Nesma on estimating

In this whitepaper Nesma presents information to enlarge your knowledge about estmating and to support your activities in this area.

The information is structured around four subjects:

1. Four steps of estimating using FPA
2. Example of estimating using fpa
3. Devil's triangle
4. Basis of Estimate

1. Four steps of estimating using FPA

More than just counting

To estimate a software development project based on the number of function points of the project, one does not automatically use the standard productivity rate for the specific development environment. Beforehand, one should determine whether there are specific circumstances that may influence (in a positive or negative sense) the productivity rate for the project. This analysis results in the expected project productivity rate.

The estimating and budgeting process consists of more activities than a function point analysis. FPA is just a supporting tool in this process. One needs to perform four steps to make a budget for a project based on FPA:

1. Determine the functional size of the information system to be developed
2. Determine the standard productivity rate for the development environment
3. Determine the expected project productivity rate
4. Budget the project

1: Determine the functional size of the information system to be developed

The amount of hours needed to develop an information system is related to the size of the information system in the same way the number of hours needed to build a wall is related to the size of the wall (number of bricks). The bigger the information system, the more hours one needs, and the more expensive it will cost. Function points are a good measure for the functional size of an information system. The functional size of an information system is expressed in a number of function points.
The way in which FPA determines the functional size of an information system is described on the Nesma website.

2: Determine the standard productivity rate for the development environment

Determining the (functional) size of the information system is not enough. The next step is, to determine the specific development environment of the information system. It is clear, that the specific development environment (software and hardware) plays an important role in the needed number of development hours, especially in the construction phase (programming and testing).

There may be a big difference in developing an information system in a third generation programming language, like Cobol, or in a fourth generation language. The same remark is valid for the various hardware platforms and architectures, such as mainframe, PC, client server, SOA environment or the cloud. It is generally necessary to maintain a standard productivity rate for each development environment: needed development hours per function point. A company should record these standard productivity rates for each phase of the system development life cycle. For the Construction phase, however, the reliability and repeatability is more consistent than for earlier phases of the project life cycle.

Standard productivity rates (specifically per development environment) are determined based on experiences with completed similar projects in the past. These project productivity rates from the past are an indicator for the expected productivity rates in new similar projects in the future. Experiences in new projects may lead to an update of the standard productivity rates. To sensibly use a standard productivity rate in future situations, a company should clearly determine which project activities are included in this rate (for more explanation, see step 4). Activities that are not included in the rate should be budgeted individually in step 4.

3: Determine the expected project productivity rate

Using a check list with so called “productivity attributes”, one should establish whether there are specific circumstances for the project under consideration that influence the productivity in positive or negative sense (for example, inexperienced/very experienced project staff, a new development tool, etc.). One should estimate the particular influence of these circumstances and “translate” them into the expected project productivity rate (hours per function point for the project under consideration). This expected project productivity rate might differ from the standard productivity rate for the specific development environment.

The step from function points to the initial project budget (needed development hours in the project) is simple: multiply the number of function points by the expected project productivity rate in hours/fp.

4: Budget the project

For each productivity rate (hours/fp) an organization should record which activities are included in this rate, and which are not. Activities that are usually not included in a productivity rate are activities that are not correlated to the size of a project or not always performed in a typical project. For example, in some projects an organization needs to have extensive talks with external third parties, but not in all projects. This could be a reason for the organization not to include this type of activity in the (standard) productivity rate.
Effort needed for activities that are not included in the productivity rate should be estimated in

another way; these hours should be added to the initial project hours (hours related to the activities that are included in the standard productivity rate) as determined in step 3, see above.

Another good way to estimate the project costs other than that based on FPA is to determine a traditional estimate from experienced staff and specialists. Comparing both estimates (FPA versus traditionally estimated) may provide a better analysis in more detail, and a better estimate.

Using the expected project hours needed, the financial rates and other expenses, one finally determines the project budget (cost).

Confront the result

Finally, Nesma advises to always calculate a budget based on a “conventional” activity estimate of specialists too, based on the Product Breakdown Structure and the Work Breakdown Structure.
Compare this specialist estimate with the FPA estimate.

Although estimating based on functional size is widely recognized as being superior to specialist estimates, the confrontation of the two approaches together results in an even better reliability of the estimate than either of the two approaches apart.

2. Example of using estimating using FPA

We follow the procedure as described in the page Four steps of estimating and illustrate how to estimate a project.

Successively the four steps are described:

1. Determine the functional size of the system to be developed
2. Determine the standard productivity rate for the development environment
3. Determine the expected project productivity rate
4. Budget the project

1. Determine the functional size of the system to be developed

Apply the way of working as described at the Nesma website. For making a good FPA count, consulting the Nesma Counting Guidelines is necessary.

Determine the number of function points in the scope of the project, based on the functional specifications of the project.
For this, we can conduct three different types of functional size analysis: either a detailed count, an estimated count, or an indicative count. These three functional size analysis types are explained at the Nesma website**.** The result of all three analysis types is a size in function points. The difference is the reliability of this size, but also the moment the analysis can be conducted. The reliability of the budget is related to the reliability of the count of the functional size. Typically the indicative count can be performed earliest, but the trade-off is the lowest reliability.

