Software Estimation using a Combination of Techniques

By Klaus Nielsen, MBA, PMI-ACP, PMI-RMP, PMP

This article covers the basics of software estimation techniques by taking a brief journey through time in order to understand the past and current states of estimation techniques. The purpose of this journey is to build upon the best practices of estimation and, in the process, understand more about the techniques and best practices from the traditional software and agile environments. The presentation of the wide range of software estimation techniques will help broaden the perspective on estimation from traditional techniques to the new agile techniques.

The article will demonstrate how traditional and agile estimation techniques may be employed in new ways. Overall, the article is based upon the foundation of best practices on software estimation (with emphasis on the use of more than one or two techniques), getting more than one opinion, estimates might start at ranges, and the value of documenting your estimates for trust and historical data. Software estimation with a combination of techniques is a bit of an art, but mostly, it is just applied best practices project management in practice.

> "Estimation is the true application of knowledge, skills, basic tools, and techniques in project management in order to meet the requirements of a particular project; with the requirements of a particular project come maturity, mix methods, and best practice."

Know the Past, Find the Future of Estimation

Estimation techniques may be considered a young discipline within information technology, but still with a bit of history that points toward the future. Several sources refer back to Peter V. Norden, Project RAND (Delphi), SEER-SEM, and Frank Freiman (PRICE — 1969) as some of the pioneers of estimation in the 1960s. In the 1970s, recognized techniques, such as SLIM, were developed. Even with a fair range of estimation techniques, Fred Brooks states in *The Mythical Man-Month* (1975) that "Our estimating techniques are poorly developed. We are uncertain of our estimates," which seems to have haunted us ever since. In the 1970s, the economically based estimation techniques from Wolverton, Walton, and Felix were published and, later, the eminent *Function Points* (1979) by Alan Allbrecht, and Barry W. Boehm's first version of *CoCoMo* (1981) saw the light of dawn. These estimation techniques described are more or less still in use in updated versions and partly covered in the article.

Estimation Challenges

Estimation is all about estimating the project constraints, which include, but are not limited to, Scope, Quality, Schedule, Budget, Resources, and Risk. The center of attention for academics and every day project managers has been on estimating time and costs, which are parts of *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)*—Fourth Edition, and include the Project Time Management and Project Cost Management Knowledge Areas. The literature of estimation may make it sound like estimation is a walk in the park, because most estimation techniques are fairly easy to understand and can be applied to relatively simple projects, but the testimonials of failures due

to wrong estimates are long and very expensive, according to Fortune 500 companies, where only 1 out of 4 projects completes on time and budget. Chris F. Kemerer claims that, in 1997, costs were two to three times the estimated and this was the norm; the last 10 to 15 years have just confirmed this fact with new and even bigger failures.

Estimation is an Ongoing Process

In most cases, top management and project managers are fully aware of the fact that precision in estimates varies over time with fairly imprecise estimates, perhaps 50% off in the initiating phase, which decreases to 25% in the planning phase, and hopefully 10% or less in the executing or later phases. This understanding is also theorized in the theory of cone of uncertainty or similar models (e.g., Roetzheim, 2000). This underlies the importance of re-evaluating the estimates as an ongoing process.

History of Software Estimations

Currently, the literature and practices around estimation techniques are fairly closely related to the development models of the traditional waterfall methodology and agile development. We tend to define project management as "the application of knowledge, skills, tools and techniques to a broad range of activities in order to meet the requirements of a particular project." This implies that the lack of knowledge of the techniques is a lack of project management skills. If we may add and alter a bit of ancient Chinese wisdom from Sun Tzu or his general, Sun Bin, to state: "If you know the techniques and know yourself, you need not fear the result of a hundred projects." The idea behind this alternation is the fact that projects are floating with uncertainties, in which we as project managers must know and use our arsenal of estimation techniques to overcome our estimation challenges in many projects. This implies using the best and most proper estimation techniques according to the situation of the project. Again, this may imply that, sometimes, traditional waterfall estimation techniques are more suitable to parts of an agile project and vice versa, which will be discussed in the next section.

