrolling
- Remote printing (via online transactions)
- Preassigned function keys
- Batch jobs submitted from online transactions
- Cursor selection of screen data
- Heavy use of reverse video, highlighting, colors underlining, and other indicators
- Hard copy user documentation of online transactions
- Mouse interface
- Pop-up windows.
- As few screens as possible to accomplish a business function
- Bilingual support (supports two languages; count as four items)
- Multilingual support (supports more than two languages; count as six items)

| Score As | Descriptions To Determine Degree of Influence |
|:---:|---|
| 0 | None of the above. |
| 1 | One to three of the above. |
| 2 | Four to five of the above. |
| 3 | Six or more of the above, but there are no specific user requirements related to efficiency. |
| 4 | Six or more of the above, and stated requirements for end-user efficiency are strong enough to require design tasks for human factors to be included (for example,  minimize key strokes, maximize defaults, use of templates). |
| 5 | Six or more of the above, and stated requirements for end-user efficiency are strong enough to require use of special tools and processes to demonstrate that the objectives have been achieved. |

## 8. Online Update

The application provides online update for the internal logical files.

| Score As | Descriptions To Determine Degree of Influence |
|:--------:|-----------------------------------------------|
| 0 | None. |
| 1 | Online update of one to three control files is included. Volume of updating is low and recovery is easy. |
| 2 | Online update of four or more control files is included. Volume of updating is low and recovery easy. |
| 3 | Online update of major internal logical files is included. |
| 4 | In addition, protection against data lost is essential and has been specially designed and programmed in the system. |
| 5 | In addition, high volumes bring cost considerations into the recovery process. Highly automated recovery procedures with minimum operator intervention are included. |

## 9. Complex Processing

Complex processing is a characteristic of the application. The following components are present.

- Sensitive control (for example, special audit processing) and/or application specific security processing

- Extensive logical processing

- Extensive mathematical processing

- Much exception processing resulting in incomplete transactions that must be processed again, for example, incomplete ATM transactions caused by TP interruption, missing data values, or failed edits

- Complex processing to handle multiple input/output possibilities, for example, multimedia, or device independence

| Score As | Descriptions To Determine Degree of Influence |
|:--------:|-----------------------------------------------|
| 0 | None of the above. |
| 1 | Any one of the above. |
| 2 | Any two of the above. |
| 3 | Any three of the above. |
| 4 | Any four of the above. |
| 5 | All five of the above. |

## 10. Reusability

The application and the code in the application have been specifically designed, developed, and supported to be usable in *other* applications.

| Score As | Descriptions To Determine Degree of Influence |
|:---:|:---|
| 0 | No reusable code. |
| 1 | Reusable code is used within the application. |
| 2 | Less than 10% of the application considered more than one user's needs. |
| 3 | Ten percent (10%) or more of the application considered more than one user's needs. |
| 4 | The application was specifically packaged and/or documented to ease re-use, and the application is customized by the user at source code level. |
| 5 | The application was specifically packaged and/or documented to ease re-use, and the application is customized for use by means of user parameter maintenance. |

## 11. Installation Ease

Conversion and installation ease are characteristics of the application.  A conversion and installation plan and/or conversion tools were provided and tested during the system test phase.

| Score As | Descriptions To Determine Degree of Influence |
|:---:|:---|
| 0 | No special considerations were stated by the user, and no special setup is required for installation. |
| 1 | No special considerations were stated by the user *but* special setup is required for installation. |
| 2 | Conversion and installation requirements were stated by the user, and conversion and installation guides were provided and tested.  The impact of conversion on the project is not considered to be important. |
| 3 | Conversion and installation requirements were stated by the user, and conversion and installation guides were provided and tested. The impact of conversion on the project is considered to be important. |
| 4 | In addition to 2 above, automated conversion and installation tools were provided and tested. |
| 5 | In addition to 3 above, automated conversion and installation tools were provided and tested. |

## 12. Operational Ease

Operational ease is characteristic of the application. Effective start-up, back-up, and recovery procedures were provided and tested during the system test phase. The application minimizes the need for manual activities, such as tape mounts, paper handling, and direct on-location manual intervention.

| Score As | Descriptions To Determine Degree of Influence |
|---|---|
| 0 | No special operational considerations other than the normal back-up procedures were stated by the user. |
| 1 - 4 | One, some, or all of the following items apply to the application. Select all that apply. Each item has a point value of one, except as noted otherwise.<br><br>Effective start-up, back-up, and recovery processes were provided, but operator intervention is required.<br><br>Effective start-up, back-up, and recovery processes were provided, but no operator intervention is required (count as two items).<br><br>The application minimizes the need for tape mounts.<br><br>The application minimizes the need for paper handling. |
| 5 | The application is designed for unattended operation. Unattended operation means *no operator intervention* is required to operate the system other than to start up or shut down the application. Automatic error recovery is a feature of the application. |

## 13. Multiple Sites

The application has been specifically designed, developed, and supported to be installed at multiple sites for multiple organizations.

| Score As | Descriptions To Determine Degree of Influence |
|---|---|
| 0 | User requirements do not require considering the needs of more than one user/installation site. |
| 1 | Needs of multiple sites were considered in the design, and the application is designed to operate only under identical hardware and software environments. |
| 2 | Needs of multiple sites were considered in the design, and the application is designed to operate only *under similar* hardware and/or software environments. |
| 3 | Needs of multiple sites were considered in the design, and the application is designed to operate *under different* hardware and/or software environments. |
| 4 | Documentation and support plan are provided and tested to support the application at multiple sites and the application is as described by 1 or 2. |
| 5 | Documentation and support plan are provided and tested to support the application at multiple sites and the application is as described by 3. |

## 14. Facilitate Change

The application has been specifically designed, developed, and supported to facilitate change.

