Chapter 1

The Estimation Process: Phases and Roles

OBJECTIVES

This chapter covers

– Two generic approaches to estimation: judgment-based and engineering based
– An overview of the process for estimating software projects
– The foundation: The productivity model
– The phases of the estimation process
– Roles and responsibilities in estimating and budgeting

1.1 INTRODUCTION

When an organization has not measured its own productivity on past projects, it is mostly in the dark about:

- how the organization is performing,
- how much a manager's performance differs from someone else's, and
- how much the assumptions made in a manager's estimation judgment differ from those made in someone else's!

In this context, which is typical in many software organizations, using productivity models originating in environments with different productivity performance ratios does not provide real value. This is all the more true when little is known about

- the quality of the data in these external repositories and
- the quality of the productivity models within the environments in which they have been built.

When an organization has collected its own data and developed its own set of capabilities for analyzing those data and documenting the quality of their productivity models, then it has developed

Software Project Estimation: The Fundamentals for Providing High Quality Information to Decision Makers, First Edition. Alain Abran.

- a key competitive advantage in market-oriented organizations and
- a key credibility advantage in organizations in noncompetitive contexts.

Estimation is not at all about coming up with a magic number to which everyone must commit at the peril of their professional career (which leads to staff members spending lots of overtime attempting to meet unrealistic deadlines.)

This chapter presents an overview of the phases of an estimation process and explains the differences between a productivity model and its use in an estimation process. It is organized as follows:

- Section 1.2 introduces two generic approaches to estimation: judgment and engineering.
- Section 1.3 provides an overview of some common practices and expectations involved in estimating software projects.
- Section 1.4 discusses the levels of uncertainty in an estimation process.
- Section 1.5 presents the key concepts of a productivity model.
- Section 1.6 explains the use of a productivity model in an estimation process.
- Section 1.7 discusses the estimation responsibilities in a business context.
- Section 1.8 explains the differences between budgeting and pricing.
- Section 1.9 provides a summary of the chapter.

1.2 GENERIC APPROACHES IN ESTIMATION MODELS: JUDGMENT OR ENGINEERING?

1.2.1 Practitioner's Approach: Judgment and Craftsmanship

In contrast to estimation with mathematical models, where explicit cost drivers are included in the models as either quantitative or categorical parameters, which are manipulated with well-described mathematical equations, the estimation technique often used in practice in industry (also referred to as the *expert judgment estimation approach*) would not typically document which parameters are taken into account, or how they are explicitly combined.

The overall estimation process in the expert judgment approach is similar to the estimation process described later in this chapter, but is much less transparent, of course, and there is no possibility of tracing back to historical data on how the expert judgment models were built. In addition, it is not feasible to gauge the performance of the expert judgment models when there are no objectively quantified and standardized data on key project variables, such as software size:

- A project might appear to be successful if it has respected the "official" budget; however, without the ability to verify that all the promised functions have been delivered, it is a mistake to claim that the estimates were correct: when only a portion of the required functions are delivered, then the expected benefits cannot all be harvested, which destroys the cost-benefit analysis that justified the launching of the project in the first place.

We can conclude from this that analyzing the performance of an expert-based estimate without a corresponding analysis of the functionality delivered is of very limited value.

Of course, the expert judgment approach is highly dependent on the specific expertise of the people participating in the estimation process, and will vary from project to project, making performance assessment challenging.

This dependency of expert judgment on expertise gives the estimation process many of the characteristics of a craft, which is mostly dependent on the abilities of the craftsmen, rather than on a thoroughly tested and well-defined engineering technique.

The decision as to which cost drivers to include, as well as the determination of the values of each interval within a cost driver for particular impact, is most often based entirely on the judgment of a group of estimation tool builders, or even a single tool builder.

Practitioners will also typically attempt to improve conventional software estimation models using a similar approach:

- The addition, modification, and/or deletion of cost drivers is based on a value judgment (also referred to as *expert judgment* or *subject matter expertise*).
- An impact factor is also assigned on this basis.

What this means is that the improvement process is typically subjective and most often undertaken without statistical analysis to support proposed changes.

1.2.2 Engineering Approach: Modest – One Variable at a Time

Building software models from an engineering perspective is based on

- Observation of past projects and quantitative data collection
    - including identification of the scale types of the variables, and taking them into account to ensure adequate use of those variables in productivity models.
- Analysis of the impact of individual variables, one at a time.
- Selection of relevant samples, and of samples of sufficient size from a statistical viewpoint.
- Documentation and analysis of the demographics of the dataset used.
- Very careful extrapolation to contexts other than those from which the data were collected.

The engineering approach consists of investigating the factors involved and studying them one at a time before combining them.

In this approach, it is not taken for granted that it is feasible for a single model to handle all sets of conditions:

- Instead, a search is conducted for models that are reasonably good within a well-identified and understood set of constraints.

This is the approach taken in this book for building the basis for productivity models:

- Look at the work–effort relationship, one variable at a time, to gain insights for each variable.

Taking this approach means initially obtaining a number of productivity models for each variable, and admitting that

- no one model will be perfect (i.e., it will not take into account the other variables *directly*) and that
- each model will teach us something about the effect of that single variable on the dependent variable, which is effort.

1.3 OVERVIEW OF SOFTWARE PROJECT ESTIMATION AND CURRENT PRACTICES

Here we present an overview of the estimation process, followed by some current practices and expectations.

1.3.1 Overview of an Estimation Process

A high-level view of a software estimation approach is depicted in Figure 1.1:

(A) On the left are the inputs to the software estimation process. These inputs typically consist of

- Product requirements:
  - ▶ the functional requirements requested by the users and allocated to the software.
  - ▶ the nonfunctional requirements, some of which will be allocated to software, and others to other parts of the system (hardware, procedures manual, etc.).
- Software development process: typically, a specific life cycle is selected (agile, iterative, etc.), along with its various components, including the development platform, the programming languages, and the project team.
- Project constraints: these are the constraints externally imposed on the project (predefined deadlines, maximum available budget, etc.).

(B) In the center is a representation of the productivity model used as the foundation of the estimation process, and includes

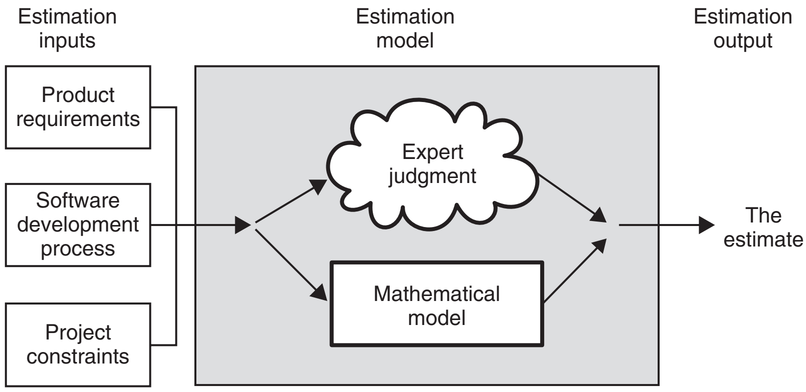

Figure 1.1 One Perception of an Estimation Process

- the "implicit" models of each of the experts participating in the estimation process (typically, the productivity model of the experts is not documented).
- mathematical models: regressions, case-based reasoning, neural networks, and so on.

