Using PSP^{SM} to Develop Software Requirements & Architectural Design

SEPG 2004

Ellen George

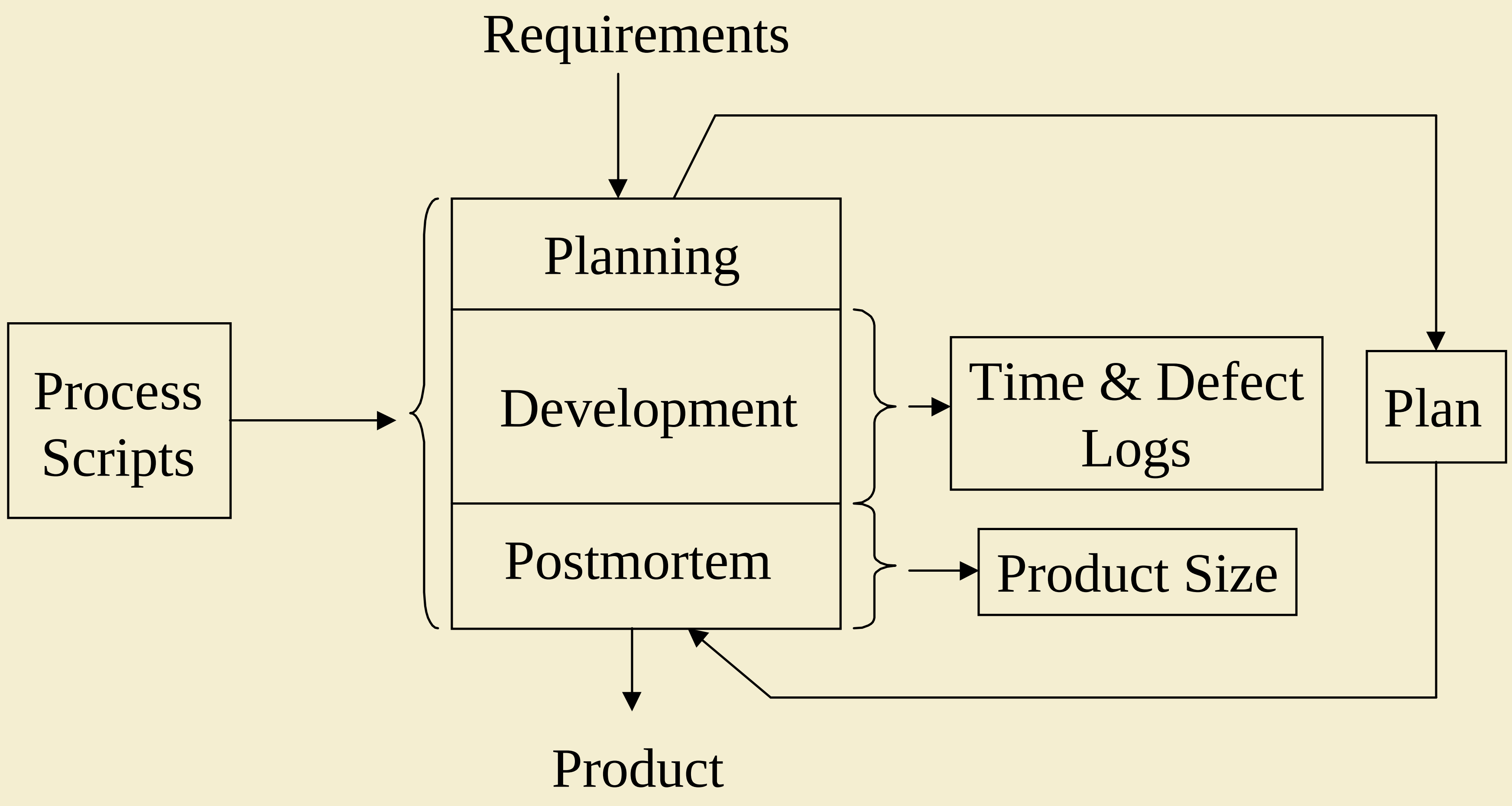
Steve Janiszewski

PS&J Software Six Sigma

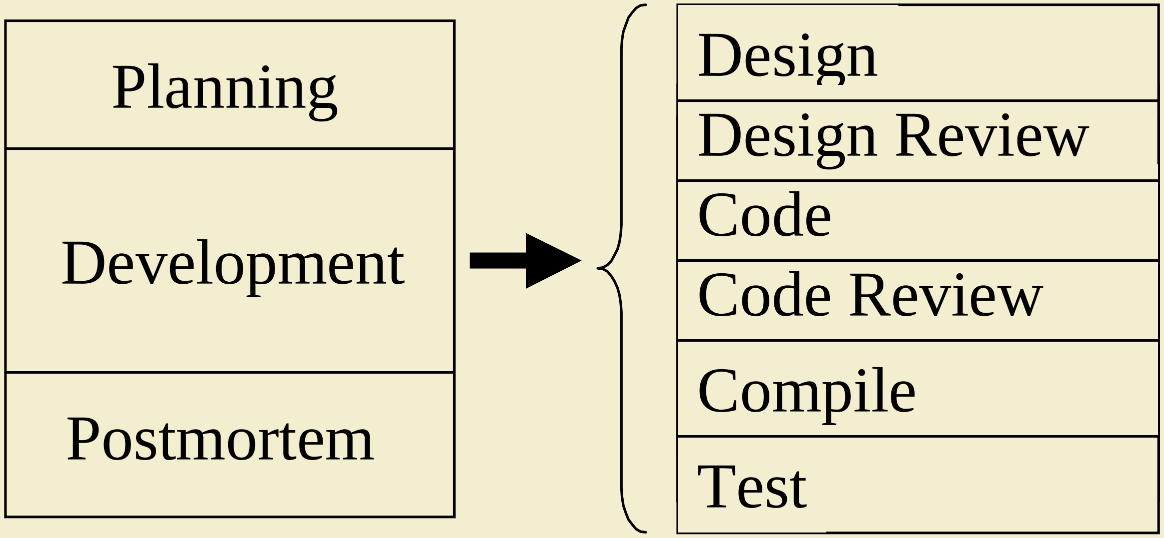
Introduction

- No guidance on the specifics of applying the Personal Software Process[SM] (PSP[SM]) to other life cycle activities is available in the standard training
- As interest in PSP[SM] grows in the community, there is a need for concrete illustrations of how to apply PSP[SM] across the entire software development life cycle
- We discuss how to extend PSP[SM] to cover other life cycle phases and illustrate with some real project examples covering requirements analysis and database architecture design