Traditional Estimation Techniques for Waterfall Methodology Projects

Traditional estimation techniques are commonly deployed as part of the waterfall methodology. The following list of estimation techniques may not be complete but does contain the estimation techniques most practitioners would consider best practice and are hopefully known. The techniques include, in random order:

- Analogy
- Effort method
- Programming techniques (lines of code)
- Expert Judgment
- Delphi/Wideband Delphi
- Program Evaluation and Review Technique (Beta or normal)
- CoCoMo 1/CoCoMo 2
- Albrects Function Points
- Parametric methods (e.g., PRICE)
- Various top-down or bottom-up techniques

Traditional Estimation Techniques Explained

Analogy

The basis of the technique for estimation by analogy is to describe (in terms of a number of variables) the project for which the estimate is to be made and then to use this description to find other similar projects that have already been completed. This means that this story point may be similar to a story point in another project, which is half the size of this story point. We simply compare story points across projects and take the similarities and variations into consideration. It is a quick and easy method to apply, which fits well into an agile context. A variation of the analogy method is the technique called "Yesterday's Weather" by Kent Beck (2000), which means that the estimate of tomorrow is going to be the same as that of today.

Program Evaluation and Review Technique (PERT)

Most estimates are just one number where the PERT formula is mentioned in the fourth edition of the *PMBOK® Guide* (PMI, 2008) as a tool and technique of two processes — Estimate Activity Durations and Estimate Costs, which give us three fairly quick estimates. The PERT Formula: (OPTIMISTIC + (4 X MOST-LIKELY) + PESSIMISTIC)/6. This estimation technique gives us three estimates relatively quickly with uncertainties in considerations and could be more applied in agile project management than the case today.

Agile Estimation Techniques for Agile Projects

Some of the traditional estimation techniques are deployed for agile projects; all in all, however, these worlds seem fairly separate for now. The agile software estimation techniques include among others:

- Relative sizing
- Wideband Delphi
- Planning poker

- Affinity estimation
- Ideal time and elapsed time
- Disaggregation
- Bottom-up/top-down estimation
- Constructive agile estimation algorithm
- Time, budget, and costs estimation

Agile Estimation Techniques Explained

Relative Sizing

Relative sizing is an agile estimating method of estimation of scales. Some practitioners prefer to apply relative sizing using Fibonacci numbers 0,1,1,2,3,5,8,13,21,34,55,89,144… but options on scale are many. The concept of relative sizing is to estimate the size of a function or user story based on the size of another one. Let's illustrate this on cars; our examples include four types of cars: Tata Nano, Toyota Prius, McLaren F1, and the Bugatti Veyron Super Sports, which we have listed in order of relative effort, meaning that the Tata Nano is the cheapest car, whereas the Bugatti is the most expensive car of the four. Then we need to consider how much more expensive the Toyota Prius is compared with the Tata Nano, where our estimate is 14 times. The same goes for the other car, which we compare with the Tata Nano. If we then estimate the Tata Nano to cost US$5000 and taking 5 days to build, then we know that the Toyota Prius will cost US$5000 x 14 = US$70,000 or take 70 days to build. We estimate this to be a quick relative sizing of items or story points compared with the other items or story points.

Wideband Delphi

The original version of the Delphi method was first applied in the 1960s and the concept was based on asking a range of experts for estimates. The experts were not located in the same location and by all means unable to communicate and influence the other expert estimates. Clearly, these methods had some strengths and weaknesses, which the later method of Wideband Delphi attempted to overcome. One of the issues with the original Delphi methods was the wide range of assumptions the experts used to estimate and, because the experts were unable to talk, all estimates had different assumptions, making them fairly weak. The Wideband Delphi is a process that a team can use to generate an estimate. The project manager chooses an estimation team and gains consensus among that team on the results. Wideband Delphi is a repeatable estimation process because it consists of a straightforward set of steps that can be performed the same way each time:

1. Choose the Team
2. Kickoff Meeting Step
3. Individual Preparation
4. Estimation Session
5. Assemble Tasks
6. Review Results

After the team of estimators has been chosen (often the agile team), the kickoff meeting will take place during which the story points or items to estimate are discussed, as well as any assumptions or other considerations. After this meeting, all estimators have time for individual preparation in which the actual initial estimates are completed by the estimators without influence of the other estimators. All estimators are then assembled in an estimation session in which the estimates are discussed. This may end the estimates if they are close and can be agreed upon or the process may run again in order to narrow down the estimates. The Delphi technique is also good for agile teams because it is

1. An iterative process that is repeated;
2. An adaptive process based upon feedback; and
3. A collaborative process based on team participants.

