

Human factors and analytical models in software estimation: an integration perspective

Roberto Meli, DPO, Italy
roberto.meli@iol.it

Abstract

In the field of software estimation, supporters of a rather pragmatic and individualistic approach - based mainly on intuition and personal evidence – usually undertake endless discussions with supporters of a more general and theoretical approach - based on the collection and statistical analysis of actual data from past finished projects. The present paper supports the idea that the utilization of both the approaches represents a better answer to the ever challenging forecasting needs. The author starts from the definition of a classification framework for estimation methods and for the relative advantages and disadvantages. It then goes through the presentation of a real case of integration between a human approach, based on structured analogy, and an analytical model for the Function Point estimation of a software application. Finally, it ends with the clarification of the interaction of such tools as Early Function Point Analysis, team based estimation, case oriented training, Experience Data Bases and AHP analysis. The final goal which had been set up was the gain of the advantages of the two distinct approaches without suffering the correspondent disadvantages. This is what has been actually achieved in some real environments.

1. Introduction

In the field of software estimation, supporters of a rather pragmatic and individualistic approach - based mainly on intuition and personal evidence – usually undertake endless discussions with supporters of a more general and theoretical approach - based on the collection and statistical analysis of actual data from past finished projects.

Both the methods present advantages and disadvantages. In the real world of software development, project managers have generally preferred the first approach whereas academics and consultants have usually pointed out the superiority of using corporate or market models derived and proved on some benchmarking available data bases.

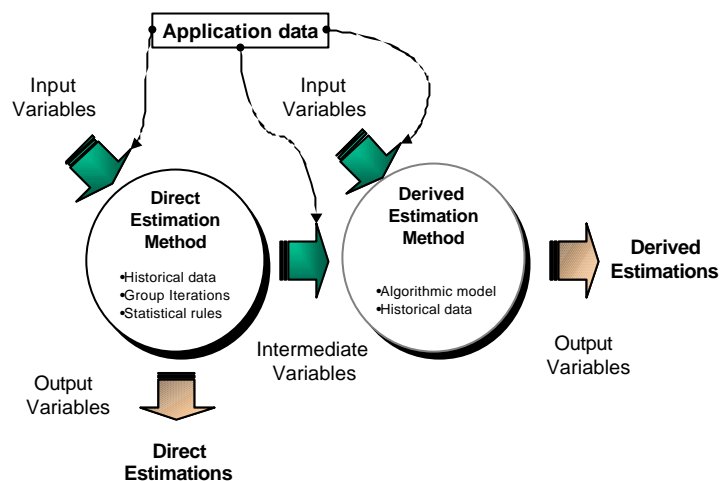
Quite recently we have perceived signals of a stronger convergence between the two approaches since many project managers have decided to learn and experiment some kind of formal models, less subjective and more shareable, whereas in the theoretical area many important “opinion makers” have started to criticize the presumed superiority of empirical models if compared to the use of the human capabilities on their own.

Times have come when an attempt in the integration of the two distinct scenarios should be carried on. The present paper starts from the definition of a classification framework for estimation methods and for the relative advantages and disadvantages. It then goes through the presentation of a real case of integration between a human approach, based on structured analogy, and an analytical model for the Function Point estimation of a software application. Finally, it ends with the clarification of the interaction of such tools as Early Function Point Analysis, team based estimation, case oriented training, Experience Data Bases and AHP analysis. The final goal which had been set

up was the gain of the advantages of the two distinct approaches without suffering the correspondent disadvantages. This is what has been actually achieved in some real environments.

2. Direct and Derived Estimation

An estimation method may be represented as an input-processing-output system where the input variables are, essentially, information on the software application to be “sized” or “estimated”, whereas the output variables may be the functional size expressed in FP or the total cost to be expended or the duration of a specific project phase etc. Both the input and the intermediate variables are measured through the use of consistent metrics. The estimation methods may be classified in two main categories: Direct Estimation and Derived Estimation. The former groups all the methods based on analogical reasoning and intuition where the result is achieved directly without the formalization of a step by step analytical process. The latter groups all the well defined algorithmic or structured methods based on theoretical or statistical models. We may classify all the existing estimation methods into the preceding categories. Table 1 reports some generic considerations about these categories.



