Software Project Estimation

Introduction

Software estimation has had a lot of attention. Yet, it is still a challenge. Much has been written about metrics and methodologies to predict project completion. Software estimation is especially challenging because every piece of useful software is unique.

What follows is a "back-of-the-napkin" approach. It is not academically rigorous but has evolved through practice. The approach herein is a kind of recipe. A recipe is a step toward good results. You achieve the best results when:

- You know your people
- You understand the requirements
- You have built solutions with the technology before
- The stakeholders have a vested interest in the project

As with all recipes, best results are achieved when the cook makes the recipe his own and puts his spin on it.

Goal of Estimation

Predict Project Costs

The essential goal of software estimation is to predict project cost. An important note is whatever number you get, it will be wrong. I have found the disconnect between estimates and actuals to be the case with rigorous estimation techniques and sloppy ones. Both rigorous and sloppy techniques regress to a mean. If both techniques achieve mediocre outcomes, then why bother with a structured approach? The reason is because structure provides a solution framework. The framework forces you to acknowledge your assumptions. It also prompts you

to think about items which are often forgotten. And, it puts a fence around the costs so you can compare to the budget.

Project cost has several components:

- Timeline required to complete the project
- Materials used in the project
- Licenses (or permits) needed for the project

Cost, effort, and the number of people needed are derived from the above.

Cost of Ownership

Ownership cost estimation is out of scope for this article. However, the candidate solution and project estimation can inform ongoing recurring costs. Such costs includes people needed to support the solution, licensing and vendor support costs, and operational costs.

Terminology

Below are terms used in this article. Note the definitions are specific to the subject of estimation, and more narrowly, the current flavor herein. You may be accustomed to other terms which connote the similar meanings.

Definitions

Cost is the expense to complete a project, inclusive of labor (effort), licenses, and materials.

Effort is the central estimation metric. Effort is the amount of work one person must expend. Effort is measured in person*hours or person*days. For example, one person*hour is the amount a work one person completes in one hour. If the work is perfectly divisible, then two people could theoretically complete the same amount of work in 30 minutes. Similarly, if only 25% of a person's attention is available to complete the task, it would take 4 hours to complete. In some tools, such as MS Project, effort is called *work*.

A **component** is a module or artifact which can be built as a unit.

A **doneness step** is an action needed to complete a component. A set of doneness steps comprise a kind of "recipe" to complete a component.

Duration is the amount of time needed to complete a doneness step.

Timeline is the overall project calendar from start to finish. Timelines are usually defined by stakeholders' business objectives.

Rate is the average cost per person per unit of time, such as $/hr or £/day.

Materials is the physical stuff needed for the solution, such as new hardware. Cloud solutions complicate material costs. Cloud providers usually charge a fee per unit of use. If developing a cloud solution, then the material cost is the expense for the cloud resources multiplied by the duration of the project.

Lifecycle stages refer to software development activities, such as requirements gathering, prototyping, developing, testing, etc.

Resources are the people on the team who will do the work. Personally, I prefer the term P*eople*. MS Project and other tools usually use the word *Resource.*

A **License** is the authorization from a vendor to use their product. Depending on the vendor's revenue model, license costs can easily slip into ownership costs. For purposes of the subject at hand, licenses should focus on the costs to develop the solution and put into production for one year.

Project Start is the date when the project officially begins, complete with funding and staffing.

Target Date is the go-live date. Staff may be needed after the target date for post-live support.

Partitionability refers to the number of parallel tracks a project can sustain. For example, painting the exterior of a house could accommodate several simultaneous painters. Painting a closet would allow for a limited number of people to work at the same time. Closet painting is less partitionable than exterior house painting

Scope is the set of requirements the project must fulfill.

Cost, Effort, Duration, and Resources

Relating Cost

I like to avoid math. However, given the topic is estimation, there are a few basic relationships to know.

