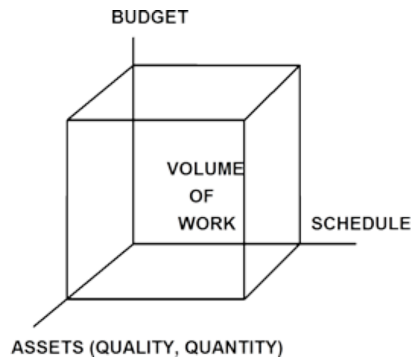
Software Cost, Schedule and Quality

As mentioned earlier, cost and schedule must be aligned to help produce a quality product as shown in the graphic below.

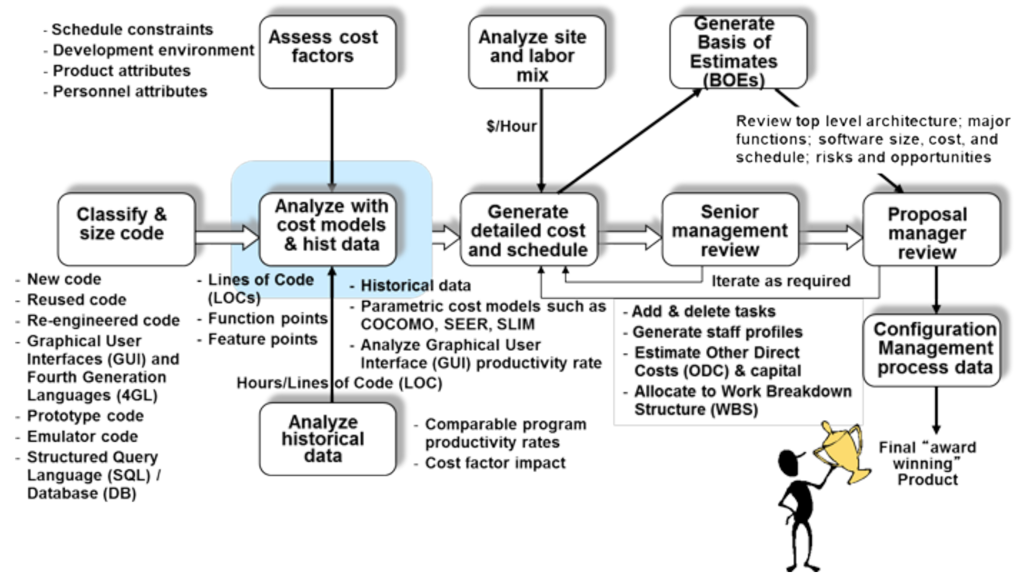


Quick Labor estimate and Rule of Thumb

This module builds on the previous modules and further discusses how to estimate software cost and schedule. Remember the labor and schedule estimates are based on:

- Software Requirements Analysis
- Software Design, Code and Unit Testing
- Software Integration and Test (SWIT)

But the software manager must also take into account and increase the estimate by some percentage to support to Systems Engineering Requirements Analysis and Design; planning and preparation to setup Software Development Environment; support to the Independent Test Teams; Software Configuration Management; management, clerical, and software process (metrics, Software Engineering Process Group (SEPG)); the development system, administration, and maintenance (operating system, tools, and COTS); and software licenses and COTS for the Target System. As we estimate the cost and schedule, remember the software estimation process.



Quick Labor and Rule of Thumb Schedule Estimates

Assumptions made during the size and effort estimation activity must be documented. For example, let's agree on the following assumptions:

- Our standard corporation tools will be used; this is our computer aided software engineering (CASE) environment
- Our development system will support one workstation (or terminal) per software engineer.
- Target hardware, COTS, Government Furnished Equipment (GFE), Customer Furnished Equipment, and reuse software will be available when needed.
- Our system will have normal software reliability; it will not be a man-rated system, one that supports human activities (e.g., NASA shuttle system).
- Our system will not need special security requirements (e.g., multi-level security activities).
- We will not have special documentation or reviews.
- Our system will have an adequate development environment in terms of quality and speed.
- We will assume a reasonable schedule and appropriate budget constraints.

Wouldn't it be great if all systems' assumptions were like these! With these assumptions, we can apply the Quick Labor Estimate which gives you the staff months needed to complete your activity and the Rule of Thumb Schedule estimate which gives you the likelihood of success (or the percent of time) to successfully implement this program in the time frame.