The following characteristics can apply for the application:

- Flexible query and report facility is provided that can handle simple requests; for example, *and/or* logic applied to only one internal logical file (count as one item).

- Flexible query and report facility is provided that can handle requests of average complexity, for example, *and/or* logic applied to more than one internal logical file (count as two items).

- Flexible query and report facility is provided that can handle complex requests, for example, *and/or* logic combinations on one or more internal logical files (count as three items).

- Business control data is kept in tables that are maintained by the user with online interactive processes, but changes take effect only on the next business day.

- Business control data is kept in tables that are maintained by the user with online interactive processes, and the changes take effect immediately (count as two items).

| Score As | Descriptions To Determine Degree of Influence |
|:---:|---|
| 0 | None of the above. |
| 1 | Any one of the above. |
| 2 | Any two of the above. |
| 3 | Any three of the above. |
| 4 | Any four of the above. |
| 5 | All five of the above. |

## *Skill Builder:*

The following questions are used to help build on the concepts discussed in this section.  They are designed to encourage thought and discussion.

1. What is the value adjustment factor if all of the general system characteristics scored a value of 5 (strong influence)?

2. What is the value adjustment factor if each of the general system characteristics has no influence (a score of 0)?

3. What is the origin of the .65 in the value adjustment factor calculation?

4. What is the possible (theoretical) range of the value adjustment factor?

# *General System Characteristics – Notes Page*

# HISTORY AND IFPUG

## Objective of Section:

To provide a brief history of Function Points and describe IFPUG.

## Brief History:

Function Point Analysis was developed first by Allan J. Albrecht in the mid 1970s. It was an attempt to overcome difficulties associated with lines of code as a measure of software size, and to assist in developing a mechanism to predict effort associated with software development. The method was first published in 1979, then later in 1983. In 1984 Albrecht refined the method and since 1986, when the International Function Point User Group (IFPUG) was set up, several versions of the Function Point Counting Practices Manual have been published by IFPUG.

## Growth and Acceptance of Function Point Analysis

The acceptance of Function Point Analysis continues to grow. This is indicated by the growth of the International Function Point User Group (IFPUG). Since 1987 membership in IFPUG has grown from 100 members to nearly 600 members in 1997. Additionally, in less than six years conference attendance has grown from 125 in 1988 to over 300 by 1997. Examination of IFPUG clearly indicates that the majority of its is members are from North America, but Function Point analysis growth outside North America is strong. This is evident by the growing number of function point organizations worldwide. There are numerous affiliate organizations of IFPUG. There are affiliate organizations in Italy, France, Germany, Austria, India, The Netherlands, Australia, Japan, and several other countries.

The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## More Information about IFPUG:

More information about joining IFPUG, conferences, committees can be obtained by contacting the IFPUG.

Website: www.IFPUG.Org
Email: Ifpug@Ifpug.org

# CALCULATING ADJUSTED FUNCTION POINT

## Objective of Section:

Describe the calculations necessary for determining the final Function Point Counts. The exercises at the end of the section help the student demonstrate that they have gained the basic knowledge required.

## Understanding the Equations:

There are three sets of equations new projects (Development), existing projects (Baseline or Application) and for enhancement projects. There are two equations for the enhancement projects. The first equation accounts for size of the enhancement project while the second equation adjusts the size of the Application.

## Forget About the Equations for a Minute:

The equations can be very cumbersome and there are many variables. Forget about the exact equations for a moment.

When you develop a new application you need to know the entire size of the project. This means you would want to include the number of function points of the application plus any other function points that need to be developed. For example, you may need to develop a mini (temporary) application to assist with conversion efforts. So in the end, you would have the number of function points for the application to be installed plus any other functions you needed to develop.

When you have an enhancement project and you are going to modify an existing production application, you are concerned about two things. The first thing is the size of the actual enhancement project. How many function points is this project? The size of this project includes any added functionality, any changed functionality, and any deleted functionality. Also in an enhancement project you may have other functionality needed that is not directly part of the enhancement project.

Normally an enhancement project is the size of any (added functionality plus any changed functionality) x the value adjustment factor. The value adjustment factor normally does not change; there is normally no conversion effort, so on and so forth.

The second concern is how did the enhancement project change the actual production application. Is the existing production application larger than before? And if it is larger by how much? This would be any added functionality.   Also you would want to know of any functionality that exist before and is larger after the enhancement.

In practice the size of the existing production application will be impacted by added functionality more.   Many organizations learn that existing application size does not change much, but they are changing existing functionality.