First, *cost* is a function of *effort*, *rate*, *materials*, and *licenses*:

$$Cost(Effort, Rate, Materials, Licenses) = (Effort * Rate) + Materials + Licenses$$

For example, if

- A contractor's rate is $150/hr
- An effort estimate is 480 hrs
- We must purchase a new server for $10,000

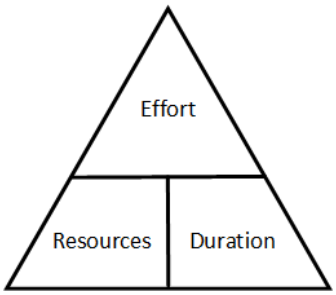
- Platform software license cost $2,000

The cost is

$$(480 \text{ hrs} * \$150/\text{hr}) + \$10{,}000 + \$2{,}000 = \$84{,}000$$

Relating Duration, Effort, and Resources

Duration, effort, and resources have a tidy little relationship as depicted below:



The key principle is the conservation of effort for a given project scope and solution. In other words, the real effort number is fixed for a given project scope. Only a change to scope or solution can affect effort. On the other hand, duration and resources can vary.

	Month 1	Month 2	Month 3	Month 4
Person 1	Effort = 16 person*months			
Person 2				
Person 3				
Person 4				

has the same effort as

	Month 1	Month 2	Month 3	Month 4	Month 5	Month 6	Month 7	Month 8
Person 1	Effort = 16 person*months							
Person 2								

Ideally, as responsible practitioners, we would build our estimate and inform our colleagues of the duration, where duration is effort divided by the number of resources:

$$Duration = Effort / Resources$$

However, businesses usually set a target date. Therefore, with a target date in hand, the duration is fixed:

$$Duration = TargetDate - ProjectStart$$

Once we have the duration and effort, can therefore estimate the number of people (resources) needed:

$$Resources = Effort / Duration$$

Herein is a key difference between an estimate and reality: This estimation approach assumes a) fungibility of resources and b) effort is evenly partitionable. In real life, neither is true. Some things just cannot be accelerated by adding people.

> "The bearing of a child takes nine months, no matter how many women are assigned."
> — **Frederick Brooks**

An estimate is not a plan. An estimate can allow imaginary fungibility and partitionability at the expense of accuracy. A practitioner can improve accuracy by adding details which account for how many people can realistically work on an activity at the same time. A plan prescribes how to build the project, accounting for all dependencies and constraints.

Estimation Steps

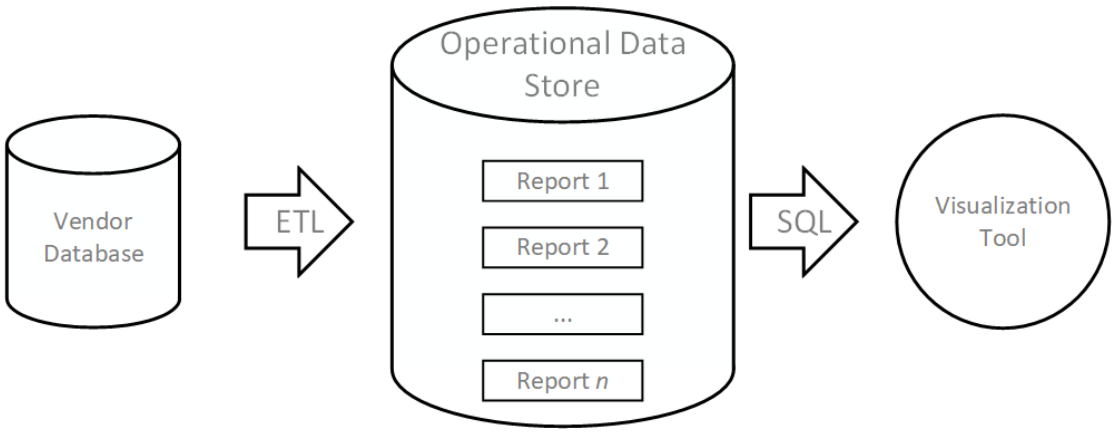
Effort is the essential part of the estimate. It is also the most challenging to figure out. This approach uses the following steps to arrive at project effort:

1) Choose a candidate solution
2) Identify the solution components
3) Account for Initiation Costs
4) Determine steps of doneness for each component type
5) Assign a duration for each step
6) Add lifecycle stage durations
7) Build the duration estimate
8) Build the effort estimate
9) Build the timeline estimate
10) Account for additional costs
11) Account for risks, contingencies, and assumptions
12) Summarize the estimate
13) Refine the estimate

1) Choose a Candidate Solution

Use your understanding of the requirements to inform the set of possible solutions. Drop infeasible solution candidates. Use the Choosing a Reporting Solution guide to help frame which solutions might work. Narrow the selection based on non-functional requirements.