To determine the quick labor estimate, we need to first determine how many hours per month each person is working on average. Let's assume an average of 160 hours / staff month. To get this, an employee works on average:

40 hours/week * 52 weeks/year = 2080 hours/year

Minus 2 weeks' vacation (80 hours) = 2000 hours/year

Minus 10 holidays (80 hours) = 1920 hours/year

Some companies subtract out hours for sick leave or additional leave which further reduces the average number of productive hours/staff month (SM).

Productive hours/staff year = **1920 hours/year**

Productive hours/SM = 1920 hours/12 months = **160 hours/SM**

Note: Companies often use the 5/4/4 accounting cycle over a three month period, that is, the first month has five (5) accounting weeks, and the second and third accounting months have four (4) accounting weeks each. Thus the accounting month and calendar month are not exactly aligned. For example, the first accounting month may run from January 1 to February 4, the second account month is from February 5 to March 4, and the third accounting month is from March 5 to April 1; thus, over a three month period 13 weeks are included. Using this method the number of hours/staff month is more granular and may vary each month. For the purpose of this course, we will generally use 150 or 160 hours/SM.

Recall the definition for productivity in terms of LOC:

LOCs = Productivity * unit of time or

Productivity = LOC/unit of time and in this case LOC/hour

The **Quick Labor Estimate** or Effort in Staff Months (SM) is defined as:

Effort = Number of LOC estimated /(Productivity * (Productive hours/SM))

= Number of LOC estimated / ((LOC/hour) * (160 hours/SM)) = Effort in SM

The **Rule of Thumb Schedule Estimate**, a widely used formula, is referred to as the Time for Development (T_{Dev}) or Time for Labor (T_L) and is defined as:

Formula	Percentile
$T_L = 2.0 * (SM)^{(1/3)}$	10%
$T_L = 2.5 * (SM)^{(1/3)}$	50% (Moderate Risk)
$T_L = 3.0 * (SM)^{(1/3)}$	90%

That is you take the cube root of the resulting Quick Labor SM estimate and multiply it by a constant to give you the chance of success to successfully implement this program in this time frame. The general idea that schedule is a cube-root function of effort is universally accepted by estimation experts, but...

Beware the Impossible Region: When the constant becomes too small (below 1.875) and therefore the time to develop becomes too short, there is a 0% chance of successfully implementing the program.

$T_L = 1.875 * (SM)^{(1/3)}$ 0%

Note: the Rule of Thumb Basic Schedule Equation is not intended for estimation of small projects or late phases of larger projects and another estimation tool should be used for these circumstances.

Example:

You are tasked to estimate the effort needed and the time to develop a software system that is estimated to have 20,000 LOC. You are able to assume moderate risk and the typical productivity for your team (or company) is 15 LOC/day.

Effort = Number of LOC estimated / (Productivity * (Productive hours/SM))

20,000 LOC / (15 LOC/day *160 hours/SM)

Since the productivity is shown in terms of LOC/day, you must convert it to LOC/hour by dividing by 8 to ensure equal terms.

Thus the equation becomes:

Effort = 20,000 LOC / ((15 LOC/day) / (8 hours/day) *160 hours/SM) = 66.67 SM = ~ 67 SM

$T_L = 2.5 * (SM)^{(1/3)} = 2.5 * (66.7)^{(1/3)} = 2.5 * 4.05 = 10.1 \sim 10$ months

That is this 20,000 LOC project will take approximately 67 SM. Fifty percent of the time, 67 SM can be successfully completed in the 10 month period. If you want to lower the level of risk, increase the constant. That is:

$T_L = 3 * (SM)^{(1/3)} = 3 * (66.7)^{(1/3)} = 3 * 4.05 = 12.2 = \sim 12$ months

Now 90% of the time, 67 SM can be successfully completed in the 12 month period.

Remember this is just an estimate and there are multiple assumptions. The Quick Labor and Rule of Thumb Schedule Estimates should only be used as single data points. To determine the best estimates, use these results with other estimating techniques, then analyze and justify the results.

