Top-Down and Bottom-Up Expert Estimation of Software Development Effort

Magne Jørgensen,
Simula Research Laboratory
P.O. Box 134
NO-1325 Lysaker, Norway
Telephone: +47 67 82 82 00
Fax: +47 67 82 82 01
magne.jorgensen@simula.no

Abstract: *Expert estimation of software development effort may follow top-down or bottom-up strategies, i.e., the total effort estimate may be based on properties of the project as a whole and distributed over project activities (top-down) or calculated as the sum of the project activity estimates (bottom-up). The explorative study reported in this paper examines differences between these two strategies based on measurement and video recording of the discussions of seven estimation teams. Each estimation team applied a top-down estimation strategy on one project and a bottom-up estimation strategy on another. The main contribution of the study is the observation that the recall of very similar previously completed projects seemed to be a pre-condition for accurate top-down strategy based estimates, i.e., the abilities of the software estimators to transfer estimation experience from less similar projects was poor. This suggests that software companies should apply the bottom-up strategy unless the estimators have experience from, or access to, very similar projects.*

Keywords: Expert estimation, top-down estimation, bottom-up estimation.

1 Introduction

This paper focuses on expert estimation of software projects. Expert estimation can be described as estimation conducted by persons recognised as experts where important parts of the estimation process is based on non-explicit, non-recoverable reasoning processes, i.e., "intuition". Expert estimation includes estimation strategies in the interval from completely unaided judgment ("gut feeling") to judgment supported by historical data, process guidelines and checklists ("structured, history-based expert estimation"). According to several empirical studies, expert estimation is the most common software estimation approach [1-4].

There are valid reasons for applying expert estimation instead of formal estimation models. For example, expert estimation may be more flexible regarding required estimation input and time spent on producing the estimate. More importantly, there is no substantial empirical evidence in favour of formal software effort estimation models. We found fifteen studies comparing the estimation accuracy of expert estimates with that of model estimates for software development and maintenance work. Of those studies, five were in favour of expert estimation [5-9], five found no difference [2, 10-13], and five were in favour of model-based estimation [14-18].

Formal software development effort estimation models are typically based on explicit parametric formulas, e.g., the COCOMO II-model [19], well-established statistical theory, e.g., linear regression based estimation models [20], or well-defined clustering algorithms, e.g., the analogy-based model described in [21]. Many properties of the estimation models are consequently explicit and relatively easy to analyse. The properties of expert estimation strategies are, on the other hand, relatively difficult to analyse and the current knowledge of expert estimation strategies is consequently poor. According to Brown and Siegler [22] psychological research on real-world quantitative expert estimation *"has not culminated in any theory of estimation, not even in a coherent framework for thinking about the process"*.

The frequency of use of expert estimation, the lack of substantial evidence in favour of formal estimation models, and the poor understanding of the properties of expert estimation processes motivate the expert estimation strategy focus of the study described in this paper. The study examines the estimation processes of the two categories of software development effort expert estimation strategies termed *top-down* and *bottom-up* estimation.

When applying top-down estimation, the total effort of a software project is estimated without a decomposition of the project into activities or other types of project parts. A possible top-down expert

estimation strategy is to compare the current project, *as a whole*, with previously completed similar, projects. The estimated total effort is then distributed over activities, applying for example guidelines about the activities' typical proportion of the total effort. In this paper we are mainly interested in the estimation of the total effort. An implication of this focus is that we study only the first step, i.e., the step resulting in an estimate of the total effort, of the complete top-down strategy.

When applying bottom-up estimation, the project work is typically divided into activities and the effort of each activity is estimated. The project's estimate of the total effort is then the sum of the effort estimates of each project activity, possibly with the addition of an effort budget to cover unexpected activities and events.

The top-down and bottom-up estimation strategies may be applied in the bidding phase, planning phase and re-estimation phase of software projects. There are, however, situations that clearly favour top-down estimation, e.g., early estimates based on vague requirement specification that do not enable detailed break-down of development work, and situations that clearly favour bottom-up estimation, e.g., re-estimation of remaining activities of a project. In most estimation situations, however, both strategies are, as far as we have experienced, reasonable options for the estimation of software project effort.

We found only two empirical studies on the differences between top-down and bottom-up software development effort expert estimation strategies. Moløkken [23] found that the software professionals applying a bottom-up strategy were more over-optimistic, i.e., less accurate, than those applying a top-down estimation strategy. Connolly and Dean [24] found that the software estimation accuracy improved with bottom-up ("decomposed") estimation instead of top-down ("holistic") estimation in one experiment, but found no improvement in another. In other words, the previous results are far from conclusive. We have not found any empirically-based attempt to describe the conditions for selecting top-down or bottom-up software effort estimation strategies, i.e., with the same focus as this paper.

The remaining part of this paper is organised as follows: Section 2 describes the design, the results, and the limitations of an exploratory empirical study on differences between top-down and bottom-up-based expert estimation. Section 3 discusses the results and their implications. Section 4 concludes and suggests further work.

2 The Study

2.1 Design

The study was conducted within a large international IT consultancy company. The Norwegian branch of that company has more than 1000 employees. Most of the employees are project managers and software developers. A variety of types of applications are developed and maintained, including financial and telecom applications.

The company was paid for the participation of seven estimation teams. The estimation teams were selected to be as close to real estimation teams as possible. Each estimation team included one experienced project manager and one or two software developers with relevant background. A senior manager of the company conducted the selection and ensured that each team had sufficient development and estimation competence for realistic estimation. In total, seven project managers and ten software developers participated in the study.

Two software development projects (project A and B) were selected as basis for the estimation tasks. Project A had just been completed by the organisation when we conducted our study, while Project B was a few weeks from completion. None of the participants in our study knew anything about the projects. The two projects were selected applying the following criteria:

- There existed relatively complete requirement specifications.
- The projects were similar with respect to how difficult they were to estimate.
- There was reliable information about actual effort of the projects.
- There was no significant deviation between the specified and the delivered products.
- The projects were typical for the organisation, i.e., several similar projects had been completed by the organisation earlier.

These five criteria were introduced to ensure that the two projects were comparable with respect to estimation information and complexity, that the estimation accuracy could be measured meaningfully, and that it was meaningful to apply the top-down estimation strategy.

The selected projects included much data input, output and administration, but did not have to implement complex algorithms. The specification of Project A (11 pages) described, among other

things, 6 file-based interfaces to other systems, 3 reports and 6 user input screens. The specification of Project B (14 pages) described, among other things, 4 file-based interfaces to other systems, 1 report and 14 user input screens. The actual effort of Project A was 1340 work-hours and that of Project B 766 work-hours. Both projects were smaller than the average projects conducted by the organisation, but they were typical with respect to type of customer, application, technology and architecture. The original estimates of the effort of Project A and B were 1251 and 824 work-hours, respectively. These estimates were based on expert estimation applying a work breakdown-structure, i.e., bottom-up estimation.