For purposes of discussion, we'll use an Operational Data Store solution.



To further aide the example, we will assume we need to create an ODS for the WideWorldImporters database. WideWorldImporters ODS must report on the following:

- Orders
- Sales
- Purchases
- Inventory

2) Identify Solution Components

Work backwards from the Requirements and the Vision. Use the candidate solution architecture as a framework for what needs to be estimated. Think through what it would take to build the solution.

Identify Component Types

The essential output of the Operational Data Store is a report. For the sake of example, a report would have the following component types:

1) Visualization
2) Storage
3) ETL
4) Scheduling/Triggering

3) Account for Initiation Costs

For each component type, identify one-time initiation or procurement steps. Below is a hypothetical example using our component types. It is all just illustration; it is not a specific template. You the practitioner must lead your team to define the one-time costs of each component type.

1) Visualization
    a. Technology selection – allow time to select the technology platform.
    b. Licensing
    c. Platform development
    d. Server allocation
    e. Installation
2) Storage
    a. Technology selection
    b. Licensing
    c. Server Allocation
    d. Installation
3) ETL
    a. Technology selection
    b. Licensing
    c. Installation
4) Scheduling/Triggering
    a. Technology selection

For each initiation / procurement step, consider

1) Project team effort
2) Shared service effort
3) User effort
4) Supporting all environments such as dev, test, and prod
5) Disaster recovery costs

ID	Item	Project Team Effort	Shared Service Effort	User Effort	Production License	Test License	Development License	Disaster Recovery	Total
1	**Visualization**								
1.1	Technology selection	5 d	5 d	3 d					
1.2	Licensing				$ 1,000.00	$ -	$ -	$ 1,000.00	
1.3	Platform development	10 d							
1.4	Server allocation		10 d		$ 10,000.00	$1,000.00	$ 1,000.00	$ 10,000.00	
1.5	Installation		5 d						
2	**Storage**								
2.1	Technology selection	5 d							
2.2	Licensing				$ 50,000.00	$ -	$ -	$ 50,000.00	
2.3	Server Allocation		10 d						
2.4	Installation		5 d						
3	**ETL**								
3.1	Technology selection	0 d							
3.2	Licensing				$ -	$ -	$ -	$ -	
3.3	Installation								
4	**Scheduling/Triggering**								
4.1	Technology selection	5 d	5 d		$ -	$ -	$ -	$ -	
		25 d	40 d	3 d	$ 61,000.00	$1,000.00	$ 1,000.00	$ 61,000.00	
	Rate	$ 150.00	$ 150.00	$ 180.00					
	Total	$ 30,000.00	$ 48,000.00	$ 4,320.00	$ 61,000.00	$1,000.00	$ 1,000.00	$ 61,000.00	$206,320.00

4) Determine Steps of Doneness

In our parlance, “Steps of Doneness” refer to the specific actions which the developer(s) must complete for each component type. Determine Steps of Doneness for each component type.

1) Visualization
    a. Query – write a query against the ODS
    b. Filter – provide a way to narrow the results
    c. Presentation – present the information according to the requirements
2) Storage
    a. Table – data structure to hold the report
    b. Indexes – indexes needed for efficient presentation
    c. Partitioning – structure for efficient storage and retrieval
3) ETL
    a. Source Extraction Query – write a query against the source system
    b. Transformation – resolve foreign key references and denormalize as needed
    c. Insert – insert into the storage table
    d. Error Handling – account for possible problems
    e. Reconciliation – provide a mechanism to ensure data quality
4) Scheduling/Triggering
    a. Control Integration – call the custom code from the scheduling service
    b. Error Handling – detect errors in the custom code
    c. Notification – notify administrators when problems occur

5) Assign Durations

Assigning a duration is the real trick. There are established methods for assigning duration, such as Wideband delphi. **The key principal is to work as a team to assign the duration.** Do not conjure up numbers alone and bestow them on the team. The developers must own the estimate.

Assign a **typical** duration for each doneness step. The typical duration will represent steps of medium complexity. Focus on just the development portion of the activity. Think back to prior experiences. What were the average durations for a single person to complete a specific doneness step?

