Case Study
Part B

Sizing the System

This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. © 2011 California Institute of Technology. Government sponsorship acknowledged.
The purpose of this exercise is to generate a size estimate for a hypothetical software project for the purposes of generating a cost estimate.
Sizing the System

In this exercise, we will:

1. **Measure the size of reference code** to be used in the size estimation process
2. **Estimate** the amount new code development, reuse, and modification
3. **Generate a probabilistic estimate of equivalent size** for the new software project
Project Description

1. JPL is developing flight software (FSW) for a flight project. It is a telecom system that can be reused by landers and rovers for communicating with earth.

2. The flight software’s can be divided into four primary function: monitoring data, data transfer, command and control, and relay communication.

3. The telecom system has some design heritage with an existing telecom system that has been developed. There is a small amount of code inheritance. All new code developed will be in C.

4. A software development environment including a test-bed exists.

5. The software is nearing its preliminary design review (PDR). The software must be delivered to ATLO in 16 months (64 weeks), with a small, though experienced (3 years C experience, but very little experience in the development tools), development staff.

6. Requirements are immature, therefore 10-20% requirement volatility is expected.

7. There is concurrent HW development. The HW is being developed by a contractor in another state.

8. This will be mission class B (Mission Critical) software.

9. The project is currently budgeted at 54 WM. IV&V is paid for at the project-level, and the cost of maintenance does not need to be included.

This example of a JPL software development project is loosely based on a real project. It is meant to illustrate the basic steps of developing a software estimate. It is not intended to serve as a source for answers to all questions that may arise regarding software estimation.
The first step in the sizing process is to identify and measure the size of reference (analogy) code

In this exercise, we will use the JPL SLiC code counter to measure the size of four different software functions that have been identified as reference modules:

1. Monitor Data (Function W)
2. Data Transfer (Function X)
3. Command and Control (Function Y)
4. Relay Communication (Function Z)
Count Reference Code

Note: Sample code (Functions W,X,Y,Z) has been loaded on the training PCs for this example

1. Double-click on the ‘Cygwin’ desktop icon

2. Enter: `cd example_data` at the prompt and press ‘ENTER’ to move to the folder with the example data
At this point, you are in the example_files folder, which contains sub-folders containing our sample reference code:

- example_w
- example_x
- example_y
- example_z

To verify that you are in the correct location, type `ls -l` at the prompt and press ENTER. You should see a window similar to below:
Now that we are in the root folder (with all reference code below us), let’s perform a count of all source:

$ ./slic -t

By default, SLiC automatically finds and counts all supported source code under the current path.
The next step is to count at the first subfolder level.

This command shows the SLOC totals for each function (folder):

```
$ ./slic --output-depth=1
```

The `output-depth` option displays totals (totaled recursively) at depth $d$ relative to the current (or explicitly specified) path.
Compute Total SLOC

- Compute Total SLOC based on
  - Monte Carlo Simulation

- **Step 1:** Open MonteCarloSizing Tool (On Desktop in folder named “QSM” and [today’s date])
- **Step 2:** Enter Size numbers from previous slide into tool in historical column
- **Step 3:** Scale your software size to the reference sizes
- **Step 4:** Run Monte Carlo Simulation
- **Step 5:** Save your results for the next exercise

This column contains the logical SLOC we estimated from SLiC
MonteCarloSizing Tool

- Excel-based tool to help you with your analogy size estimates
- Incorporates uncertainty by allowing distributional inputs
- Choose between point estimates, uniform inputs (Low and High), or triangular inputs (Low, Most Likely, and High)
- Uses Monte Carlo techniques to aggregate size and compute total equivalent size
- Can choose number of iterations per Monte Carlo run – 9,999 iterations is recommended
- Other features: Function Point Calculator that allows distributional inputs for uncertainty
Scaling the Software Size

- Estimate Size Distribution parameters
  - Convert to logical lines if needed
  - Derive ML based on analogous functions from completed software systems
  - Adjust estimate for differences between current fn and analogous fn
  - Estimate low and high estimates based on best and worst case scenarios and document basis of estimate

Read the basis of estimate to fill in missing size estimates
Choose between point estimates, uniform inputs (Low and High), or triangular inputs (Low, Most Likely, and High)
MonteCarloSizing Tool Output

- MonteCarloSizing Tool outputs a Low, Most Likely, and High Equivalent Size estimate
- Save your results for the next exercise

Note: Output numbers will vary slightly due to randomness of draws