Is your project's best estimation method Agile or conventional?

Accurate estimation can make or break your project

Dina S. Ahmed

This article elaborates the importance of good estimation practices. It shows and categorizes a set of estimation techniques. It provides a high level comparison between Agile and conventional techniques and finally recommends some general estimation tips. Estimation can be an efficient practice with better awareness and collaboration.

Introduction

One of the major skills required for any project leader is having good sense of effort estimation. Mastering estimation is essential as it highly impacts projects delivery deadlines. Inaccurate estimation dramatically impacts the cost and profitability of any project. It also influences the perception and trust of all team members and customers towards the project leader. Knowing how to estimate is essential for all team members because they all have a part in the estimation process. There is no doubt of how crucial it is for the whole team to build up this capability. Yet, we have to acknowledge the challenges that we face during estimation process in the software industry where it is almost impossible to simply replicate the same project and assume that by analogy all estimates will sustain. Each project always comes with a new set of requirements, risks, technologies, team members, and so on. Estimation challenges can be summarized as:

- Extreme estimation that can be too optimistic or too pessimistic.
- An individual's incorrect perception that estimation is a commitment.
- Resistance to adapt to new emerging factors that could impact estimates.
- Lack of expertise or historical information that can provide initial ground for estimates.
- Uncertainty of the project scope or business requirements.
- Poor risk mitigation plan.
- Unmeasured ad hoc activities like meetings, holidays, demos, prototypes.

Learn more. Develop more. Connect more.

The new developerWorks Premium membership program provides an all-access pass to powerful development tools and resources, including 500 top technical titles (dozens specifically for web developers) through Safari Books Online, deep discounts on premier developer events, video replays of recent O'Reilly conferences, and more. Sign up today.
Lack of using updated tools and processes to track estimates.

In order to gain this skill, each individual needs to be aware of what options they have available to them. What are the different techniques that can be used and when should they be used? This article shows various estimation techniques that help control projects and improve decision making. The article starts with an overview for some popular conventional methods, then shows some agile techniques. The article ends with a summarized comparison between agile development versus conventional methods. You'll also learn a set of ingredients to make you a better estimator.

**Conventional estimation methods**

Many project estimates are built using the guidance of previous project histories or the expertise of subject matter experts (SMEs). In the following section you will learn the methods that focus on the similarities between projects.

**Historical information**

**PROBE (Proxy Based Estimating)**

This technique was created by Watts Humphrey of the Software Engineering Institute at Carnegie Mellon University as part of the PSP (Personal Software Process). It is based on the idea that If an engineer is building a component similar to one he built previously, then it will take about the same effort as it did in the past. In the PROBE technique, a repository is kept with all detailed information for each component that was built before. Then when a new project needs to be estimated, it is broken down into components that match the historical information. A formula is then used to calculate the estimate for each task.

**COCOMO II**

In 1980, Constructive Cost Model (COCOMO) was developed. It is based on analyzing the result of statistical study for 63 software development projects. Ten years later, an updated version called COCOMO II was introduced that covers modern development life cycles and uses a wider set of data. The new model takes a set of input variables and calculates the target estimates based on the previously studied projects.

**Group estimation**

**Wide Band Delphi** is a process developed in 1940. It depends on group estimation where the project manager selects a moderator and an estimation team. The moderator is unbiased and manages the meeting. The moderator also ensures that everyone participates. The project manager must be a member of the estimation team so that the team is aware of the priority of the requirements.

The process starts with a kickoff meeting where the team is familiarized with the goal of the project. The output of the session is a high-level list of tasks and a set of assumptions written by the moderator and distributed to the team. Each member of the estimation team individually creates a set of preparation results, as seen in Figure 1, in order to prepare for the next estimation session meeting. Additional tasks or assumptions can always be added for further discussion. In
the estimation session, the moderator distributes an empty form to each estimator. Each estimator fills in the estimation form based on his or her preparation results.