Assign a level of effort for high and low complexity. Again, working as a team, think how to adjust the typical (or medium) complexity durations to get to low and high complexity. Any rule will work so long as the team buys into it. For example, low complexity durations can be 50% of the medium complexity. High complexity durations can be 200% of the typical durations.

Component Type	Doneness Step	01-Low	02-Med	03-High
Visualization	a. Query – write a query against the ODS	1.5 d	3.0 d	6.0 d
Visualization	b. Filter – provide a way to narrow the results	2.0 d	4.0 d	8.0 d
Visualization	c. Presentation – present the information according to the requirements	0.5 d	1.0 d	2.0 d
Storage	a. Table – data structure to hold the report	2.0 d	4.0 d	8.0 d
Storage	b. Indexes – indexes needed for efficient presentation	2.0 d	4.0 d	8.0 d
Storage	c. Partitioning – structure for efficient storage and retrieval	1.0 d	2.0 d	4.0 d
ETL	a. Source Extraction Query – write a query against the source system	1.0 d	2.0 d	4.0 d
ETL	b. Transformation – resolve foreign key references and denormalize as needed	0.5 d	1.0 d	2.0 d
ETL	c. Insert – insert into the storage table	2.0 d	4.0 d	8.0 d
ETL	d. Error Handling – account for possible problems	2.5 d	5.0 d	10.0 d
ETL	e. Reconciliation – provide a mechanism to ensure data quality	2.5 d	5.0 d	10.0 d
Scheduling/Triggering	a. Control Integration – call the custom code from the scheduling service	2.0 d	4.0 d	8.0 d
Scheduling/Triggering	b. Error Handling – detect errors in the custom code	2.0 d	4.0 d	8.0 d
Scheduling/Triggering	c. Notification – notify administrators when problems occur	0.5 d	1.0 d	2.0 d

Don't get too hung up on assigning durations. You will iterate further as the estimate comes together. Just get something on the page.

Agree on your time units. Some folks like to measure time in hours. Others prefer days. Do what makes sense for the team and the project. Be wary of single tasks taking more than forty hours or 5 days. One-week tasks are not wrong, just make sure you know what goes into them.

6) Account for Methodology and Lifecycle Stages

Project lifecycle stages are activities like eliciting requirements, technical design, development, and testing. Decide up front on how to account for lifecycle stages. Methodologies such as Agile or Waterfall may affect how to account for lifecycle stages.

In a waterfall approach, each component may have requirements, design, development, and testing stages. Decide if the metrics refer to just the development stage or if it includes all stages.

Agile methods address lifecycle stages differently. Steps of doneness may include activities such as prototyping and obtaining user feedback, which are design and testing activities respectively. In fact, an estimate for an agile project may be a pivot of a waterfall estimate. Agile practitioners break projects into sprints or iterations. Each sprint contains all the components to implement a small number of stories.

We could add the lifecycle activities as doneness steps. For example, it may make sense to add steps for requirements-gathering, designing, developing, etc. In my experience, adding life-cycle tends to inflate the estimate and prevents the team from thinking through what it takes to deliver individual components. In practice we do not elicit separate requirements for each component type. When the estimate accounts for time in terms of lifecycle stages, the result over-counts the time allocated to the lifecycle activities.

Instead, the approach presented here estimates lifecycle activities as a function of development effort. For example, the grid below shows requirements take 10% of the effort needed to develop a given component type. The allocation is additive.

	Lifecycle Allocations			
Component Type	**01-Rqmts**	**02-Design**	**03-Dev**	**04-Test**
ETL	10%	10%	100%	35%
Scheduling/Triggering	10%	10%	100%	35%
Storage	10%	10%	100%	35%
Visualization	10%	10%	100%	35%