When estimating the effort of the projects, the participants had the following information available:

- *The requirement specifications of the projects*. An important condition for the use of real requirement specifications of the organisation was that we did not publish the content, i.e., we cannot include the specifications as appendices of this paper.
- *The estimation process instructions*. The estimation instructions are included as Appendix 1 (top-down estimation instructions) and Appendix 2 (bottom-up estimation instructions). The work break-down structure, which guided the bottom-up estimation, is described in Appendix 3. This work-break-down structure was identical to the decomposition structure applied by most of the company's projects.
- *The company's on-line database of completed projects*. This project database was not designed for the purpose of finding similar projects, e.g., there was no search window designed for this purpose, and it turned out to be difficult to gain any benefit from its use. We ensured that information about the projects to be estimated was not applied by the estimation teams.
- *Other available people and documents in the company*. We allowed the estimators to phone people in their own company, e.g., other software developers with relevant experience, and to collect documents from their own offices or computers. Some of the estimation teams benefited from this opportunity. The estimation teams were not allowed to apply information from people that had participated in the project.

The information available to the estimation teams was similar to the information they normally had. The main differences from a completely realistic estimation process were, we believed, the limited time available (four hours to estimate two projects), the presence of estimation instructions, the presence of an "experiment leader" answering questions related to the instructions, the lack of contact with the customer, and the video recording of the estimation discussions. The possible implications of these and other limitations of the study are discussed in Section 2.3.

The seven estimation teams were divided into two groups: G1 and G2. First, the estimation teams estimated Project A, then Project B. The G1 estimation teams were instructed to apply a bottom-up estimation strategy on Project A and a top-down estimation strategy on Project B. The G2 estimation teams applied the strategies in the opposite sequence, see Table 1.

Table 1: Estimation Processes

Group	**Project A**	**Project B**
G1	Bottom-up	Top-down
G2	Top-down	Bottom-up

The estimation of a project included the following steps:

1. *Preparation (individual work)*: Each member of the estimation team read and tried to understand the requirement specification. No team discussions at this stage. Time usage: 30 minutes.
2. *Estimation (team work)*: Each estimation team discussed and agreed on the minimum, most likely and maximum effort of the project. In addition to these estimates, the teams provided the subjective probability that the actual effort would be included in the minimum-maximum interval. Time usage: Approx. 90 minutes for the bottom-up -based estimation and approx. 60 minutes for the top-down -based estimation. This difference in use of time between the two strategies was meant to reflect the difference in workload, e.g., the top-down estimates did not have to identify activities and distribute the total effort to the individual activities. The team work and discussions were video-taped.
3. *Questionnaire Completion (individual work)*: Each participant completed a short questionnaire related to the estimates and their own experience level. Time usage: 5 minutes. The questionnaire is included in Appendix 4.

This study is explorative and we did not state any hypotheses or well-defined research questions in advance. Our main research goal was to explore relationships between estimation strategy and estimation performance. The first part of the analyses is related to the quantitative data, i.e., reporting the estimates and the answers from the questionnaires. The second part analyses the actual estimation processes through the teams' estimation discussions. The analyses in the second part are based on guidelines from think-aloud protocol research [25], e.g., techniques for collecting and transcribing verbal data, and studies on group processes [26], e.g., methods for analyses of group communication. We were not able to find any other study analysing team estimation discussions in software contexts and we had, for this reason, to develop our own guidelines for coding and categorization of the verbal data from the group discussions.

2.2 Results

2.2.1 Quantitative Data

Tables 2 - 4 include the estimation data and questionnaire answers of all the estimation teams. As described earlier, the actual effort of Project A was 1340 work-hours and that of Project B 766 work-hours. As measure of estimation accuracy we use the magnitude of relative error (MRE), where

MRE = abs(actual effort – estimated effort)/actual effort.

The "Hit"-column indicates whether the minimum-maximum interval included the actual effort ("Yes") or not ("No). The "Hit rate" is the proportion of "Yes"-values, i.e.,

Hit rate = number of "Yes"-values / (number of "Yes-values" + number of "No"-values).

The relative width of the minimum-maximum interval is measured as:

RWidth = (Maximum effort – Minimum effort) / Most likely effort.

A discussion of the measurements follows each table. Note that not all the results we report are directly related to top-down or bottom-up aspects of the estimation strategies. We include these results because we believe they may be useful for better understanding of the strategies of team based expert estimation.

In the following analyses we have not included statistical tests of significance of differences, e.g., t-tests of difference in mean estimation accuracy. The main reason for this choice is that we designed this study to be an explorative in-depth analysis of expert estimation processes relationships. In other words our main research question is "When can we expect top-down to better/worse than bottom-up estimation?", not "Is top-down better/worse than bottom-up estimation?" A statistical test of significance of differences according to the second (or first) research question would require a well-defined population of people, tasks and projects and a random sample from this population, i.e., a study design different from our.

The "blocks" we analyse include few observations, e.g., the block "Estimation of Project A applying the top-down estimation strategy" include only 3 observations. The use of standard deviation as a measure of variation within a block with few observations can be misleading and we have for this reason not included this measure of variance. Instead, we recommend the reader to investigate the raw data together with the information from the qualitative part (Section 2.2.2) to evaluate the validity of the findings of this study.

Table 2: Estimation Data

Team	Project	Process (BU = Bottom-up, TD = Top-down)	Mini-mum effort (work-hours)	Most likely effort (work-hours)	Maxi-mum effort (work-hours)	Over/ under-estimate	Subjective estimate of probability of hit (Hit = "Yes")	MRE	Hit	RWidth
1	A	BU	1392	1920	2640	Over	50%[1]	43%	No	0.65
2	A	BU	917	1054	1311	Under	86%	21%	No	0.37
3	A	BU	960	1152	1920	Under	90%	14%	Yes	0.83
4	A	BU	1144	1496	1867	Under	90%	12%	Yes	0.48
5	A	TD	1600	2200	2800	Over	80%	64%	No	0.55
6	A	TD	500	540	700	Under	90%	60%	No	0.37
7	A	TD	1300	1500	2000	Over	90%	12%	Yes	0.47
1	B	TD	941	1344	2016	Over	70%	75%	No	0.80
2	B	TD	500	960	3600	Under	99%	25%	Yes	3.23
3	B	TD	600	860	1620	Under	90%	12%	Yes	1.19
4	B	TD	1500	1600	2000	Over	80%	109%	No	0.31
5	B	BU	882	1168	1566	Over	80%	52%	No	0.59
6	B	BU	668	760	1056	Under	90%	1%	Yes	0.51
7	B	BU	820	1070	1330	Over	80%	40%	No	0.48
						Mean values	**86%**	**39%**	**43%** (hit rate)	**0.80**

