Constructing Sizing Methods

SLIM-ESTIMATE® HOW TO SERIES

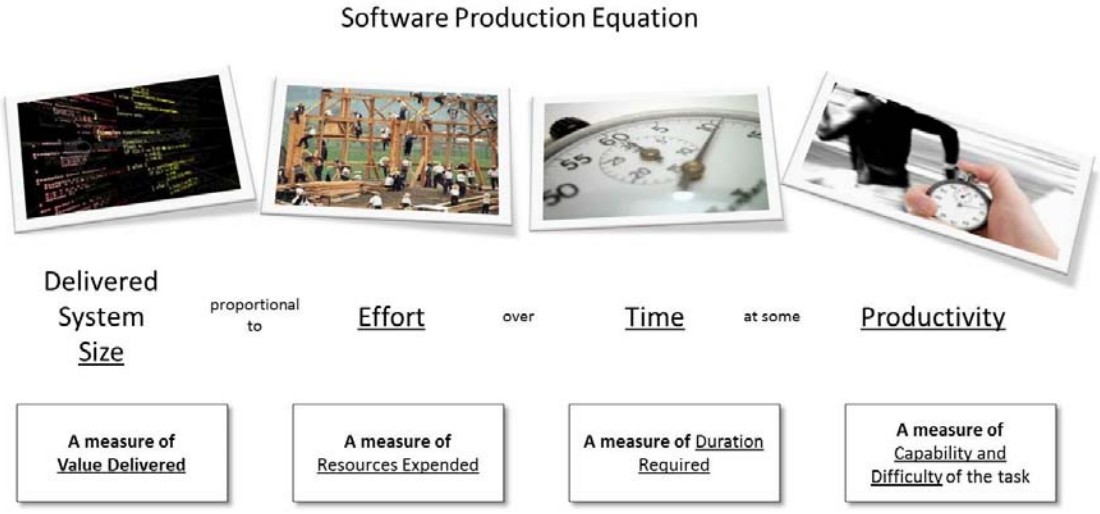
QSM®

Contents

Overview

Documents in the *SLIM-Estimate® How To* series walk you through SLIM's estimation process. Each guide tells you what the goal is, what steps are required to achieve the goal, and what information you configure or enter. This document explains the importance of product sizing, and how the SLIM methodology differs from other estimation techniques, particularly effort-based sizing. Some important things to understand about product and project sizing in SLIM-Estimate® include:

- SLIM-Estimate determines total effort and duration for the entire project, start to finish. It is a top-down approach that rolls up detailed estimates of all the system parts that must be constructed, tested, and delivered, regardless of the individual tasks that may need to be performed.
- Size is a measure of the amount of work to be done. It accounts for complexity. A simple web page requires less work than an online shopping cart. SLIM-Estimate incorporates this relative complexity into its sizing techniques.
- Size is not:
    - How much your project is going to cost
    - How long it will take to complete
    - How much effort will be required to complete it
    - How many staff will be assigned to it
- SLIM's Software Production Equation shows that the time and effort required to develop the system are driven system size and productivity. Size and Productivity are estimation outputs, whereas time and duration are estimation outputs.



What Is Size?

Size is a numerical measure of those things that a project needs to create or fulfill in order to be considered complete. In other words, the product(s) the project was formed to produce and deliver. Some examples of size are:

- Lines of code that will be created
- Screens, reports, interfaces

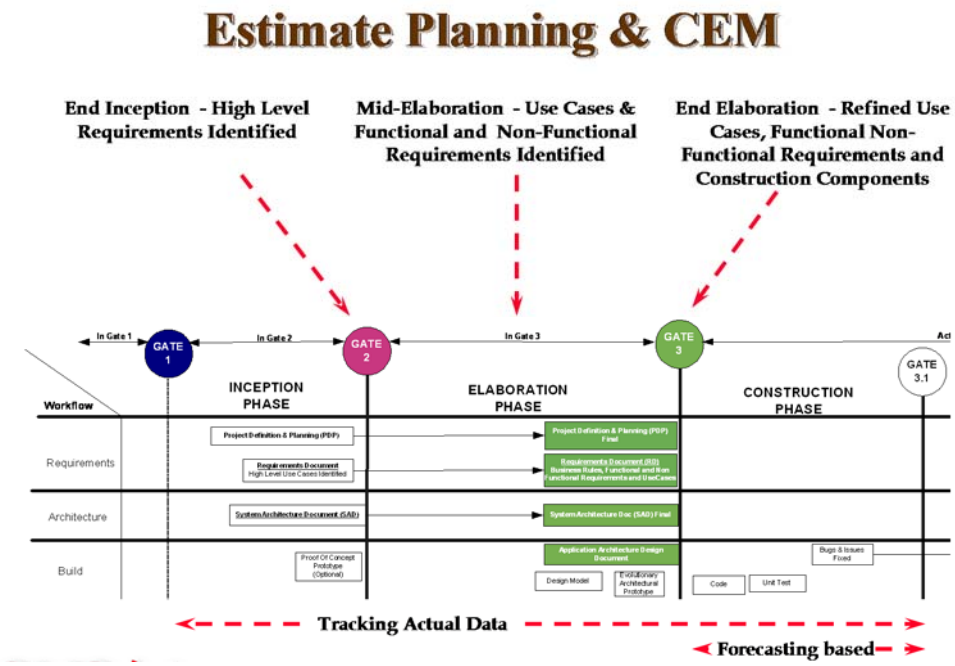
- Use Cases
- Features
- Story Points
- Software configuration and customization steps

Multiple units of measure can be used to quantify product size.

Selecting Size Measures

When a project is first being considered, there is normally only a high level understanding of what it will accomplish. At this stage only abstract size measures such as high level requirements are available. Later on, say when a technical design has been completed, more concrete measures such as screens, reports, interfaces, and programs can be estimated. When selecting a size measure to use for estimating it is important to keep it aligned with where you are in the project lifecycle. If you are estimating early in that life cycle, the size measure will be abstract. More concrete measures will be available later in the life cycle.

The diagram below shows three points for gathering size metrics for a typical software development life cycle (labeled here as CEM - Corporate Enterprise Methodology). The initial estimate performed prior to the start of the project may have used requirements to measure size. At the conclusion of the Inception Phase, the size estimate could be updated with a count of the final requirements count (Gate 2). Subsequent size estimates can be performed mid-way through the Elaboration Phase, and at the end of this phase. The respective units of measure may be objects and construction components. All of these measures can then be used in tracking all product development schedules throughout the project.



SLIM-Estimate®'s solution log enables you to capture the results of individual size estimates. You are then able to immediately see the change in project schedule and effort resulting from changes in scope and/or early estimation errors revealed by the more detailed project estimates. Additionally, you will have a record of estimation process accuracy plus sound measures for sizing future projects. Capturing the total software size at the end of the project completes the estimation process. You establish sound historical data that can be entered into SLIM-DataManager®, along with total duration and effort, to compute the actual PI.

Function Units

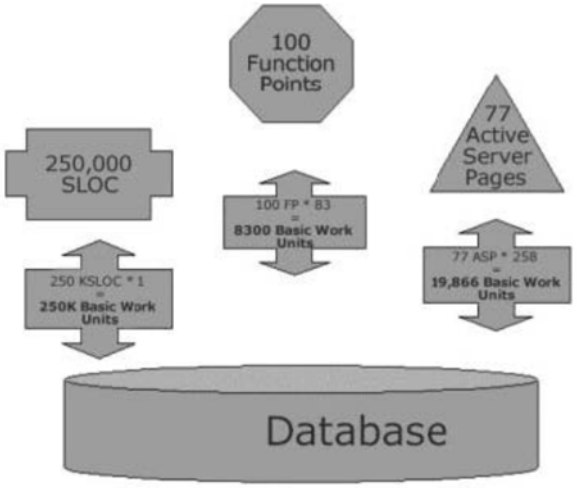
Early in the estimation process, we need a size unit that users can understand to express what they need the product to do. SLIM calls this measure the **Function Unit**. Just as it sounds, it is the overall measure of functionality built into the product. Examples are requirements, story points, and features.