Planning Poker

Planning poker is one of those agile estimation methods that may cause some veteran traditional project managers to smile a bit because it is an estimation method used while playing cards. Planning poker is a Delphi approach that works in the following ways (Desharnais et al., 2010):

- Individual stories are presented for estimation.
- After a period of discussion, each participant chooses from his or her own deck the numbered card that represents his or her estimate of how much work is involved in the story under discussion.
- All estimates are kept private until each participant has chosen a card.
- Finally, all estimates are revealed and the discussion can begin again.

Even though planning poker is a full planning process that combines estimation with identifying the scope of the project and the tasks required to complete the software and similar to relative sizing, planning poker is an estimating technique used to achieve consensus on work estimates. The estimators use a set of cards with Fibonacci numbers of 0, 1,1,2,3,5,8,13,21,34,55,89,144. A story point or item is presented for the group of estimators who discuss the content of the story point. Cards are shown and the estimators are

able to discuss and come to an agreement on the relative size of the story point.

Affinity Estimation

Affinity estimating is an estimation method that is easy to remember because we may recall it as t-shirt sizes, ranging from S, M, L, to XL or by the sizes of our coffee mugs.

An alternative is an estimation technique by Lowell Lindstrøm, which is also suitable for arriving at a high-level estimate of a big feature list called the product backlog, which is often more than 20 items.

The steps are:

1. Silent Relative Sizing
2. Wikipedia-like Editing of Wall
3. Place items into relative size buckets
4. Product Owner "Challenge"
5. Get it into an Electronic Tool

The first step in the process is silent relative sizing in which team members silently take a subset from the product backlog and place them on the board, in order of small to large. This step may take 5 to 10 minutes. The second step is an editing process in which the backlog is re-read and discussed and changes may occur. This step may take 20 to 40 minutes to complete. Now the items on the board will be placed in buckets of varying sizes (i.e., small, medium, and large). The product owner challenge is an exercise during which the product owner and his or her supporters play the devil's advocate. This implies asking all kinds of questions to the content of the board and having the team rethink their order. The questions should not be disrespectful, just challenge the decisions and order on the board. If the team decides on the challenge to reorder, they can do so. This step may take up to an hour. The last step is documentation of the results, often into an electronic tool.

Constructive Agile Estimation Algorithm

The Constructive Agile Estimation Algorithm (CAEA) is an algorithmic approach for the estimation of cost, size, and duration of a project. It incorporates various vital factors, namely: performance, complex processing, configuration, security, data transfer, operation ease, project domain, and multiple sites. All these factors require extra efforts in development of the software, thereby increasing the cost, size, and duration of the project. In this section, most estimation methods are non-algorithmic approaches, which heavily rely on expert opinion and historical data to be estimated precisely. Algorithmic approaches are useful, particularly when historical data and expert estimators are not available; some may argue that an algorithmic approach for estimation of agile projects leads to a step toward the engineering practices, thereby establishing the fact that these methods are not ad-hoc methods. Research by Bhalearao (Bhalerao et al., 2009) analyzes the usefulness of CAEA on various application domain projects of small sizes. The Constructive Agile Estimation Algorithm consists of three calculations, first new story point = story points + 0.1 * unadjusted values or NSP = SP + 0.1 * UV. The renaming part of the CAEA algorithmic approaches calculates the size and duration of the project.

Traditional Estimation Techniques for Agile Projects

The concepts behind most of the agile estimation techniques are based on a lightweight approach that favors interaction and communication as part of the process. Often we estimate velocity, story points, size, or effort. When examining the traditional estimation techniques analogy, the Pert and Delphi techniques come to mind as full-fitting alternatives and supplements. Going into greater details using functions point techniques and similar IT supported techniques may also work for agile estimations; for some part, they work against some of the basic ideas of agile methods but, in general, they could work well.

Agile Estimation Techniques for Traditional Projects

Many of the agile estimation techniques are lightweight and quick to use, which for many traditional waterfall methodology projects, may work well in the initiation phase with quick and rough estimates or even later in the planning phase. Agile estimation techniques, which have been used with success on traditional projects are affinity or relative sizing estimation in the area of change management and in the initiating phase for rough estimates on time or costs.