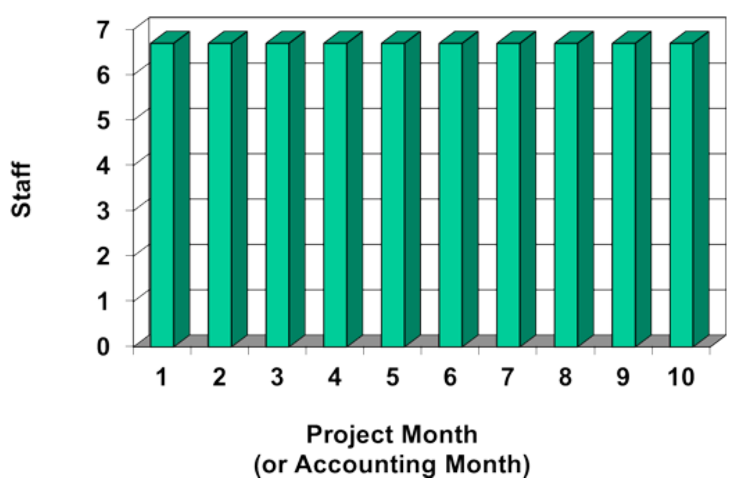
Staffing Tips

Level Loaded Staffing versus Ramp Up and Down Staffing

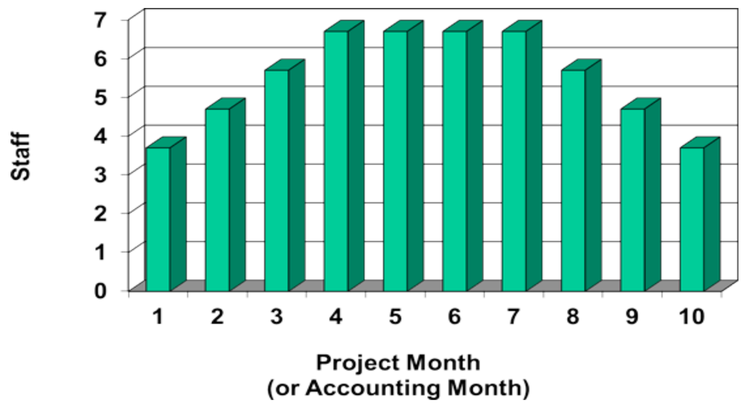
In our example the effort estimate is 67 staff months (SM) and schedule estimate is 10 calendar months. Full Time Equivalent (FTE) is a method used to measure a staff member's participation on a project. Thus,

67 SM / 10 months = 6.7 staff or 6.7 FTE

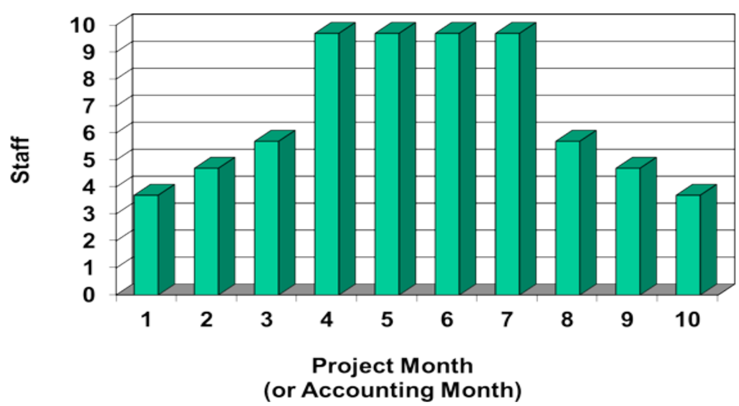
The **level loading staffing approach** uses 6.7 team members from project start to finish. At least one staff member is part-time working 70% of their time on this project.



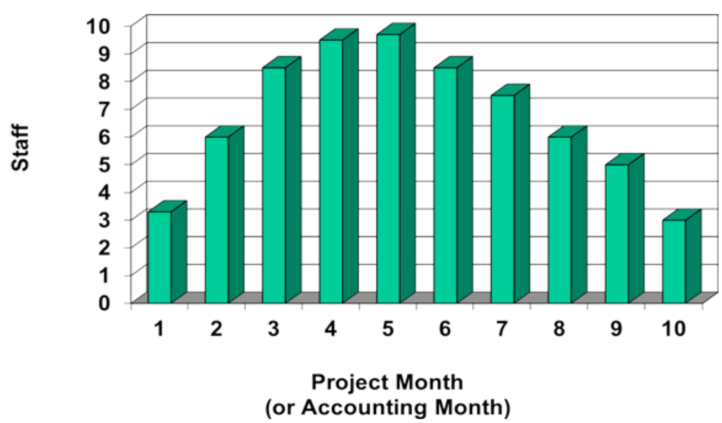
An alternative to level loading staff is to transition staff onto the program (ramping up to start of code) and transition them off as you begin integration. Transitioning staff on and off reflects the following profile:



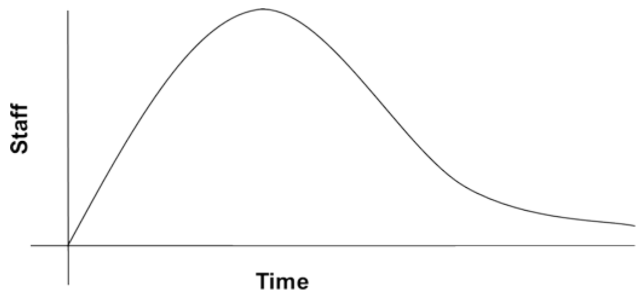
However, this profile does not fulfil the effort estimation of 67 SM. We are short 3 SMs in months 1 and 10, 2 SMs in months 2 and 9, 1 SM in months 3 and 8; that is we are short a total of 12 SMs. Hence we must alter the profile to expend 12 more SMs during months 4 through 7.



But this creates an increase from month 3 to month 4 of four staff personnel (from 5.7 to 9.7 staff) nearly doubling the size of the project. The process is iterated until you smooth the profile and comply with resource availability (or hiring capacity) within your organization. You attempt additional smoothing keeping the total staff months at 67.



The current literature suggests ramping up and down with a profile that approximates a Rayleigh distribution, which is similar to a bell curve skewed to the left.



Rayleigh Curve

It is important to determine your staffing strategy for each project; however, take into account that there are pros and cons with each approach.

Level Loaded Staffing Strategy	Ramp Up & Down Staffing Strategy
All team members are on project from start, allowing junior members to gain experience with analysis, design, integration and testing.	Senior staff perform the front and back end work for which they are most qualified including analysis, design, integration and testing.
Junior team members are more prepared for start of code, unit test, and integration.	Senior staff are not slowed down by effort required to train and mentor junior staff.
Typically it is difficult to staff all positions instantaneously, especially for larger programs.	There is the opportunity to search for resources over a longer period of time.
There may be some activities more suited for junior staff during early phases of program, for example, prototyping.	There is the opportunity to phase personnel to other efforts as the program begins to wind down.
There may be some activities more suited for junior staff during final phases, for example, software debugging.	There is more front end planning effort required with modern methods and languages, for example with using Object Orientation and C++.

Skill Mix Profile

The skill mix profile must also be considered when forming your team. Typically the planning and requirements phases up front and the integration and testing phases at the back end require more experienced personnel, while the middle development phase uses less experienced staff. The staffing ratios will fluctuate depending on the life cycle phase but in general you should typically assume there is about 10% senior staff, 40% mid-level staff, and 50% junior staff on a program. The chart defines the typical staff profile mix. In most organizations, a Master's Degree is equal to two years' experience.

Job Code	Job Description	% of Staff

Job Code	Job Description	% of Staff
5	• Technical leader or advisor leading a large team • BS plus 12 or more years' experience • MS plus 10 or more years' experience	10
4	• Technical leader leading a small team • BS plus 9 or more years' experience • MS plus 7 or more years' experience	20
3	• Individual contributor • BS plus 5 or more years' experience • MS plus 3 or more years' experience	20
2	• Individual contributor • BS plus 3 or more years' experience • MS plus 1 or more years' experience	25
1	• Entry level programmer • BS plus minimal experience • MS and no experience required	25

Total Labor Cost Comparison

It is important to understand total labor cost and its variables.