7) Build the Duration Estimate

ID	Level	Item	Functional Area	Component Type	Complexity	Life-Cycle	Iteration	Duration
1	1	Orders					Iteration 1	
1.1	2	Visualization	Orders				Iteration 1	
1.1.1	3	01-Rqmts	Orders	Visualization	03-High	01-Rqmts	Iteration 1	4.0 d
1.1.2	3	02-Design	Orders	Visualization	03-High	02-Design	Iteration 1	1.6 d
1.1.3	3	03-Dev	Orders	Visualization	03-High	03-Dev	Iteration 1	16.0 d
1.1.4	3	04-Test	Orders	Visualization	03-High	04-Test	Iteration 1	5.6 d
1.2	2	Storage	Orders				Iteration 1	
1.2.1	3	01-Rqmts	Orders	Storage	02-Med	01-Rqmts	Iteration 1	1.0 d
1.2.2	3	02-Design	Orders	Storage	02-Med	02-Design	Iteration 1	1.0 d
1.2.3	3	03-Dev	Orders	Storage	02-Med	03-Dev	Iteration 1	10.0 d
1.2.4	3	04-Test	Orders	Storage	02-Med	04-Test	Iteration 1	1.0 d
1.3	2	ETL	Orders				Iteration 1	
1.3.1	3	01-Rqmts	Orders	ETL	03-High	01-Rqmts	Iteration 1	3.4 d
1.3.2	3	02-Design	Orders	ETL	03-High	02-Design	Iteration 1	8.5 d
1.3.3	3	03-Dev	Orders	ETL	03-High	03-Dev	Iteration 1	34.0 d
1.3.4	3	04-Test	Orders	ETL	03-High	04-Test	Iteration 1	11.9 d
1.4	2	Scheduling/Triggering	Orders				Iteration 1	
1.4.1	3	01-Rqmts	Orders	Scheduling/Triggering	01-Low	01-Rqmts	Iteration 1	0.5 d
1.4.2	3	02-Design	Orders	Scheduling/Triggering	01-Low	02-Design	Iteration 1	0.5 d
1.4.3	3	03-Dev	Orders	Scheduling/Triggering	01-Low	03-Dev	Iteration 1	4.5 d
1.4.4	3	04-Test	Orders	Scheduling/Triggering	01-Low	04-Test	Iteration 1	0.5 d
2	1	Sales					Iteration 2	
2.1	2	Visualization	Sales				Iteration 2	
2.1.1	3	01-Rqmts	Sales	Visualization	02-Med	01-Rqmts	Iteration 2	2.0 d
2.1.2	3	02-Design	Sales	Visualization	02-Med	02-Design	Iteration 2	0.8 d

Now the estimate takes shape. List out each item on a worksheet along with component type and complexity. Each line item is a **task**.

Durations were given by component type and complexity. Therefore, create a lookup which finds the duration based on component type and complexity. Break out the item by lifecycle. You can further assign the task to a phase, sprint, or iteration.

Earlier, we came up with lifecycle duration factors. The duration factor is based on lifecycle and component type. The pseudo math below illustrates the relationship:

$$Duration_{task} =$$

$$BaseDuration(\ componentType, Complexity) * LifecycleFactor(\ ComponentType, LifeCycle)$$

8) Build the Effort Estimate

Effort is the amount of work a person must complete on the project. A person's project work depends on their role. Each role's involvement will depend on the lifecycle stage of each item. Identify the roles on the project and their level of participation in each lifecycle stage.

Role	01-Rqmts	02-Design	03-Dev	04-Test
BA	100%	25%	25%	100%
Dev	25%	100%	100%	100%
User	100%	0%	25%	100%

For each lifecycle, indicate how much each role will contribute. In the example above, business analysts (BA's) are heavily involved in requirements gathering and testing, but less so in design and development. Developers have basically the inverse allocation. Developers participate mostly in design and development. Users partner with BA's, so their participation follows the BA's.