Use a Mix of Estimation Techniques

Whether agile or traditional projects and estimation techniques, there is no one technique that can provide you with the decisive final answer with certainty. This may sound harsh to some agile practitioners as it may slow down the process, because we need the individuals and interactions as highlighted in the values of the *Agile Software Development Manifesto*. In early 2000, we conducted some surveys in Denmark on the application of estimation techniques and noticed that the larger and the more mature IT companies tend to use a wide range of estimation techniques in various phases of projects, and that some techniques were developed in some phases but not in others, which underlies the

importance of using more techniques and at various phases. Estimation is not one-size-fits-all!

Estimation Tips

- Some of the worse examples of poor estimates are when practitioners conduct the estimate by themselves on their desktops. We need to know our estimation techniques as professionals, but estimating is all about people and communication or we will be faced with GIGO — 'Garbage In, Garbage Out.' Estimation best practice is a team effort.
- Management and project managers tend to view estimates as final numbers (consider Mark Durrenberger in 1999 or agile practitioners using ranges), perhaps just for a start. Early in the initiating or planning phase, a range of US$100,000 to US$110,000 may be sufficient and also point out the uncertainty in the estimates, whereas in later phases or with more knowledge, it may not seem as appropriate to estimate ranges as the final costs or time schedule of the project.
- Early on, most estimates are fairly inaccurate; often, however, we tend to believe in them if they are documented so we can understand the process behind the estimates and the data implied. This also goes for assumptions. With this in mind, in some cases, we might even correct the estimates or give a second opinion; however, all in all, documentation makes estimates more trustworthy. Lack of documentation combined with a weird looking estimate, right or wrong, may start a debate that could have been avoided. If you don't document for yourself and your stakeholders, do it for your colleges, because they will need the historical data in the future.
- Estimating something all new and sometimes chaotic is tough, but with the help of others it might not be as bad as it sounds. Historical data are available in most companies and not always visible, but if you have the data it may save you time and produce a better outcome, because some estimation techniques use some sort of historical data from analogy to function points, weighting, and similar techniques. Some veteran project managers may claim that no two projects or situations are similar, but most projects managers would still prefer 'flawed' data over no data.

Conclusions

Some project managers still claim that estimating is all about art — a good project manager knows how to estimate from his or her own experience, common sense, and with a little help from some friends. Others may claim estimation is mostly a science, because we apply a range of estimation techniques in settings with a variety of variables we need to take into consideration. Estimation is the true application of knowledge, skills, basic tools, and techniques in project management in order to meet the requirements of a particular project; with the requirements of a particular project come maturity, mix methods, and best practice.

About the Author

Klaus Nielsen, MBA, PMI-ACP, PMI-RMP, PMP is the managing director at Global Business Development in Denmark and an associate lecturer in project and program management at the IT University of Copenhagen, Denmark. Klaus has over 10 years of project management experience in managing and delivering complex, high-visibility information systems projects. He can be reached at kni@itu.dk

References

Beck, K. (2000). *Extreme programming explained: Embrace change*. Indianapolis, IN: Addison-Wesley.

Bhalerao, S., & Ingle, M. (2009). *Agile estimation using CAEA: A comparative study of agile projects*. 2009 International Conference on Computer Engineering and Applications. IPCSIT (vol. 2 2011). Singapore: IACSIT Press.

Desharnais, J.M., Buglione, L., & Kocatürk, B. (2010). *Using the COSMIC method to estimate agile user stories*. Istanbul, Turkey: Boaziçi University ACM.

Durrenberger, M.R. (1999). True estimates reduce project risk, *PM Network*, 45–48.

Peixoto, C. E. L., Audy, J-L-N, & Prikladnicki, R. (2010). *The importance of the use of an estimation process*. Cape Town, South Africa: Computer Science School, Pontifícia Universidade Católica do Rio Grande do Sul, Brazil. SDG'10, 8 May 2010.

Project Management Institute. (2008). *A guide to the project management body of knowledge* (*PMBOK® guide*) Fourth edition. Newtown Square, PA: Author.

Roetzheim, W.H. (2000). Estimating internet and intranet development effort. *Trends in Software Management*, issue 1–4.

Stellman, A., & Greene, J. (2011). *Applied software project management* — estimation. Sebastopol, CA: O'Reilly.