**Figure 1. Preparation results**

![Preparation results](image)

As seen in Figure 2, each estimator puts the list of tasks in each row with the corresponding estimates and assumptions of the estimation form. Then the moderator gathers this information and writes it onto a white board for the first round.

**Figure 2. Estimation form**

<table>
<thead>
<tr>
<th>WBS</th>
<th>Task Name</th>
<th>Est. Hours</th>
<th>Delta 1</th>
<th>Delta 2</th>
<th>Delta 3</th>
<th>Total</th>
<th>Assumptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Detailed requirements</td>
<td>16</td>
<td>+5</td>
<td>+2</td>
<td></td>
<td></td>
<td>Two meeting with customer</td>
</tr>
<tr>
<td>2</td>
<td>Design Proof of concept</td>
<td>24</td>
<td>+2</td>
<td>-3</td>
<td>-1</td>
<td></td>
<td>Demo for customer to reach consensus</td>
</tr>
<tr>
<td>3</td>
<td>Rework for design</td>
<td>8</td>
<td>+2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Low level design</td>
<td>16</td>
<td>+1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Development environment setup</td>
<td>24</td>
<td>+3</td>
<td></td>
<td>+2</td>
<td>+1</td>
<td>Purchase for specific software</td>
</tr>
<tr>
<td>6</td>
<td>Back end Implementation</td>
<td>40</td>
<td>+7</td>
<td></td>
<td>+3</td>
<td></td>
<td>Having the environment ready by IS team</td>
</tr>
<tr>
<td>7</td>
<td>Executing Testing</td>
<td>20</td>
<td>-2</td>
<td></td>
<td></td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td></td>
<td><strong>Total</strong></td>
<td><strong>148</strong></td>
<td><strong>166</strong></td>
<td></td>
<td><strong>171</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The discussion between the estimators involves each one estimator adjusting the estimation by using the delta column. The moderator writes the new estimation on the white board for each round. Notice that in each round the agreement for the estimates become closer, as seen in Figure 3. This is expected because the team members' points of view are being clarified.
After the estimation session, the project manager gathers all the output in a table and lists all estimates, best scenario and worst scenario. Big estimate differences are marked for further discussion. A final review meeting is called by the project manager to share with the estimators the summarized results and allow for more discussion or consensus.

**Parametric estimation**

Parametric estimation methods use the relationship between historical data and other parameters in order to reach an estimate through the mathematical formula. A simple example is: if the estimate to write one method is 2 hours, then it would require 100 hours to write 50 methods. The different kinds of parametric estimation are introduced in this section.

**Use Case Point (UCP)**

This technique was developed in 1993. It is based on use cases used in Unified Modeling Language (UML) to estimate the size of software. UCP evaluates many elements like use case actors, technical complexity, and environment complexity. It then puts them all together into a formula to calculate the overall sizing.

\[
UCP = (UUCW + UAW) \times TCF \times ECF
\]

Formula key:

- **UUCW**: Unadjusted Use Case Weight
- **UAW**: Unadjusted Actor Weight
- **TCF**: Technical Complexity Factor
- **ECF**: Environmental Complexity Factor

For more information, see [Use case point an estimation approach](#).

**Function Point Analysis (FPA)**

As a concept it is very similar to Use Case Point. The functional requirements are categorized into one of five categories: outputs, inquiries, inputs, internal files, and external interfaces. The
function is then assigned a number of function points based on its complexity. Using the following mathematical formula, the estimates are calculated.

\[ FP = UAF \times VAF \]

**Formula key:**
- **FP**: Function Point
- **UAF**: Unadjusted Function Point
- **VAF**: Value Adjustment Factor

For more information, see [Fundamentals of FPA](#).

**Three–Point Estimates**

This method tackles the uncertainty of estimation by using the Program Evaluation and Review Technique (PERT). The PERT technique calculates the estimate based on three estimates and analyzes the output by putting them into a mathematical formula.

\[ E = \frac{(O + 4M + P)}{6} \]

**Formula key:**
- **E**: Estimate
- **O**: An optimistic best case scenario
- **P**: A pessimistic worst case scenario
- **M**: Most likely scenario

For more information read about the [PERT Technique](#).