PSP^{SM} Process Flow



PSP^{SM} Across the Life Cycle



- **PSP^{SM} can be generalized to other life cycle activities by**
  - **substituting different product development & evaluation activities;**
  - **changing the size metric;**
  - **modifying estimating algorithm;**
  - **defining an appropriate defect type standard.**

Defining the Product

- Step number one is clearly defining the product produced by each life cycle phase
  - Standard PSP℠ 's product is new & modified lines of code (LOC)
  - Coding standard and line counting standard precisely define a LOC and how it should be counted
- A good product standard is the first requirement for applying PSP℠ to other activities
  - Should control product content and format
  - Products produced by the same activity should not be too dissimilar or it will be impossible to pick a useful size metric

Creating a Good Product Spec

- Template based product standards are frequently a good way to control content and format
  - Requirement to complete all template elements controls content
  - Format of the template controls the format of the product
- Adopting a commercial standard like a requirements specification language, Entity Relationship Diagrams (ERDs) or Universal Modeling Language (UML) can a good way to put structure into diagrammatic designs but frequently needs some additional specification of content and format

Excessive Product Variability

- A loose product standard will allow so much variability that it will be impossible to find a useful size metric for the product
- Content can vary from person to person and may not be consistent from product to product for the same person
- Format will vary from person to person and may not be consistent from product to product for the same person
- The standard can allow so many optional elements that there will be a large variation in content and format from product to product

Optional Content

- Example: A design standard that says design may be documented by textual description or UML diagram
- Example: A design standard that includes a required class diagram, optional textual description, an optional state diagram, and optional activity diagram
- Example: A design standard that simply says to use UML with no guidance on content or format
- Eliminate excessive optional content or break up products with lots of optional content into optional products!

Product Size Metrics

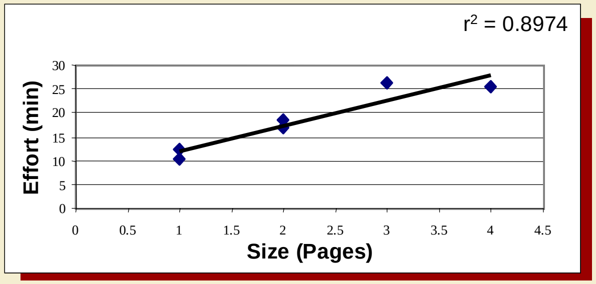
- Measuring productivity [unit product size/hr] and product quality [defects/unit product size] require a product size metric
- A good size metric will have three characteristics
  - the effort required to produce the product will be proportional to its size
  - the number of defects injected in producing the product will be proportional to its size
  - it is easy to count, preferably via automation
- The “best” size metrics have the highest degree of linearity
- If there are multiple size metrics with comparable characteristics, it is a matter of convention to pick one and use it

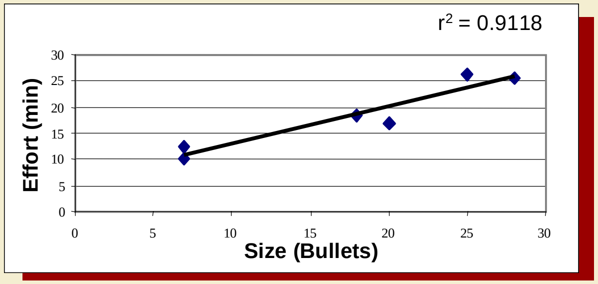
Picking a Size Metric

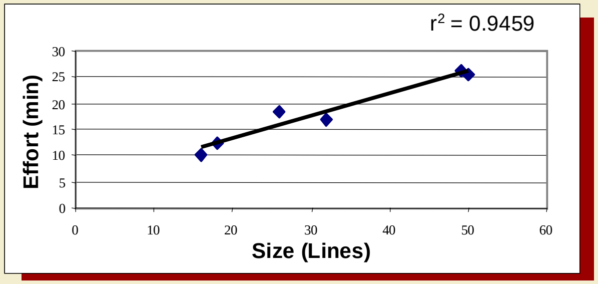
- Measure the effort required by a developer to create 5 – 10 products with sizes that span the typical range of product sizes
- Make a list of candidate size metrics & measure the size of the products with each metric
- Perform a linear regression of the effort on each of the size for each metric and select the metric that has the best fit
- Verify that the same metric works for other team member’s data
- If no candidate metrics provides an adequate fit, consider revising the product standard before looking at regression on multiple variables or higher order regression

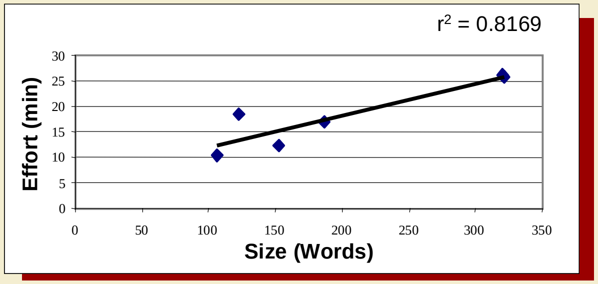
Selecting a Size Metric - Example

- Product: Product training course modules
- Product standard tightly controlled both format & content
- Format control automated via PowerPoint® master slide feature
- Homogenous product - mainly text without a significant number of complex diagrams
- All candidate metrics performed reasonably well, probably because the standard caused them to be correlated
- Lines was selected as size metric



