Project Lifecycle Pro

IT Project Estimation – A Practical No-Nonsense Guide

This whitpaper covers the process of IT project estimation related to developing or implementing software. This is a practical guide based on what has worked for me over the last thirty years. I started my career performing far too precise analysis which missed some of the bigger picture. Now I identify the critical elements while performing less precise analysis.

My approach performs IT project estimation in several steps. First, a high-level estimate is created with a 20% contingency during the plan stage. Next, a detailed estimate is created with a 10% contingency on the first iteration of the analyze and design stages. Finally, story points are used to estimate the Agile sprints.

I also typically create both a top-down and bottom-up estimate. The top-down is used in the early planning stages and it shifts to a more detailed bottom-up estimate as more information is learned about the project. A bottom-up estimate may also need to be adjusted based on the practicalities of labor staffing. Meaning, you determine you need 3.2 developers to meet the release dates but you might have to staff full-time resources so you must budget for 4 full-time developers.

Plan Stage IT Project Estimation

The goal of this stage is to perform a high level IT project estimation so management can evaluate the financial return and budget for the project (i.e. assess the business case). This should all be desktop analysis done on the back of an envelope. Not looking for precision at this point, but looking for a range. Is it going to cost $10K, $100K, $1M?

Software Package License Costs

If a software package will have to be purchased, I perform the IT project estimation during the Plan Stage based on the vendor's standard license costs. A short list of possible of vendors is developed and I either issue a request for information (RFI) to the vendors or pull standard pricing right off their websites. If the software will require hiring a software consulting partner to help with implementation, then I get typical costs from the software vendor (some offer standard packages) or consulting partner recommendations which can then provide standard pricing quotes.

Labor Costs

During the Plan Stage labor costs are developed in collaboration with IT managers and possibly external development firms. An experienced project manager and/or IT manager should be able to review a project description and initial technical design to estimate if a project will be 3 months, 6 months, a year or more in length. Key to the IT project estimation will be the initial technical design

which should identify the major technical components required, such as, APIs, new user interfaces, etc. This information will typically be captured in a context diagram, technical component inventory or the work breakdown structure of the initial project plan.

Generating the initial labor estimate then boils down to multiplying the typical team size by the number of months estimated to complete the project. Hourly wage rates are varied based on whether the resource will be internal or external. So management ends up getting a high level estimate. For example, $1.2M ($1M + 20% contingency) with a total project length of 12 months.

Analyze/Design Stage IT Project Estimation

The Analyze and Design Stages produce the detailed business requirements and user stories, as well as, the technical requirements and user stories. In a large Agile project, the Analyze and Design stages may be iterated to develop detailed user stories just-in-time for the sprints. However, the goal should be to finalize the overall project budget and schedule at the end of the first iteration of the Analyze/Design Stage. This is so management can make a fully informed decision as to whether to approve the beginning of development and formation of the full project team.

But how to generate an accurate complete IT project estimation when all the requirements and users stories have not been completed? The answer is to estimate at the component level. By this point, the team should know how many and what type of technical components must be built. Technical components are items such as, APIs, user interfaces, database tables, reports, etc.

Task-Level IT Project Estimation is Too Complex

Early in my career, I'd attempt to create super complex project plans while trying to assign labor hours for every task. This excessive detail gave the illusion of improved accuracy, but it was a fool's dream. It also made the project plans extremely complex in that every component being built needed four tasks (analyze, design, build and test) with labor estimates.

The project plan should be simple and streamlined and not attempt to capture every user story as a task. The work breakdown structure should be: project stage, release and major component. The project plan should focus on an overall timeline and only contain the detailed tasks that will be managed outside the Agile Kanban board, such as, change management, marketing, business case development, etc.

Component-Level IT Project Estimation is 'Just Right'

I now perform IT project estimation by either using an Excel spreadsheet tool or an estimation tool like in Project Lifecycle Pro to estimate at the component level. Each standard technical component type is assigned twelve labor standards. A labor standard is assigned for analyze, design, build and test for each of three effort rankings: high, medium, low. For example a report ranked 'low' might have a build labor standard of 1 hour. While a report ranked 'high' might have a build labor standard of 4 hours. These labor standards are stored as a tab in the estimation spreadsheet and then populated using VLOOKUP for each of the component records.

The IT project estimation is performed in a meeting with the product owner and IT manager or architects. Each technical component type which must be built is ranked by effort as either: high, medium or low. The standard labor hours are then automatically applied to the analyze, design, build and test tasks. The team reviews the standard labor values and makes manual adjustments as necessary.

Estimate Non-Critical Components in Bulk

Definition of the technical components can be iterated. Non-critical components such as reports can be estimated as a total rather than having to identify every specific report needed at the beginning of the project. For example, you could simply guesstimate that about ten reports will have to be built. Then multiply the standard labor estimate per report by ten. However, you will want to specifically identify critical components before you lock down the IT project estimation. For example, are you going to modify an existing user interface to capture additional data, or are you going to have to build an entirely new user interface and workflow?

Labor Totaled by Role

Once all the components are ranked by effort, the total labor hours can be totaled for analyze, design, build and test. These tasks also correspond to the roles on the team. Analyze is usually performed by the business analyst or product owner to document functional requirements. Design is performed by the architects or IT leads to document technical requirements. Build is performed by developers, and includes unit testing. Test is the acceptance testing performed by business analysts or QA department. So now, the project manager has a total estimate of labor for business analysts, architects, developers and QA.

IT Project Estimation by Release

Finally, the list of technical components to be built are assigned to releases. And thus, the labor estimate can now also be summed by release. The project manager and product owner can now begin release planning. Given the planned staffing of the team and the hours estimated for the release a release date can be forecast. Of course, the team staffing or the release scope can also be refined to hit target release dates.