(C) On the right is the estimation output normally expected, which constitutes

- an estimate of the amount of effort (or cost, or project duration) required to deliver software that will meet the requirements specified in input at the specified level of quality.

1.3.2 Poor Estimation Practices

In the literature, there is a large body of knowledge on project estimation in general, and on software project estimation in particular; however, in practice, the software industry is plagued by a number of poor estimation practices, such as those illustrated in Figure 1.2:

(A) The estimation inputs:

- There is only a very brief description by the customer of the software system expected, usually at a very high level (i.e., poorly defined) and, of course, poorly documented. How many times are software staff asked to provide estimates based on a half-page description of user requirements? This type of estimation input is referred to as a "wish list" in Figure 1.2. Such a list will inevitably change over time, and most probably expand at an unpredictable rate.
- In the hope of offsetting this lack of description of what is expected by users, a software manager will want to take as many cost drivers as possible into account, expecting in this way to lower his estimation risks.

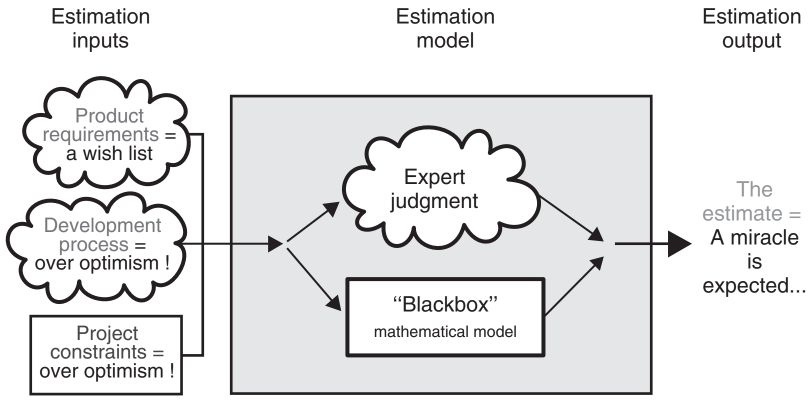

Figure 1.2 Some of the Poor Estimation Practices Observed in Industry

(B) The estimation model:

- A formal or an informal model to mix (in a black-box manner) these ill-defined requirements together through the use of readily available:
  - ▶ local experience: internal or external experience (the expert judgment approach) or
  - ▶ mathematical models described in books or hidden in estimation tools.

(C) The estimation output is made up of

- a single estimate, which is made up of the mandated project budget that must be respected, along with the requirement that the expected functionality be produced within a prescribed period of time;
  - ▶ Note: this figure does not take into account unplanned overtime, for which the development team will not be paid!
- an overly optimistic attitude, which is very common among software professionals, that the development team will outperform any previous historical performance and overcome all constraints in a timely manner; and
- accountability on the part of the software engineer or project manager providing the estimate, in terms of meeting customer expectations and respecting the project budget allocated by senior management.

To summarize, in this worst practice, both customers and senior management expect that their staff (and suppliers) will commit to delivering the expected software functionality on time and on budget, and all this without having themselves worked out the details of what they expect as a well working product and the uncertainties inherent to any new project.

In other words, on the one hand, miracles are expected by customers and senior management, and, on the other, too many software staff, when providing single-point estimates, behave as if they are in the business of continuously delivering miracles!

Some of the Best Estimation Practices in Industry

Mature software organizations consider estimation as a process that gives them a business advantage over their competitors: to acquire this competitive advantage, they have invested in their estimation process to master the key factors, including:

- investment in gathering project requirements and in understanding their qualities;
- use of international standards for software measurement;
- continuous quantitative measurement throughout the project life cycle;
- quantitative analysis of their past performance: that is, how productive were they in terms of delivering past projects and meeting project objectives;
- in-depth analysis of their estimation performance (actual vs estimated).

Some of the Worst Estimation Practices in Industry

- Wishful thinking and single-point estimates.
- Use of estimation black boxes (expert judgment and/or undocumented mathematical models).
- Reliance on others' numbers: no investment in their estimation process to develop a sustainable competitive advantage.

1.3.3 Examples of Poor Estimation Practices

The following are some examples of poor estimation practices – see also Figure 1.3.

(A) Inputs to estimation models:

- Product requirements = Wish list:
  - No measurement of the functional requirements themselves, using international standards.
  - Use of post-project KLOC (thousands of lines of code) without considering the mix of programming languages and their different characteristics.
  - Size units often considered almost irrelevant.
  - Guesstimate of KLOC based on poor requirements and a poor understanding of the relationships between requirements and KLOC in various programming languages.

(B) Development process:

- Individual descriptive factors transformed into quantitative impact factors without knowledge of their accuracy and variance.
- No objective quantitative knowledge of the impact of the project variables in their own development environment.
- Total reliance on outside numbers without strong supporting evidence.

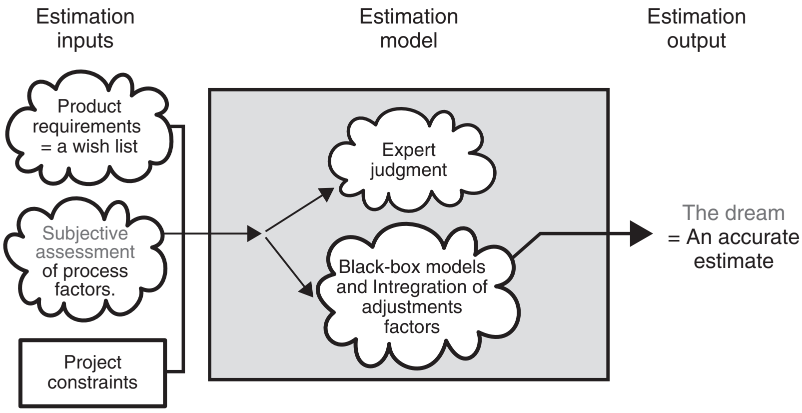

Figure 1.3 The Dream: An "Accurate" Estimate

(C) Productivity model:

- Unknown estimation performance of the so-called experts in the expert-based estimation approach.
- No verification that the assumptions necessary for each statistical technique have been met (e.g., The "normality" distribution of the variables for regression models).
- Too many variables and not enough data points for sound statistical analysis.
- No verification of the size of the software to be delivered in the analysis of the performance of the expert-based approach.
- And so on.