ID	Level	Item	Functional Area	Component Type	Complexity	Life-Cycle	Iteration	Duration	BA	Dev	User
1	1	Orders					Iteration 1				
1.1	2	Visualization	Orders				Iteration 1				
1.1.1	3	01-Rqmts	Orders	Visualization	03-High	01-Rqmts	Iteration 1	4.0 d	4.0 d	1.0 d	4.0 d
1.1.2	3	02-Design	Orders	Visualization	03-High	02-Design	Iteration 1	1.6 d	0.4 d	1.6 d	0.0 d
1.1.3	3	03-Dev	Orders	Visualization	03-High	03-Dev	Iteration 1	16.0 d	4.0 d	16.0 d	4.0 d
1.1.4	3	04-Test	Orders	Visualization	03-High	04-Test	Iteration 1	5.6 d	5.6 d	5.6 d	5.6 d
1.2	2	Storage	Orders				Iteration 1				
1.2.1	3	01-Rqmts	Orders	Storage	02-Med	01-Rqmts	Iteration 1	1.0 d	1.0 d	0.3 d	1.0 d
1.2.2	3	02-Design	Orders	Storage	02-Med	02-Design	Iteration 1	1.0 d	0.3 d	1.0 d	0.0 d
1.2.3	3	03-Dev	Orders	Storage	02-Med	03-Dev	Iteration 1	10.0 d	2.5 d	10.0 d	2.5 d
1.2.4	3	04-Test	Orders	Storage	02-Med	04-Test	Iteration 1	1.0 d	1.0 d	1.0 d	1.0 d
1.3	2	ETL	Orders				Iteration 1				
1.3.1	3	01-Rqmts	Orders	ETL	03-High	01-Rqmts	Iteration 1	3.4 d	3.4 d	0.9 d	3.4 d
1.3.2	3	02-Design	Orders	ETL	03-High	02-Design	Iteration 1	8.5 d	2.1 d	8.5 d	0.0 d
1.3.3	3	03-Dev	Orders	ETL	03-High	03-Dev	Iteration 1	34.0 d	8.5 d	34.0 d	8.5 d
1.3.4	3	04-Test	Orders	ETL	03-High	04-Test	Iteration 1	11.9 d	11.9 d	11.9 d	11.9 d
1.4	2	Scheduling/Triggering	Orders				Iteration 1				
1.4.1	3	01-Rqmts	Orders	Scheduling/Triggering	01-Low	01-Rqmts	Iteration 1	0.5 d	0.5 d	0.1 d	0.5 d
1.4.2	3	02-Design	Orders	Scheduling/Triggering	01-Low	02-Design	Iteration 1	0.5 d	0.1 d	0.5 d	0.0 d
1.4.3	3	03-Dev	Orders	Scheduling/Triggering	01-Low	03-Dev	Iteration 1	4.5 d	1.1 d	4.5 d	1.1 d
1.4.4	3	04-Test	Orders	Scheduling/Triggering	01-Low	04-Test	Iteration 1	0.5 d	0.5 d	0.5 d	0.5 d
2	1	Sales					Iteration 2				
2.1	2	Visualization	Sales				Iteration 2				
2.1.1	3	01-Rqmts	Sales	Visualization	02-Med	01-Rqmts	Iteration 2	2.0 d	2.0 d	0.5 d	2.0 d

Effort then becomes a function of role type and the lifecycle stage.

$$\text{Effort}_{\text{role}} =$$

$$\text{RoleParticipationFactor(RoleType, LifeCycle)} * \text{Duration}$$

9) Build the Timeline Estimate

There are two ways to figure out the timeline. You can calculate the timeline based on resources and effort. Or, there will be a target date. The latter is more common.

To calculate the timeline, divide the effort by the number of resources.

To accommodate a target date, adjust the number of resources until the timeline fits. Getting the estimate to meet the target date may require an unrealistic number of resources. Forcing the timeline to fit by adding too many resources indicates either the target date is not feasible, the project is fraught with risks, or the step durations need some additional thought.

Break the timeline down by lifecycle stage, phase, sprint, or iteration. Define a project start date. Summarize the duration as workdays for each lifecycle stage or iteration. Excel pivots are great for summarizing.

Resources	2			
Level	3			
Row Labels	**Sum of Effort**	Duration	Start	Finish
01-Rqmts	30.4 d	15.2 d	01-Jan 2019	22-Jan 2019
02-Design	43.6 d	21.8 d	22-Jan 2019	20-Feb 2019
03-Dev	232.0 d	116.0 d	20-Feb 2019	05-Aug 2019
04-Test	69.2 d	34.6 d	05-Aug 2019	20-Sep 2019
Grand Total	**375.2 d**	187.6 d		

10) Account for Additional Costs

Fixed Costs and Oversight

Certain costs are strictly a function of the timeline. For example:

- Project Leadership – project management, architecture, customer relationship, etc.
- Shared Services - database administrators, networking services, virtual hardware support, etc.
- Rent – office lease, data center lease, equipment lease, etc.
- Recurring charges – telecom/Internet/VPN access, server certificates, cloud-based tools, etc.