Total Labor Cost = $/hour * hours

$/Hour is based on employee salary which is loaded with:

- Fringe (or benefits): represents approximately 40%
- Overhead (or building & utilities): represents approximately 60%
- General and Administrative (G&A) (or corporate staff): represents approximately 10%

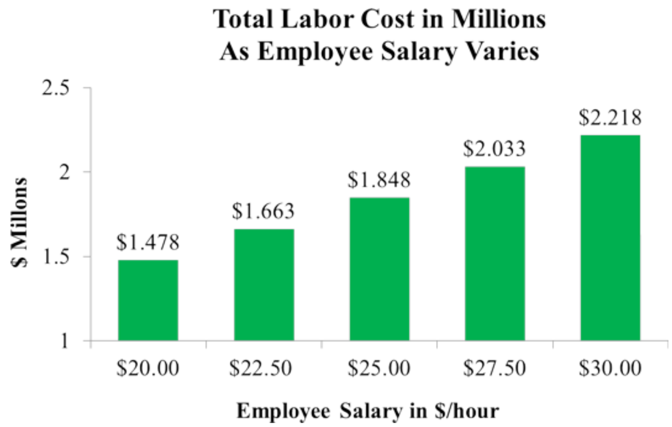
Salary burdens compound so one unit of employee salary becomes: 1 * 1.4 * 1.6 * 1.1 = 2.5 cost to the company

Hours are based on the:

- Productivity rate (LOC/day or LOC/hour), and
- LOC

As employee salary, fringe, overhead, G&A, productivity rate, and LOC vary, the total labor cost varies. Let's assume employee salary is $25/hour, fringe is 40%, overhead is 60%, G&A is 10%, your project's size is 30,000 LOC, and productivity rate is 8 LOC/day. The following graphs illustrate the effects as each variable changes.

The effect of **Employee Salary** on Total Labor Cost as the cost of employees' salary varies plus/minus 20% from $20/hour = $41.6K/year to $30/hour = $62.4K/year is shown in this graph.

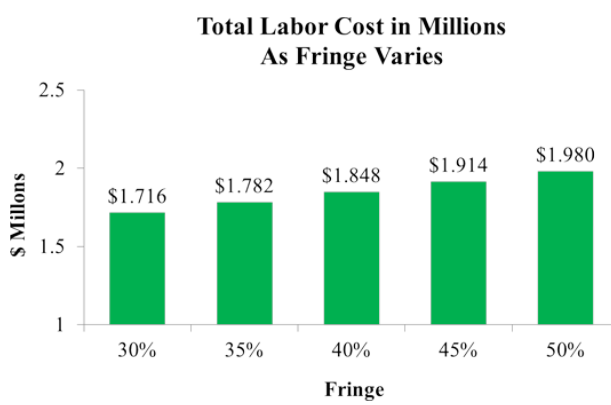


As you vary employees' salary, these results vary around $1.85M as seen in the following equation:

Total Labor Cost = $/hour * hours
= (Employee Salary * Fringe * Overhead * G&A) * (LOC / Productivity)
= ($25/hour * 1.4 * 1.6 * 1.1) * (30K LOC / (8 LOC/day / 8 hour/day))
= ($61.6/hour) * (30K hours)
Total Labor Cost = $1,848,000 = $1.848M

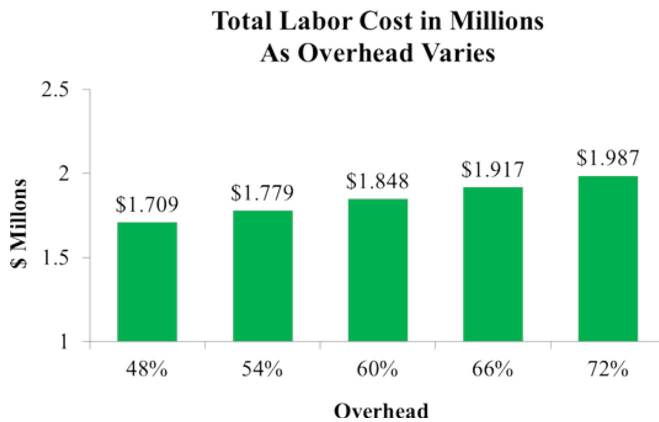
Thus as the employees' salary increases from $20/hour while holding the variables the same, total labor cost increases.

The effect of **Fringe** on Total Labor Cost as the cost of fringe varies plus/minus 25% from 30% to 50% is shown in this graph.