(D) Estimation output:

- The dream: an accurate estimate.
- Limited analysis of the range of candidate values and candidate causes of variations in the estimates.
- Limited documentation of the quality of their estimation outcomes.

1.3.4 The Reality: A Tally of Failures

Software project estimation is a recurrent and important activity in large and small organizations across the world, and a large amount of research has been performed on software project estimation over the past 50 years and a large number of models proposed to industry. The bottom line is, how well is software estimation performing in industry?

The answer is, not very impressively [Jorgensen and Molokken 2006; Jorgensen and Shepperd 2007; Petersen 2011]:

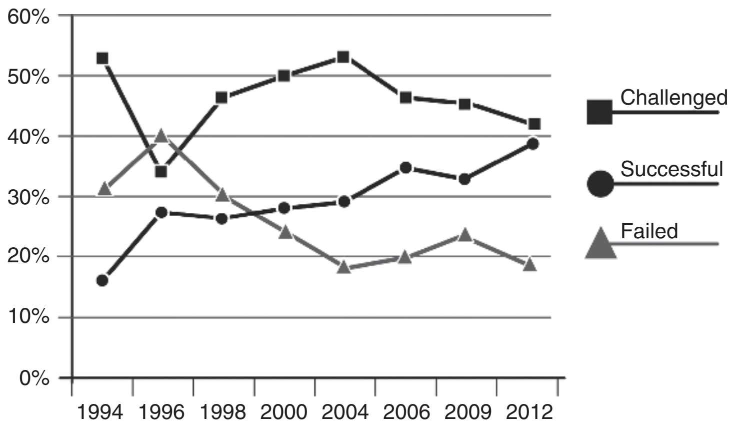

Figure 1.4 Project Success Trends Based on Standish Group Data [Adapted from Miranda 2010].

- Figure 1.4, constructed using data from the Standish Group Chaos Reports cited by [Eveleens and Verhoef 2010], shows that, over the 30-year period, barely 30% of software projects have been delivered on time and on budget:
  - Since the publication of the first Standish Group report in 1995, the software development community has been making some progress in its ability to complete development projects on time and on budget, but almost 70% of software projects still finish late and over budget, or are cancelled.
- The 2008 study by El Eman and Koru [2008] puts the average number of challenged and failed projects at 50%.

1.4 LEVELS OF UNCERTAINTY IN AN ESTIMATION PROCESS

1.4.1 The Cone of Uncertainty

The well-known *cone of uncertainty* attempts to represent the range of expected variations in models across the project life cycle – see Figure 1.5.

At the early, feasibility stage, which is about future projects (i.e., $t = 0$):

- The project estimate can err on the side of underestimation by as much as 400%, or on the side of overestimation by 25% of the estimate.

At the end of a project (i.e., t = the end of the project):

- The information on effort, duration, and costs (i.e., the dependent variables) is now known relatively accurately (with respect to the quality of the data collection process for effort recording).

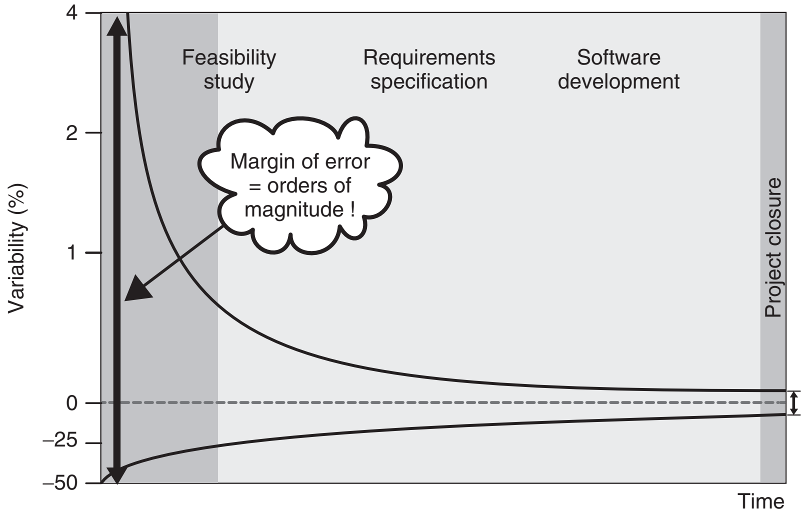

Figure 1.5 Uncertainty Levels in the Project Life Cycle [Adapted from Boehm et al. 2000, Figure 1.2, p. 10]

- The information on the cost drivers (independent variables) are also relatively well known, since they have all been observed in practice – the variables are therefore considered to be "fixed" without uncertainty (many of these are non-quantitative, such as the type of development process, programming language, and development platform.)
- However, the relationships across these dependent variables and the independent variable are far from being common knowledge. *Even in this context of no uncertainty at the level of each variable at the end of a project*, there is no model today that can perfectly replicate the size–effort relationship, and there remains uncertainty in the productivity model itself.

We refer to this stage as the *productivity model stage* (at t = the end of project). The reason why the cone of uncertainty at the extreme right of Figure 1.5 does not infer full accuracy is because all the values in this cone are tentative values provided mostly by expert judgment.

1.4.2 Uncertainty in a Productivity Model

A rough graphical two-dimensional representation of the performance of productivity models (i.e., in a context of completed projects) is depicted in Figure 1.6, where the size of completed projects is plotted along the horizontal axis and the actual effort for the completed projects is plotted along the vertical axis. Each point in this figure corresponds to a completed project (in terms of its size and actual effort) and the slope corresponds to the statistical equation that would best represent this set of completed projects, that is, the corresponding productivity model.

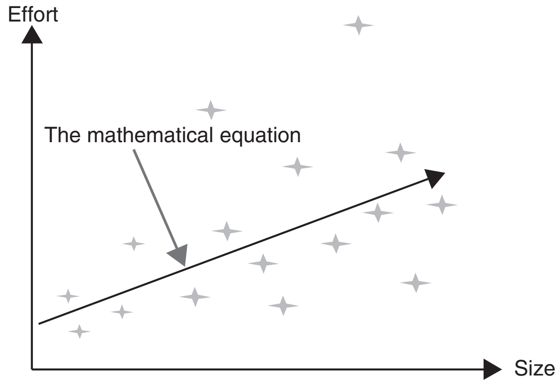

Figure 1.6 A Model with One Independent Variable

- In other words, the productivity model represents the modeling of the relationships across the two variables in this figure, that is, between the independent variable (the size of the software) and the dependent variable (completed project effort).

It can be observed in Figure 1.6 that most of the actual data do not fall exactly on the mathematical equation (i.e., the slope line), but at some distance from it. This means that the productivity model does not accurately model the size–effort relationship: some actual data are close to the line, with other data quite far apart, even though there was no uncertainty on the inputs to the estimation process.