## *Definition:*

The final Function Point Count is obtained by multiplying the VAF times the Unadjusted Function Point (UAF).   The standard function point equation is:

$$FP = UAF * VAF$$

Where:
UAF = Unadjusted Function Points
VAF = Value Adjustment Factor

## *Unadjusted Function Point:*

| Type of Component | Complexity of Components | | | |
|---|---|---|---|---|
| | Low | Average | High | Total |
| External Inputs | ___ x 3 = ___ | ___ x 4 = ___ | ___ x 6 = ___ | |
| External Outputs | ___ x 4 = ___ | ___ x 5 = ___ | ___ x 7 = ___ | |
| External Inquiries | ___ x 3 = ___ | ___ x 4 = ___ | ___ x 6 = ___ | |
| Internal Logical Files | ___ x 7 = ___ | ___ x 10 = ___ | ___ x 15 = ___ | |
| External Interface Files | ___ x 5 = ___ | ___ x 7 = ___ | ___ x 10 = ___ | |
| | | Total Number of  Unadjusted Function Points | | _____ |

## *Development Project Function Point Calculation:*

Use the following formula to calculate the development project function point count.   Notice there is an additional term CFP which is conversion function points.   Often when a new application is replacing an old application, the data must be converted.   Sometimes a "mini application" needs to be developed to assist in the conversion.  This mini application does not exist after the new application is up and running.  This is why development function point calculation is different the application function point count (see next below).

$$DFP = (UFP + CFP) * VAF$$

Where:

DFP is the development project function point count

UFP is the unadjusted function point count

CFP is the function points added by the conversion unadjusted function point count

VAF is the value adjustment factor

## *Application Function Point Count (Baseline):*

Use the following formula to establish the initial function point count for an existing application. The user is receiving functionality. There are no changes to the existing functionality or deletions of unneeded functionality. The application function point count *does not* include conversion requirements.

$$AFP = ADD * VAF$$

Additionally, this equation is used to establish the function point count for an application at any point in time.

Where:

AFP is the initial application function point count.

ADD is the unadjusted function point count of those functions that were installed by the development project. Since many enhancement projects (that were not counted) have been installed in the application, the ADD in this case represents all functionality that exists within the application boundary at a particular point in time.

VAF is the value adjustment factor of the application.

## *Enhancement Project Function Point Calculation:*

Use the following formula to calculating the size for enhancement projects.

$$EFP = [(ADD + CHGA + CFP) * VAFA] + (DEL * VAFB)$$

Where:

EFP is the enhancement project function point count.

ADD is the unadjusted function point count of those functions that were added by the enhancement project.

CHGA is the unadjusted function point count of those functions that were modified by the enhancement project. This number reflects the functions *after* the modifications.

CFP is the function point count added by the conversion.

VAFA is the value adjustment factor of the application *after* the enhancement project.

DEL is the unadjusted function point count of those functions that were deleted by the enhancement project. It is important to consider the absolute value of the DEL not the negative value.

VAFB is the value adjustment factor of the application *before* the enhancement project.


In practice:

EFP = [(ADD + CHGA + CFP) * VAFA] + (DEL* VAFB)

In practice VAFA = VAFB = VAF, so the equation becomes

EFP = (ADD + CHGA + CFP+ DEL)* VAF)

Also normally CFP = 0, so the equation simplifies further

**EFP = ((ADD + CHGA + DEL)* VAF)**


## Simplification of the equation:

To examine the equation in detail let's assume that VAFA = VAFB = 1 and CFP = 0.
Hence EFP = (ADD + CHGA + DEL).
That is, the size of an enhancement project is a summation of all added functionality, changed functionality and any deleted functionality.

In theory and in practice, each piece of the formula must be adjusted by the appropriate Value Adjustment factor. Assume now that VAFA ≠ VAFB. The added and changed after is adjusted by the VAFA, but the deleted is adjusted by the VAFB. Additionally, if CFP ≠ 0 then it should be adjusted by VAFA.

# *Application After Enhancement Project:*


**AFP = [(UFPB + ADD + CHGA) - (CHGB + DEL)] * VAFA**


Where:
UFPB = Unadjusted Function Point Count Before Enhancement.
AFP = Application Function Point Count
DEL = is the number of function points deleted (the negative value).
All other acronyms are the same as before.


Of course, an enhancement calculation can add and/or Delete functionality from the UFPB. Added functionality can be due to new components or added functionality can be due to increase in size of existing components. For example, an existing external input could go from a low to an average – valued at 3 to 4.


In Practice
Normally VAFA = VAFB = VAF, so the equation can be re-arranged

AFP = (UFPB+ADD + CHGA-CHGB – DEL)* VAF

Let's assume that CHGA = CHGB and DEL = 0.
Then AFP= (UFPB + ADD) * VAFA

## *Skill Builder:*

The following questions are used to help build on the concepts discussed in this section. They are designed to encourage thought and discussion.

1.  An application has a base unadjusted function point count of 500, a value adjustment factor of 1.10. What is the adjusted function point count?

2.  An application has 100 unadjusted function points and a value adjustment factor of 1.02. An enhancement project adds 25 function points, deletes 20 function points, and changes 15 function points (in this case assume CHGB = CHGA). The new value adjustment factor is 1.05.

3.  What is the new (after the enhancement) adjusted function point count?

4.  What is the enhancement function point count?

An application has the following:
10 Low External Inputs, 12 High External Outputs, 20 Low Internal Logical Files, 15 High External Interface Files, 12 Average External Inquiries, and a value adjustment factor of 1.10.

