Software Estimation – How to go about it?

Murali Chemuturi

Background

When there is lack of clarity on a subject, a plethora of literature is brought out on the subject. More often than not, the literature and the books on the subject – ignore the basics and address the advanced topics - confound the confusion rather than clear it. As a developer of software estimation tools, a consultant on the subject, and a trainer on the subject, this confusion often confronted me. Most people know one or more software sizing techniques – most have a reasonable understanding of Function Point Analysis or Use Case Points estimation techniques – but are stymied when it comes to making a real life software estimate.

Software estimation is carried out more than once in the life a software development project. The project acquisition stage is the one where the detail available to estimator is minimal. The confusion of estimation arises in this stage. All other stages, progressively, increase the detail and hence reduce uncertainty (and confusion) in estimation.

Therefore, I am attempting to throw some light on software estimation – basics - as attempted at project acquisition (or project approval) stage.

What most people either - do not know or do not have – a clear understanding of is – how to apply these techniques to a real life situation – that is when the boss / marketing department asks for a software estimate against an RFP (Request for Proposal) or a PIN (Project Initiation Note). The following, as I found – are the questions from the estimators, not only beginners but also experienced ones.

1. How to arrive at all the components that need to be included in the estimate – before the design is completed?
2. How to arrive at the complexities for each of the components?
3. How do I get the productivity figure and from where?
4. How do I handle uncertainty?
5. What software sizing technique do I use?

I will address them in the following sections.

Breaking the project down into components and constructing the WBS

It is possible that there exists a detailed specification of the project for which software estimation needs to be carried out. That is an excellent scenario – the breaking down becomes easier.

The other end of the continuum is that the spec is limited to a small paragraph in an email message requesting a fixed bid. The breaking down is most difficult.

We normally go about estimation, somewhere in between these two scenarios. How do we go about it?

I will address the from the stand point of worst-case scenario.

The request for estimate would be some thing like this –

1. We need an estimate for developing a simple material management system for so and so company
2. We need an estimate for a no-frills web site that is functional for a business that wants to get on the net mainly to allow customers to browse thru catalogues and place orders and track status of order. Management reports are needed
3. We are presently using one write (or quick books or whatever) and as our operations have grown, looking to upgrade it to next level. Ability to use it over internet is desirable
4. A non-profit organization is looking to upgrade the existing operations into a collaborative environment. Flexible on technology to be used. Internet usage needs to be built in. Presently we are using MS-Office suite for our HR and Finance activities and also bring in our client interfacing activities into the ambit of the new system
5. So on and so forth

Of course, this is the worst-case scenario. How do we go about it?

First – read carefully – ignore all adjectives –

1. They want a Material Management System. They did not specify any exceptions – so assume that all necessary functionalities are expected.
2. They want a fully functional e-commerce web site. “No frills” generally means, no frills in your proposal - they expect your lowest price
3. They want web based accounting (or HR or Marketing or …) system
4. Non-profit means “not making any profit” – expect lower price. The system requirements are in Excel sheets all over the organization – you need to collect them, study them and develop the software.

Now that the scope is known, the project needs to be broken down into its constituent components – this is how to go about it –

- Adopt a top-to-down approach
- First – breakdown the project by one level –
  - Material Management into Purchasing, Storing and analysis. E-commerce web site into Loading the catalogues, Browsing catalogues, Shopping cart, order processing with payment processing, MIS etc.
  - Then breakdown each major function into its constituent components
  - Continue to breakdown, from one level to next level, until no further breakdown is possible
  - Remember to list them down in an Excel sheet or a similar spreadsheet
  - For each of the components, decide, how they get their inputs and the outputs they give out and how they convert inputs into outputs. Also list out any intermediate components the conversion process may need

There - you have the list of components. However, the catch is that one needs to be knowledgeable about the domain at hand. If we have it – excellent – but – if we do not have? Now where do we get the info to carry out the above-cited breakdown?

Here are some alternatives –

1. You worked on a similar project and you know – this is the ideal scenario
2. Catch hold of a software developer colleague and get him to transfer his knowledge to you

3. Try your boss – he may not be able to debug your program, but amazingly, bosses are knowledgeable in these matters
4. Catch hold of a functional expert in your organization (surely, your organization has a Finance manager, a HR manager, a Marketing manager and so on) or an outside expert and learn from him. Go back to them after you completed the breakdown and get the expert to check for completeness
5. Internet is amazing – it seems to possess amazing knowledge and demo software for all and sundry functional areas. Use your favorite search engine and search for the application at hand and download a few demos and try them out. It will make you knowledgeable enough so that you can carry out the breakdown the project into its constituent components.

