A STRATEGY FOR A CREDIBLE & AUDITABLE ESTIMATION PROCESS USING THE ISBSG INTERNATIONAL DATA REPOSITORY

Alain Abran, Reiner Dumke, Jean-Marc Desharnais, Iphigénie Ndiaye and Christian KOLBE

August 3, 2002

Accepted for publication at IWSM 2002 – Magdeburg (Germany)

1- Introduction: the business context

The software estimation process must provide credible input for business decision-making. Most of the time, business managers must rely on incomplete information to make decisions: there is almost always some information lacking or too expensive to gather within the time frame of the decision--making process. Within their field of expertise, decision-makers can make valuable expert judgments on the missing or incomplete information. However, many decision-makers have not mastered the information technology domain and need expert support to fill in the gaps in their knowledge of the subject.

For this reason, the software estimation process must provide decision-makers not only with estimates (the “the numbers”), but also information on the quality and confidence level of that estimate. The key assumptions and the key uncertainties inherent in the estimation process must be conveyed to business managers to help them make informed business decisions on the basis of the estimates provided to them. For example, decision-makers should have a feel for the quality and accuracy of the inputs, as well as for the estimation models used for deriving the estimates.

The estimation process must be credible from a business perspective, and the outcomes of the estimation process must include statements on the credibility of its various components. Furthermore, since major investment decisions are made based on these estimates, the full estimation process should be auditable.

This paper highlights the key elements that make an estimation process both credible and auditable. This includes a discussion on the quality of the input measures (products, processes and resources), the reliability of the productivity models built into an estimation process, the other inputs to the estimation process (key assumptions, constraints and expert judgments) and the type of decisions that should be taken on the basis of the confidence level of the outcomes of the estimation process. This is illustrated with examples from the multi-organizational project repository of the International Software Benchmarking Standards Group – ISBSG.

2- The estimation process

A high-level view of the estimation process is presented in Figure 1, complete with inputs and outputs. This process can be further decomposed into two major sub-processes: a productivity simulation sub-process and an adjustment sub-process.

The productivity simulation sub-process includes a productivity simulation model which takes as input Measures of resources, processes and products, and which provides as output Simulation Results. The adjustment sub-process takes the simulation results as input, together with information on uncertainty factors and risk assessment results, and provides as output the outcomes of the full estimation process.

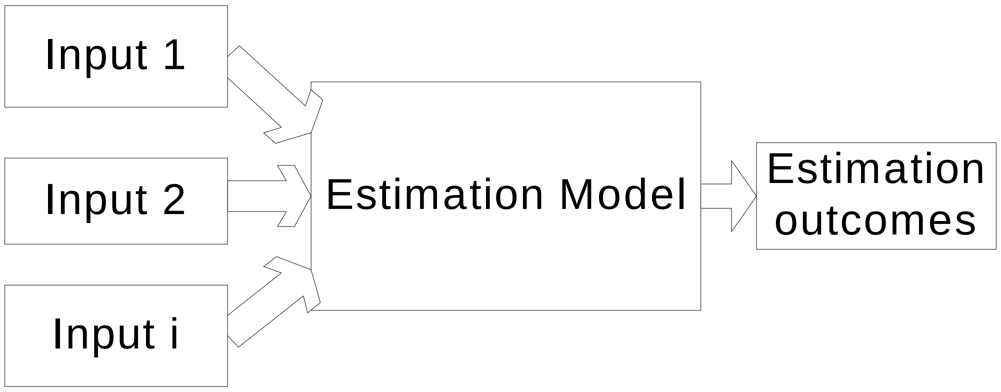


Figure 1: High-level model of an estimation process.

The credibility of the whole estimation process must be based on the credibility of each of its sub-processes and their components. The final result of the estimation process cannot be more reliable than the reliability of each component, and is as weak as its weakest component. Therefore, the quality of each component must be made known to the decision-makers for prudent use of the outcomes of an estimation process. Decisions should not be based only on the results of a simulation model.

3- Simulation sub-process

3.1 Quality of the input measures

Measures are required as input to the simulation sub-process. This includes measures of the resources, of the processes and of the products [FEN 92] (Figure 2).