5.  What is the unadjusted function point count?

6.  What is the adjusted function point count?

# CASE STUDIES

## *Objective of Section:*

The Case Studies require the student to put together several pieces of knowledge together to solve the case study.   The case studies insure that the student is grasping and understanding not only individual components but also the components as they relate to each other.

While the exercises at the end of each section are intended to be guided practice, the case studies are intended to be independent practice.  The student should be able to work solve the case study working alone or in a small group without instructor guidance.

## Crossword Puzzle
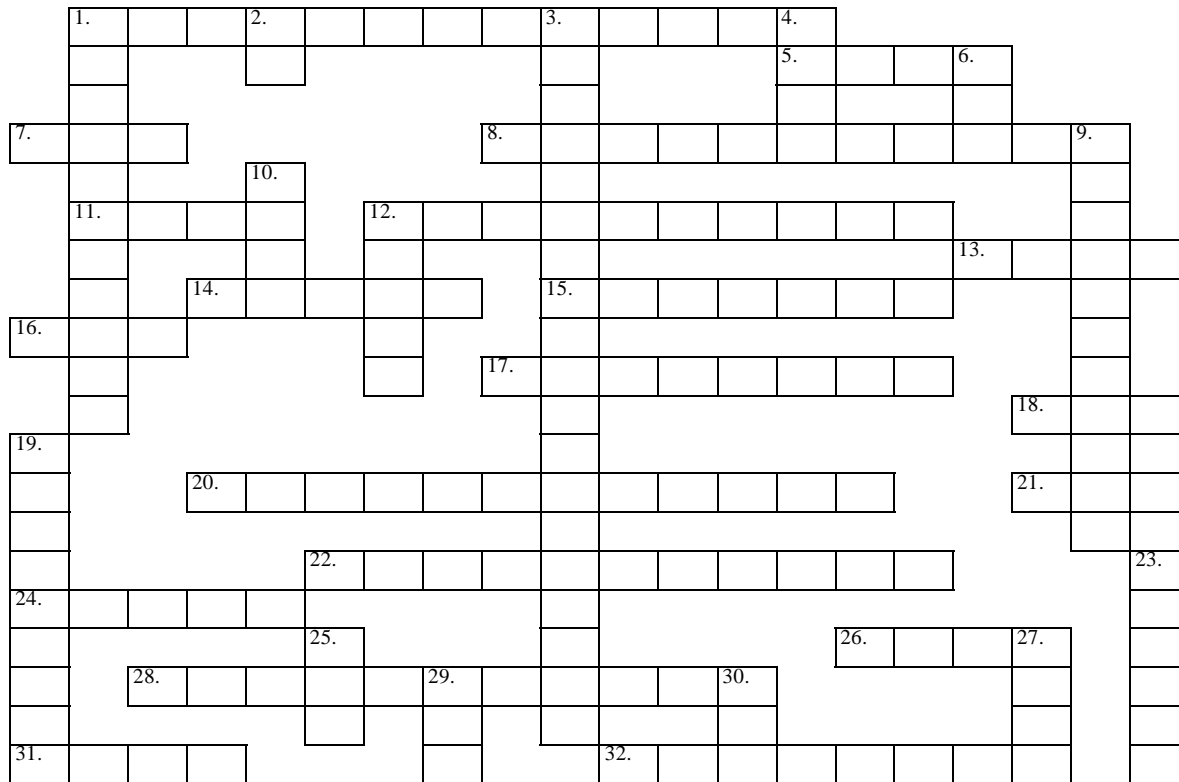
**Across**

1. Updates an ILF
5. Mountains in Northern Italy

7. The summation of the GSC's divided by 100 + .65
8. An EO contains this
11. Good Bye (Italian)
12. Ability to modify data through an elementary process

13. Contains Logical Information
14. Dracula's title
15. Not physical but
16. A unique user recognizable field
17. Establishes what functions are included in the function point count
18. Read or maintained by transaction
20. What EI's, EO's and EQ's are called
21. Another measure of software size (Abbreviation)
22. The specification, construction, testing, and delivery of a new information systems
24. International Function Point User Group (Abbreviation)
26. Function points should be counted from the " " view
28. Another word for Reused (No. 10 of 14)
31. Function points are not hard they are ..
32. The first function point count

**Down**

1. A change to a baseline application
2. From inside to outside of the boundary, contains no derived data
3. Logical Groups of Data inside boundary
4. Not to give but to…
6. Mediterranean ____
9. Collection of automated procedures and data supporting a business objective
10. Flying alone (a single person)
12. Brooks thinks this is mythical
19. A characteristic of an entity
23. Another name for a software bug
25. The set of questions that evaluate the overall complexity of an application
27. The eternal city
29. Not hello, but good ___
30. Sí (English meaning)

## Collection Letter

<today_date>

Dear <title>  <last_name>,

Our records indicate that you are past due <num_of_days>.  If you do not pay within <pay_day>, then we will kindly repo your <color><style_ automobile>.

<greeting>.

Warm Regards,
<repo_man>

### Example letter

December 18, 1999

Dear Mr. Harmon,

Our records indicate that you are past due 255 days. If you do not pay within 5 days from the date of this letter, then we will kindly repo your red Ford 150 Truck.