**Agile estimation methods**

One core distinction between estimation in traditional methods versus agile methods is in the use of the relative estimation concept. There is no absolute estimation in days or hours in agile estimation. Instead, story points are used. The rationale behind this is that many studies demonstrated how humans are more accurate when using comparisons in estimating. For example, you can carry two bags and estimate that one bag is heavier than the other one but you cannot be sure about how much each bag actually weighs. In the following sections, you'll learn some of the most popular estimation methods in agile estimation.

**Planning poker**

Planning poker is one of the most popular agile estimation techniques. It guarantees that all members actively participate through playing a game. The game starts with a set of cards being given to each member on the team. Each set of cards contain a distinct number following to a great extent Fibonacci sequence: 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, and so on. As seen in Figure 4, these numbers represent a relative evaluation for the story. Zero means the story is too trivial to be tracked; twenty means that the story needs to be broken into smaller ones. The rationale behind using Fibonacci is the intent to create uncertainty which will encourage discussion especially with large numbers. Now back to the game. Each member of the team starts estimating a certain
story, then each one reveals the card that contains that story estimate from his or her point of view. Members who estimated the highest and lowest values explain their argument. In return these explanations make the whole team rethink about the reasoning, then share experience and assumptions. The whole team re-evaluates the story again and again until they reach a consensus. For more information, read Planning poker.

**Figure 4. Sample of planning poker cards**

![Planning Poker Cards](image)

**T-shirt sizing**

In this method, a set of size cards are distributed. Each size card contains a specific size, which is: Extra small (XS); Small (S); Medium (M); Large (L) and Extra large (XL). The cards are distributed horizontally on a table. The team collaborates together in adding each story under the appropriate size category. After all of the user stories are sorted out, the team puts, for each size, an equivalent story point, as seen in Table 1. For example: XS is 1 story point, S is 2 story points, M is 4 story points, and so on. This method ensures that all stories are mapped to story points.

**Table 1. T-Shirt sizing mapped in story points**

<table>
<thead>
<tr>
<th>T-Shirt Size</th>
<th>Story point</th>
</tr>
</thead>
<tbody>
<tr>
<td>XS</td>
<td>1</td>
</tr>
<tr>
<td>S</td>
<td>2</td>
</tr>
<tr>
<td>M</td>
<td>4</td>
</tr>
<tr>
<td>L</td>
<td>6</td>
</tr>
<tr>
<td>XL</td>
<td>10</td>
</tr>
</tbody>
</table>

Similarly, other methods encourage using more creative and fun sizing methods. For example, using the weight of dogs where Chihuahua is the smallest and Great Dane is the largest weight. Another uses animal sizes where, for example, the mouse is the smallest size and the elephant is the largest.

**Estimating large number of stories**

What if you need to quickly estimate a large number of stories? In such a case, use relative mass estimation. It is a method that produces a rapid way to categorize and estimate large backlogs.
of stories. To use this method, you'll have a card for each story. With a set of story cards on the table, the moderator picks up one of the cards and asks the team, “Is this story considered large, medium or small?” Depending on the answer of the team, that card is placed in a specific spot on the table. If it is large then the card goes on the left side of the table. If it is small, then it goes on the right side of the table. If it is medium the card is placed in the center of the table. Going to the next card, the same question is asked and the new story is placed relative to the one before. This process continues until all of the stories are sorted on the table.

Another similar method that targets estimating large backlogs rapidly is called Wall estimation. This method uses a big empty wall to sort the stories. Stories at the top represent highest priority while stories at the bottom represent lower priority. This also works horizontally on the wall to determine story size. Stories on the left are the smallest while those on the right are the largest.

**Comparison between estimation in agile methods versus conventional methods**

The first questions that come to anyone's mind are, "Which method should I use?" and "Which method is the best?" Well, this is a debatable question and there is actually no right answer for that. There are many studies in this area that proves that it depends on many factors such as the type of the project. The factors are shown in Table 2.