SLIM-Estimate calculations use what is called the **Basic Unit of Work**. In the past, Source Lines of Code (SLOC) was a nearly universal measure of work for software projects spanning different technologies, languages, and development paradigms. But the advent of modern diagramming tools, GUI languages, and programming environments make lines of code less useful as a measure of work performed. Developers who use diagramming tools may find that a combination of GUI actions better represents the work needed to translate a given set of requirements into software. Those who spend their time configuring database tables may wish to identify the smallest unit of work applicable to database construction and build from there.

It doesn't matter what the Basic Unit of Work is called. *What matters is that the estimator identify the programming tasks (or steps) to be performed that carries approximately the same amount of time and effort as writing an executable line of code*. The goal is to preserve a common frame of reference while allowing users the flexibility to choose the sizing method that most accurately reflects the actual work being performed: translating abstract requirements into a concrete, functioning software system.

Gearing Factor

We need a way to map these two size units, Basic Unit of Work and Function Unit, to one another. SLIM calls this conversion, or mapping value a **Gearing Factor**. It is similar to any unit conversion factor, such as 2.54 centimeters per inch.

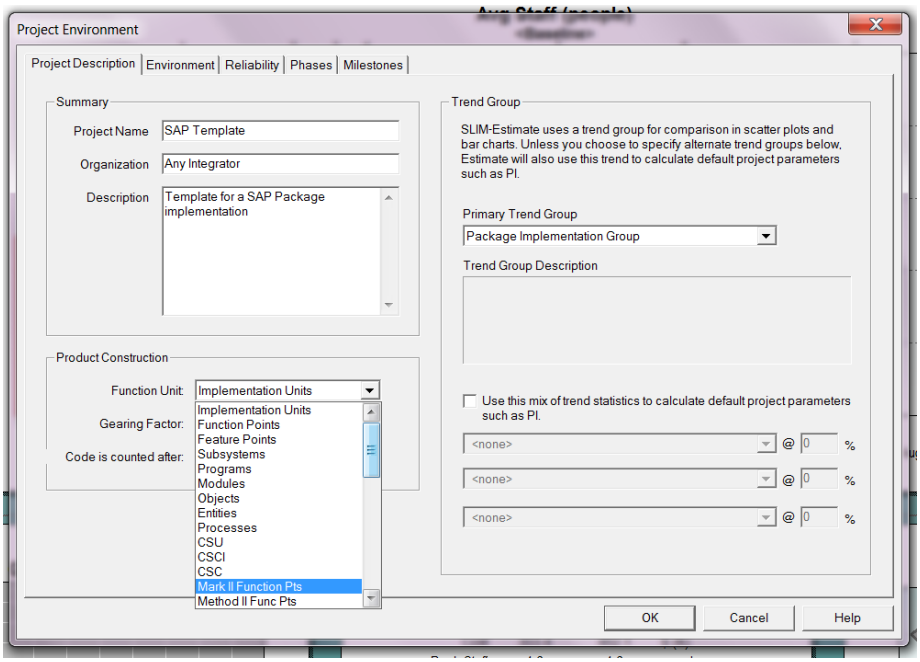


It is expressed as the average number of Basic Units of Work in your chosen Function Unit (Basic Work Units/Function Unit)

Example gearing factors include the following:

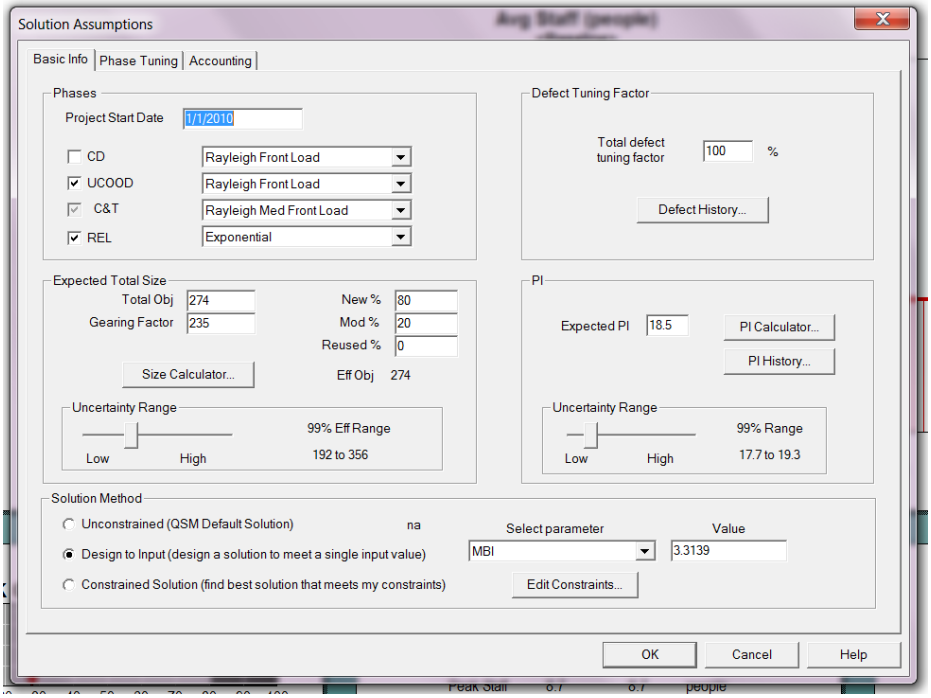
- 500 Implementation Units/Requirement
- 350 SLOC/Story Point
- 20 Screens/Module

Sometimes projects are estimated using the Basic Unit of Work. In this case, the Gearing Factor is simply 1. One of the best ways to derive gearing factors is from completed software project data. Gearing factors can be calculated at the end of a project and the resulting factors used to estimate new projects. Gearing factors can also be estimated or sampled during the sizing process. SLIM-Estimate® templates often designate Implementation Units as a generic unit of measure any project can use. Select the Basic Unit of Work using the *Tools | Global Options* screen. The screen below shows the list of choices available for designating the Function Unit on the *Tools | Project Environments* screen. You may add your own size unit if you like.



Expected Total Size

The main parameters required to calculate a project estimate are entered using the *Estimate | Solution Assumptions* screen. Enter the Expected Total Size, Gearing Factor, and the range of uncertainty about the size estimate on the *Basic Information* tab.



The total software size can be broken into categories of new, modified, and reused, to describe or model the type of work being done to build the application. This is important, remembering that ultimately we want to capture the amount of work, so we can accurately predict the time and effort required to complete the work. Creating new functionality is often easier than modifying existing functionality. And although reusing existing "code" can save time in the long run, work must be done to integrate and test the features of the reuse with the newly created and modified portions.