Please have a Merry Christmas and prosperous New Year.

Warm Regards
Rocky Balboa

### Questions and other information

- ❖ The number of past due days (num of days) is date of letter minus the due date.  Due date derived from the *Payment File*
- ❖ *Pay day* is calculated.
- ❖ *Repo Man* is read from the *Employee File*
- ❖ Title and Last Name are read from the *Customer File*
- ❖ The *greeting* is based upon the date of the letter and an appropriate message from the *Greeting File*.

What are the data elements?
Is this letter an EO or an EQ, why?
How many FTR's?

## *Control Inputs*

1. How many data elements are on the "Checking Preferences" control Screen?

2. How many data elements are on the "General Preferences" control Screen?

3. The how many control inputs are represented by the menu items to the right?

4. If the "default" reads values from a control file, then how is "default" treated?

Accounting

Checking

Finance Charge

General

Iconbar

Menus

**Checking Preferences**

- ☐ Print account names on voucher
- ☐ Change check date when check is printed
- ☐ Start with payee field on check
- ☑ Warn about duplicate check numbers

✔ OK
✗ Cancel
? Help
Default

**General Preferences**

**Desktop**
- ◉ Save when closing company
- ○ Save current desktop
- ○ Don't save the desktop

**Time Format**
Show portions of an hour as
- ◉ Decimal (10.20)
- ○ Minutes (10:12)

- ☐ Pressing Enter moves between fields
- ☐ Automatically place decimal point
- ☑ Automatically recall last transaction for this name
- ☑ Warn when editing a transaction
- ☑ Warn when deleting a transaction or unused list item
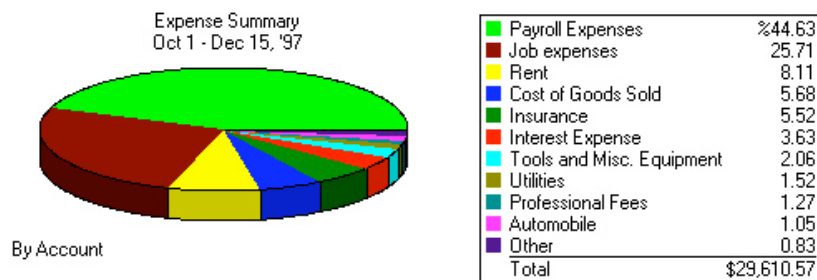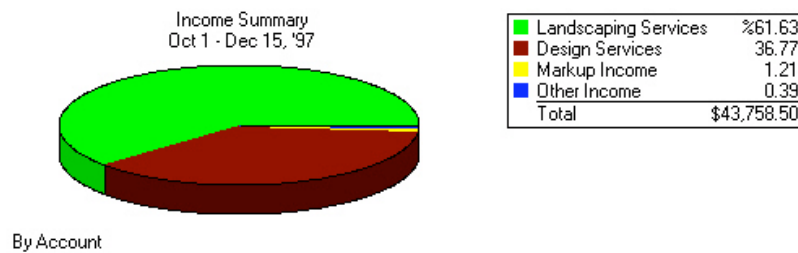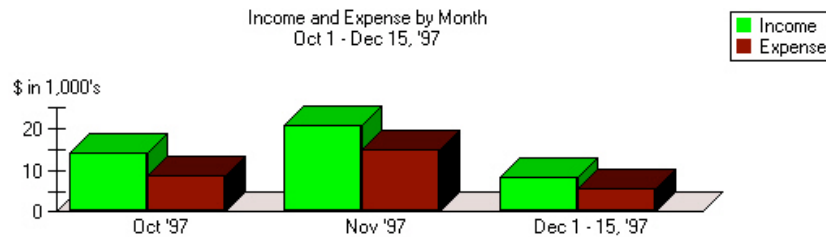- ☑ Beep when recording a transaction
- ☐ Never update name information when saving transactions
- ☐ Turn on all one time messages
- ☑ Hide Qcards for all windows

✔ OK
✗ Cancel
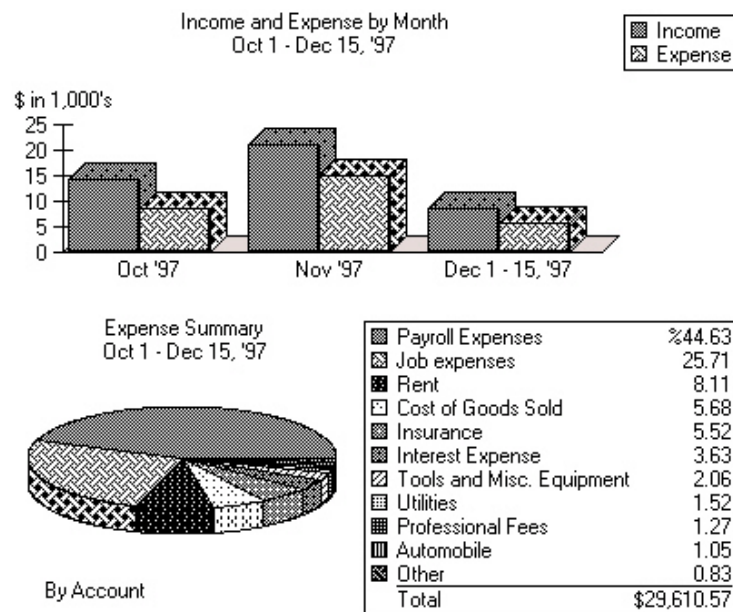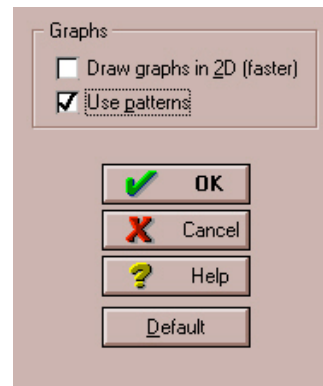? Help
Default