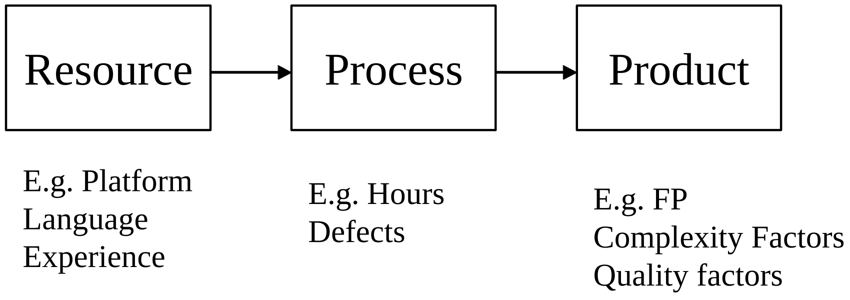


Figure 2: Inputs to the productivity simulation model

Some measures might be quantitative, such as those of functional size [ALB84] [ABR01], while others are descriptive (the project development mode -- Prototyping, JAD and RAD -- qualitative measures, the values of which are on a nominal scale type).

The degree of accuracy of these measures must also be known and should be documented for each and all types of measures. In Figure 3, this document is referred to as an Audit Report.

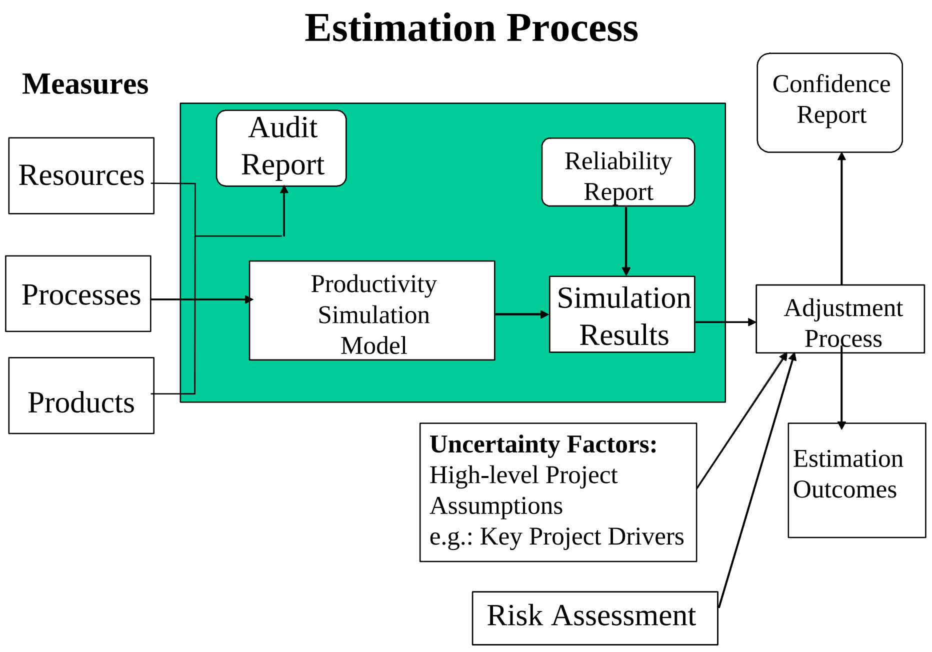


Figure 3: Key components of an estimation process

3.2 Quality of the productivity simulation model(s)

Furthermore, the simulation model (or models, when more than one is used) should be calibrated to the business Group that has to make a business decision and commit resources. In other fields of engineering and business, information on industry averages might be used for high-level planning of resources; these industry averages can be used for performance comparison, but should not be used as a basis for commitment in terms of resources and schedules for individual products. Business would much prefer locally tailored models derived from their specific environment.

Therefore, in the situation of an externally supplied productivity simulation model, there should be both a **calibration** and a **validation** (Figure 4) of the simulation model with respect to the corporate historical database [DES93].

This implies that there should be documented evidence of the effectiveness and reliability of the productivity simulation model, based on both the data set that contributed to the design of the **productivity simulation model** (calibration) and additional data to validate the **quality of the model** (validation). The explanatory power of the model, in the absence of uncertainty, should be documented and credible. This is referred to in Figure 3 as the Reliability Report.