2.1. Direct estimation methods

These are also known as Expert Opinion Methods. They involve consulting one or more experts, who provide a direct guess of the estimation required, based mainly on past personal experience, intuition and unconscious processes. Quite frequently an expert knows what is the right value but is unable to explain why it is so. The mental process is similar to an iceberg, the biggest part of the reasoning being under the surface of the conscience. A direct estimation may be improved by the use of some techniques like Delphi method (or its Shang variant) or the Analogy. The former is an iterative group technique that allows the participants to improve the individual estimates through the cooperation of different points of view in the estimation effort. The latter involves reasoning by analogy with one or more completed software applications to relate their actual values to an estimate of a similar new application. Analogy can be made between high or low level components of the software application. The Expert Opinion approach to estimation may results in very accurate estimates, although it is entirely dependent on the experience and the psychological characteristics of the expert(s). Sometimes, in case of many experts collaborating to the same task, it may be difficult for the estimates to converge to a unique value. The estimate itself may be influenced by subjective factors, as personal relationships, contractual aspects, and so on.

2.1.1. Delphi or Shang techniques

These are among the most commonly used procedures, by which individual predictions are combined. In brief, they provide iterated cycles of anonymous estimations by each expert, until the

estimates converge to an acceptable range. The result is a group estimate arrived at by consensus. The group estimate is typically a better overall estimate than any individual prediction.

2.1.2. Three-point estimation technique

This is a technique to improve direct estimation, when more values are provided by estimators. Given the Minimum, the Most Likely, and the Maximum Value for the size, the estimate is:

$$\text{Est.Value} = (\text{Min} + 4 \times \text{MostLikely} + \text{Max}) / 6$$

with standard deviation:

$$s = (\text{Max} - \text{Min}) / 6$$

2.1.3. Simple analogy method

The most common way to apply a Simple Analogy Estimation for software is to look for a known system in a historical database, that is "similar" (in analogical sense) to the application under estimation. The found implemented system provides a quick estimate of the new project value. Further investigation likely leads to Structured Analogy (see next section)

2.1.4. Structured analogy methods

These are more formal approaches than the Simple Analogy Method. The estimator compares the proposed application to one or more existing applications: s/he typically identifies the type of application, establish an initial prediction, and then refine the prediction within the original range. Passing from a Simple Analogy to a Structured Analogy, differences and similarities are identified and used explicitly in a mathematical way to adjust the estimate. A concept of "distance" among systems may be defined and used to prioritise the choices.

2.2. Derived estimation methods

These methods, also known as Algorithmic Model Methods, provide one or more transformation algorithms which produce a software estimate as a function of a number of variables which relate to some software attributes. Generally, these methods are related to a decomposition process. By decomposing an application into its major functions, estimation can be performed in a stepwise fashion. The difference between a direct estimation and a derived estimation is mainly that in the second case the measurements or the estimations are not made directly on the final values but on different software attributes which are supposed to be correlated with the final values themselves.

There are many Algorithmic Model Methods for estimating FP size, effort, duration, staff etc. mostly because of statistical researches and benchmarking results. Not every method is cited in technical literature, but many of them are widely known in practice.

3. A comparison between direct and derived methods

Direct estimation is an activity characterized by a high level of psychological aspects that may influence the quality of the final results. It is quite difficult to assess the methods themselves because it is almost impossible to distinguish the method from the people involved. When we say that an expert judgment technique is performing very well, we are actually saying that the people involved in the experiment are performing well. There is no guarantee that a different group, even with the same kind of experience, would perform at the same level or even that the same group will perform at the same way in the future.

On the other side an explicit and documented "pure" derived technique may be separated from the people that are actually using it. Its performance is not particularly influenced by the different

individuals who will eventually use it. The algorithmic model needs to be maintained by a centralized team.

For this reason is not possible to assess, in a definitive way, which method is better than the other: they both stand as usable workbenches and the choice should be based on the strengths and weaknesses that in a particular situation are more important than others.