The current performance targets often mentioned in the literature for such productivity models (with one or multiple independent variables) for modeling the size–effort relationship are something like those in Figure 1.7:

- That is, 80% of the projects fall within 20% of the distance from the equation line, and 20% of the projects outside of this distance (but within an unspecified upper range of variation).

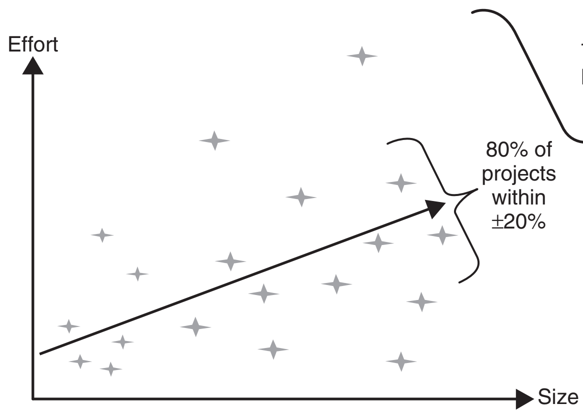

Figure 1.7 Model Accuracy Targets.

The context (and the data collected) used to build a productivity model differs significantly from that in which estimation must be performed: in practice, a project must be estimated fairly early in the project life cycle (i.e., *a priori*), at a time of high uncertainty, in terms of both what software functions must be developed and how they will be developed.

In the next two sections, we discuss both productivity models in more detail, and then their application in an estimation process.

1.5 PRODUCTIVITY MODELS

Researchers typically build their mathematical models using data from completed projects.

- This means that they start with a set of known facts, about which there is no uncertainty – see Figure 1.8.
- Therefore, most of the so-called estimation models in the literature are actually *productivity* models.

The inputs to the models to be built are

- Product requirements: the software that has been built and delivered.
  - The software can be measured very precisely based on the actual software delivered.
  - The software characteristics can also be described, using whatever classification schemes are available.
- The software development process is completed, and can also be described and categorized without uncertainty:

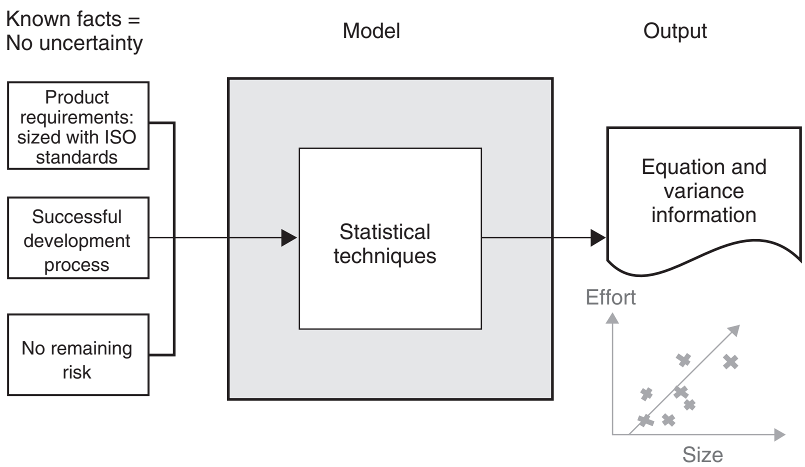

Figure 1.8 The Context of a Productivity Model

  - Resources: the relevant staff experience in a business domain, their development expertise, their availability during the project, and so on.
  - Process: the development methodology, the development environment, and so on.
- The project constraints are now precisely known, with no remaining uncertainty and no remaining risks: they are constant (i.e., no longer variable).

In summary, these inputs from completed projects can constitute either

- quantitative information (such as software functional size, perhaps measured using an international standard of measurement – such as Function Points or COSMIC Function Points) or
- nominative information (such as the programming language) or nominal categories of information (such as categories of case tools), or ordinal variables (such as levels of complexity, from very simple to very complex).

(A) Mathematical equation models
Estimators have at their disposal a large number of mathematical techniques to help them to determine quantitatively, from a number of completed projects, the relationships between the dependent variable of interest (for instance, project effort or project duration) and the independent variables (the product size and the various cost drivers).

- For instance, the small graph at the bottom right of Figure 1.8 (and Figure 1.6) represents the relationships between the size of the software projects completed and the effort required to deliver these projects:
  - The horizontal axis represents the size of the software delivered (i.e., the past).
  - The vertical axis represents the effort expended for each project.
  - The stars each represent a project tuple (size and effort).
  - The slope of the graph represents the regression line that corresponds best to the set of points (i.e., the relationships between the independent variable – project size – and the dependent variable – project effort). This regression line, obtained by a statistical model, represents the productivity of projects delivered in the context of the points composing this specific dataset, and so corresponds to the productivity of the past projects for which there is no longer any uncertainty.

Some of the key benefits of these mathematical models of the productivity of past projects are the following:

- The variables in these datasets are described using a documented set of conventions.
- The performance of these mathematical models can be described and analyzed.
  - For instance, with the regression model in Figure 1.8, the delta of each point to the equation line can be calculated to figure out the “quality” of these models.

- Anybody can use these models for estimating future projects, and, provided the same information is inserted in these models, the same number will come out (in that sense, the models are "objective"). In practice, the estimate will vary when the inputs vary.

Productivity models are therefore *models of past projects built from known information* with

- quantitative variables measured accurately based on what has been implemented in the software (but still with some degree of imprecision in their measurement),
- quantitative variables collected during the project life cycle and archived in the project recording system, or
- other descriptive variables of known information assessed subjectively by project experts, and for which there is no intrinsic uncertainty, the projects being completed.

(B) Expert judgment approach
The expert judgment approach is generally informal, not documented, and is informally derived from past experience based on subjective recollection of past projects, very often without reference to precise quantitative information on the software delivered, nor on precise information on cost drivers.
The only precise information available would typically concern the dependent variables (effort and duration), but not the independent variables (e.g., product size, in particular in terms of the functionality delivered).
In addition, there is usually no precise information on the productivity of past projects and no graphical representation of the performance of a set of projects.

1.6 THE ESTIMATION PROCESS

1.6.1 The Context of the Estimation Process

The typical estimation context is characterized by

- *the imprecise nature of the requirements at estimation time early in the life cycle*:
  - imprecision of the requirements,
  - ambiguities and omissions in the requirements,
  - instability of the requirements across the project life cycle,
  - and so on.

  Of course, all the above-mentioned statements make it impossible to measure the requirements size accurately at that time, when size can at best be approximated.
- *uncertainty about a number of factors that could impact the project*:
  - the experience of the project manager,

  - whether or not the new development environment will perform as advertised by its vendor,
  - and so on.
- *a number of risks*:
  - users changing their minds about the requirements,
  - an inability to hire competent staff within the planned time frame,
  - loss of key staff,
  - and so on.