Estimate fixed costs after calculating the duration.

Post-Live Support

Determine the roles needed to support the project once it goes live. For example, a business analyst may be needed for two months. Some contracts and statements of work may specify support level requirements independent of the effort estimate.

Some methods for calculating post-live support are:

1) Add support as a fixed cost after project go-live. The cost can be a fixed number of roles for a fixed duration. For example, two developers for 4 weeks after go-live.

2) Calculate support as a function of effort. Support requirements based on the amount of effort required to deliver the project is hard to mathematically justify, but some may find the method intuitive.

3) Calculate support based on the architecture. Allocate some number of roles per unique component type for a duration. Roles based on component type or technology ensures there is enough expertise to handle whatever need arises.

4) Calculate support based on defect rates. Defect rate counting can get tricky. Minimum quality (or maximum number of defects) for go-live may be spelled out in a contract or statement of work. Define defects in objective terms.

Agree with your stakeholders about support requirements at the start of the project. Decide on the method to account for support. Include the additional costs in the estimate based on the selected method.

Availability

The team is almost certainly not going to be able to plow through the timeline without time off. Create a team-size adjusted buffer to account for things like:

- Vacations
- Sick Days
- Personal Time Off

You can use some method, such as prorating existing policies against the timeline. Whatever you do, account for it in a realistic manner.

11) Account for Risks, Contingency and Assumptions

The following method accounts for risk. Risks may manifest as higher than expected defect rates, slower than expected progress, or other unexpected challenges.

Uncertainty Risk

Uncertainty can be thought of as known-unknowns, with apologies to Donald Rumsfield. The following technique gauges uncertainty around the project by posing questions about the team and the candidate solution. Responses to the questions are binary. Each negative response increases risk by a small factor. Quantify total uncertainty by adding up the factors. Work with your team to create the question list and decide the percentages.

ID	Risk	Yes	No	Response	Risk Factor	Risk Spread
1	Does the team understand the domain?	the team knows the business	the team is not familiar with the business	N	5%	5%
2	Is the team familiar with the tools and technology?	team has experience with the tools and technology	the team has not used the tools or the technology in the past	N	5%	5%
3	Are the requirements stable?	the business requirements are stable and not likely to change significantly	the organization is still figuring things out	N	5%	5%
4	Is the solution well-understood?	the architect, team, and business users understand the solution and are in agreement	the solution is in flux, for any reason	N	5%	5%
5	Are the stakeholders available to support the project?	there is a clear mandate for the project and the stakeholders agree with the direction	there is disagreement among the stakeholders concerning project direction	N	5%	5%
						25%

Use the factors to create an estimate range. Below are some sample questions and percentages.

LowEstimate = Estimate – (Estimate*RiskSpread/2)

HighEstimate = Estimate + (Estimate*RiskSpread/2)

Contingency Risk

Contingency accounts for unknown-unknowns. Such things include higher than expected defect rates, sudden change in project direction, or unplanned resource shortage. Usually contingency is a factor used to multiply the top-level costs.

Assumptions

Identify key assumptions you expect to be true. An assumption can be something like “Will use existing technology platform” or “Will onboard new resources within an 8-week timeframe”. A useful assumption is one in which if it proves to be false, the estimate will change. If the estimate is the same either way, then the assumption is not worth noting. Ordinarily, assumptions are not accompanied by effort. The estimate will only be affected when an assumption proves to be false.

12) Summarize the Estimate

Summarize the estimate by putting all the key inputs and results on one page. The summarization page can be a little like a tax form. It pulls all the calculations together.

Project Information

Start Date – identify when the project is expected to begin. The estimation sample assumes the project is fully staffed and funded by the start date.

Team Size - state the expected number of people (other than leadership). The number of people on the project is an key factor. Divide the team size into the effort to calculate the project timeline.

Budget Amount – indicate the expense amount set aside for the project

Duration

Estimated Duration – the sum of effort across all roles divided by the Team Size. Remember an estimate is not a plan, but merely an guess of effort, time, and cost. Estimates do not address specifics needed for an execution plan.