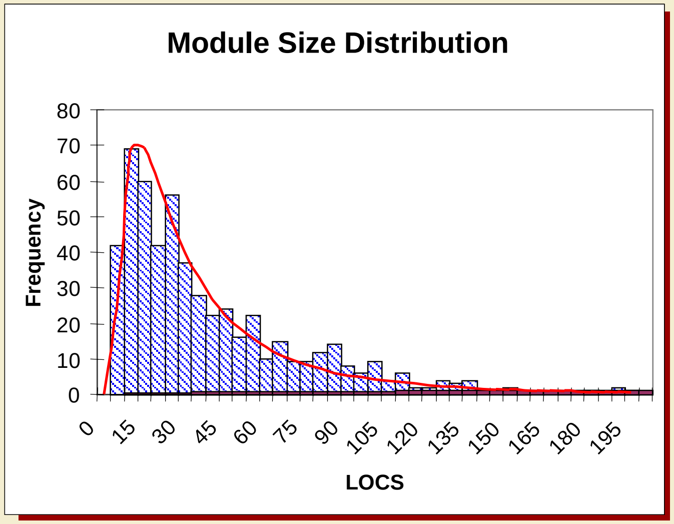


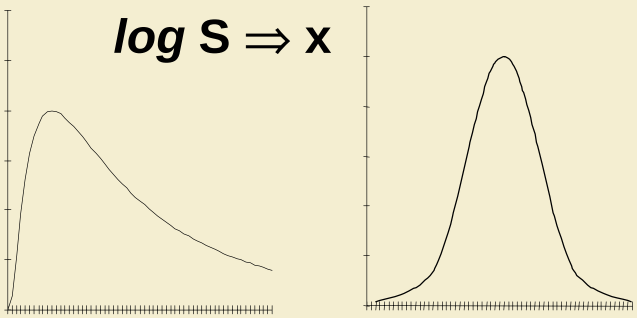




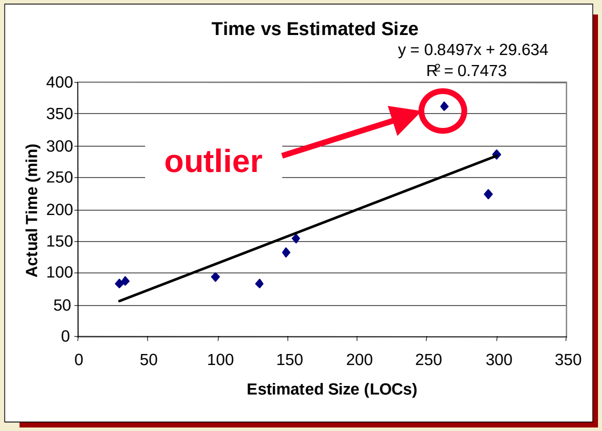
Estimating Product Size

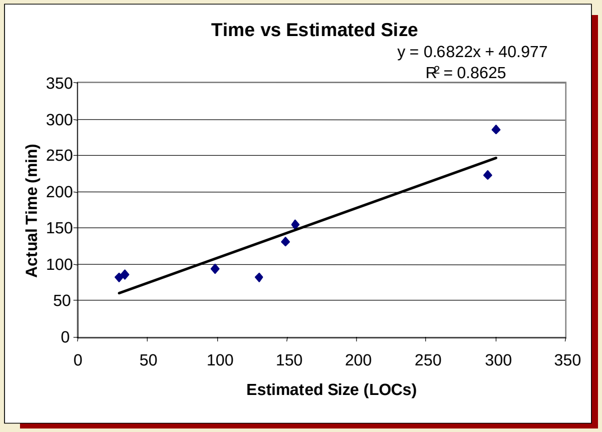
- PSP℠ uses proxy-based estimates
  - Historical distribution of product sizes of different types is used to estimate the size of similar products
  - For a normal distribution with mean μ and variance σ^2, a medium size product is estimated to be μ units, a large product $\mu + \sigma$ units, etc
- For code, the distribution of average class size/method closely follows a log normal distribution, and a transformation is used to treat it as a normal distribution

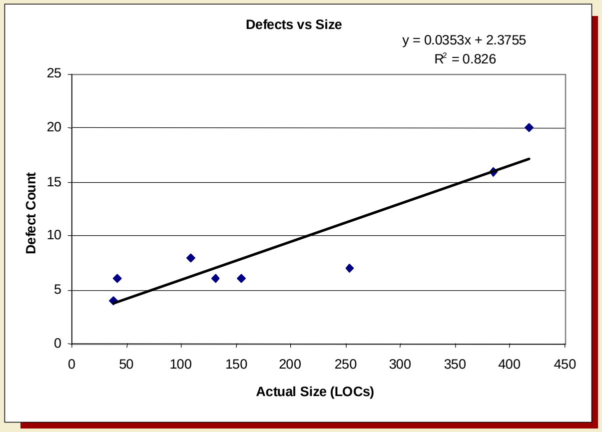




Estimating Effort and Defects







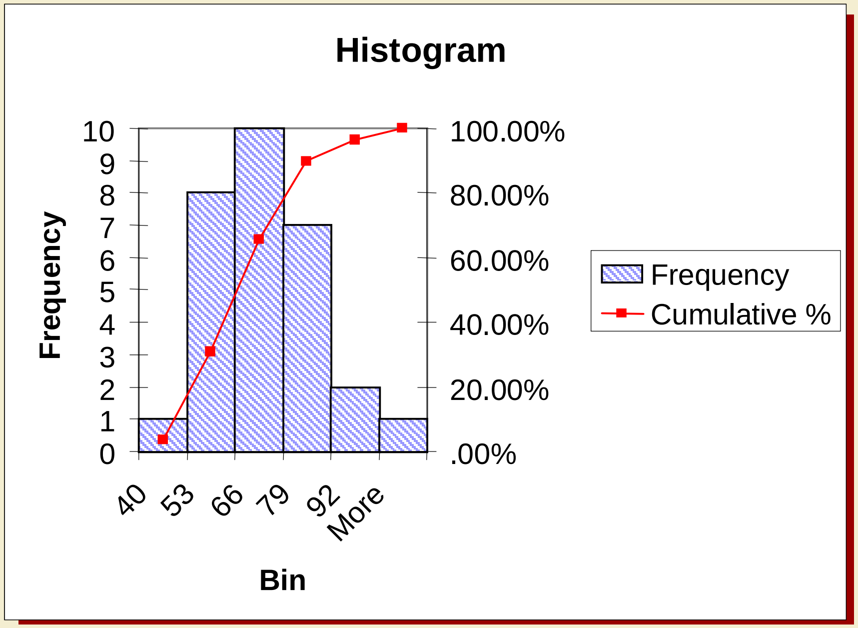
- Linear regression is used to estimate development effort from the size estimate and a second time to correct the size estimate
  - typically necessary to identify and eliminate outliers from the data set before proceeding
- Expected number of defects in a new product estimated from historical defect densities for similar products
- Applicable to most software related products

Requirements Size Metric

- Case Study: requirements specification for a large distributed system
- Requirements standard based on a template requiring specific types of textual information
- Brainstorming identified candidate size metrics: requirements, pages, paragraphs, words
- Size-Effort correlation used to screen potential size metrics
- Preliminary data indicated that any of the proposed size metrics could produce a good correlation with effort
- “Words” was tentatively selected as the size metric based on ease of automated counting
- Algorithms to estimate the size of and effort to write a section of a requirements spec were developed & validated