If we look at the psychological component of a direct estimation we may consider that there are lots of sources of possible biases in the estimation results. One of them is the psychological pressure under which an estimation is usually required and must be prepared. This situation arises when the estimation process is linked to a negotiation process. When we know what are the expectations regarding an estimate – cost, effort and duration, for example - we are heavily influenced by that expectation. An expected value for the estimation represents a strong anchor for the final result of the estimation process.

Method Category	Strengths	Weaknesses
Direct Guess (individual or Delphi)	<p>Suitable for atypical projects</p> <p>May result in very accurate and shared estimates</p> <p>It is based on the past but may be consciously oriented to the future</p>	<p>Quality of estimation is usually no better than “quality” of estimators</p> <p>It is difficult to document</p> <p>It may be difficult to justify the results</p> <p>It is impossible to share the interior knowledge with others</p> <p>It is heavily dependent on the expertise of the estimator(s)</p> <p>It is based on rare and costly human resources</p> <p>Estimation capability (and resources) may be quickly lost and slowly gained</p> <p>Estimations are influenced by psychological aspects</p>
Simple & Structured Analogy	<p>Based on representative experience</p> <p>Tends to consider all software factors</p> <p>It is possible to document some of the results and decisions taken</p> <p>It is quite simple to teach the technique</p> <p>It is based on the past but may be consciously oriented to the future</p>	<p>Quality of estimation is usually no better than “quality” of estimators</p> <p>It depends on the availability and quality of (local) historical data</p> <p>It depends on the expertise of the estimator(s)</p> <p>It requires the definition of an explicit model suited for the comparisons</p>

Algorithmic Model	<p>It is based on objective, repeatable, analyzable formulae</p> <p>It may be calibrated to historical data</p> <p>It may be easily taught to and used by beginners</p> <p>It is public and standard</p> <p>It implies a methodical identification and evaluation of influencing factors</p> <p>It is usually based on objective metrics</p> <p>Its use leads to a quick negotiation on the resources</p>	<p>It is based on and oriented to the past</p> <p>It is not suitable for atypical projects</p> <p>Disregards components not explicitly included in the model</p> <p>There is the risk of avoiding responsibility in the estimation process (“this is the tool’s job”)</p>
-------------------	---	---

Table 1. General comparison of software estimation methods.

4. Case Study: Function Point Estimation

To show a case study for the integration between the direct and derived estimation approaches I have chosen the Function Point sizing process.

One important advantage to be taken into account in considering a Function Point measurement is that it is possible to determine its value at an early stage of a software project, i.e. when detailed functional users' requirements of a business application are evident and available. Unfortunately, for the purposes of Project Management, this is not early enough, since the level of detail needed to apply IFPUG standard counting rules implies that a large portion of the project has already been seen through (Functional Specification accounts for 15 to 40 % of the total work effort). Otherwise, it would be impossible to identify External Inputs, External Outputs, External Inquiries, Internal Logical Files, and External Interface Files without running into significant evaluation errors. In fact, if we consider the importance of estimation with respect to the management of the project, we end up with a curious and perhaps paradoxical phenomenon: measurement is very useful when we do not have enough elements to obtain it (Feasibility Study), but when we can identify it with absolute accuracy (just before the final product is ready) it is no longer necessary (at least for the purposes of predicting effort).

This means that we need to have at our disposal an FP measurement value estimation that can already be used after a carefully-produced Feasibility Study. This document roughly define what is going to be subjected in the end to a more in-depth detailed analysis, which can in turn make real FP counting possible. In fact, experts are increasingly using the term "counting" with regard to using IFPUG standard rules for identifying FP values, and "estimation" for all alternative techniques for forecasting the same FP values.

5. Early Function Point Analysis

Early FP technique – presented since 1997 and largely used in several operational environments - combines different estimating approaches in order to provide better estimates of a software system size. This method makes use of both analogical and analytical classification of functionalities. In addition, it lets the estimator use different levels of detail for different branches of the system (multilevel approach): the overall global uncertainty level in the estimate (which is a range, i.e. a set of minimum, more likely, and maximum values) is the weighted sum of the individual components' uncertainty levels. The multilevel approach makes it possible to exploit all the knowledge the

estimator has on a particular branch of the system being estimated, without asking questions difficult to answer in an early stage or, on the contrary, leaving some detailed information on a portion of the system unused. Finally, the Early FP method provides its estimates through analytically and statistically validated tables of values.