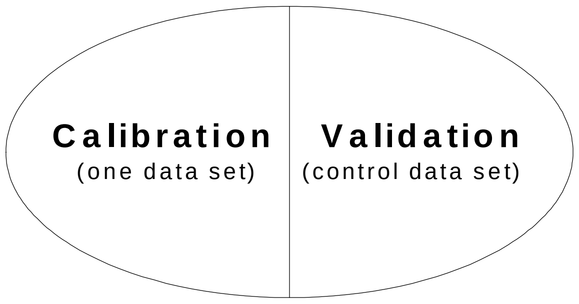


Figure 4: Calibration and Validation

4- Simulation using the ISBSG international repository

The construction of an estimation model, whatever estimation method is used, usually requires a set of completed projects from which an estimation model is derived and which is used thereafter as the basis for the estimation of future projects. However, in most organizations there is often no structured set of historical data about past projects, which explains their inability to build their own models based on the characteristics of past projects. Organizations without such historical data traditionally had four alternatives when they wanted to improve their estimation process:

1- Collect data from past projects and build estimation models using their own historical data sets – this is particularly useful if the projects to be estimated have a high degree of similarity with past projects. This, of course, requires the availability of high quality information about past projects documented in a similar and structured way.

2- Take the time required to collect project information from current projects, and wait until completion of enough projects to build sufficiently reliable estimation models with a reasonable sample size. This, of course, requires time, and most often managers cannot afford to wait.

3- If their upcoming projects bear little similarity to their own past projects, an alternative is to access data repositories containing projects similar to the ones they are embarking on and derive estimation models from these. A key difficulty, until fairly recently, has been the lack of market availability of repositories of a projected project type.

4- Purchase a commercial estimation tool from a vendor claiming that the tool basis includes historical projects of the same type as their upcoming projects. This is a quick solution, but often an expensive one.

While alternatives 1 and 2 are under the total control of an organization, alternatives 3 and 4 depend on an outside party. Until fairly recently, for those organizations without their own historical data sets for building estimation models themselves, and who could not afford the long lead time to do so, only alternative 4 was widely available, with all the associated constraints of not knowing either the basis of the estimation or the quality of the estimates derived from sources not available for independent scrutiny. In this paper we refer to these commercial tools as black-box estimation tools.

In the mid-1990s, various national software measurement associations got together to address the limitations of alternatives 1 to 4, specifically to overcome the problems of both the availability and the transparency of data for estimation and benchmarking purposes and to offer the software community a more comprehensive alternative in the form of a publicly available multi-organizational data repository. This led to the formation of the *International Software Benchmarking Standards Group* [ISBSG] for the development and management of a multi-organizational repository of software project data. This repository is now available to organizations, for a minimal fee, and any organization can use it for estimation purposes. Such a repository can also, of course, be used for the analysis of software estimation tools already on the market [ABR02].

Many software engineering data sets are heterogeneous, have wedge-shaped distributions [ABR94] [ABR96] [KIT84] and can, of course, have outliers which have an impact on the construction of the models and on their quality. Therefore, each sub-sample was analyzed for the presence of outliers, and well as for visually recognizable point patterns which could provide an indication that a single simple linear representation would not be a good representation of the data set. For instance, a set of projects within one interval of functional size might have one behavior with respect to effort, and another size interval could have a different behavior. If such recognizable patterns were identified, then the sample would be sub-divided into two smaller samples -- if there were enough data points, of course. Both the samples with outliers and those without them were analyzed. This research methodology was used on each sample identified by a programming language.

Two prototypes were built to facilitate the development of estimation (or simulation) models with the ISBSG repository: the first with Visual Basic on a stand-alone PC [STR98], and the second using Web technology to access both the data and the software from anywhere with an Internet connection [KOL01].

Table 1 [NDI01] presents the set of linear regression models derived directly from this ISBSG repository: 12 samples including outliers on the left-hand side, and 18 samples excluding outliers on the right-hand side, and subdivided by size intervals where warranted by graphical analysis. In Table 1, the coefficient of regression (R^2) is presented for each model. Analysis of the R^2 of the estimation models for the samples (with and without outliers) illustrates how the outliers can impact the general behaviour of the size-effort relationship. For example, the R^2 of 0,62 for the C++ sample with outliers (left-hand side, Table 1) could lead us to believe that the size-effort relationship is strong; however, a few outliers have a major undue influence on the regression modelling, and their exclusion leads to models where there is almost no size-effort relationship, with an R^2 of either 0,11 or 0,06 for the sets of intervals and quite distinct models. By contrast, outliers can hide a significant size-effort relationship for the majority of the data points: for example, the PL1 sample with outliers has a very low R^2 of 0,23. However, when some are excluded from the sample, the size-effort relationship has an R^2 of 0,64 for the 80 to 450 FP interval. The subdivision of samples (with and without outliers) gives different linear models, as well as different strengths in the size-effort relationship: with

COBOL II, for instance, the R^2 is 0,45 for the 80 to 180 FP size interval, while it is 0,61 for the 180 to 500 FP interval, thereby giving managers more fine-tuned, and more representative, models of the information contained in the data sets.