## *Graphical Information*

Income and Expense by Month
Oct 1 - Dec 15, '97

- Income
- Expense

$ in 1,000's

Income Summary
Oct 1 - Dec 15, '97

| | Landscaping Services | %61.63 |
|---|---|---|
| | Design Services | 36.77 |
| | Markup Income | 1.21 |
| | Other Income | 0.39 |
| | Total | $43,758.50 |

By Account

Expense Summary
Oct 1 - Dec 15, '97

| | Payroll Expenses | %44.63 |
|---|---|---|
| | Job expenses | 25.71 |
| | Rent | 8.11 |
| | Cost of Goods Sold | 5.68 |
| | Insurance | 5.52 |
| | Interest Expense | 3.63 |
| | Tools and Misc. Equipment | 2.06 |
| | Utilities | 1.52 |
| | Professional Fees | 1.27 |
| | Automobile | 1.05 |
| | Other | 0.83 |
| | Total | $29,610.57 |

By Account

1. What are the external outputs?

2. What are the data elements for each EO?

3. How are the legends treated?

## *Graphs Part II*

There are two data ILF's that contain information needed to produce the graph.  There is an additional control file which alters the way the graph looks.

1.  Is there a control EI and control ILF for "graphs"?

2.  How many total FTR's are referenced for the graphs?

3.  Does this graph represent another EO?

Graphs
- ☐ Draw graphs in 2D (faster)
- ☑ Use patterns

✔ OK
✗ Cancel
? Help
Default

Income and Expense by Month
Oct 1 - Dec 15, '97

▨ Income
▨ Expense

$ in 1,000's

Oct '97   Nov '97   Dec 1 - 15, '97

Expense Summary
Oct 1 - Dec 15, '97

| | | |
|---|---|---|
| ▨ | Payroll Expenses | %44.63 |
| ▨ | Job expenses | 25.71 |
| ▨ | Rent | 8.11 |
| ▣ | Cost of Goods Sold | 5.68 |
| ▨ | Insurance | 5.52 |
| ▨ | Interest Expense | 3.63 |
| ▨ | Tools and Misc. Equipment | 2.06 |
| ▦ | Utilities | 1.52 |
| ▦ | Professional Fees | 1.27 |
| ▥ | Automobile | 1.05 |
| ▨ | Other | 0.83 |
| | Total | $29,610.57 |

By Account

## *The Weather Application*

Release 1.0

The following application was designed to capture temperature and rainfall by city and state. There is only one input screen, one file and one report. Each field on the following input screen can be modified (add, changed or deleted). The add and change functions are different. All previous entries viewed by using the scroll bar. Assume a VAF of 1.0.



| Weather Storage File | | | | |
|---|---|---|---|---|
| City | State | Temperature | Rain Fall | Date |

| Average Temperature and Rain Fall by City and State | | | | |
|---|---|---|---|---|
| | | Temperature | Rain Fall | Date |
| City 1 | State 1 | | | |
| | *Detail Readings for City 1* | | | |
| | Averages | | | |
| City 2 | State 2 | | | |
| | *Detail Readings for City2* | | | |
| | Averages | | | |

Based on the weather application fill in the following table.  The exercise is designed to identify the exact number of data elements.

| Component (EI,EXTERNAL OUTPUT, EQ, ILF and EIF) | Number of Data Elements? | What are the data elements? |
|---|---|---|
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

What is the total unadjusted number of function points?

# *Adding A New Customer*

The following two screens are used to add a new customer to an application.  The customer is not considered added until both Address information and Additional Information is completed.

The OK and Next buttons both save information to the file.



**Figure 6**

There are four drop down list boxes on the Additional Info tab (Type, Terms, Rep and Tax Item). The first three (Type, Terms and Rep) are read from files that are maintained by the application. Tax item is hard coded. Please ignore the "Define Fields" button.



**Figure 7**

The drop down lists Type, Rep and Terms are displayed at the end of this case study.

For this part of the application please answer the following questions.

1. How many external inputs are there?

2. How many total data elements are there on the external input?

3. What are the data elements?

4. In terms of function points what are Type, Terms and Rep (see next page)?

5. In terms of function points how are Type and Terms treated the second time they appear?

6. The Rep and Terms drop down box are used again when invoices are created.

## *Enhanced Weather Application*

Release 2.0

Release 2.0 is an enhancement to "The Weather Application" Release 1.0. The user wants the ability to save temperature as either Celsius or Fahrenheit. To accomplish this a radio button is added to the input screen, which allows the user to select either Celsius or Fahrenheit. An additional field is added to the file, and an additional field is added to the reports. Assume that the value adjustment factor increases to 1.14.