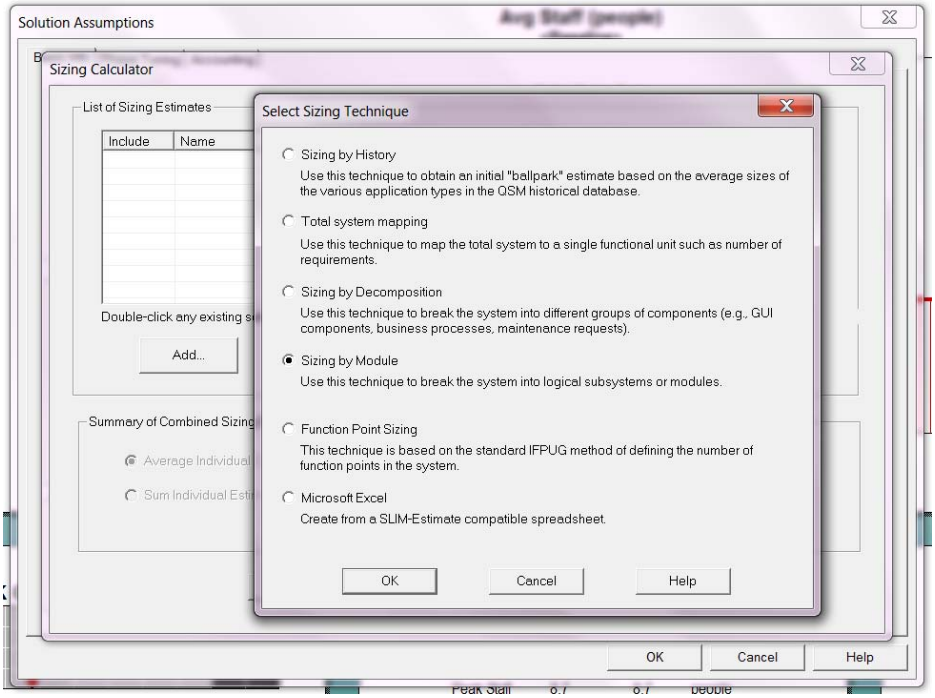
The final system's Total Size consists of all components that have been built, modified and used without modification

SLIM's Effective Size is based only on the components which are built or modified.
The Unmodified components add complexity rather than work

The **Effective Total Size** in SLIM consists of the percent of **New** and **Modified** features only. Reuse percent is part of the PI calculation. Enter the values for each portion of the total system for each category. Then enter the total size and associated gearing factor in the appropriate fields. You will notice that the label for size will reflect the measure you entered for the Function Unit (ex: Total Obj on the screen above).

Using the Sizing Calculator

Because estimating size is the biggest challenge in software estimating, QSM recommends using more than one sizing technique. This tends to produce better estimates, as they can be used to validate one against the other. Select the *Sizing Calculator* button on the *Solution Assumptions* screen, and then selecting the *Add* button. The list of techniques is ordered from top to bottom by the amount of data required to specify the estimate. For example, **Sizing by History** does not require that you enter a numerical value; rather it presents a range of potential sizes that correspond to the QSM database trend lines selected on the *Project Environments* tab. As you slide along the bar from *Very Small* to *Very Large*, SLIM displays the average size and associated uncertainty range from the database.

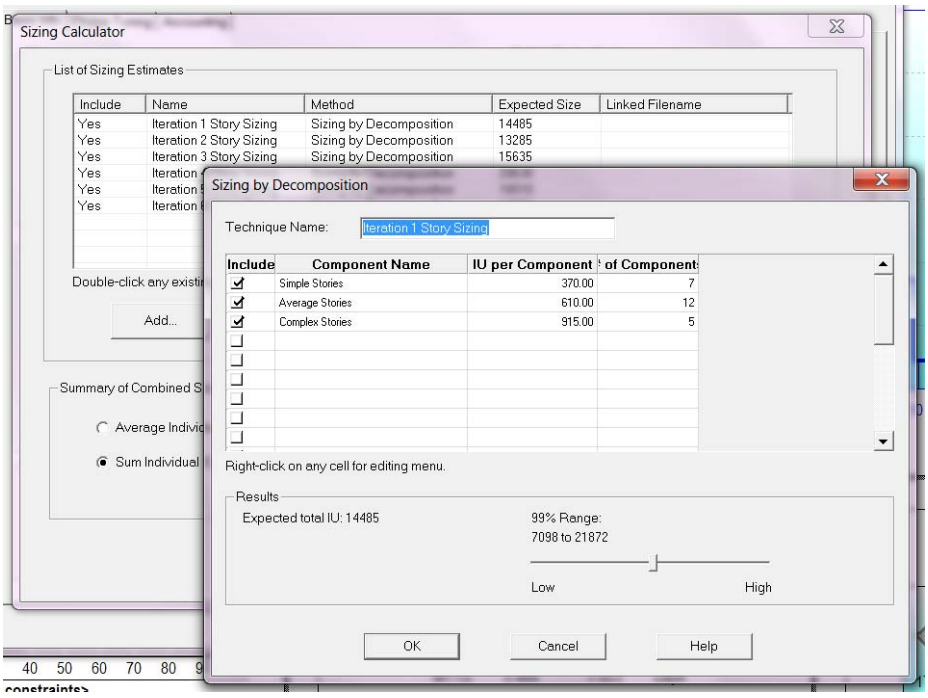


There are six sizing techniques:

- Sizing by History (Rough estimate)
- Total System Mapping (Single unit of measure such as requirements)
- Sizing by Decomposition (Breaking the system into groups of components)
- Sizing by Module (Breaking the system into logical subsystems)
- Function Point Sizing (Using the standard IFPUG method)
- Microsoft Excel (Creating a SLIM-Estimate compatible worksheet)

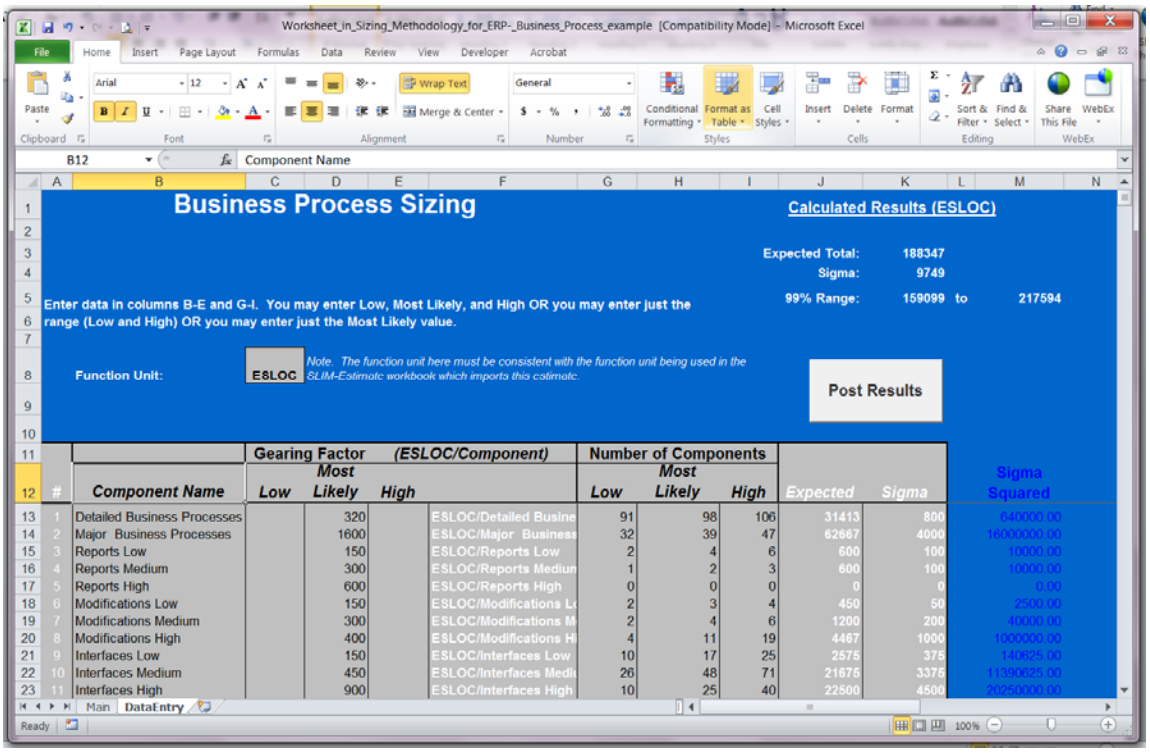
You can use more than one technique to size the system, in fact, QSM recommends using multiple sizing techniques. You have the choice to either sum or average the different estimates. The choice will defend on whether or not the estimates include the whole system or just parts.