Now you know how to breakdown a project into its constituent components and the list is made.

This list is normally called the WBS (Work Breakdown Structure).

Complexity of components

Almost all the software size estimation techniques necessitate rating the complexity of the component (or transaction as some techniques prefer to call). Those techniques also specify rules for complexity rating in a great detail – sometimes, even going to micro level, to controls on the screen or number of data elements in a file / table. It is OK to comply with these rules, once an application is developed and available to retrofit the estimate. But, when you are estimating at the time of project acquisition stage (that too, without any guarantee that the project would be awarded to you and therefore your objective is to spend as little time as absolutely necessary to come out with the estimate), it would be impractical to be able to use those complexity rules in Toto.

Here is what I suggest –

1. Use your programming experience and best judgment and rate the complexity for each of the components of the WBS
2. Review each of the complexities, critically examining, and modify as you see fit and proper
3. Get the complexities reviewed by a senior person, not necessarily your boss but a person who has at least two years of more experience than you
4. More reviews add more accuracy to the estimate.
5. Remember, that a Login screen is of average complexity in some size estimation techniques and in some applications!

Now, it may appear that this is not a foolproof method of deciding complexities of components – I agree.

However, I suggest this method - keeping in mind, the detail available and the effort and resources that can be committed to this activity during project acquisition stage.

Now fit this WBS in the chosen software sizing technique.

Appropriate software sizing technique

How do you select the appropriate technique? Here are some suggestions –

1. The technique your organization has standardized on
2. The technique with which you are most familiar and comfortable in using. In case you are not familiar with any of the techniques,
    a. Try and use Function points – by far the most popular and used technique – first though – you need learn it – there are many free resources on the Internet, for you to download and learn
    b. Use Task based estimation this is just estimating the time required for each of the components on the WBS and compute the total effort needed for the project
3. The technique your customer requested or is familiar with
4. The technique for which you have reliable (to the extent possible) productivity data
5. Take suggestion from your boss

Applying the productivity figure

Once you have the software size, apply Productivity figure – person hours per unit of software size – and obtain the effort required for the project.

Now, where do we get this Productivity figure? Here are some sources –

1. Your organizational data – it would be available with Quality Assurance department or SEPG (Software Engineering process Group) or Knowledge Repository, or PMO (Project Management Office)
2. Industry benchmarks – possible sources are - IFPUG (International Function Point User Group), ISBSG (International Software Benchmarking Standards Group) nearest SPIN (Software Process Improvement Network), SPMN (Software Project Managers Network).
3. Search the Internet – who knows, this data may be available on a site like www.effortestimator.com !
4. Discuss with your boss or the marketing persons in your organization. An experienced marketing person usually has an uncanny knack of being able to arrive at the duration and effort required to execute a project.
5. Use your best hunch, if all else is not possible

Uncertainty in estimation

As the word – Estimation – itself suggests, there is inherent uncertainty. What I suggest here minimizes it – not eliminates - it.

1. Always give three estimates of software size – best-case, normal-case and worst-case scenarios. Best case is 90% of your estimate and worst-case is 110% of your estimate and normal-case is your original estimate.
2. When applying productivity figure, again apply the same three values.
    a. To the worst-case software size apply the worst-case productivity figure – that is 110% of the normal productivity figure
    b. To the best-case software size apply the best-case productivity figure – that is 90% of your normal productivity figure
    c. To the normal software size, apply the normal productivity figure
3. Thus you have three scenarios –
    a. Worst-case scenario – when software size is highest and productivity is lowest
    b. Best-case scenario – when the software size is lowest and productivity is highest
    c. Normal-case scenario – when the software size and productivity are normal
4. Give these figures to whoever requested the estimates.

What we are precisely communicating is –

1. The project is likely to be executed in the normal-case scenario
2. When every thing goes in an excellent manner, we could execute the project in the best-case scenario
3. When every thing goes wrong, we could execute the project in the worst-case scenario

Remember that pricing the project and committing the delivery schedule is a commercial decision dictated more by the client-requirements and to some extent by the competition. It is possible that marketing commits a best-case scenario and the project be executed in the worst-case scenario! But – we need to plan for making sure that we meet the commitments made to the client by infusing more resources or better methodologies or any other way.

Project execution has a crucial role and should not be undervalued in ensuring that estimates are met. What needs to be achieved during project acquisition stage is to make the best possible attempt to make a realistic estimate and also give the best and worst-case scenarios for marketing to make an educated pricing and delivery commitments to the client.

The above methodology, hopefully ensures that.

Feel free to send your feedback to murali@chemuturi.com