How many "enhancement" function points does this represent?

What is the baseline function point of release 2.0?

# *BikeWare*

## Release 1.0

BikeWare is a software product designed for competitive bike riders.   BikeWare captures and stores a variety of information.   BikeWare is for a single rider only.  The rider wants to be able to change, add or delete information about a ride or rider.  The following information is either entered by the rider or calculated.  All **bold** items are stored. The following information is grouped logically into two major groups (ride and rider):

*Ride Information*

     **Average Speed**

     **Bike Chill Factor**

          T = Temperature during the Ride

          W = Average Speed

          X = .303439 * sqr (W) - .0202886 * W

          Bikechill = Int (91.9 - (91.4 - T) * (X + .474266))

     **Cadence**

     **Calories Burned** $= \text{Exponential}^{((.092037\ *\ Average\ Speed)\ -\ 4.26))\ *\ (Duration\ of\ Ride)\ *Weight\ of\ Rider)}$

     **Date of the Ride**

     **Distance of the Ride**

     **Duration of Ride**

     **Temperature during the Ride**

*Rider Information*

     **Age** (age of rider in years)

     **Weight** (weight of rider)

     **Sex** (sex either male or female)

*Graphs*

Four separate graphs (see below) can be created by days, by weeks or by months for each item below.  A different set of calculations will be used depending of the graph is days, weeks or a months graph.   Each graph is available on line or as a hard copy and processing logic is different.

     **Distance of Ride**

     **Average Speed**

     **Duration of Ride**

     **Calories Burned**

For BikeWare determine the following information:

Identify the external inputs, how many data elements and how many files will be referenced?

How many files type referenced are there for the add, the change and the delete?  Is it always the same?
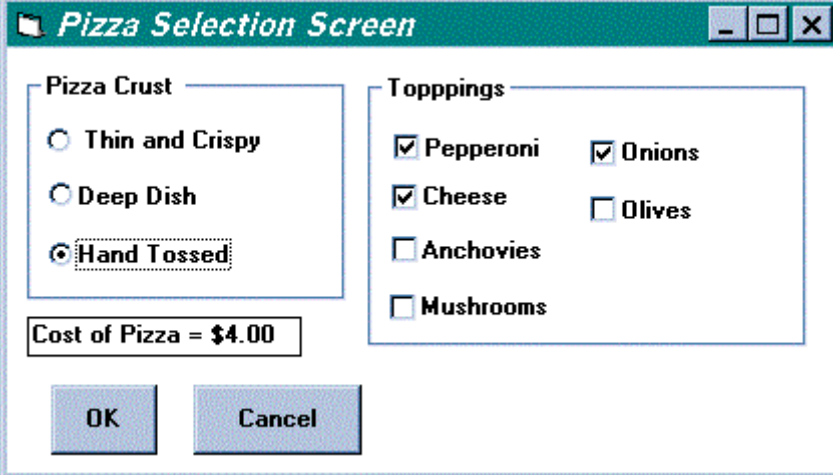
How many internal logical files are there and what are the data elements?

How many external outputs?  Describe the external outputs also?

How many data elements for each external output?

# *Pizza Screen Design*

**Option 1**



Toppings are read from another application (kitchen application). If the topping is not available it is not displayed.

The cost of the Pizza is calculated automatically.

**Figure 8**

When the OK button is clicked the Toppings, Pizza Crust Type and Cost of Pizza are saved.

**Option 2**

The Items in the drop down box are hard code – not read from a file.
Available Toppings are read from another application (kitchen application).
When a Topping is selected from Available Toppings it is copied to Selected Toppings

**Figure 9**

The Cost of the Pizza is automatically calculated.
When the OK button is clicked the Selected Toppings, Pizza Crust Type and Cost of Pizza are the saved.

What are the differences if any between Option 1 and Option 2?  Please fill in the table below.

| Option 1 | | Option 2 | |
|-----------|---------------|-----------|---------------|
| Component | Data Elements | Component | Data Elements |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

Note:  Components are external input, external inquiries, internal logical files,  and external interface files

# www.PIZZACLUB.COM

## Part 1

WWW.PizzaClub allows customers to order pizza via the Internet.  The following is only one screen of many screens.

1.  Once the customer has accessed www.PizzaClub.Com they fill out this screen.
2.  When the customer clicks on the form the information is saved to a file.
3.  If any of the fields are not filled out (populated) the customer receives and error message telling them "All fields must be populated".