Requirements – Size Distribution (1)

- **After more data was available, the next step was to develop a size estimating algorithm**
- **Histogram provides a preliminary assessment of shape of requirements size distribution function**



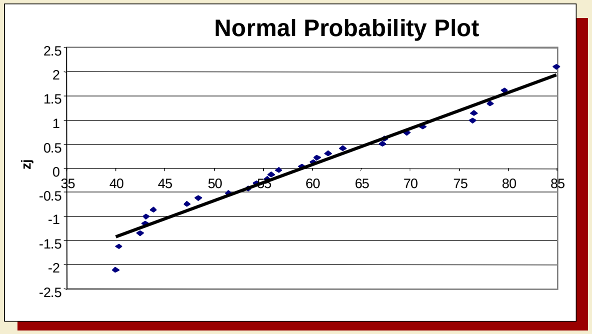
- **Not clear that it is log normal distribution like those we see for module sizes**
  - **Template format kept the minimum size of a requirement at about 40 words so distribution looks a lot more symmetrical**

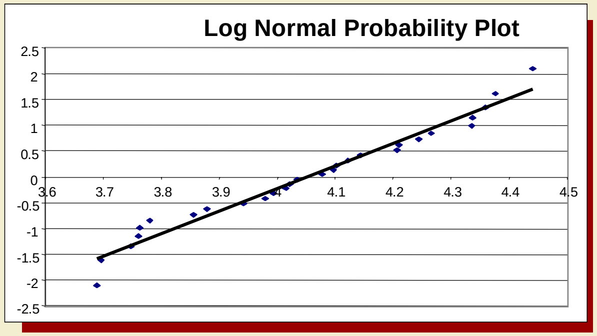
Requirements – Size Distribution (2)

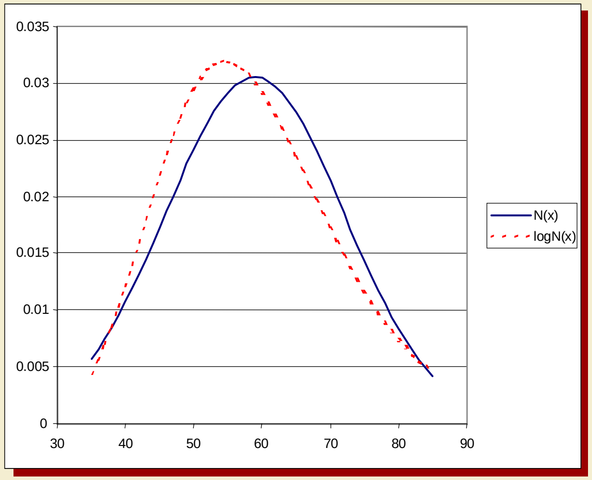
- Probability plots marginally better for normal distribution
- χ^2 test can't reject either with p-values of 0.95 and 0.93 respectively!
- Both density functions look quite similar for our data set
- Both produce similar size estimates
  - most significant difference shows up in the size of a very small requirement
  - Either are adequate as the basis of a size estimating algorithm