- *Strategy vs. estimation accuracy*: The bottom-up estimation process led to, on average, more accurate estimates for both Project A and Project B. The mean MRE of the bottom-up-based estimates on Project A was 23%, while the mean MRE of the top-down-based estimates was 45%. The corresponding values for project B was 31% (bottom-up) and 55% (top-down). Section 2.3 discusses possible reasons for this difference.
- *Strategy vs. assessment of estimation uncertainty*:
  - *Discussion of uncertainty measures*: While the estimated effort can be compared with the actual effort, there is no such corresponding value for the subjective estimate of the probability of hit. For example, provided that the probability of hit is assessed to be less than 100% it is not necessarily an incorrect assessment when the actual effort is outside the minimum-maximum interval. In the long run, however, an X% probability of hit should lead to X% of the minimum-maximum intervals including the actual effort, i.e., a "Hit rate" of X%. In our experience, most software organisations that assess the uncertainty of effort estimates either provide minimum-maximum intervals without assessment of the probability of hit, or provide minimum-maximum intervals where the probability of hit is pre-specified, e.g., the estimators are asked to provide the minimum-maximum intervals to fit the (pre-specified) 90% probability of hit. In our study, we let the estimators decide *both* the minimum-maximum interval and the probability of hit. This enables a study of differences in probability assessments as well as widths of the minimum-maximum intervals, but complicates the analyses of uncertainty assessment accuracy. The following uncertainty analyses should therefore be interpreted carefully. Note that the important criterion for accuracy of the uncertainty assessments is based on correspondence between "Hit rate" and probability assessment.
  - *Differences*: An inspection of the "Hit rates", estimated probabilities of hit and, relative widths of the minimum-maximum intervals shows no systematic difference in the accuracy of the uncertainty assessment due to estimation strategy. For example, the correspondence between "Hit rate" and estimated probability of hit seems to be equally poor, but similar to results reported elsewhere [27], for both estimation strategies. While the "Hit rate" was 43% (same "Hit rate" for both strategies), the mean estimated probability of hit was as high as 86% (same mean estimated probability for both strategies). The relative width seems to be similar for the two estimation strategies, as well. The median RWidth was 0.55 for the top-down estimates and 0.51 for the bottom-up estimates.

[1] An analysis of the team discussion shows that this value was based on a misunderstanding and should be interpreted as *the probability that the actual effort exceeds the estimated effort*. This probability value is, for this reason, excluded from the analysis.

- *Strategy vs. magnitude of estimates*: The average bottom-up-based estimate was almost the same as the average top-down-based estimate for Project A (1406 versus 1413 work-hours), but somewhat lower for Project B (999 versus 1091 work-hours). This weakly suggest that the top-down-based estimates on average were higher than the bottom-up-based estimates, i.e., a finding in accordance with that reported in [23]. An analysis of the number of projects under- and over-estimating shows similar results. Four of the bottom-up-based and six of the top-down-based estimates were over-estimates. However, the differences in our study are small and more studies are needed to understand if/when top-down estimation strategies lead to, on average, higher estimates than bottom-up estimation strategies.
- *Strategy vs. estimation bias*: Both the top-down and the bottom-up-based estimates were, on average, too high. A potential reason for this is that Project A and B were smaller than most projects completed by the company. As suggested by the results described in [9] over-estimation may be common for small projects. This estimation bias is consistent with the "regression toward the mean" effect [28], i.e., that it is a *rational* behaviour to move ("regress") the estimate towards the *mean* actual effort of a large set of previously completed projects or activities when the uncertainty of the estimate is high. In the most extreme estimation uncertainty situation, i.e., when the knowledge about the required effort is zero and an estimate has to be provided, it is meaningful to use the mean actual effort of all relevant previous projects or activities as estimate. We discuss the implications of "regression towards the mean" for software effort estimation in more detail in [29]. The "regression towards the mean"-effect is probably present for both top-down and bottom-up-based estimates. The only difference in effect is related to whether the estimates regress towards the mean effort on relevant, previously completed, whole projects or project activities. The set of previously completed and recalled development *activities* is probably larger than the set of previously completed and recalled projects. Consequently, the process of estimation in high uncertainty situations, i.e., situations when it is rational to use the mean effort of many previously completed projects or project activities as estimates, may be more robust (lower probability of high inaccuracies) for the estimation of activities compared with whole projects. More studies are needed to validate this hypothesis.
- *Number of team members vs. estimation performance*: Two of the teams (Team 2 and 4) had three project members to cover all needed estimation and development competence for meaningful estimation of project A and B. Informal analysis of the estimation performance showed no difference in estimation performance due to the addition of one team member. It is, however, to expect that more team members on the long run will remember more activities and find better project analogies. An important topic for further study is consequently how many and which experts to include to achieve optimal estimation results.

Table 3: Number of Similar Projects Found (only measured for the top-down-based estimates)

Team	Project	Number of Project Analogies Found
1	B	0
2	B	0
3	B	2
4	B	1 (but, did not remember the actual effort of that project)
5	A	0
6	A	1 (but, did not remember the actual effort of that project)
7	A	3

- *Use of project analogies vs. estimation accuracy (Table 3)*: Only four of the estimation teams found project analogies, i.e., previously completed, estimation relevant projects when applying the top-down estimation strategy. Two of those teams (Team 4 and 6) indicated during the team discussions and in the questionnaire that they did not remember or were able to re-find (applying the project data base) the actual effort of those projects. Consequently, only two of the four teams that found project analogies (Team 3 and 7) could benefit much from them. The project effort estimates of these two teams were among the most accurate (12% and 12% deviation), i.e., the top-down estimation strategy led to accurate estimates only when project analogies where the effort was known were found. As reported earlier, the organization's project data base was difficult to use for the purpose of finding similar projects and none of the teams found project analogies applying that database, i.e., all project analogies were recalled from memory. More discussion about the importance of and difficulties of finding project analogies is provided in Section 2.2.2.

Table 4: Knowledge Level and Belief in Estimate (individual answers)

Team	Project	Process (BU = Bottom-up, TD = Top-down)	Knowledge About How to Solve the Design and Programming in the Project			Personal Belief in Estimate		
			Project Leader	Developer 1	Developer 2	Project Leader	Developer 1	Developer 2
1	A	BU	Some	Low		OK	Too low	
2	A	BU	Very Much	Much	Very Much	OK	OK	OK
3	A	BU	Some	Much		Too low	OK	
4	A	BU	Some	Low	Very Much	Too low	Too low	OK
5	A	TD	Very little	Much		OK	OK	
6	A	TD	Much	Very Much		OK	OK	
7	A	TD	Some	Much		OK	OK	
1	B	TD	Some	Little		Too Low	OK	
2	B	TD	Much	Much	Some	OK	OK	Too Low
3	B	TD	Little	Much		OK	OK	
4	B	TD	Low	Much	Very Much	OK	OK	OK
5	B	BU	Low	Much		Too low	Too low	
6	B	BU	Some	Some		OK	OK	
7	B	BU	Some	Much		OK	OK	