Estimated End Date – The duration in business (working) days added to the start date. For example, Excel includes the WORKDAY function which calculates dates using only business days.

Target Date – The original date set by the stakeholders for when the project must be complete.

Timeline Gap – The difference between the Estimated End Date and the Target Date measured in days.

Timeline Gap% - The Timeline Gap divided by the estimated duration. A large gap is a red flag. Anything greater than 20% should give cause for concern.

Initial Cost

Initiation Cost – the sum of all the costs to get the project started.

Development Effort Expense

Role – list the roles needed on the project. These are the project roles from the effort estimate.

Hours – the total amount of effort for each role on the project.

Rate – either a per-role rate or a blended rate.

Total – the total cost across all the roles

Oversight Expense

Role – list the roles needed on the project. These are the Project Manager, Architect, or other supervisory roles.

Hours – the total amount of effort for each role on the project.

Rate – either a per-role rate or a blended rate.

Total – the total cost across all the roles

Base Total Estimate

Base Total Effort – Sum of the Development Effort and the oversight effort

Base Total Cost – the sum of the Initial Cost, Effort Expense, and Oversight Expense

Risk Factor

Risk Factor Percentage – sum the risk factor coefficients

Risk Factor Cost – multiply the Risk Factor Percentage by the Base Estimate

Contingency Percentage – display the Contingency Percentage

Contingency Cost – multiply the Contingency Percentage by the Base Estimate

Total Risk Adjustment – Add the Risk Factor Cost and the Contingency Cost

Total Cost

Low Estimate – subtract ½ of the Risk Adjustment from the Base Estimate

High Estimate – add ½ of the Risk Adjustment to the Base Estimate

Budget Gap – Low – difference between the Low Estimate and the Budget

Budget Gap – High – difference between the High Estimate and the Budget

Project Information

Start Date	1/1/2019
Team Size	4

Duration

Estimated Duration	023.5 d
Availability Buffer	3.4 d
Estimated End Date	2/6/2019

Initial Cost

Initiation Cost	$ 206,320.00

Effort Expense

Role	Hours	Rate	Total
BA	633	150.00 USD/hr	$ 94,950.00
Dev	1000.8	150.00 USD/hr	$150,120.00
User	611.2	180.00 USD/hr	$110,016.00

Oversight Expense

Arch	187.6	150.00 USD/hr	$ 28,140.00
PM	187.6	150.00 USD/hr	$ 28,140.00

Risk Factor

Risk %	25%

Total Cost

	Low	High
Project Cost Range	$ 540,475.25	$ 694,896.75

13) Refine the Estimate

Sense Check

Take a step back and decide if the estimate makes sense at all. If it takes two years to build four reports, then tighten the metrics. If you can meet the target date but the risk range is high or you need to hire an unrealistic number of people, you may need to revisit the scope.

After sense-checking the metrics, examine the gap between the target date and the estimated date. If the gap is still greater than 20%, the project may have too much risk to go forward. Likewise, evaluate the gap between the budget and estimated cost. Too high of a cost gap may also mean the project has too much risk.

Review and Feedback

Walk through the estimate with your team before sharing it with the stakeholders. Make sure you have buy-in from the folks doing the work. One thing if note is the adage "work expands to fill the box". Start with a tight estimate and inspire your team to rise to the challenge. Increase estimates only with good justification. I know I perform better when I have just a little less than what I think I need.

Execution Plan

An estimate is not a plan. The estimate accounts for effort and cost. A plan accounts for sequential dependencies and actual steps to complete the project. Assemble a plan based on the doneness steps and actual resources (names of people). Find the predecessors for each steps and level the plan. The completion date may blow out compared to the estimate. Use the discrepancy to refine the estimate or the plan.

Conclusion

There are a million ways to estimate software. This article describes only one. It is a crude, back-of-the-napkin approach. However, use it and you will get out of it what you put into it. Spend time thinking about the doneness steps. Work with your team to create the estimates. Incorporate past performance into the estimates. Sense check the results. Be realistic but don't sandbag.

Further Reading

There is a lot of great work in the discipline of software estimation. Learn as much as you can. Check out the material below.

Steve McConnel's book

Wideband Delphi Method

Software Project Estimation: The Fundamentals for Providing High Quality Information to Decision Makers