**Table 2. A comparison matrix between agile methods and conventional methods**

<table>
<thead>
<tr>
<th></th>
<th>Agile methods</th>
<th>Conventional methods</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Project size</strong></td>
<td>Small and Medium projects</td>
<td>Large projects</td>
</tr>
<tr>
<td><strong>Software lifecycle</strong></td>
<td>Agile lifecycle. For example, Extreme Programming (XP) and scrum</td>
<td>Traditional lifecycle. For example, waterfall and spiral</td>
</tr>
<tr>
<td><strong>Estimation team size</strong></td>
<td>Usually from five to nine.</td>
<td>Could be less than five. In a subject matter expert case it could be one member.</td>
</tr>
<tr>
<td><strong>Team collaboration</strong></td>
<td>High involvement of all team members such as product owner, development team, testing team.</td>
<td>Not all methods encourage participation of all team members.</td>
</tr>
<tr>
<td><strong>Historical information</strong></td>
<td>Estimation methods do not mandates the existence of historical information.</td>
<td>Most of the methods depend on historical information.</td>
</tr>
<tr>
<td><strong>Time to estimate</strong></td>
<td>May take longer time since it is based on reaching consensus for the estimates.</td>
<td>May take less time compared to agile methods.</td>
</tr>
<tr>
<td><strong>Pointing system</strong></td>
<td>Complexity of the points. For example, the story point sizes.</td>
<td>Parametric points. For example, Functional Point Analysis and Use Case Points.</td>
</tr>
</tbody>
</table>

**Ingredients for better estimation**

Creating good estimations is essential in order to drive your project to success. Estimation is one of the most important elements for planning. The better the estimation, the better the control for the deliverables and less wasted cost. So how do you measure if your estimates are good? Compare
the variance between estimation and actual. By definition, a good estimate is within 25% of the actual result 75% of the time (Conte, Dunsmore, and Shen 1986).
Regardless of which estimation technique you choose to follow, there are some ingredients that definitely make it better.

- **Estimation techniques awareness.** You need to be familiar with all estimation methods in order to know which method is the best fit for your project.
- **Use small tasks units.** The more you break down your tasks into smaller pieces, the more insight you have about how long it will take to be completed. Typically a task's estimated completion time should be within two days of the actual completion time.
- **Team collaboration.** Involving your team in estimation erases any vagueness around the task. It helps in exchanging information and expertise thus reaching better estimation. It also increases the ownership of the estimate and the task.
- **Ask the right questions.** Every task contains ambiguity. You need to ask the right questions then try to find answers in order to eliminate this ambiguity. Put assumptions aside in order to create a common understanding for the task boundaries and risks.
- **Keep records of your actual effort.** Learning from previous estimates is definitely an added value. You need to keep records for the actual estimates for historical reference.

**Conclusion**

To become a better estimator you need to understand and explore different estimation techniques. It is important to know the options available and which one meets the needs of your project. This article provided a comprehensive list of estimation techniques under both agile and conventional methods. It gave you just enough information to understand the basics and get familiar with each technique. The matrix comparing agile versus conventional methods will help you decide which method to use. With the general recommendations and overview of techniques, you will be a better estimator.

**Acknowledgments**

I would like to thank Amr Yassin, Michael Schenk and Sally Fikry for reviewing and providing feedback on this article. It was definitely a pleasure working with them.
Is your project's best estimation method Agile or conventional?

Related topics

- *Software Cost Estimation with COCOMO II*, Barry Boehm et al. (Prentice Hall PTR, 2000)
- *Analysis of Effort Estimation Model in Traditional and Agile*, Manjula, R., and R. Thirumalai Selvi
- *A comparison between agile and traditional software development methodologies,* Awad, M. A., (University of Western Australia, 2005)
- *Agile estimating and planning*, Cohn, Mike. (Pearson Education, 2005)

© Copyright IBM Corporation 2015

Trademarks
(www.ibm.com/developerworks/ibm/trademarks/)