- *Knowledge vs. assessment of estimation uncertainty (Table 4)*: As reported earlier, the estimation strategy seemingly had little impact on the accuracy of the uncertainty assessment. The self-assessed level of knowledge of how to complete the project, however, may be more important for the uncertainty assessment. While the mean RWidth of team estimates with at least one participant with very much knowledge (four team estimates) was 0.38, the mean RWidth of the remaining ten team estimates was as high as 0.93. The mean estimated probability of hit was almost the same for the two categories, i.e., 87% for the team estimates with a least one participant with very much knowledge and 85% for the other team estimates. Narrower minimum-maximum intervals as a consequence of more knowledge may be a rational behaviour if the increased knowledge led to better estimates of most likely effort. However, as suggested in the following analyses, the self-assessed level of knowledge seems to be a poor indicator of the estimation ability of software professionals.
- *Knowledge vs. over-confidence*: As reported earlier, the minimum-maximum intervals were, on average, too narrow to reflect the stated confidence levels (subjective probabilities of hit). Similar over-confidence results are reported in many other studies, e.g., [24, 30]. The level of over-confidence was higher in situations were at least one of the team members assessed his/her knowledge to be very high, i.e., high self-assessed technical skill was not an indicator of proper level of confidence. For example, only 25% of the minimum-maximum intervals included the actual effort in the situation of very much knowledge of at least one team member (four team estimates), versus 50% of the minimum-maximum intervals otherwise (ten team estimates).
- *Knowledge vs. estimation accuracy*: There were no clear connections between the team members' self-assessed level of knowledge and the estimation accuracy. Consequently, there may be important differences between "knowing how to solve a task" and "knowing how much it will cost". Similar differences for software maintenance task estimation are reported in [17]. An alternative potential explanation is that the software professionals' ability to assess their own knowledge is low.
- *Knowledge vs. impact on the estimate*: In all but two cases, the team member with the highest level of self-assessed knowledge believed that the estimate was OK. This indicates, not surprisingly, that the level of relevant technical knowledge is a good indicator of how much a person influence the estimate. The "rank", i.e., project leader versus software developer, seems to be less important.
- *Estimation team dynamics*: Both team members in Team 5 estimating Project B believed that the estimate they agreed on was too low! This may point at poor communication skills of the team members, and/or group dynamics effects leading to a decrease in individual estimation accuracy responsibility [26]. We include more discussion about the estimate of Team 5 on Project B in Section 2.2.2.

2.2.2 Team Discussions

Our analysis of the estimation discussions started with a word-by-word transcription of the discussions into text-documents, including important visual information as comments to the discussion. The transcriptions included about 180 pages of text. We had no pre-defined hypotheses, models or templates to support our analysis when we started reading the transcriptions. Instead, we tried to develop an understanding of the estimation processes from repeated reading of the transcription of the team discussions. Based on the reading we divided the discussions themes into nine categories:

1. **Estimation process discussions (Category: Process)**, i.e., discussions related to the team's own estimation process. *Example: Discussions about how to start the estimation work.*
2. **Specification and project understanding (Category: Understand),** i.e., discussions related to the understanding of the requirement specification and project context. The decomposition discussions, e.g., discussions leading to a break-down of the project into activities, were included here, because these discussions were very much related to understanding the construction of the software. *Example: Discussions about how many and which reports or user screens the project has to develop.*
3. **Search for similar projects (Category: SearchProj)**, i.e., discussions related to finding similar, previously completed, software projects. *Example: Discussions about whether a previously completed project is sufficiently similar to be used as input to an estimate of the current project, or not.*
4. **Search for similar project parts (Category: SearchPart)**, i.e., discussions related to finding similar, previously completed project parts. *Example: Discussions about the complexity of the user screens in the current project compared with the complexity of previously completed user screens in other projects.*
5. **Estimate analysis phase related activities (Category: EstimAna)**, i.e., discussions related to providing effort estimates of activities of a project's analysis phase. *Example: Discussions about the amount of analysis work that remains.*
6. **Estimate design phase related activities (Category: EstimDes)**, i.e., discussions related to providing effort estimates of activities of a project's design phase. *Example: Discussions about the amount of effort to design a particular report.*
7. **Estimate programming phase related activities (Category: EstimProg)**, i.e., discussions related to providing effort estimates of activities of a project's programming phase. *Example: Discussions about the amount of effort to program a particular user screen.*
8. **Estimate "other" activities (Category: EstimOther)**, i.e., discussions related to providing effort estimates of activities other than analysis, design, and programming. This includes, e.g., the estimation of test, project management, and documentation effort. *Example: Discussions about the required project administration work.*
9. **Estimation of total effort (Category: EstimTot)**, i.e., discussions related to the estimation of the total project effort. This category includes, among other things, application of projects analogies (found in the SearchProj-discussion) to estimate the current project as a whole, calculation of the sum of the activity estimates, discussions about potentially forgotten activities, discussions on the uncertainty of the total estimate, discussions about whether the total effort estimate feels "reasonable", and final adjustments of the total effort estimate. *Example: Discussions where the actual effort of a previously completed, similar project is applied to provide an estimate of the total effort of the current project.* **NB**: When the effort of an activity was calculated as a *proportion* of another, instead of estimating the activity directly, we decided that the estimation discussion belonged to the EstimTot category. On the other hand, if the effort of an activity was estimated *directly*, either as an intuition-based estimate or through application of similar activities completed in other projects, we decided that the discussion belonged to the relevant Estim-category, i.e., EstimAna, EstimDes, EstimProg, or EstimOther.

We marked each text sequence with respect to these categories. To increase the robustness of this categorisation process, we let another researcher conduct an independent coding of the transcribed text applying the same categories. The differences between the analyses were examined and the reasons for the differences discussed. Based on these examinations and discussions a final coding was developed.

Table 5 describes the differences between top-down and bottom-up estimation regarding the proportion of the text (counted as proportion of words) related to the different categories. An analysis of the video-tapes suggests that there was a high correlation between time spent on and proportion of words belonging to the different discussion categories, i.e. the proportion displayed in Table 5 seems to provide an accurate picture of where the estimation discussion time was spent.

Table 5: Discussion Category Distributions

Discussion category	Top-Down Estimation	Bottom-Up Estimation
1) Process	9%	7%
2) Understand	31%	49%
3) SearchProj	19%	0%
4) SearchPart	6%	1%
5) EstimAna	0%	2%
6) EstimDes	0%	1%
7) EstimProg	8%	16%
8) EstimOther	0%	4%
9) EstimTot	27%	20%