	Normal	LogNormal
vs	29.7	36.8
s	45.1	46.0
m	60.5	57.6
l	76.0	72.0
vl	91.4	90.0

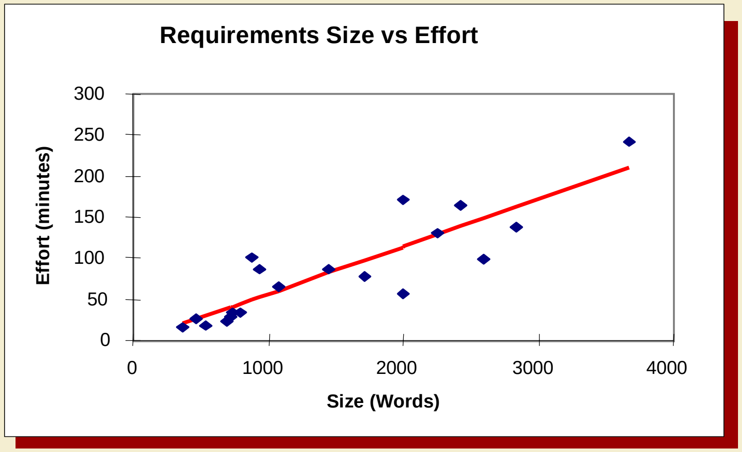
- Log Normal was selected

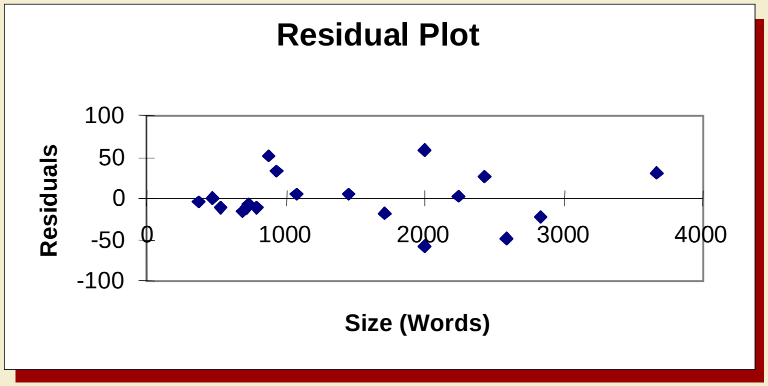






Requirements - Size Time Correlation



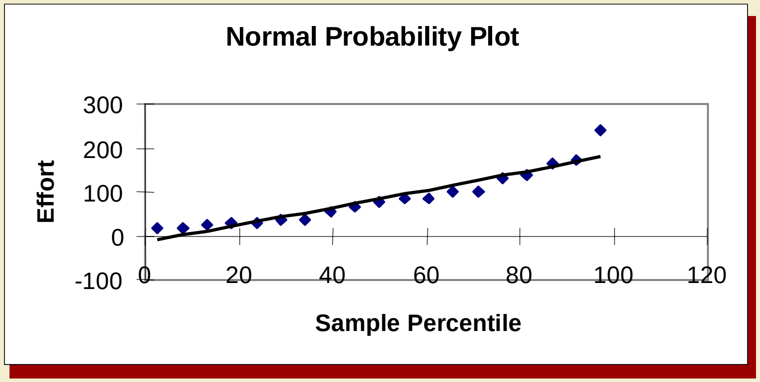


$r^2 = 0.77$

slope = 0.06 min/word

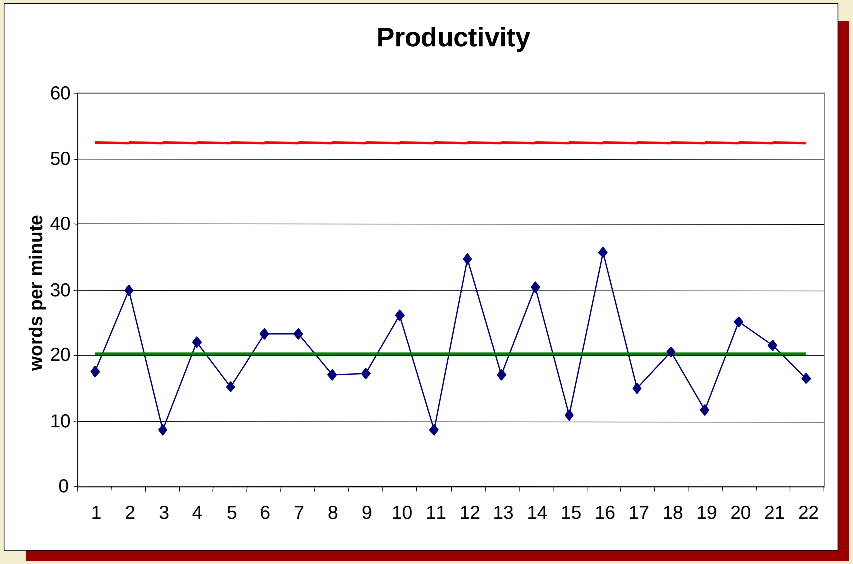
intercept ≈ 0

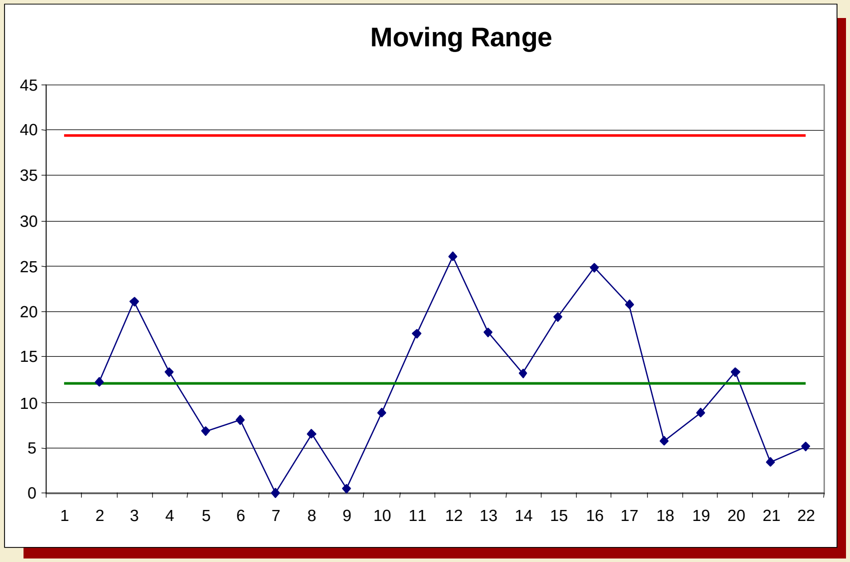
p-Value = 9×10^{-7}



- **Good correlations between size and effort for personal data resulted in straight forward application of PROBE for requirements in the 0 – 5000 word range.**

Requirements – Process Stability





- **Personal productivity data indicates a stable process with a mean of about 20 words/minute and a standard deviation of about 10.7 words/minute**

Some Observations

- PROXY based size estimating works best as a personal metric since it is sensitive to writing style
- The PROXY classification scheme can be standardized easily however
- Keep the number of product element types in the classification scheme to a minimum
  - Don't have multiple types that have essentially the same statistics
- If you don't have enough data, you can combine types until you have enough to split them apart

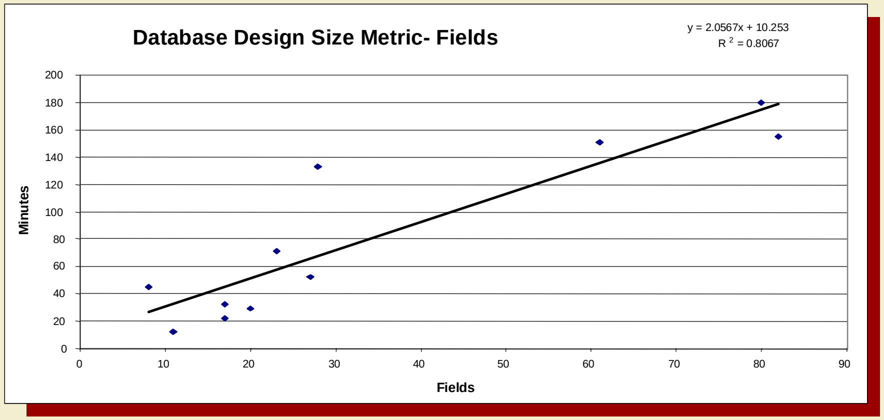
Some More Observations

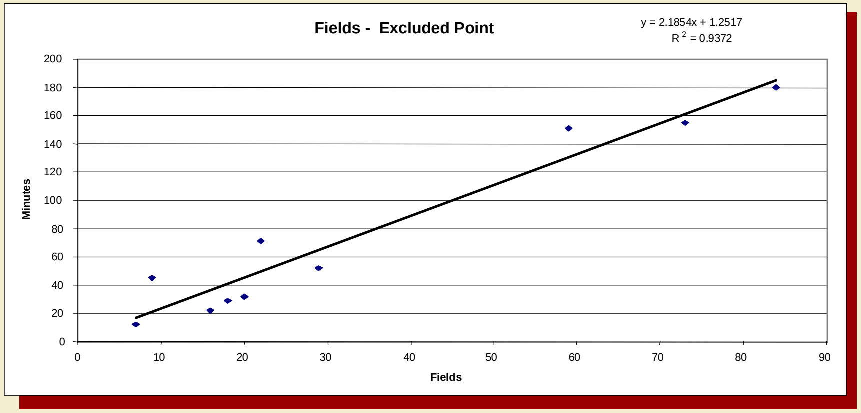
- **Estimating data sets should be representative of their author's work and style**
- **Estimating data sets should**
  - **have at least 5 points; 10 or more is much better**
  - **they should have a high r^2 - at least 0.5, 0.75 or more is better**
  - **generate regression equations that have reasonable slopes and intercepts**
  - **be checked for stratification and partitioned if necessary**
- **Outliers should be identified and eliminated**
  - **can be identified using run charts or prediction intervals**
- **Estimating data sets should be validated based on a history of producing reasonable results**