For example, ten technical components have been prioritized for Release 1. The total estimated build labor is 500 hours. The team is assigned three developers. At 80% utilization, each developer can work 32 hours per week (the remaining eight hours is for vacation, holidays, meetings, or any other non-working time). 500 hours divided by 96 hours per week (32 x 3) results in 5.2 weeks, rounded up to 6 weeks. Now the project manager has a release estimate of 6 weeks or 3 sprints, and can give management a realistic possible release date. The project manager can also begin coordinating marketing and change management efforts around the target release date.

Analyze/Design Stage Gate

At the end of the first iteration of analysis and design, a stage gate should be held for management to approve the beginning of the build stage. Up to this point, project costs should have been fairly low but once build begins costs will increase significantly. If a software package is being purchased, then licenses will have to be purchased to support the install and configuration. If software is being developed, then the internal or external developers will have to be staffed along with QA resources.

For management to make the decision to proceed at a much higher spend rate, they need the following information. They will need a timeline for the releases: when are they going to get something for their money?. They need a staffing plan: how many people will have to be dedicated to the project and for how long? They will need a total cost estimate within a plus/minus ten percent: how much is it going to cost? Finally, they need a business case comparing the costs to benefits: will they get a fair return on their investment?

Build Stage IT Project Estimation

Agile software development is all about building functionality in the correct priority and working as efficiently as possible. Agile is about getting the functionality built, not wasting time planning and budgeting. So now I need to marry the overall IT project estimation done above with running the day-to-day sprints. And I need to do this while minimizing the distraction for the developers.

Group User Stories by Component and Release

The first step is to assign user stories to components. This can be done by making each component an epic or creating custom labels per component. As user stories are created the author should be assigning the component value. Now, user stories can easily be sorted by component and prioritized accordingly.

JIRA has standard functionality to define a release by assigning user stories. Continuing the example from above, we have ten components planned for the first release. Let's assume we have an average of three user stories per component. We have the user stories assigned to the components so we can prioritize the thirty user stories towards the top of the backlog. Finally, we assign the thirty user stories to the first release in JIRA.

Sprint Planning and Story Points

Please note this is the first time in this estimation process that the developers will be pulled away from coding to plan. The first thing I do when planning a sprint is meet with the product owner to ensure that the product backlog is correctly prioritized for at least the next sprint. I hold this meeting a couple of days prior to the team sprint planning meeting.

The entire team is assembled for the sprint planning meeting. The product owner starts the meeting by explaining their rationale for prioritizing the user stories and what they feel is the minimal viable

product for the next release. The product owner also shares the estimated release date but emphasizes that we will operate under Agile principles and that the release date will not be finalized until we are into the final sprint.

The team walks through each user story in order. Story points are assigned to each story, but the majority of the conversation should focus on ensuring the developers understand each user story and soliciting their input on the best approach. For example, the user story could have made a false assumption on the best technical approach. Or, developers may suggest changing the priority of some stories because they are related and will be more efficient to code together. Of course, the developers are ultimately determining what amount of work is feasible for the sprint.

Monitoring and Adjusting the IT Project Estimation

The sprint planning meeting is the first opportunity to adjust the labor estimates. If we continue the example, thirty stories were planned in three sprints for the first release. If only five stories make it into the first sprint, then it is likely all the stories are not going to be completed in three sprints. I probably wouldn't start communicating a new release date at this point, but I'd make sure no commitments were made against the release date. I'd tell marketing not to sign any advertising contracts, or training not to schedule any training dates yet. Of course, I could also request additional developers if the release date has to be met.

During the sprint, the burndown chart will be the ongoing guide to forecast the possible release date. It will be fairly obvious whether the original release date is realistic based on the burndown chart progress. JIRA also shows the status of the stories assigned to the release. If 66% of the release stories are completed by the end of the second sprint, then the release date is probably sound.

Adjusting and Maintaining Labor Standards

As part of the sprint retrospective, the project manager should review if the labor standards used for the component-level estimate where accurate. If the labor standards were significantly off, then this is a good time to share the initial estimate for a component with the developers and discuss why the estimate was wrong. Perhaps it was an issue that could not have been foreseen, like the new functionality unearthed a bug in existing code. Or perhaps, the wrong people were ranking the complexity of the component.

In Conclusion

I hope that I've made the case that component-level IT project estimation achieves the right balance of providing management high-level guidance while avoiding unneeded detail. Remember that your estimates are going to revert to the mean, so the positive and negative variances should average out to a reasonable estimate. Management only cares about the total cost so they can budget and plan resources. They mostly care about cost overruns which are addressed by adding

20% contingency during the plan stage and 10% later. You can always give back funds later in the project as things become more certain.

The project team should be focused on building the solution as efficiently as possible not on worrying about meeting estimates or detailed reporting of their time. You shouldn't care if one report took 30 minutes to build and another took 1.5 hours as long as they average out to your standards. So there isn't any need for developers to track their time to this level. You really don't even care if component estimates are accurate as a whole, you care if your overall budget estimate or release dates are going to vary enough that management must be informed. Meaning, you are going to need more money or other groups dependent upon the release date must change their plans.

IT Project Estimation Related Info

If you are looking for a project management tool for IT project estimation and currently have Salesforce, please consider my Salesforce project management app: Project Lifecycle Pro (PLP). It is a comprehensive project management office (PMO) and project portfolio management (PPM) tool. PLP supports the IT project estimation process I describe above. You maintain a labor standards table within PLP which is customized for your component types and populated with your labor standards. Components are inventoried within PLP and tied to releases and requirements. The components are also ranked by complexity which automatically applies your labor standards and sums the totals to the release. One really nice feature is that the labor standards can be applied or overridden manually for individual components at the click of a button.