- *Strategy vs. understanding*: There were more understanding-related discussions when applying the bottom-up strategy, i.e., 49% vs. 31% of the total discussion. For example, we observed that the bottom-up estimation strategy led to much more discussion about the implication of the requirement specification compared with the top-down estimation strategy. This may be an important property of the bottom-up estimation strategy. The estimators are forced to understand more about the requirement specification implication.
- *Top-down focus on SearchProj*: As reported earlier, most of the estimation teams that applied the top-down estimation strategy did not find any previously completed projects useful for the estimation of the current project. Table 5 shows that the unsuccessful search for similar projects was not an implication of little focus on this task. As much as 19% of the discussions were related to searches for similar projects. An interesting observation was that the estimators had major problems comparing the size of previously completed projects with the new projects. The lack of established size measures, such as "function points" or "use case points", meant that they had to base size discussion on number of user screens, interfaces, etc. These size measures were, however, not standardised and the size discussions were frequently ending without any conclusion on the degree of similarity with respect to size.
- *Violation of top-down instructions*: If our top-down estimation instructions had been followed completely, the SearchPart and all the Estim-categories should have been 0% for the top-down estimation. Many of the teams, however, found it difficult to follow our instructions completely. There were consequently discussions related to searches for similar, previously completed, activities (SearchPart = 6%), and direct estimation of the programming activities (EstimProg = 8%). This finding is further discussed, in relation to the teams' processes, in the comments to Table 6.
- *Bottom-up estimation focus on EstimProg*: When following the bottom-up estimation strategy, the majority of the "Estim"-discussions were related to programming activities (EstimProg). This suggests that the bottom-up estimation strategy is focused on the estimation of the programming activities. The remaining activities are either calculated as proportion of the effort on programming (the discussions are then categorised as EstimTot) or estimated without much discussion (EstimAna = 2%, EstimDes = 1%, EstimOther = 4%). There is more on this in the comments to Table 7. The proportion of "SearchPart" when following the bottom-up estimation strategy (SearchPart = 1%) is surprisingly low, i.e., we would expect that the bottom-up strategy stimulated to SearchPart-discussions. A potential explanation is that bottom-up estimation is based on mental searches for part-analogies, e.g., pieces of programs that were similar to the one to be estimated, but that the estimators are unable to articulate the results of this search and the outcome, i.e., the search for part-analogies may be very much "intuition"-based. We discuss this topic in more detail in relation to Table 7.

Table 6 (top-down estimation) and Table 7 (bottom-up estimation) describe the main estimation steps of each team. An estimation step is characterised by a continuous sequence of words that belongs to the same discussion category. There is a change to the next step when the discussion category changes. Note that Tables 6 and 7 describes the *main* steps only, i.e., there were a few minor changes in discussion categories that are not included in the descriptions.

Table 6: Top-Down Estimation Steps

Team 1	Team 2	Team 3	Team 4	Team 5	Team 6	Team 7
Understand	Process	Process	Understand	Process	Process	Process
SearchProj	SearchProj	SearchProj	SearchProj	Understand	SearchProj	Understand
Understand	Understand	Understand	Understand	EstimProg	SearchPart	SearchProj
SearchProj	Process	SearchProj	SearchProj	Process	SearchProj	Understand
Understand	Understand	Process	SearchPart	SearchProj	Understand	SearchProj
SearchProj	EstimTot	SearchProj	SearchProj	Process	EstimProg	EstimTot
EstimTot	SearchProj	EstimTot	Understand	EstimTot	Understand	Process
Understand	Understand		SearchProj	SearchProj	EstimProg	SearchProj
EstimTot	EstimProg		EstimTot	EstimTot	Process	EstimTot
	SearchProj		Process	Process	EstimProg	SearchProj
	Understand		EstimTot	EstimTot	EstimTot	EstimTot
	EstimTot		SearchProj			
			EstimTot			

- *Top-down estimation process (Table 6)*: The typical top-down estimation process was a repeated SearchProj–Understand discussion sequence, with instances of estimation process (Process) discussions in-between. When that repeated sequence resulted in one or more useful project analogies (Teams 3, and 7), an estimate of the project was provided (EstimTot). If no useful project analogies were found, the final estimates were typically based on an incomplete bottom-up strategy (indicated by the presence of EstimProg and/or SearchPart of Teams 2, 4, 5, and 6) and finally a "gut feeling" of the required total effort. The division of the projects into parts, and search for analogies for the project parts, may be a natural strategy when the uncertainty regarding the meaningfulness of the pure top-down strategy is high, i.e., when no good project analogies were found.

Table 7: Bottom-Up Estimation Steps

Team 1	Team 2	Team 3	Team 4	Team 5	Team 6	Team 7
Process	Process	Process	Process	Process	Process	Process
Understand	Understand	Understand	Understand	EstimTot	Understand	Understand
Process	Process	EstimAna	EstimAna	Process	EstimProg	Process
Understand	EstimProg	Process	EstimDes	Understand	Understand	Understand
Process	Understand	EstimProg	EstimProg	EstimAna	EstimProg	EstimProg
Understand	EstimProg	Understand	EstimOther	Understand	Understand	Understand
EstimDes	Process	SearchPart	EstimTot	EstimDes	EstimProg	EstimProg
EstimProg	EstimProg	Understand		EstimProg	Understand	EstimTot
Understand	Understand	EstimProg		EstimOther	EstimProg	Process
EstimProg	EstimProg	Understand		EstimTot	Understand	EstimTot
Process	Understand	EstimProg		EstimOther	EstimProg	
Understand	EstimProg	Process		Understand	Understand	
EstimProg	Understand	EstimProg		EstimOther	EstimProg	
Understand	EstimProg	EstimOther		EstimTot	Understand	
EstimProg	Understand	EstimDes			EstimProg	
Process	EstimProg	EstimAna			EstimTot	
EstimProg	Understand	EstimOther				
EstimTot	EstimProg	EstimTot				
	Understand					
	Process					
	EstimDesign					
	Process					
	Understand					
	EstimProg					
	Understand					
	EstimProg					
	Understand					
	EstimProg					
	EstimOther					
	EstimTot					

- *Bottom-up estimation process (Table 7)*: There were, roughly, two types of bottom-up estimation processes: "Sequence", as exemplified by the process of Team 4, and "Inside-out", as exemplified by the process of Team 2. The teams applying the "Sequence"-process provided the activity estimates in approximately the same sequence the project would have been executed, i.e., the estimation process started with the estimation of the analysis activities, then the design activities, etc. The "Inside-out"-process, on the other hand, implied that the estimation teams started with the estimation of the programming activities (the "inside" of the project) and then estimated the other activities as proportions of the programming effort. We could not find any relation between the type of bottom-up estimation process and estimation accuracy.
- *Lack of explicit reference to previous experience*: There was not much explicit reference to the previous experience leading to an effort estimate of, for example, programming activities. As can be seen in Table 7 (category SearchPart), only the discussions of Team 3 included explicit reference to previous estimation relevant experience on project parts[2]. A typical example of the lack of explicit reference to previous experience when estimating the programming effort is the following sequence from Team 6's discussion (our explanations are inside the <…>):

  "Project Leader: *I believe that this will take about 40 work-hours.*

  Software Developer: *It will be difficult to use that little. You need a work-day … maybe two.* <One work-day for each of the 8 user screens>

  Project Leader: *OK. Let say 14 * 8* <14 work-hours * 8 user screen>. *That means 112 work-hours. Do you believe this is possible?*

  Software Developer: *It depends on the complexity of the user screens.*

  Project Leader: *Shall we say 100 work-hours?*

  Software Developer: *OK."*

  Although this discussions, hopefully, is based on a lot of previous experience, no references are made to that experience. The lack of explicit reference to previous experience may be a result of the estimation time restrictions. However, we have observed the same lack of emphasis on discussing the experience leading to an estimate when observing estimation processes in software projects, i.e., we believe that the above discussion is typical for many software estimation contexts.
- *Estimation "anchor"*: Team 5 started the process with a brief discussion about their "gut feeling" about the total estimates, indicated by the early EstimTot discussion occurrence of that team. This initial "gut feeling" was very inaccurate, suggesting an effort estimate of more than 2.000 work-hours on Project B, i.e., more than twice the actual effort of the project to be estimated (766 work-hours). Although the subsequent discussion did not refer to that initial estimate, it is possible that the initial estimate worked as an "anchor" and explains the too high estimate (1168 work-hours). This is further supported by the finding that both members of that estimation team believed that their estimate was too low although it in reality was much too high, see Table 4. We discuss the potential impact from estimation anchors in [31].