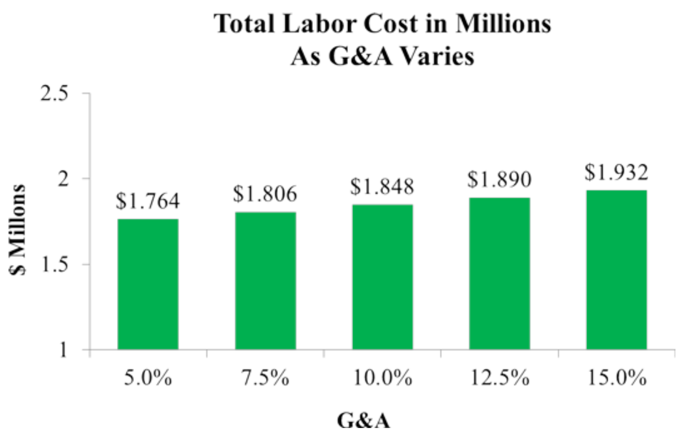
As the fringe increases and the other variables remain the same, the total labor cost increases. It is for this reason that many companies work hard to reduce the cost of fringe benefits.

The effect of **Overhead** on Total Labor Cost as the cost of overhead varies plus/minus 20% from 48% to 72% is shown in this graph.



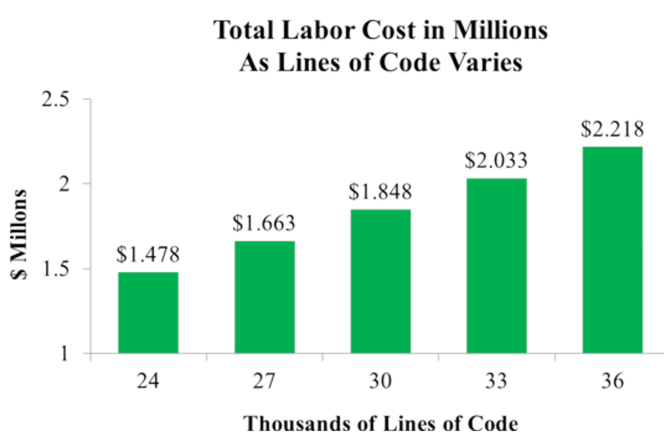
Similarly as the overhead costs increase and the other variables remain the same, the total labor cost increases. Again many companies work hard to reduce overhead costs.

The effect of **G&A** on Total Labor Cost as the cost of G&A varies plus/minus 50% from 5% to 15% is shown in this graph.



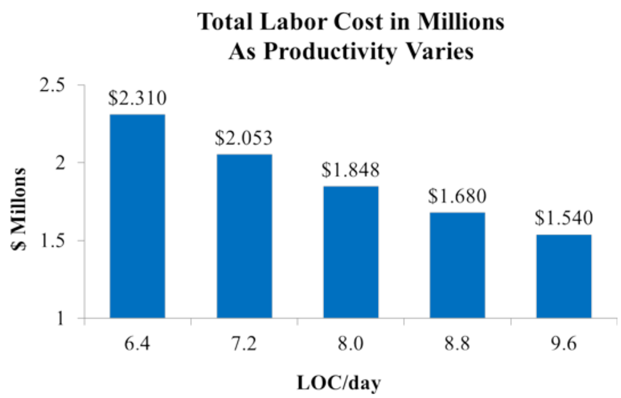
As the G&A costs increase and the other variables remain the same, the total labor cost increases. Again many companies work hard to reduce G&A costs.

The effect of LOC on Total Labor Cost as the number of LOC varies plus/minus 20% from 24,000 LOC to 36,000 LOC is shown in this graph.



As size of the program increases in terms of LOC and the other variables remain the same, the total labor cost increases.

Lastly the effect of **Productivity Rate** on Total Labor Cost as the productivity rate varies plus/minus 20% from 6.4 LOC/day to 9.6 LOC/day is shown in this graph.



Not surprisingly as employees become more productive, the total labor cost decreases. For this reason many companies implement process improvement plans.

It is employee salary, the number of LOC, and the productivity rate that have the largest impacts on total labor cost. For these reasons, software program/project managers work hard to ensure that they properly staff their program or project, build a quality product, increase productivity, and keep costs low, while ensuring that they deliver the product and its component on time. You should now have a better appreciation of the implications when a single variable changes, but in reality, it is not single variables, but multiple variables that are changing simultaneously making the total cost of labor very complicated.