Sizing by Decomposition is a very common sizing technique, because it allows multiple gearing factors to be used. Each component can be classified into complexity buckets, such as simple, modified, and complex in order to calculate relative gearing factors. These individual estimates are summed to get the total size for each component. The screen below shows an example of decomposition by Iteration, with Stories as the designated function unit. Simple, Average, and Complex Stories have been counted and relative gearing factors assigned.



The **Microsoft Excel** option is the most flexible, accommodating the greatest level of detail. Both the gearing factor and size can be expressed as a 3-point estimate, increasing the precision of uncertainty calculations. This method is also great for incorporating data from other applications where metrics data may be stored.

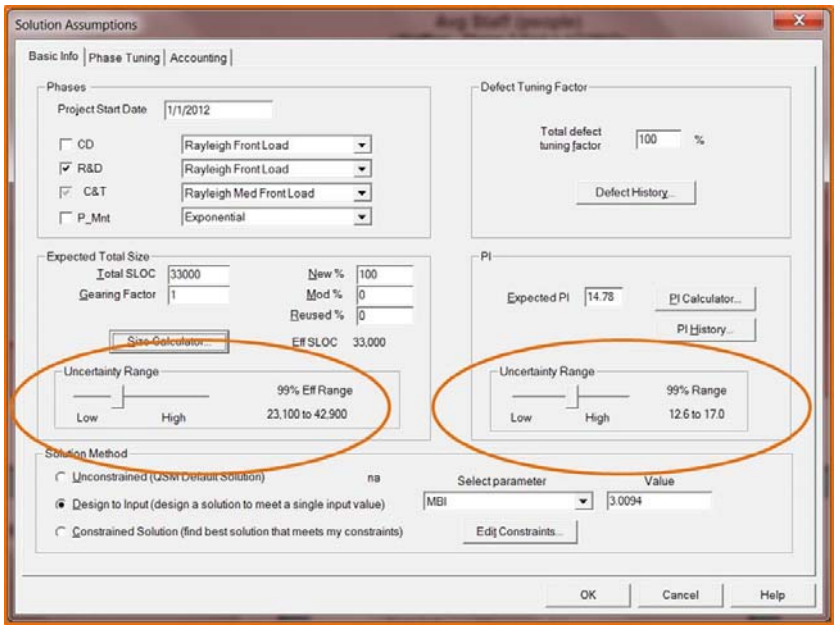
Several sample spreadsheets are installed with SLIM-Estimate, and can be found in the QSM/Toos80/Samples directory. If you use the spreadsheet method, be sure to check the User Manual for requirements on how to either modify the spreadsheet samples provided, or supply a custom spreadsheet.



Uncertainly Range

A single-value estimate usually raises more questions than it answers. You do not know if that value represents the best or worst case scenario. SLIM-Estimate® not only provides the ability to calculate an effective total size with multiple techniques, it allows you to quantify the uncertainty of the estimate.

The Solution Assumptions screen Basic Tab contains an *Uncertainty Range* slider bar for both **Effective Total Size** and **PI**. As you move the slider bar from left to right you are providing a 99% confidence interval for your size estimate. The difference between these lowest and highest values varies from 0 at the extreme left side of the scale (indicating you are certain of the exact system size) to a maximum variance at the right end of the scale. Position the slider bar so that you are 99% confident that the true system size will not fall outside the displayed size range. The size uncertainty range is used to calculate the standard deviation for your size and productivity estimates. Monte Carlo Simulation is used to sample these distributions to generate the effort, cost, and schedule probabilities presented for each solution.



What are the Next Steps?

You can start estimating system size by selecting one or more techniques that match your development life cycle. Look at the project deliverables and identify the function units you can count from requirements and design documents. Find out if you can count lines of code or other completed product artifact from either a configuration management or other process tool. The goal is to calculate gearing factors for the function units you selected. You may be able to gather enough historical data to calculate the gearing factor, or you may start by using QSM supplied valued. If you have any questions, contact your QSM representative or email QSM Support.