2.3 Limitations

The study's realism with respect to estimation task and participants is in our opinion high. The limitations are probably more related to:

- There may have been unwanted reduction in realism imposed by our estimation instructions and experimental setting, i.e., the estimation situation in our study deviated from their normal estimation situation.
- The transfer of our findings to other estimation tasks, conditions and participants may be difficult. For example, to which extent is it possible to transfer our results to situations with larger projects, more time spent on the estimation process, and less complete requirement specifications?

[2] The accuracy of that team's estimate was high, only 14% deviation between estimated and actual effort.

The team discussions and our informal discussions with some of the participants after they had finished suggested that they felt that the main limitations in realism related to the estimation instructions were that:

- They could not contact the customer to ask for more information. However, there had been situations were the customer was not available for more information, or did not possess the desired information.
- They would normally spend more time on the estimation task. In particular, they would normally spend more time when applying the bottom-up estimation strategy. There had been, however, situations where they completed the estimation process under similar strict time limitations, e.g., when they had to provide estimation input for a bid on a small software development project.
- Most of the participants were not used to follow a top-down estimation strategy. All the participants had, on the other hand, experience from use of the bottom-up effort estimation strategy.

These limitations in realism may explain the, on average, somewhat lower estimation accuracy compared with the original estimates of Project A and B. In addition, the lower experience with top-down estimation may explain the lower estimation accuracy of those effort estimates compared with the bottom-up estimates. We should therefore be careful when transferring the estimation accuracy results to situations different from that in our study. The observations from the team discussions may, on the other hand, be easier to transfer to other estimation contexts. For example, we believe that the finding that the software professionals had to find *very similar*, previously completed projects to benefit from the top-down estimation process is not a result of the experimental conditions.

As reported earlier the organization's project database turned out to be of little value when searching for project analogies. This means that the top-down-based estimation strategy may have performed better in situations where a searchable, high-quality project database is present.

None of the participants indicated that the presence of a video camera reduced the realism.

The best argument for the realism of our study may be that most of the estimates were reasonably accurate, e.g., the mean estimation deviation was about 30%. This level of estimation accuracy is not unusual, e.g., Standish Group (www.standishgroup.com) reports a mean estimation deviation as high as 89%.

Obviously, it would have improved the study if we had included more than seven estimation teams, more than two projects to be estimated and allowed the participants to spend more time on the estimation of the projects. However, since the company was paid close to their ordinary hourly fee for the software professionals' participation, a larger scale study would have been very expensive. Instead of increasing the scale of the study, we decided to increase the "depth" of the study through video recording and analysis of estimation team discussions. We believe that increased depth of a study can be just as, or even more, useful than increased scale in order to understand the *reasons* for high or low estimation accuracy. A better understanding increases the possibility of transfer of the results to other estimation contexts.

In sum, our results are mainly valid for team-based estimation of small projects, with strong restrictions on time spent on estimation, and little previous experience on applying the top-down estimation strategy. Some of our results, e.g., the systematic over-estimation of effort and the higher accuracy of bottom-up estimation strategies, may not be typical for other contexts. However, many of the results from the team discussions in combination with the estimation accuracy measurements, we believe, may be valid for other expert estimation contexts where top-down or bottom-up expert estimation strategies are applied. There is, however, a need for more studies on understanding the implications of expert estimation strategies to increase the confidence in the presented results.

3 Discussion of Differences

From the analysis of the accuracy measurement, team discussions and other related studies we believe that the following are essential differences between the bottom-up and top-down estimation strategies:

- In the absence of estimation models and "rules-of-thumb", as in our study, top-down-based expert estimation has to rely on the recall of similar, previously completed projects and bottom-up estimation on the recall of similar, previously completed parts of projects [24]. The team discussions and estimation accuracy results of our study suggest that the estimators found the recall of similar project *parts* much easier than the recall of similar *whole* projects. This result may, to some extent, be a consequence of the use of technical skilled estimators in our study. For

example, Moløkken [23] found that estimators in *non-technical* roles preferred a top-down estimation strategy. Moløkken reports that 75% of the estimators in non-technical roles selected a top-down estimation strategy, while 90% of the estimators in technical roles selected a bottom-up estimation strategy. In Moløkken's study the top-down-based effort estimates were, on average, the most accurate. In other words, the estimation strategy preference and accuracy seem to depend on the type of background. The results of our study may, therefore, mainly be relevant for situations with estimators with technical backgrounds

- Applying a typical, phase-based, work break-down structure as basis for bottom-up estimation enables accurate estimates only when the estimators have knowledge about these activities and do not forget important activities (or forget to add a sufficiently large budget to cover unexpected activities). This means that the applicability of the bottom-up estimation strategy is restricted to situations where there are estimators with sufficient knowledge about how to construct the software. Even then, it is easy to forget activities and/or under-estimate the degree of unexpected events. The weak tendency towards lower estimates when applying the bottom-up strategy found in our and Moløkken's [23] study may be a consequence of forgotten activities and/or under-estimation of degree of unexpected events. Ironically, in situations with bias towards over-estimation, as in our study, forgetting activities may lead to *higher* estimation accuracy of the bottom-up estimation strategy compared with the top-down estimation strategy. This means that our results of improved estimation accuracy of the bottom-up estimation strategy may in particular apply to estimation situations with bias towards over-estimation.
- Estimators following the top-down estimation strategy may be able to provide reasonably accurate estimates with very low costs and without much technical expertise applying high-level information about previously completed projects, as exemplified in [23]. A typical bottom-up-based estimation process, on the other hand, requires both technical expertise and more estimation time. This difference favours the bottom-up estimation in situations where it is useful to *force* the estimators to spend a minimum of time on understanding the implications of the requirement specification, and the top-down estimation in situations where it is essential to produce an estimate with low cost and/or without spending the effort of busy technical experts.
- Bottom-up estimation seems to lead to an increase in the proportion of time spent on understanding the implications of the requirement specification, at least in situations with strong estimation time limitations. This understanding of the requirement specification is clearly useful to solve other project goals, e.g., project planning and execution goals. Top-down estimation may not to the same extent contribute to other project goals.
- Top-down estimation may induce more distributional (history-based) thinking than bottom-up estimation [32], i.e., top-down estimation may stimulate a broader examination of previous experience compared with bottom-up estimation. The discussion category distribution in Table 5, i.e., the proportion of discussion spent on searches for analogies (SearchProj and SearchPart), supports this difference in degree of distributional thinking. A consequence of more distributional thinking may be less over-optimism [32]. Less over-optimism with top-down estimation is reported in [23], as well. In our study there was an underlying tendency to over-pessimism and the previous results on over-optimism are therefore difficult to transfer. In addition, the actual use of the history obviously depends on the *availability* of relevant experience, not only the estimation strategy. It is, therefore, not possible to conclude that estimators applying top-down estimation actually apply more of the history, only that the process seems to *stimulate* more history-based thinking.
- A pre-condition for meaningful bottom-up estimation is that the decomposed tasks can be estimated more accurately than the whole task, i.e., the actual decomposition is important to achieve the benefits from the bottom-up estimation strategy. For example, some human judgment studies report that decomposition was not useful for low-uncertainty estimation tasks, only for high-uncertainty estimation tasks [33, 34]. In addition, there are observations suggesting that a bottom-up estimation may "activate" too much knowledge (including non-relevant knowledge) when the decomposition structure is too detailed or implies a non-optimal break-down [35]. For this reason, predefined decompositions, e.g., a predefined work breakdown structure as applied in our study, activating only relevant knowledge may be a pre-condition for improvement from the use of the bottom-up estimation model.