From the right-hand side of Table 1, the samples can be classified into three groups with respect to their size-effort relationship (excluding the sub-samples with too few data points for the large upper intervals):

A- languages with an $R^2 < 0,29$, representing a weak relationship between functional size and effort: Access, C, C++, Powerbuilder and SQL;

B- languages with an $R^2 > 0,39$, representing a strong relationship between functional size and effort: Cobol, Cobol II, Natural, Oracle, PL1, Telon and Visual Basic;

C- language samples for which there are either not enough data points in the sample ($N<10$) or for which the interval is much too large for the number of data points, thereby making the samples very sensitive to the size of a candidate outlier within this size range (upper intervals for C++, Cobol II, Natural, PL1 and SQL). This illustrates that regression models under these conditions are not easy to interpret empirically. This can be illustrated with the following example:

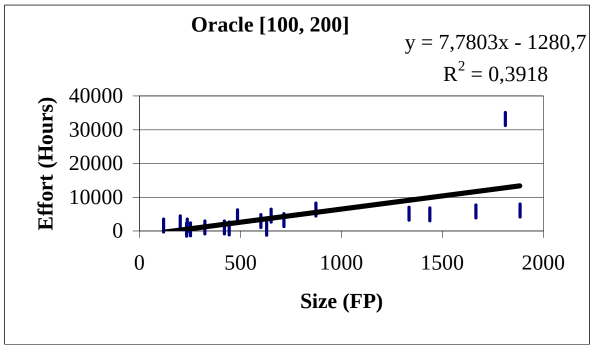
> Within the higher interval, for instance for PL1 with 5 data points between 450 FP and 2500 FP, the slope is reasonable -- the fixed cost is negative (but minimal), but the R^2 of 0,86 can only be indicative and tentative, since the sample range is much too sparsely populated between 450 and 2500 FP for the results to be otherwise.

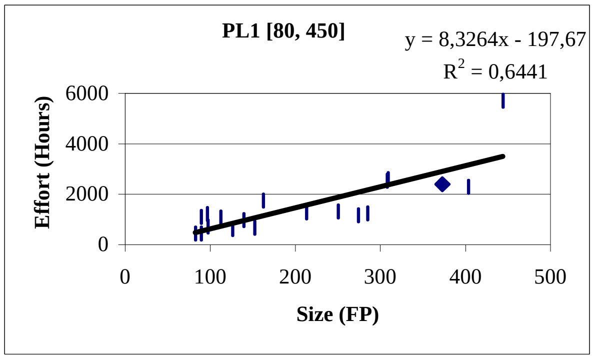
Some estimation models built directly from the ISBSG are presented graphically below for: Oracle (100, 2000), PL1 (80, 450), Powerbuilder and Telon (Figures 5a to 5d).

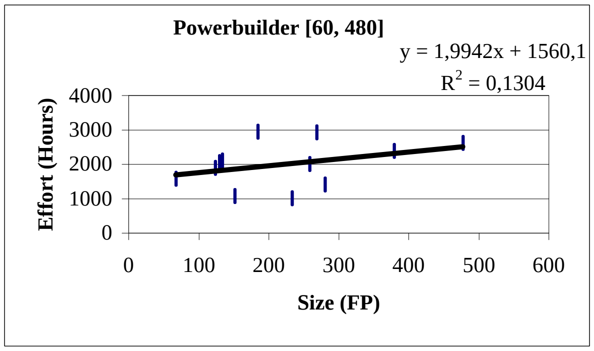
It can also be observed that the samples for each programming language have very distinct estimation models, with differences in both the coefficients and the fixed (or error) terms, thereby indicating a different production function for each, and, of course, dispersion across this production function. More specifically, for the samples with a large number of data points, the coefficients (or slope) will vary from 0,3 for Access to 16,4 for Cobol II (80-180). There are also, of course, variations in their constants or error terms. This means that when the programming languages are known, such information should be used for deriving the relevant production function to be used in an estimation context, rather than using a single model, or a two-level model by generic environment (mainframe, PC, etc.).