Estimating future software projects is often, in practice, carried out in such a context, when the information

- is incomplete,
- contains a number of unknowns, and
- is associated with a number of risks.

In this chapter, those needs are addressed through an engineering process to develop an *estimation process* to handle the constraints mentioned above, which are

- incompleteness,
- uncertainty, and
- risks.

1.6.2 The Foundation: The Productivity Model

The productivity model developed from past projects is at the core of an estimation process, whether this model

- is described formally through mathematical equations or
- is hidden beneath the experience-based knowledge of people using the expert judgment approach to software estimation.

Next, the productivity model in Figure 1.8 is used in an estimation context (i.e., the left-hand side of the cone of uncertainty depicted in Figure 1.5) for future projects when

- the inputs (including the size of the product requirements and cost drivers) are unknown precisely and have themselves potentially large ranges of variation and uncertainty.

The expected ranges of variation of the inputs (i.e., future projects) on the horizontal axis will definitively impact the range of variation of the variable estimated in output (for instance, project effort or project duration on the vertical axis) leading to larger candidate variations than the initial productivity models built from past projects.

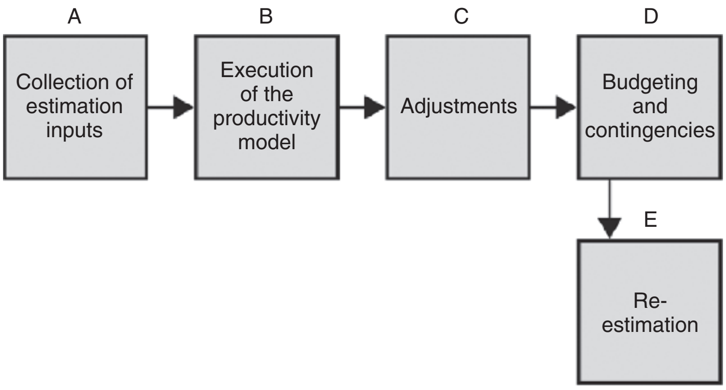

Figure 1.9 The Estimation Process

1.6.3 The Full Estimation Process

The estimation process includes the following five major phases – see Figure 1.9:

(A) Collection of the inputs to the estimation process:

- measurement of the product requirements (or, most often, estimation or approximation of the size of the requirements) and
- assumptions for most of the other cost drivers.

(B) Use of a productivity model (as a kind of simulation model).

(C) An adjustment process to take into account variables and information not included in the productivity model, including:

- identification of uncertainty factors and
- risk assessment.

(D) A budget decision on a single-number budget (at the project and portfolio levels).

(E) Re-estimation when required by project monitoring and control.

Each of these phases is described below in greater detail.

Phase (A) Collection of the estimation inputs – see Figure 1.10

Analysis of the project information and data collection for identifying the cost drivers (Resources, Process, and Products) to be used as inputs to a specific project to be estimated.

Estimation of the values of the cost drivers identified.

- ▶ At the time an estimate is prepared, the nature of these inputs is uncertain, which is why they have to be estimated.
- ▶ The uncertainty associated with these inputs should be documented for use in Phase B.

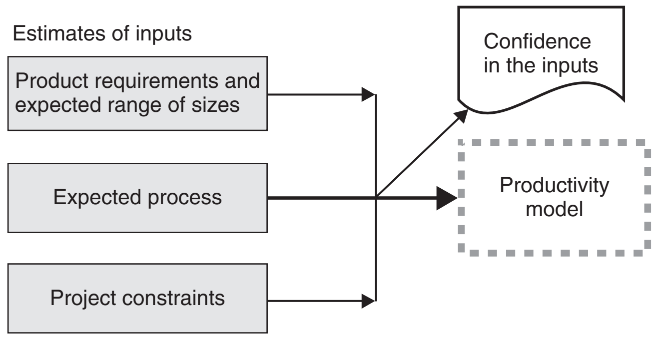

Figure 1.10 Phase A: collection of the Inputs for the Estimation Process

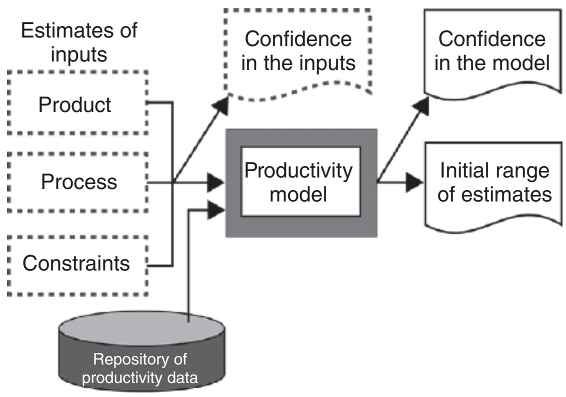

Figure 1.11 Phase B: Execution of the Productivity Model in an Estimation Process

Phase (B) Execution of the productivity model – see Figure 1.11

Execution of the productivity model in an estimation context typically involves two steps:

1. Use of the productivity model (as a kind of simulation model), usually considering only the estimated values of the inputs (and not their ranges of uncertainty).
   - **a.** The productivity model equation will produce a theoretical single estimate on the line represented by the equation.
   - **b.** The information on the performance of the productivity model is used to identify the expected range of variations (based on the historical data used to build the model).
2. Use of the information about the uncertainty and candidate ranges of variation of the estimated inputs to adjust the estimated ranges of the output of Step 1 above. This will generally increase the expected range of variation of the estimates from the productivity model.

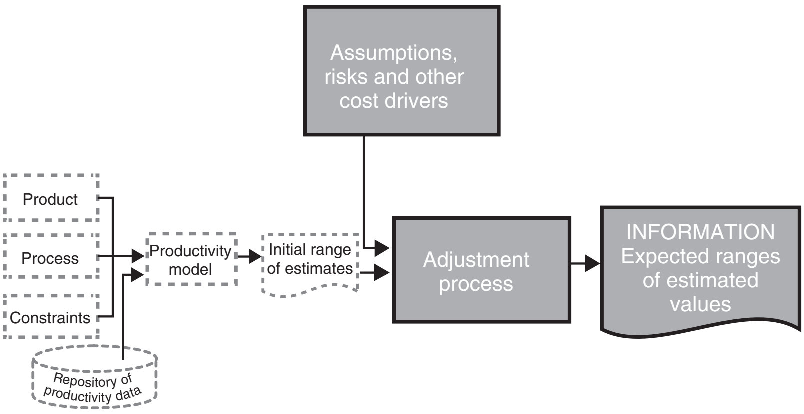

Figure 1.12 Phase C: The Adjustments

Phase (C) The adjustment process – see Figure 1.12

An estimation process is not limited to the blind use of the output of a productivity model:

- On the one hand, the core productivity model typically includes only a limited number of variables, that is, those that are explicitly included as independent variables, in the mathematical equations of such models.
- On the other hand, there are other factors for which there might not have been historical data, as well as a whole set of risk factors that might impact the project over its life cycle (often, many of these factors can be described in a mostly quantitative manner).
  - ▶ Software estimators have to identify such factors, as they may impact the project and need to be considered in an adjustment process.