4 Conclusions and Further Work

There are situations where expert estimation of software development effort should follow a top-down estimation strategy and situations where it is better to follow a bottom-up estimation strategy. The most important contribution of the current study may be the observation that the recall of very similar, previously completed, projects seemed to be a pre-condition for accurate top-down-based effort estimates. This observation implies that the abilities of the software estimators to transfer estimation experience from less similar, e.g., twice as large projects, are poor. The application of experience from less similar projects seems to be easier when the estimators follow a bottom-up estimation strategy. A possible consequence of this finding is that software companies cannot expect accurate top-down estimates if the estimators have no experience from, or access to, very similar projects. In those cases, the more time consuming bottom-up estimation strategy should be considered to improve the accuracy of the effort estimate.

The most important advantages of the top-down estimation strategy may be that it can be applied without much knowledge about how to build the software, e.g., by company managers providing project bids, that it does not require much time to produce effort estimates, and that it seems to stimulate to more explicit use of the previous experience, i.e., more history-based thinking.

A potential improvement of the top-down estimation strategy is to provide easier access to similar, previously completed projects. In the current study the estimators had to rely on their own memory, a project data base with poor search facilities, and contact with other available personnel to recall similar projects. A project data base enabling searches for similar projects may imply that more estimation teams would have found similar projects when following the top-down estimation instructions and, consequently, improved the estimation accuracy. We have initiated work on such estimation support at the company participating in the study and plan new studies on how a searchable project data base impacts the accuracy of the top-down in comparison with the bottom-up-based effort estimates.

Acknowledgement: Thanks to professor in psychology at the University of Oslo, Karl Halvor Teigen, for interesting discussions on human judgment and the provision of relevant studies.

References

1. J. Hihn and H. Habib-Agahi. *Cost estimation of software intensive projects: A survey of current practices*. Proceedings of *International Conference on Software Engineering*. 1991: IEEE Comput. Soc. Press, Los Alamitos, CA, USA. p. 276-287
2. F.J. Heemstra and R.J. Kusters, Function point analysis: Evaluation of a software cost estimation model. *European Journal of Information Systems*, 1991. **1**(4): p. 223-237.
3. J. Paynter. *Project estimation using screenflow engineering*. Proceedings of *International Conference on Software Engineering: Education and Practice*. 1996. Dunedin, New Zealand: IEEE Comput. Soc. Press, Los Alamitos, CA, USA. p. 150-159
4. J. Hill, L.C. Thomas, and D.E. Allen, Experts' estimates of task durations in software development projects. *International Journal of Project Management*, 2000. **18**(1): p. 13-21.
5. R.J. Kusters, M.J.I.M. Genuchten, and F.J. Heemstra, Are software cost-estimation models accurate? *Information and Software Technology*, 1990. **32**(3): p. 187-190.
6. S.S. Vicinanza, T. Mukhopadhyay, and M.J. Prietula, Software effort estimation: An exploratory study of expert performance. *Information systems research*, 1991. **2**(4): p. 243-262.
7. T. Mukhopadhyay, S.S. Vicinanza, and M.J. Prietula, Examining the feasibility of a case-based reasoning model for software effort estimation. *MIS Quarterly*, 1992. **16**(2): p. 155-171.
8. A. Pengelly, Performance of effort estimating techniques in current development environments. *Software Engineering Journal*, 1995. **10**(5): p. 162-170.
9. B. Kitchenham, et al., A case study of maintenance estimation accuracy. *To appear in: Journal of Systems and Software*, 2002.
10. A.L. Lederer and J. Prasad, Software management and cost estimating error. *Journal of Systems and Software*, 2000. **50**(1): p. 33-42.
11. N. Ohlsson, C. Wohlin, and B. Regnell, A project effort estimation study. *Information and Software Technology*, 1998. **40**(14): p. 831-839.