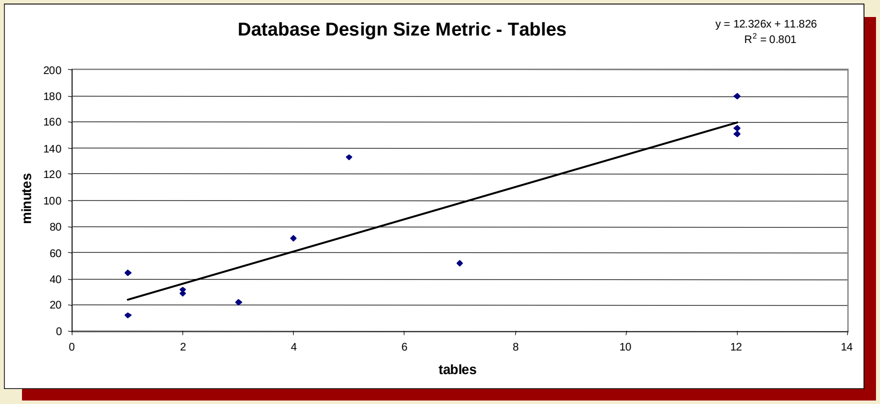
Database Table Size Metric

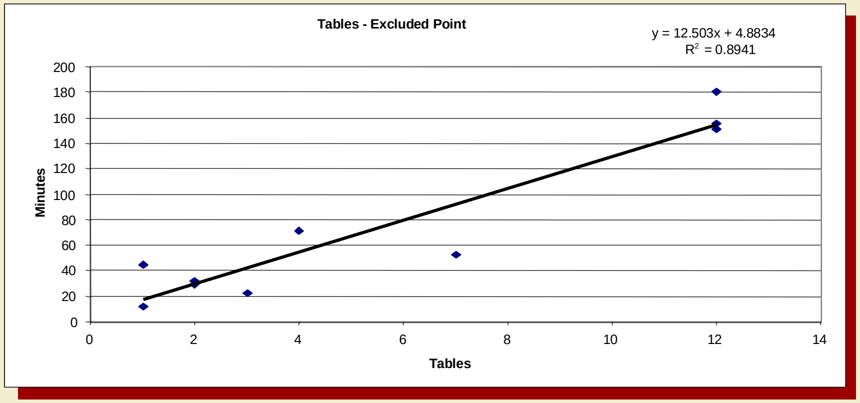
- Case Study: design of a medium size database consisting of a about 20 data modules, each module having multiple tables, relationships, & validation rules
- Design standard called for capturing the design with Entity Relationship Diagrams (ERDs).
  - Included templates for data type definition, validation rules, relationships, triggers, etc.
  - Capable of generating SQL automatically
- Brainstorming identified candidate size metrics: tables, fields, LOCs (SQL)
- Size-Effort correlation used to screen potential size metrics and “fields” was tentatively selected as the size metric
- Algorithms to estimate the size of and effort to design a data module were developed & validated

Selecting a Data Module Size Metric



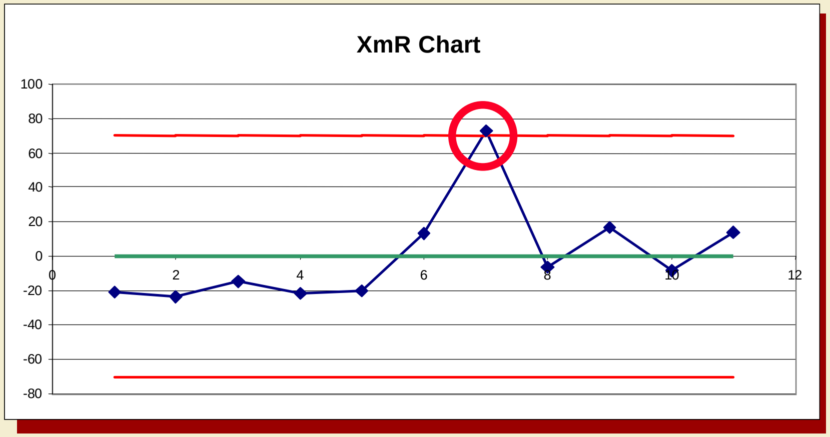


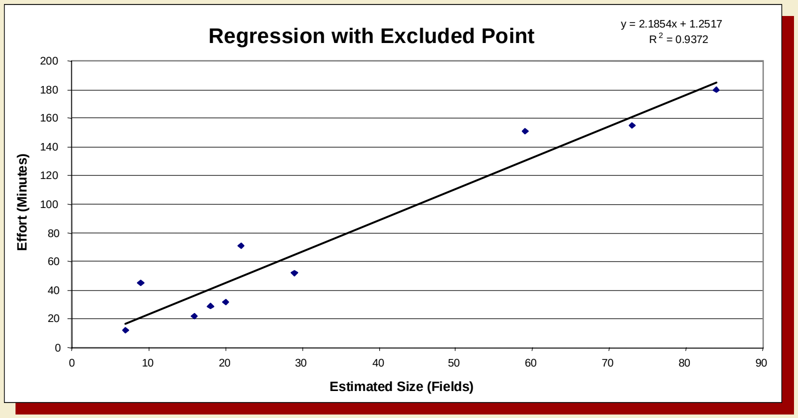


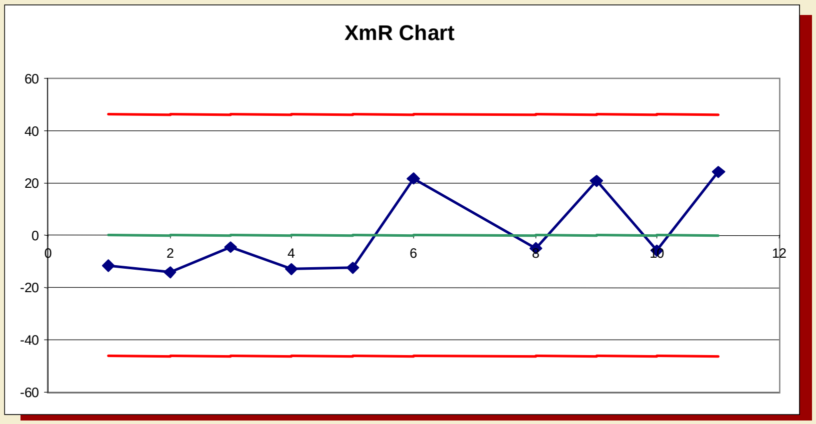


- “Tables” and “Fields” yielded a very similar r^2 until an obvious outlier was eliminated
- “Fields” then gave better correlation and ultimately performed better in estimation