For this example, suppose we want to budget a project with a functional size of 85 function points.

2: Determine the standard productivity rate for the development environment

Suppose the system will be written in Java. The functionality is moderately complex and the requested duration is 10 weeks. Phases to include are technical design, coding, unit testing, systems testing, and support of the user organization during the user acceptance test and writing of the user manual.

Based on the history base with experience project data the company knows the standard productivity rate for this kind of project in this technical environment is 10 hours per function point (fictitious).

3: Determine the expected project productivity rate

The specific characteristics of this project are analyzed using a checklist with so called "productivity attributes".
This analysis results in the conclusion that there are no specific conditions present that will influence the standard productivity rate. Hence the project productivity rate is set to the standard productivity rate: 10 hours per function point.

4: Budget the project

Suppose the standard productivity figures of the company include all project activities, except writing of the user manual. Then this activity needs to be budgeted separately. Suppose the estimate for writing of the user manual is 40 hours.
The budget of the project is then calculated as follows:

Hours for activities included in the standard productivity rate: 85 fp * 10 hour/fp	850 hours
Hours for additional activities (not included in the standard productivity rate)	40 hours
Project budget (in hours)	890 hours
Project budget (in Euro), assuming a blended hourly rate of €100: 890 * €100 =	€ 89.000

To this project budget might need to be added other costs like travel expenses and costs for purchase of equipment.

Confront the result

Finally, Nesma advises to also calculate a budget based on a "conventional" activity estimate of specialists. Compare this specialist estimate with the FPA estimate.

If such an expert estimate would be much higher, e.g. 2000 hours, then first scrutinize the expert estimate. What are the specific reasons that this project would be more expensive than similar projects in the past?
If there are such reasons known to the experts, and this is verified by the investigation, then there

is good reason to raise the project budget. Else it is best to stick with the FPA estimate.
The same applies when the expert estimate is way lower than the FPA estimate.
If the expert and FPA estimates are not far apart, it could be a good strategy to set the budget in-between the two estimates.

Always keep in mind that the experts certainly do have a valuable opinion, and based on their inside knowledge might be right. However the FPA estimate is based on the organizations own historic performance. This is a very good predictor for future similar projects.

3. Devils triangle

Estimating is more than calculating hours

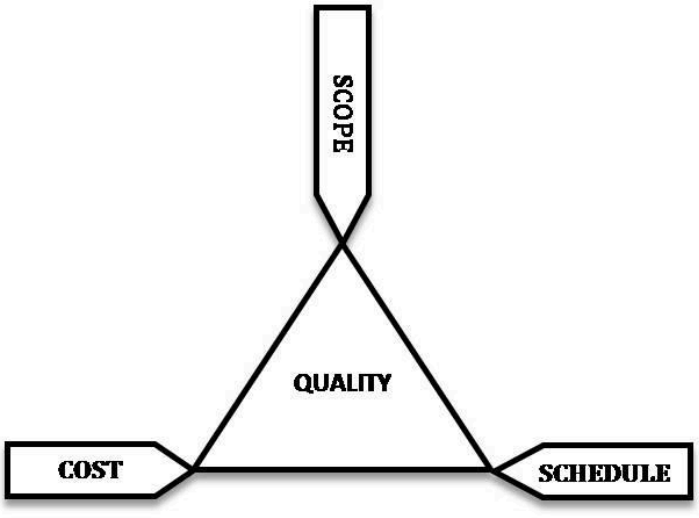
Estimating hours and cost is not the whole story, even with using FPA. One should not forget the quality of the product that is delivered, and the scheduled duration of the work.
Especially if the schedule is under pressure, the number of hours needed tends to rise.
Unfortunately at the same time the level of quality tends to drop, requiring even more effort hours to get right. To deliver the product at the agreed date and quality might require much more effort than in a schedule without time pressure. How to calculate this right?

The core metrics

Four main factors are involved in estimating a software project:

- Scope (size);
- Schedule;
- Quality;
- Cost (effort).

A well-known visualization of these factors is the ‘project triangle’:



The above project metrics dimensions are also known as the core metrics. Scope is the (functional) size of the project. For costs, you can also read 'effort hours', as cost is calculated by multiplying the number of effort hours by a (blended) tariff.
Mutual depending relationships exist between all four of these core metrics. All of them can be put as result in the middle of the triangle, and will then be influenced by the other three in the corners acting as constraints. If one of the constraints in the corners changes, the result in the middle will also change.
These four core metrics are attributes of the ***product*** to be delivered, as requested by the client of the project as part of the project requirements (constraints). In principle, a project team cannot change these; they agree to deliver these as the result of their work.

For the record: a fifth core metric exists, usually not shown in the triangle but also having mutual impact on the other four. This is of course the efficiency by which the work is executed by the project team. It generally is called:

- Productivity rate.

Productivity rate is not product-related but ***execution*** related: the effectiveness / efficiency with which the project team transforms its effort (cost) into the delivered product.

Using the core metrics

Estimating must include the value of all ***five*** core metrics for sensible results. The result of this is that there is no such thing as ***one*** good outcome for estimation. It is way more complex. And that is the reason the 'project triangle' is also called the 'devils triangle'.