The key factors in an Early FP estimate are: Macrofunctions, Functions, Microfunctions, Functional Primitives and Logical Data Groups. In conceptual terms, Functional Primitives correspond to the elementary processes of the standard FP Analysis, i.e. EI, EO and EQ, while Macrofunctions, Functions, and Microfunctions are different aggregation of more than one Functional Primitive at different detail level. Logical Data Groups correspond to standard Logical Files, but without any differentiation between "external" and "internal" data, and with some levels suitable for aggregation of more than one logical file.

Each "object" is assigned a set of FP values (min, more likely, max) based on analytical tables, then the values are summed up to provide the UFP estimate (these assignments are subject to improvement on the basis of statistical analysis for actual projects, as from the ISBSG research); as for the most of the other estimation methods, the Adjustment Factor is determined similarly to the standard IFPUG method. Estimates provided by this method may be denoted as "detailed", "intermediate" or "summary", depending of the detail level chosen (or forced) for the early classification of functionalities. The reader may refer to the cited papers in the reference for exact definition, guidelines and numerical indicators of this method.

The reliability of the EFP method is directly proportional to validity of the analytical tables on which is based and on the estimator's capability to "recognize" the components of the system as part of one of the proposed classes. This skill may sharpen through practice by comparing the different counts obtained using standard and Early FP rules. However, the Early FP technique has proved to be quite effective, providing a response within $\pm 10\%$ of the real FP value in most real cases, while the savings in time (and costs) can be between 50% and 90%, with respect to corresponding standard counts.

6. An integration perspective

The Early Function Point Technique is a good example of how it is possible to practically integrate the analytical and the analogical approach.

The first one is based on the capability of the estimator to model and make appropriate logical partitions of a given software object. The relationships among the different parts are then identified and appropriately considered. In the conceptual framework adopted, every typology of objects (functionality or data) is linked to the others typology by mean of an analytical table which give precise rules to be used in the classification phase of the technique. Once identified an object as belonging to a particular typology, its FP contribution is automatically assigned.

The second approach is based on the estimator's capability to "recognize" new software objects as similar to other software objects already known and actually classified. The analogy may be developed with respect to different target objects: a general archetypal model or a concrete software item. In the first case the estimator is required to compare the new piece of software (or its requirements) to a set of general standard abstract items like Macrofunctions, Functions, Microfunctions etc. based on the estimated composition in terms of subcomponents. In the second case the estimator is required to compare the new piece of software (or its requirements) to a set of concrete and existent software items already classified and measured. The comparison may permit to identify the software object as belonging to the same or a different class from the one represented by the known item.

In both cases the new object is identified, classified and weighted. Looking at the similarities and the differences between the “known” and the “unknown”, it is possible to derive some properties of the new object starting from the actual properties of similar existing objects.

While the analytical process activates the left part of the brain – logic, rationality, symbolic digital manipulation - the analogical process stimulates the right part of the brain - the associative, intuitive, creative and lateral thinking. An equilibrated mixture of the two brain components may guarantee more consistent and precise results for the estimation needs.

7. How to improve analogical attitudes

This section will introduce some means to improve the quality of the analogical component of the estimation.

7.1. Team based estimation

A team based estimation is an experience of cooperative work where the final result is achieved by the interaction among differently experienced participants. Each of them is representative of a different background, point of view and interests. The team may be based on a composition of business application-experts and estimation-experts who will positively interact with each other to gain a common view of the estimation exercise. Every occurrence of a team based estimation is an opportunity to train junior estimator in the application of the methods and models adopted.

7.2. Case oriented training

This is a more traditional way to improve people’s capability. It is centered on the presentation and analysis of real, concrete case studies taken from the common organizational and technical environment. In this way junior or even senior estimators may develop the appropriate skill in a controlled and experimental environment.