Outlier Management

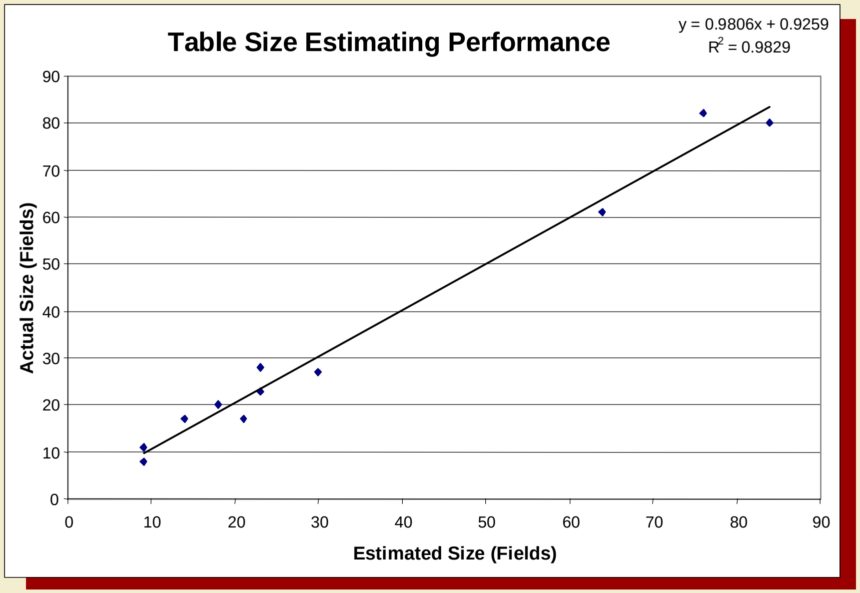


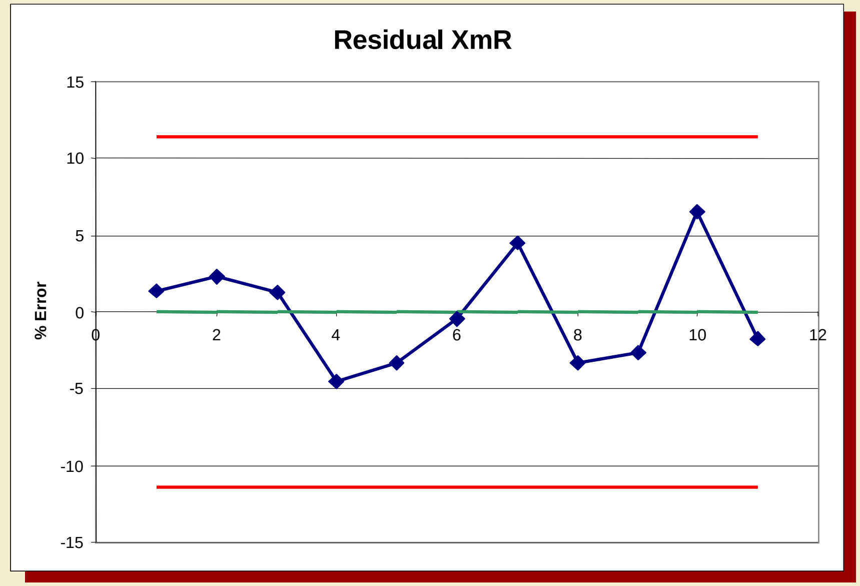




- XmR chart confirms 7th point is an outlier
  - Removing it significantly improves r^2
- Possible presence of a stratification variable or process shift