12. F. Walkerden and R. Jeffery, An empirical study of analogy-based software effort estimation. *Journal of Empirical Software Engineering*, 1999. **4**(2): p. 135-158.
13. P. Bowden, M. Hargreaves, and C.S. Langensiepen, Estimation support by lexical analysis of requirements documents. *Journal of Systems and Software*, 2000. **51**(2): p. 87-98.
14. K. Atkinson and M. Shepperd. *Using function points to find cost analogies*. Proceedings of *European Software Cost Modelling Meeting*. 1994. Ivrea, Italy.
15. M. Jørgensen. *An empirical evaluation of the MkII FPA estimation model*. Proceedings of *Norwegian Informatics Conference*. 1997. Voss, Norway: Tapir, Oslo. p. 7-18
16. I. Myrtveit and E. Stensrud, A controlled experiment to assess the benefits of estimating with analogy and regression models. *IEEE Transactions on Software Engineering*, 1999. **25**: p. 510-525.
17. M. Jørgensen and D.I.K. Sjøberg, Impact of experience on maintenance skills. *Journal of Software Maintenance and Evolution: Research and practice*, 2002. **14**(2): p. 123-146.
18. F. Niessink and H. van Vliet. *Predicting maintenance effort with function points*. Proceedings of *International conference on software maintenance*. 1997. Bari, Italy: IEEE Comput. Soc, Los Alamitos, CA, USA. p. 32 - 39
19. B. Boehm, et al., *Software cost estimation with Cocomo II*. 2000, New Jersey: Prentice-Hall.
20. Y. Yokoyama and M. Kodaira. *Software cost and quality analysis by statistical approaches*. Proceedings of *International Conference on Software Engineering*. 1998. Kyoto, Japan: IEEE Comput. Soc., Los Alamitos, USA. p. 465-467
21. M. Shepperd and C. Schofield, Estimating software project effort using analogies. *IEEE Transactions on Software Engineering*, 1997. **23**(11): p. 736-743.
22. N.R. Brown and R.S. Siegler, The role of availability in the estimation of national populations. *Memory and Cognition*, 1993. **20**: p. 406-412.
23. K. Moløkken, *Expert estimation of Web-development effort: Individual biases and group processes (Master Thesis)*, in *Department of Informatics*. 2002, University of Oslo.
24. T. Connolly and D. Dean, Decomposed versus holistic estimates of effort required for software writing tasks. *Management Science*, 1997. **43**(7): p. 1029-1045.
25. K.A. Ericsson and H.A. Simon, *Protocol analysis: Verbal reports as data (Revise Ed.)*. 1993, Cambridgde, MA: The MIT Press.
26. D.R. Forsyth, *Group dynamics*. 1999: Wadsworth Publishing Company.
27. M. Jørgensen, K.H. Teigen, and K. Moløkken, Better sure than safe? Overconfidence in judgment based software development effort prediction intervals. *To appear in Journal of Systems and Software*, 2003.
28. D.T. Campbell and D.A. Kenny, *A primer on regression artifacts*. 1999: The Guilford Press.
29. M. Jørgensen, U. Indahl, and D. Sjøberg, Software effort estimation and regression toward the mean. *To appear in Journal of Systems and Software*, 2003.
30. M. Jørgensen and K.H. Teigen. *Uncertainty Intervals versus Interval Uncertainty: An Alternative Method for Eliciting Effort Prediction Intervals in Software Development Projects*. Proceedings of *International Conference on Project Management (ProMac)*. 2002. Singapore. p. 343-352
31. M. Jørgensen and D.I.K. Sjøberg, Impact of software effort estimation on software work. *Journal of Information and Software Technology*, 2001. **43**: p. 939-948.
32. R. Buehler, D. Griffin, and M. Ross, Exploring the "Planning fallacy": Why people underestimate their task completion times. *Journal of Personality and Social Psychology*, 1994. **67**(3): p. 366-381.
33. D.G. MacGregor, *Decomposition for judgmental forecasting and estimation*, in *Principles of forecasting: A handbook for researchers and practitioners*, J.S. Armstrong, Editor. 2001, Kluwer Academic Publishers: Boston. p. 107-123.
34. J.S. Armstrong, W.B. Denniston Jr., and M.M. Gordon, The use of the decomposition principle in making judgments. *Organizational-Behavior-and-Human-Decision-Processes.*, 1975. **14**(2): p. 257-263.
35. D.G. MacGregor and S. Lichtenstein, Problem structuring aids for quantitative estimation. *Journal of Behavioral Decision Making*, 1991. **4**(2): p. 101-116.

Appendix 1: Bottom-Up Instructions

<Group 1 received these instructions for Project A, Group 2 for Project B.>

INSTRUCTIONS:

Important: Use a "bottom-up" estimation process, i.e., an estimation process based on a break-down of the project in activities and estimation of these activities individually. Use the work break-down structure described in <*Appendix 3*>. You are allowed to further detail the break-down structure.

Description of the estimation context:

Your company has already got he contract of developing the software described in <*the requirement specification – not included in this paper of confidentiality reasons*>. The task of your estimation team is to estimate the effort for the purpose of the planning of the project.

Most of the analysis phase is already completed and shall not be included in the estimate. In addition to the most likely effort, you are supposed to provide the minimum (best case) and maximum (worst case) effort, and the probability that the actual effort will be between the minimum and the maximum effort.

Example: You believe that the most likely use of effort for a project is X work-hours, that the minimum effort is as little as Y work-hours, and that the maximum effort is Z work-hours. You estimate that it is P percent likely that the actual effort is between Y and Z.

You do not know who the project members will be. Assume that the participants are normally skilled employees of your company.

SEQUENCE OF THE ESTIMATION WORK:

Step 1 (individual work): Read and understand the requirement specification.

Step 2 (team work): Divide the project work into activities according to the specified work break-down structure and agree on estimates of the most likely effort, the minimum and, the maximum effort of each of the activities. Then, agree on the estimated, minimum, and maximum effort of the total project. The inclusion of the probability that the actual effort is between minimum and maximum is only necessary for the estimates of the total project.

Step 3 (individual work): Complete questionnaire. <*Appendix 4*>

Appendix 2: Top-Down Instructions

<Group 1 received these instructions for Project B, Group 2 for Project A.>

INSTRUCTIONS

Important: When estimating the project you are not allowed to break the project into activities. Instead, you are supposed to try to estimate the project through a comparison with previously completed (preferably similar) projects. In other words, you are supposed to use an analogy-based (top-down) estimation process. Discussions and all written notes shall reflect that process.

This estimation process may be unfamiliar to many of you, but do you best. The projects you compare with do not need to be very similar to the project to be estimated to be used. You may, for example, compare characteristics such as number of screens, number of tables of the current project with previously completed projects.

Description of the estimation context:

Your company has already got he contract of developing the software described in <*the requirement specification – not included in this paper of confidentiality reasons*>. The task of your estimation team is to estimate the effort for the purpose of the planning of the project.

Most of the analysis phase is already completed and shall not be included in the estimate. In addition to the most likely effort, you are supposed to provide the minimum (best case) and maximum (worst case) effort, and the probability that the actual effort will be in between the minimum and the maximum effort.

Example: You believe that the most likely use of effort for a project is X work-hours, that the minimum effort is as little as Y work-hours, and that the maximum effort is Z work-hours. You estimate that it is P percent likely that the actual effort is between Y and Z.

You do not know who the project members will be. Assume that the participants are normally skilled employees of your company.

SEQUENCE OF THE ESTIMATION WORK:

Step 1 (individual work): Read and understand the requirement specification.

Step 2 (team work): Discuss and agree on an estimate on the most likely effort, the minimum effort, the maximum effort and the probability that the actual effort will be inside the minimum-maximum interval). Describe important assumptions.

Important: Emphasise to find earlier projects (preferably similar), and to use this in the estimation process.

Step 3 (individual work): Complete questionnaire. <*Appendix 4*>

Appendix 3: Work Break-Down Instructions

<Group 1 received these instructions for Project A, Group 2 for Project B.>

Work break-down structure:

1. Administration
2. Meetings
3. Analysis (not already completed)
4. Design
5. Programming
6. Data base work
7. Test
8. Documentation
9. Installation/system integration

Appendix 4: Questionnaire

<to be completed, individually, after the team had agreed on a project effort estimate>

1) How much knowledge do you have about how to conduct the design and programming of the project? (select only one of the following options):

a) Very much
b) Much
c) Some
d) Little
e) Very little

2) The estimate is a value you agreed on in the team. Do YOU think that the estimate was (select only one of the following options):

a) much too low
b) too low (but, not much too low)
c) OK
d) too high (but, not much too high)
e) much too high

<Question 3 was only included for the top-down estimation task, i.e., Project B for Group 1 and Project A for Group 2.>

3) How many, previously completed, projects did you find and use as input to the estimation of the project?