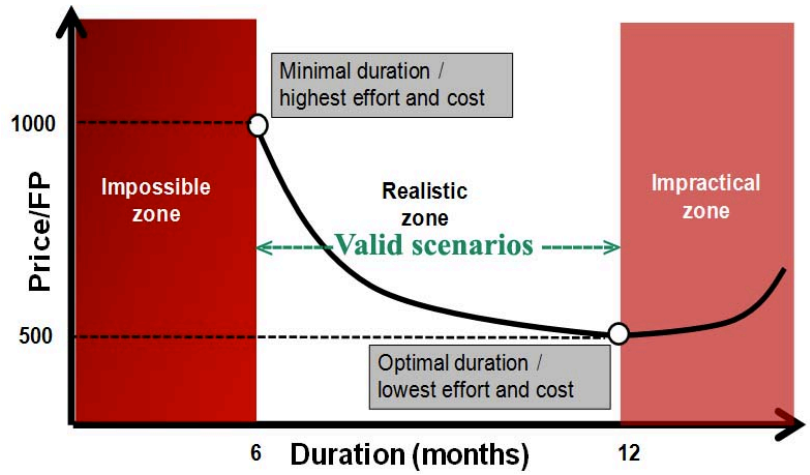
The practical consequence of the mutual depending relationships between the core metrics is the following.
When one of the three constraints in the corners is changing somehow, the other two constraints also have to change. This is necessary in order to have the result in the middle (quality in the example above) to remain constant. If not, the quality will be impacted. Schedule (duration) and Cost (effort) are usually underestimated because the scope is not clearly defined at the start of many software projects. For this reason it is important to make sure the quality will be of a sufficient level at delivery.
For ***Classical projects*** it is quite common that the (functional) scope has a tendency to expand. If the level of quality is maintained then either the cost or the schedule will have to expand too. Usually both do. This explains why Classical IT projects have a tendency to overshoot both in cost and schedule.

In contrast to Classical projects, for ***Agile projects*** the schedule and cost are fixed. Therefore, for Agile projects the quality and scope are the dependent results. The situation is completely similar. If the quality is also maintained, then the delivered scope is the variable result of an Agile project and has a tendency to fall short of the expectations at the start of the project.

Scenarios in estimating

Because of the potential large impact of the mutual depending relationships of these core metrics, it is important to be aware of them when estimating. Different sets of assumed values of the core metrics will result in different scenarios with different outcomes for the projects duration, quality and cost, and size for Agile.

However, this must be seen as an opportunity rather than a problem. This creates an extra dimension of freedom of choice for the result of the project. It makes it possible to optimize the project for lowest cost, shortest duration (time to market), best quality, or a sensible combination of these, to be determined by the needs of the organization.
A whole range of valid scenarios does exist, as illustrated by the following figure:



Because of this potential large impact, Nesma strongly recommends to include their agreed or expected values into the contract when outsourcing a project.

The devils triangle is the main reason that tooling is recommended to be used when estimating. These tools also facilitate the scenario-building and decision process.

4. Basis of Estimate

AACE International's Total Cost Management (TCM) Framework identifies a basis of estimate (BOE) document as a required component of a cost estimate.

In the TCM Framework, the BOE is characterized as the one deliverable that defines the scope of the engagement and ultimately becomes the basis for change management. Note: In the software services industries, the term 'engagement' is commonly used and synonymous with 'project'. When prepared correctly, any person with (capital) project experience can use the BOE to understand and assess the estimate, independent of any other supporting documentation. A well-written BOE achieves those goals by clearly and concisely stating the purpose of the

estimate being prepared (i.e. cost/effort/duration study, project options, funding, etc.), the project scope, cost basis, allowances, assumptions, exclusions, cost risks and opportunities, contingencies, and any deviations from standard practices. For software services the effort expended is the main driver for cost and duration. In addition the BOE is a documented record of pertinent communications that have occurred and agreements that have been made between the estimator and other stakeholders.

Content of the BOE

A well prepared BOE will:

- Document the overall engagement scope.
- Communicate the estimator's knowledge of the engagement by demonstrating an understanding of scope, quality and duration as it relates to cost.
- Provide a record of all the hypothesis and assumptions taken into account for deriving the BOE.
- Alert the stakeholders to potential cost risks and opportunities.
- Provide a record of key communications made during estimate preparation.
- Provide a record of all documents used to prepare the estimate.
- Act as a source of support during dispute resolutions.
- Establish the initial baseline for scope, quantities, effort, duration and cost for use in engagement control.
- Provide the historical relationships between baselined estimates throughout the project lifecycle.
- Facilitate the review and validation of the estimates.

This Recommended Practice is intended to be a guideline, not a standard. It is understood that not all organizations that prepare estimates employ the same processes and practices, and therefore, may opt to use this information either in part or in its entirety. However, in all cases this RP supports creating consistent estimate documentation that provides a high degree of traceability and repeatability for the estimate.

The Recommended Practice for a basis of estimate (BOE) document for software estimation can be found in the publications list on the Nesma website.