4.  What are the data elements?
5.  How many unadjusted function points does this screen and one file represent.

**Netscape - [Clubs Of America -- Membership Order Form]**

File   Edit   View   Go   Bookmarks   Options   Directory   Window   Help

| Back | Forward | Home | Edit | Reload | Images | Open | Print | Find | Stop |

Go to: http://www.pizzaclub.com/

**Where do you want you Pizza Delivered?**

Name       David Longstreet

Address    2207 West Walnut

City              State              Zip Code

Blue Springs      Missouri           64015

Phone Number    (816) 220-1091        Submit Order    Cancel Order

Email Address    david@SoftwareMetrics.Com

If you prefer you can Call 1-800-MAMA-MIA

Netscape

**Figure 10**

Part 2

Www.PizzaClub.com is going to be enhanced.   Instead of allowing the customer to type city and state, they will input the zip code number.  The application will search the zip code file and then automatically populate City and State.  The customer can override the populated fields.  The zip code file is maintained by another application.
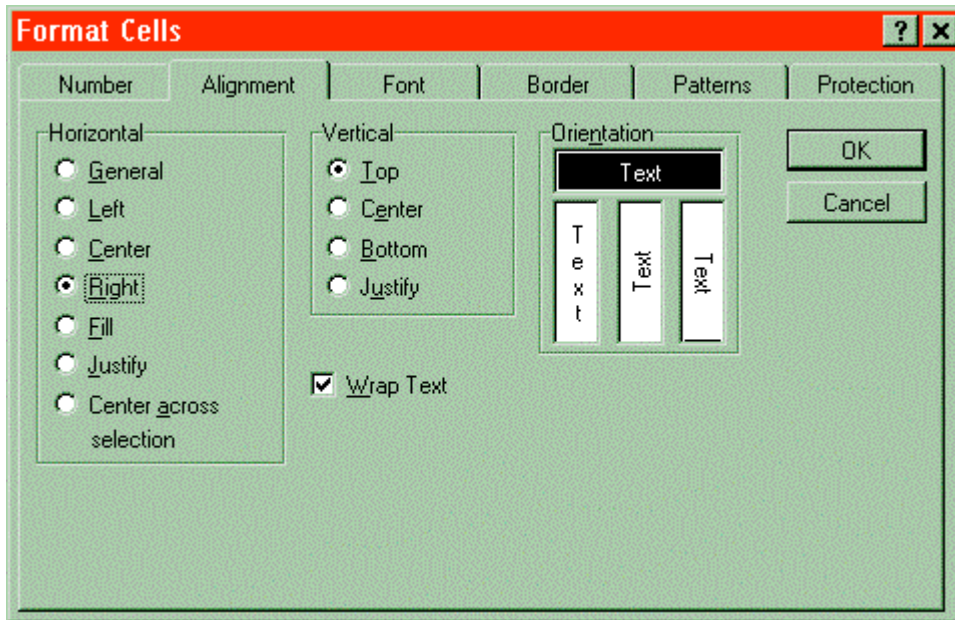
How many unadjusted function points does this enhancement represent?

What are the new components?

What data elements are impacted?

# *Control Information*

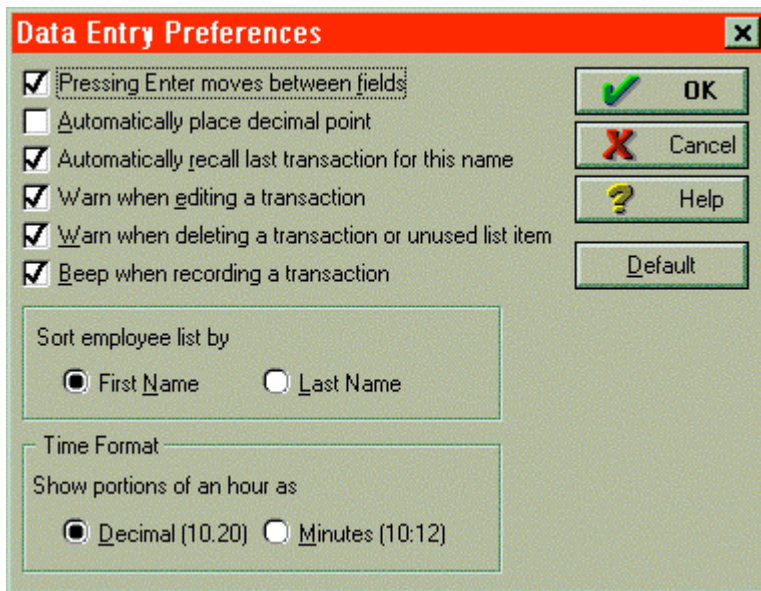What are the data elements in the following control screen (alignment)?



**Figure 11 - Control Screen**

How many data elements are there in the following "Data Entry Preferences" control screen?

If this control screen updates one internal logical file, then how many unadjusted function points does this represent?

**Figure 12**

## Word Problem 1

Let's assume your productivity rate is 10 hours per function point (it takes 10 hours of work to deliver a function point). Additionally, assume your cost per hour is $60. Therefore, the cost to deliver 1 function point is $600.

1. How much would it cost to develop an application with 5,000 function points?

2. Let's assume you anticipate a Maintenance Rate is $100/Function Point. How much needs to be budgeted to cover Maintenance Expenses for the first year?

3. Assume that the application will be operational for 6 years (application life expectancy is 6 years). Maintenance Costs will be fixed at $100/Function Point Per Year. What is the total expected cost of the application-- including all development and maintenance costs?

4. Should maintenance costs be considered when developing an application? How do you determine expected maintenance costs?

**Word Problem 2**

Assume the same cost per function point as before ($600).

Suppose a vendor, a tool vendor, and claims that his tool will increase productivity by 50 percent -- cut your cost per function point in half. Assume the following to be true:

You are planning on implementing 1,000 function points over the next year.
You want the tool to pay for itself within 1 year.

1. What is the maximum amount you would be willing to pay for this tool -- You want to break even.