Table 1: Direct ISBSG regression models (with and without outliers)

	Samples with outliers				**Samples without outliers, and within size intervals**			
Language	**N= 377**	**Size Interval**	**Linear Regression Equation (where x = FP units)**	**R²**	**N= 302**	**Functional Size Interval**	**Linear Regression Equation (where x = FP units)**	**R²**
Access	17	200-1500	Y=-0,10x+840,8	0,01	11	200-800	Y= 0,30x + 623,5	0,19
C	15	40-2500	Y= 4,05x + 4288	0,19	9	200-800	Y= 2,34x + 2951,7	0,29
C++	21	70-1500	Y=13,43x + 1346,4	0,62	12	70-500	Y= 11,53x + 1197,1	0,11
					5	750-1250	Y= -6,57x + 23003	0,06
Cobol	106	0-5000	Y=4,94x + 5269,3	0,36	60	60-400	Y= 10,83x + 299,1	0,44
					32	401-3500	Y= 12,32x – 14,1	0,64
Cobol II	21	80-2000	Y= 27,80x – 3593	0,96	9	80-180	Y= 16,39x – 92,3	0,45
					6	180-500	Y= 26,73x – 3340,8	0,61
Natural	41	20-3500	Y=10,05x – 648,9	0,85	30	20-620	Y= 6,13x + 264,9	0,47
					9	620-3500	Y= 10,53x – 1404,9	0,74
Oracle	26	110-4300	Y=6,20x + 509,7	0,42	19	100-2000	Y= 7,78x – 1280,7	0,39
PL/1	29	80-2600	Y=11,06x + 46,7	0,23	19	80-450	Y= 8,32x – 197,6	0,64
					5	451-2550	Y= 5,49x – 65,3	0,86
Power builder	18	60-900	Y=12,99x – 380,2	0,66	12	60-400	Y= 1,99x + 1560,1	0,13
SQL	20	280-4400	Y=7,57x – 271,2	0,60	11	280-800	Y= -0,42x + 3831,2	0,00
					8	801-4500	Y= 9,23x – 6064,3	0,67
Telon	23	70-1100	Y=7,42x + 650,9	0,85	18	70-650	Y= 5,50x + 1046,1	0,75
Visual Basic	34	30-2300	Y=9,69x + 661,5	0,56	24	30-600	Y= 7,24x + 52,4	0,46







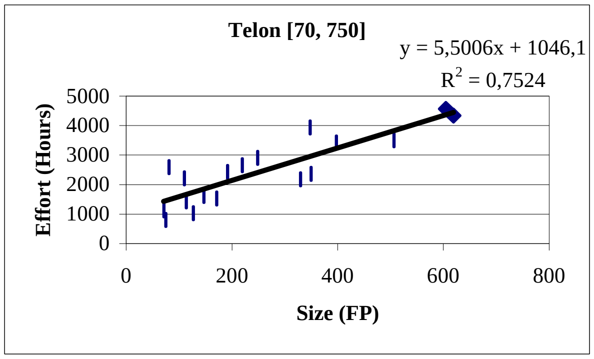


Figure 5 (a to d): directly derived ISBSG estimation models

5- Adjustment sub-process

5.1 Other variables

Once the output of the productivity simulation model has been derived, then additional inputs should be taken into account in order to make adjustments to the simulation results – Figure 3. Such additional inputs can be:

- the degree of completeness of the inputs (usually based on the project life-cycle phases)
- the project key assumptions, such as:
    - the key costs drivers;
    - the key opportunity drivers;
    - the key constraint drivers.
- the project risks (identified in a risk assessment study);
- the consequences of the iterative nature of estimates throughout the life-cycle of project phases

5.2 Scope management

Industry experience has shown that project scope will change throughout the project life-cycle, and this must be managed. A value-added estimation process will provide the basis for managing changes in project scope. On the one hand, measures of product functional size will provide the basis for scope creep identification and sizing later on in the project life-cycle. On the other hand, the projected unit cost derived from the results of the simulation model (total project cost divided by size) can provide a technique for negotiation of project price increase/reduction. This unit cost basis should be agreed upon prior to moving to the next project phase.

5.3 Continuous improvement to the estimation process

Upon project completion, comparison of “actual” versus “estimated” costs (not only in terms of Effort, but of all other Product measures) should be looked at to provide feedback on the quality of the estimation process; this could provide valuable information for improving any one of the estimation process steps. Actual data should be fed back into the simulation model for further refinement.

5.4 Guarantees of accuracy

Within the current state of the art in software estimation, it is not feasible to guarantee the accuracy of the outcomes of an estimation process for individual projects, or even for groups of projects.