7.3. Experience Data Base

An Experience Data Base (EDB) is an automated catalog of software items considered at all the different levels of the application hierarchy (Macrofunction, Function, etc.). Every item is classified and weighted in terms of subcomponents and Function Points. The useful attributes that may be represented for each record are: code, name, content description, keywords, EFPA classification typology, technical environment, language used, link to the appropriate requirements, etc.. Using different extraction criteria and filtering information is possible to select a set of software items similar to the new one with their correspondent FP value. We may then use the single values or averaging them till we are confident in the classification result. EDB may be based on local or market data, domain specific or general items, etc.

7.4. AHP technique

This technique is particularly promising as a methodological support to the estimator in the application of analogy. It is based on a pair wise comparison of a mixture of known and unknown items in such a way that a final ordering of the objects compared may be achieved. In this way it is possible to assess the unknown objects in terms of their classification or directly in terms of FPs.

8. Conclusions

People and methods are tremendously important to gain a high quality in the estimation results. Instead of conceiving them as alternative means we should consider them as synergic resources. This

may happen if we are supported by a conceptual framework of integration and a set of operational rules to follow. Early Function Point technique has proven to be a good example of cooperation between them in conceptual and practical aspects.

9. References

Some of the papers referenced may be downloaded from:

<http://web.tin.it/dpo> or <http://www.dpo.it>

- [1] Briand, L. C., El Emam, K., Surmann, D., and Wieczorek, I., "An assessment and Comparison of Common Software Estimation Modeling Techniques", International Software Engineering Research Network Technical Report, ISERN-98-27, 1998.
- [2] Hughes, R. T., "Expert judgement as an estimating method", Information and Software Technology, 38: 1996, pp. 67-75.
- [3] Kolodner, J. L., "Case-based reasoning", Morgan Kaufmann, San Mateo, California, USA, 1993.
- [4] R. Meli, "Early Function Points: a new estimation method for software projects", ESCOM 97, Maggio 1997, Berlino
- [5] R. Meli, "Early and Extended Function Point: a new method for Function Points Estimation", IFPUG Fall Conference, 15-19 settembre 1997, Scottsdale, Arizona USA
- [6] R. Meli, "Risks, requirements and estimation of a software project", ESCOM-SCOPE 99, Herstmonceux Castle, East Sussex, England, April 27-29, 1999
- [7] R. Meli, L. Santillo, "Function point estimation methods: a comparative overview", FESMA 98 - The European Software Measurement Conference – Amsterdam, October 6-8, 1999
- [8] Mukhopadhyay, T., Vicinanza, S., and Prietula, M. J., "Estimating the feasibility of a case-based reasoning model for software effort estimation", MIS Quarterly, 16(2), 1992, pp. 155-171.
- [9] L. Santillo, R. Meli, "Early Function Points: some practical experiences of use", ESCOM-ENCRESS 98, Maggio 1998, Roma
- [10] Shepperd, M. and Schofield, C., "Estimating software project effort using analogies", IEEE Transactions on Software Engineering, 23(12), 1997, pp. 736-743.
- [11] Shepperd, M., Schofield, C., and Kitchenham, B., "Effort estimation using analogy", Proceedings of the 18th International Conference on Software Engineering, Berlin, Germany, 1996.
- [12] Stensrud, E. and Myrtveit, I., "Human performance estimating with analogy and regression models: An empirical validation", Proceedings of the 5 th International Symposium on Software Metrics, Bethesda, Maryland, USA, 1998.
- [13] Vicinanza, S.S, Mukhopadhyay, T. and Prietula, M.J., "Software effort estimation: An exploratory study of expert performance", Information Systems Research, 2(4), 1991, pp. 243-262.
- [14] Walkerden, F. and Jeffery, R., "Software cost estimation: A review of models, process and practice", Advances in Computers, 44, 1997, pp. 59-125.
- [15] Walkerden, F. and Jeffery, R., "An empirical study of analogy-based software effort estimation", Centre for Advanced Empirical Software Research, Technical Report 98/8, 1998.