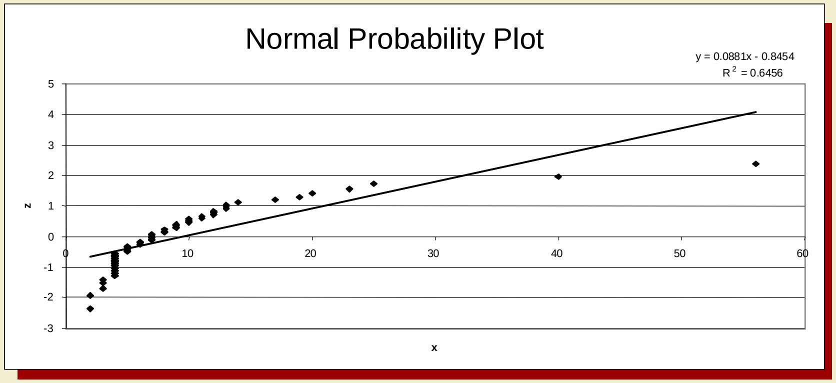
Size Correlation Analysis

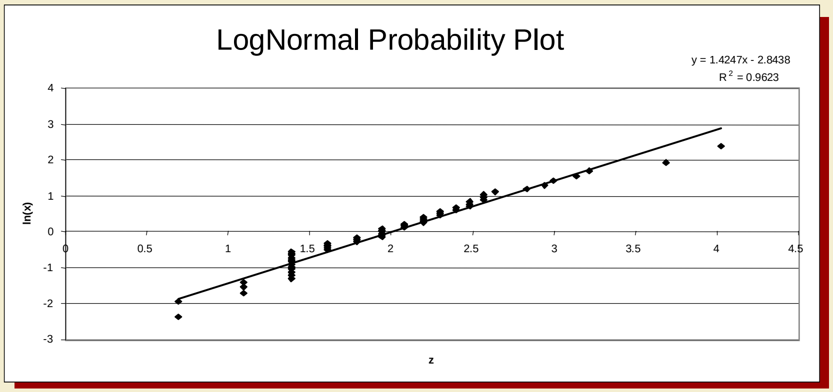


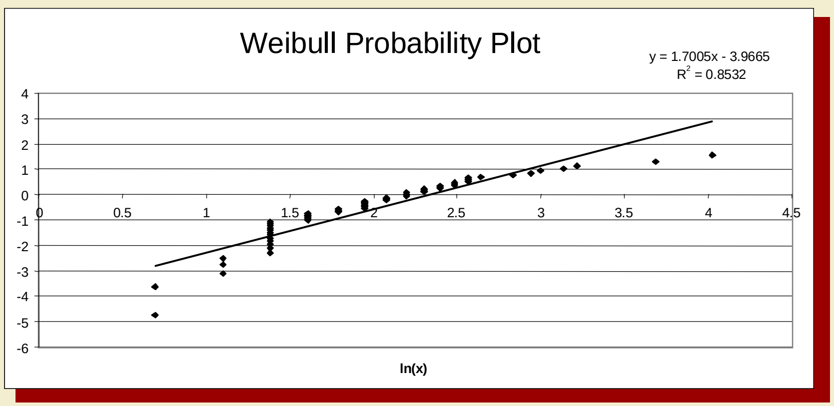


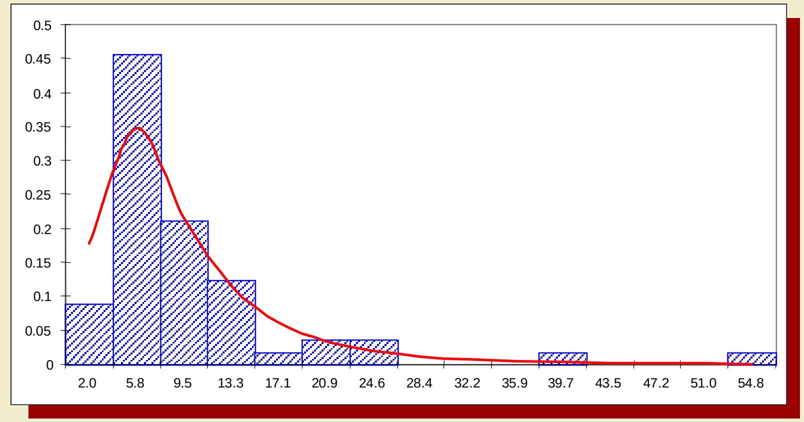
- High r^2 indicates good correlation
- Small average error and symmetric distribution of residuals indicates unbiased estimator
- XmR charts indicates a relatively stable situation

Database Table Size Distribution







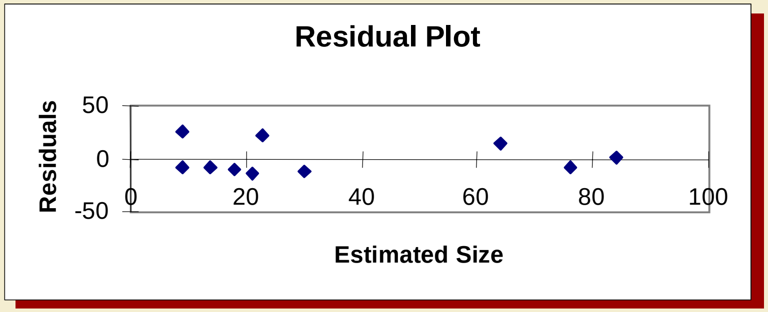


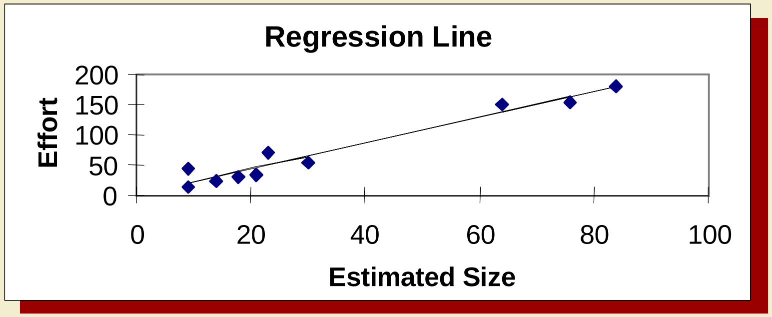
- **LogNormal has the best fit**
- **Provides reasonable estimate although it deviates significantly for small values**

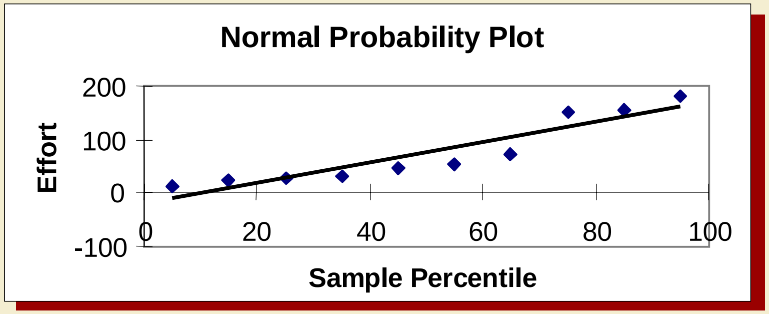
	Normal	LogNormal	Weibull
vs =	-8.604213818	1.862993874	1.121367549
s =	0.496138705	3.70297303	3.669157559
m =	9.596491228	7.360200937	8.306123544
l =	18.69684375	14.62947675	14.75270276
vl =	27.79719627	29.0782265	22.53279402
ChiSquared =	41.10526316	5.736842105	6.578947368
p-Value =	7.7278E-07	0.570790863	0.47399392

Database Design Size-Time Correlation

- **Good correlation between size and effort for personal data resulted in straight forward application of PROBE for requirements in the 0 – 100 field range.**







$r^2 = 0.95$
slope = 2.13 min/field
intercept = 0.7 min
p-Value = 2.4×10^{-6}

References

Look through the presentation CD for our talk on
"*Applying Functional TSP to a Maintenance Project*"
presented at this conference on Wednesday, March 10 at 1:30 PM.

For additional information visit our web site or contact us at:

Ellen George 201- 358-8828
EllenGeorge@SoftwareSixSigma.com

Steve Janiszewski 201- 947-0150
SteveJaniszewski@SoftwareSixSigma.com

www.SoftwareSixSigma.com