If an estimator were to provide such guarantees, the expectations raised at the customer site should be managed and the estimator should be held accountable for these guarantees. Such a guaranteed commitment can only be based on actual performance of the projects when compared to estimates.

To facilitate follow-up of the guarantees, the estimates should be stated not only on full project estimates, but also on unit cost (for example, estimated unit cost per Function

Point per project estimated). This would allow for validation and control of the estimation process, even for a minimal or major increase or reduction of project scope.

Since the validation process cannot occur before project completion, the estimator's contract should include penalty clauses for not meeting the guarantees, and the estimator should be insured through commercial, bound clauses that could be exercised up until project completion.

6- Conclusion: The business context

In project estimation, the business context is much larger and is not restricted to either a single project or to the software project perspective.

It cannot be expected that the outcome of the previous software estimation sub-process be the unique contributor to the business decision-making process.

From a project perspective, there should be a business/market estimation report made in parallel to the software estimation process, and it should be redone for each estimation iteration throughout the project life-cycle.

From a business perspective, the portfolio of all projects must also be taken into account: prior to making a decision on a specific project, business managers must consider estimated costs, estimated benefits and estimated risks of all projects, and individual project decisions must be made in the context of a strategy that optimizes the corporate outcome while minimizing the risks across all projects.

Business objectives, practices and policies must be taken into account as well when making business decisions. This might reveal that the software cost estimate is not precisely mapped to the business estimates.

For example, in a high-risk project situation with a potential for very high benefits, decision-makers may want to add contingency funding provisions to ensure project completion; even in an over-budget project situation; contingency funding might not be communicated to project management.

Similarly, to win market share, a business decision may be made to complete a project at a loss. In such situations, the project estimates are not lowered, but the project loss situation is recognized. Unfortunately, such situations are not often formally identified in the software estimation process, whereas the segregation of the business estimation and the technical estimation is not a recognized, documented practice: this leads to a situation where perfectly valid technical estimates may too often be significantly lowered for the purposes of business strategy, thereby leading to lower, as well as unrealistic and unachievable, project estimates.

From a longer term corporate perspective, the two types of estimates should be identified and managed separately. This will clarify the decision-making responsibilities and, over time, will facilitate improvements to both types of estimation process.

The technical software estimation process is not a substitute for a full business estimation process: it is only a contributor to the full extent of its specialized expertise in terms of providing decision-makers with their professional advice on the estimation of project costs, project uncertainties and project risks.

7- References

[ABR94] Abran, A., 1994, Analyse du processus de mesure des points de fonction, PhD thesis, École polytechnique de Montréal, Canada, pp 324.

[ABR96] Abran, A. and P.N. Robillard, 1996. “Function Point Analysis: An Empirical Study of *IEEE Transactions on Software Engineering*, Vol. 22, no 12, 1996, p. 895-909.

[ABR01] Abran, A., Desharnais J.-M., Oligny S., St-Pierre D., Symons C., COSMIC-FFP Measurement Manual, Version 2.1, Technical Report, Université du Québec à Montréal, Canada, April 2001.

[ABR02] Abran, A., Ndiaye, I., Bourque, P., Contribution of Software Size in Effort Estimation, to be submitted.

[ALB84] Albrecht, A. J., IBM CIS&A Guideline 313, AD/M Productivity Measurement and Estimate Validation, IBM, 1984.

[DES93] Desharnais, J.M., Validation Process for Industry Benchmarking Data, Software Engineering Laboratory in Applied Metrics, Invited Paper, IEEE Conference on Software Maintenance, Sept.-Oct 1993, Montréal.

[FEN92] Fenton, N.E., Software Metrics: A Rigorous Approach, Chapman & Hall, 1992, pp. 337.

[ISBSG] International Software Benchmarking Standards Group, http://isbsg.org

[KIT94], Kitchenham, B.A., Taylor, N.R. "Software cost models". *ICL technical journal*, Vol. 4, no 1, May, p. 73-102.

[KOL01] Kolbe, Christian., ???

[NDI01] Ndiaye, Iphigénie, Mesure de la qualité des estimations du progiciel d'estimation SLIM, Département d'informatique, Montréal, Université du Québec à Montréal, 2001.

[STR98] Stroian, Vasile, “Un environnement automatisé pour un processus transparent d'estimation fondé sur la base de données du International Software Benchmarking Standards Group (ISBSG)” in *Département d'informatique*. Montréal: Université du Québec à Montréal, 1999, pp. 47.