An adjustment process will take into account variables and information not included yet in the estimation process, including:

- ▶ identification of other cost drivers (i.e., those not included in the productivity model),
- ▶ identification of uncertainty elements,
- ▶ identification of risks and probabilities of occurrence, and
- ▶ identification of key project assumptions.

Note that this process is usually performed on the basis of expert judgment, and would normally affect not only the theoretical estimation of the productivity model, but also its upper and lower limits of estimation, and could provide qualitative information, such as:

- an optimistic estimate (a lowest cost or duration),
- a most likely estimate (with a low probability of occurrence), and
- a pessimist estimate (the highest expected cost or duration).

The output of the estimation process is therefore a set of values, that is, *a set of information* which will be used in the next phase for budgeting and project resource allocation.

Phase (D) The budget decision – see Figure 1.13

The next phase in the estimation process involves selecting a specific value or set of values (on effort and duration) from the ranges proposed and to allocate it to the project, and is the phase in which a decision is made on a project budget.

Of course, the selection of a specific value, often incorrectly referred to as an "estimate," will depend on the strategies of the business manager (i.e., the decision-maker):

- The risk-averter will select a value in the upper range (i.e., a pessimistic scenario).
- The risk-taker will select a value in the lower range (i.e., an optimistic scenario).
- The middle-of-the-road manager will analyze the ranges and their probabilities, and then select a project budget, at the same time setting aside a contingency fund to take into account the probability that he might have selected too low a value.
  - The management of this contingency fund is usually handled at the portfolio level – see also Chapter 3).

Decisions on a specific project budget (incorrectly referred to in practice as a "project estimate") should not be based only on the results of a productivity model.

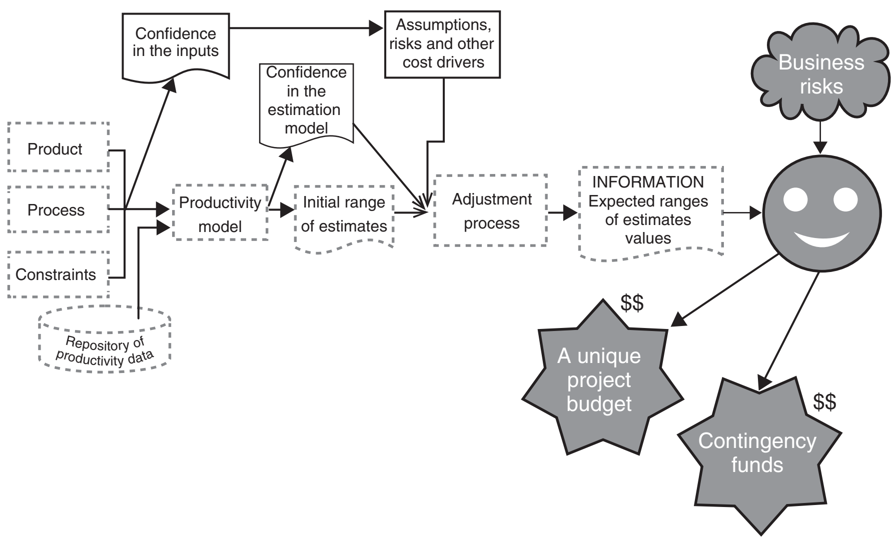

Figure 1.13 Phase D: Budgeting Decision

- The final result of the estimation process cannot be more reliable than the reliability of each subprocess and 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 outcome of an estimation process.

Additional concepts related to estimating and budgeting are discussed in Section 1.7.

Phase (E) The re-estimation process – see Figure 1.14
Since uncertainty is inherent to the estimation process, projects must be monitored to verify whether or not they are progressing as planned, with respect to budget, schedule, and expected quality. Whenever major departures from planning materialize, project re-estimation must occur [Fairley 2009; Miranda and Abran 2008]. This is discussed in greater detail in Chapters 3 and 13.

Phase (F) Estimation Process Improvements – see Figures 1.15 and 1.16
At the project level, the direct responsibilities of the project managers cover the five estimation phases described, at which point they move on to the next project.

There is an additional phase typically undertaken at the organizational level, and not at the project level, which involves analyzing the performance of the estimation process itself with respect to the initial estimates once the projects have been completed, and to improve the various phases of the estimation process, from Phase A to Phase E. This we refer to as Phase F: estimation process improvements – see Figure 1.15 for the positioning of this phase and Figure 1.16 for a summary illustration of what this phase includes in terms of inputs and outputs.

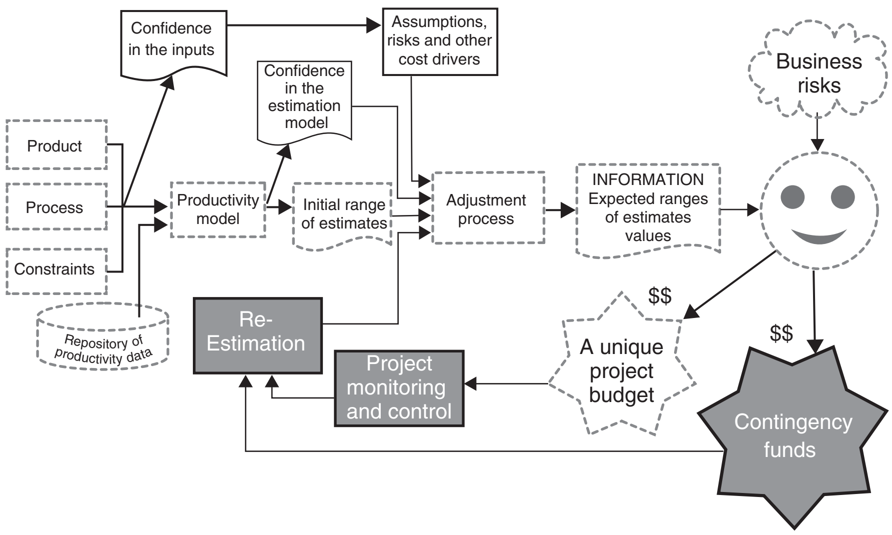

Figure 1.14 Phase E: Re-Estimation

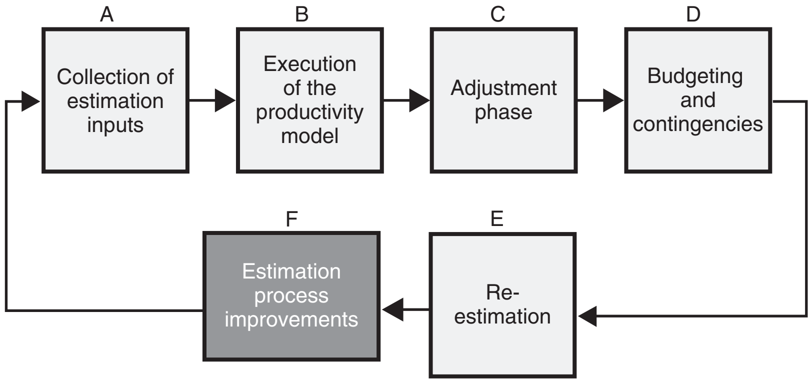

Figure 1.15 Feedback Loop of the Estimation Process

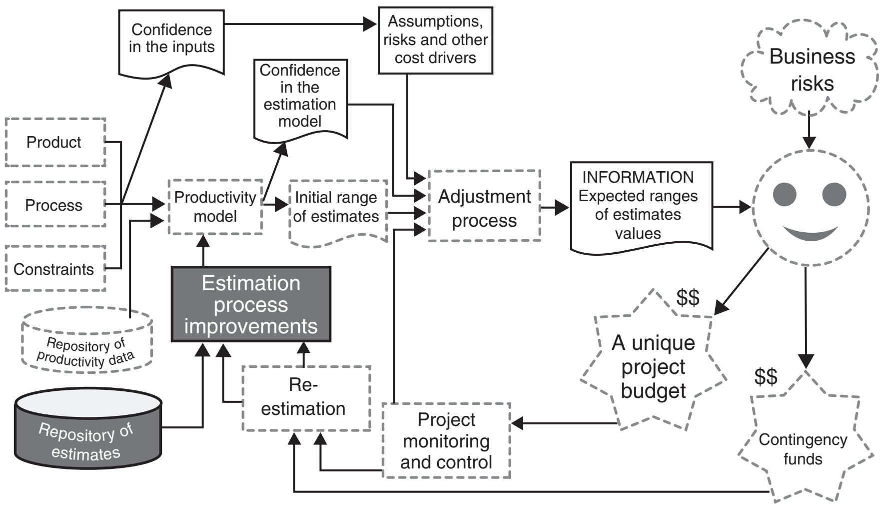

Figure 1.16 Phase F: Estimation Process Improvements

1.7 BUDGETING AND ESTIMATING: ROLES AND RESPONSIBILITIES

1.7.1 Project Budgeting: Levels of Responsibility

The technical part of the estimation process generally leads to a number of scenarios, probabilities, and "estimates."

At this point, a decision must be taken about a specific value, which is commonly referred to as the "project budget" or "project price" in a fixed price management mode.

- The project budget is a single value selected from a large range of candidate values identified by the software estimator!
- This internal project budget, selected by upper management, is then given as the "target" to the project manager (and his team).
- The external project price is determined by upper management, and provided to the customer within a business context, and may be expressed in terms of "time and materials" or as a "fixed price," for instance.

Single-Point Software Project Estimates = A Poor Estimation Culture

Currently, practitioners and managers in the software community provide a "single-point estimate."

However, this practice constitutes a widespread misuse of the concept of estimation, the object of which is to provide a plausible range of values (from a minimum to a maximum, and all the intermediate values – each with a relatively low probability of occurrence), and is the responsibility of the estimator – more on this in Chapters 2 and 3.

Another misuse of the concept of estimation is its improper association with the selection of a specific budget value (which is the role of the manager – see Sections 1.7.2 and 1.7.3), while at the same time risk-taking and the setting aside of contingency funds are addressed at a higher administrative level than that of project manager – more on this in Chapter 3.

Of course, a budget value has a greater probability of being respected than any specific estimate within a range of estimates, mainly because compromises are worked out during the project life cycle, such as reducing the number of functions delivered or lowering quality by skipping some reviews or tests.

Even though a budget value is a single figure, it combines several concepts; for instance, the estimated cost and effort involved in producing a number of deliverables at specified levels of quality at a specified time. The fact that, at the end of the project, the actual cost and effort equal the amounts budgeted does not confirm that the estimates were correct. What is not considered is the possibility that a large number of the required functions might have been delayed to future project phases, and that a great many quality issues might have been shifted to the maintenance category, increasing those costs down the road.

The selection (and allocation) strategy of a project budget will depend on the management culture of the organization and on the industrial context.

(A) *An overly optimistic culture* (or aggressively commercial culture)
In many cases, the basis for the selection of a project budget is the "price to win" (i.e., quotation of the lowest possible price, in order to secure project approval), even though the probability of meeting this budget is almost nonexistent.

- An organization might underbid a project (i.e., put forward a less than reasonable budget) and expect to lose money (i.e., the actual costs will be greater than the accepted budget), but with an expectation of larger profits in subsequent projects.

- An organization might underbid a project initially (i.e., put forward a less than reasonable budget at first), but expect to renegotiate the budget (or prices) upward, based on a number of factors with a high degree of probability of occurrence (such as the addition of functions not initially specified in the bidding process, and at a fairly high cost).

(B) *A very conservative culture*
In a governmental organization with a large number of decision-making levels and long delays for approval, management might present a budget that includes a large number of contingencies in order to avoid going back through the command chain should they have underbudgeted at some point. This would happen, for instance, in organizations operating in a noncompetitive environment (such as a commercial monopoly or government agency).

(C) *Any culture between these extremes!*

1.7.2 The Estimator

The role (and responsibility) of the software estimator in the software project estimation process is to

(A) Build the productivity model(s). This includes collecting data from past projects, building explicit models of relationships across dependent and independent variables, and documenting the quality of productivity models.

- ▸ When the organization does not have data from its past projects, the estimators must find alternative solutions (such as accessing industry data, or gaining access to commercially available estimation tools, and analyzing their performance).

(B) Carry out Phases A–C of the estimation process described in Figures 1.9–1.12, which consists of

- ▸ collecting data for the project being estimated and documenting them,
- ▸ feeding these data into the quantitative productivity models as input and documenting the expected ranges of solutions,
- ▸ carrying out the adjustment process described in Figure 1.12, and
- ▸ providing this information to decision-makers.

1.7.3 The Manager (Decision-Taker and Overseer)

The manager's responsibility is to take risks and expect to be held accountable for managing them, while minimizing these risks by obtaining as much information as possible with the resources available.

The manager has then to make an informed decision by selecting the "optimal" budget for a specific project in a specific context:

- from the ranges of values provided by the productivity model and the corresponding estimation process

and by committing to a single estimate for a project. This responsibility

- is not in any way an engineering responsibility and
- is clearly a management responsibility.

When a manager forces his technical staff to commit to a single estimate, he is transferring what should be his responsibility to them. This is the inherent risk of decision-making in a context of uncertainty and risk:

- The manager is making the estimator accountable for a decision that he should be taking himself, based on the information that the estimator will have provided.

When a software staffer commits to a single estimate, he is overstepping both his domain of expertise and his domains of responsibility:

- He is acting like a manager, becoming accountable for the risks taken, and he is not being adequately paid for these management duties!

In practice, the business estimation process is much broader than the estimation process, and is not restricted to either a single project or to the software project perspective.

- The outcome of a previous software estimation subprocess cannot not be the only contributor to the decision-making process.

From an organizational perspective, the portfolio of all projects must be taken into account, and, before making a decision on a specific project, managers must also take into account

- the estimated costs,
- the estimated benefits, and
- the estimated risks of all projects.

Decisions on individual projects must be made in the context of a strategy that optimizes the corporate outcome, while minimizing the risks across all projects.

The manager's (i.e., the decision-taker's) additional responsibilities are the following:

- Implementing an estimation process (such as the one described in this chapter), which includes:
  - allocating resources for data collection and data analysis for building the initial productivity model,

  - allocating resources for integrating this productivity model into the design of the full estimation process,
  - allocating resources for training on the use of the full estimation process, and
- assigning skilled and trained resources to the estimation process whenever an estimate is required for a specific project.

> *Example of a High-Risk Project*
>
> In a high-risk project situation, with the potential for major benefits, decision-makers will want to provide for contingency funding to ensure project completion, in case the project goes over budget.
>
> Such contingency funding might not be communicated to project management.

This is discussed further in Chapter 2.

1.8 PRICING STRATEGIES

In addition to the estimating and budgeting practices and concepts described in the previous sections, a number of additional practices are referred to (incorrectly) as "estimating" techniques, such as the "win market share" the so-called "estimation technique" – see box below.

> *Example of a Pricing Strategy: Win Market Share*
>
> To win market share, a business decision may be made to underbid for a project by presenting the customer with a "project budget" that may be considerably lower than the expected project cost.
>
> Such a market strategy may hide two other business sub-strategies:
>
> **(A)** The potential for loss is recognized ahead of time to support the long-term customer relationship, in the form of later and more lucrative projects.
>
> **(B)** The supplier has realized that, in the course of his project, he has additional ways to increase project costs to recover from the underbid estimates.
>
> This can lead to a situation where perfectly valid ranges of technical estimates are ignored in order to become aligned with business strategies, resulting in project budgets that are unrealistic and unachievable.

1.8.1 Customers-Suppliers: The Risk Transfer Game in Estimation

Almost any customer of a software project is ideally looking for a project at a fixed cost and guaranteed to be *on time* and *on budget*, while implicitly expecting that all the quality targets will be met as well – if not exceeded.

In practice, except in highly competitive markets and in the presence of a great deal of freely available information on the economic factors, this does not happen often because there is an asymmetry of information between customers and producers.

Two generic pricing modes are observed in the software development industry – with a number of variations:

(A) *Time and materials billing mode*
Under this economic pricing model, the customer pays for the effort expended on his project by the software development staff, at an agreed price per staff member throughout the development life cycle. This means that, even though a budget may be allocated ahead of time by the supplier, that supplier is not bound contractually to deliver the described software functions within this budget, by this deadline, and with those quality levels. The supplier is bound to best practices but not to unknown budget figures. In this case, it is the customer who takes on all the budget-related risks. Therefore, providing for overbudgeting is entirely the responsibility of the customer: the customer is basically taking all the business risks.

(B) *Fixed price contract*
Under this economic pricing model, it is the supplier that is legally bound to deliver all of the functionality within the specified budget, deadlines, and quality levels. In such a model, the suppliers is taking all the risks, and correspondingly should have included within the contract, upfront within the agreed price, high contingencies to handle such risks. In such a context, the customer theoretically transfers all the risks to the provider, at a cost of course.

In a context where the economic information between customers and producers is well balanced, the risks across both modes are well managed, but in practice this is not often the case in software development.

1.9 SUMMARY – ESTIMATING PROCESS, ROLES, AND RESPONSIBILITIES

Estimating a fixed effort budget with a fixed duration *accurately* and early on in the budgeting process is not feasible from an engineering perspective:

- The software inputs to the productivity models are far from dependable, and may vary considerably over the project life cycle.
- The available productivity models, built with information from projects completed, are not sophisticated enough to provide a high degree of explanatory power with few independent variables.
- There is, most of the time and in most software organizations, no well-structured feedback loop to improve the foundations of the estimation process.
- Software technology itself is continuously changing, resulting in some of the historical foundations of the productivity models becoming outdated.

Notwithstanding all the above,

- many users still insist that software projects be priced at a fixed cost and be guaranteed to be completed on time and on budget

and

- many project managers commit to completing software projects at a fixed cost and guaranteeing that they will be completed on time and on budget!

This illustrates that beyond the estimation process, there is a business estimation process, distinct from the engineering estimation process.

Business objectives, practices, and policies must also be taken into account when making business decisions.

- Consequently, there are often major differences between the sets of engineering-based estimates and those of the business estimates.

From a corporate perspective, the following two types of estimates should be identified and managed separately:

- engineering estimates and
- business estimates.

This would clarify the decision-making responsibilities and, over time, facilitate improvements to full estimation process.

From an engineering perspective, the software estimation process:

- should not replace the business estimation process

but

- should be a contributor to the full extent of its specialized expertise in terms of providing decision-makers with their professional engineering advice on the estimation of project costs, project uncertainties, and project risks.

This chapter has presented the components that must be implemented to develop a strategy for a *credible and auditable* estimation process.

Key Lessons Learned

In this chapter, we have discussed the fact that the goal of an estimation process should not be to provide a single hard figure, but rather to provide

- information about ranges of plausible values,
- feedback about how good this information is,
- limitations of the information used as input to the estimation process,
- limitations of the information provided as output of the estimation process, and
- analysis and mitigation of risks by documenting the assumptions made about the inputs, and the use of these inputs, in the estimation process.

The realistic expectations of an estimation process must be clarified, as well as what constitutes

- the technical responsibility from an engineering perspective (i.e., the provision of information based on a rigorous process) and
- the managerial responsibility of making a decision on a single-project estimate (from the array of information provided by the productivity model and its context of use for a particular project).

EXERCISES

1. If you do not have quantitative information on your organization's performance in software project delivery, can you expect to have good estimates for the next project? Explain your answer.
2. What are the two broad approaches to software estimation, and what are their differences?
3. Identify some of the *worst* practices with regard to *inputting* to an estimation process.
4. Identify some of the *best* practices with regard to *inputting* to an estimation process.
5. Identify some of the *poor* practices in handling the *outputs* of an estimation process.
6. What do industry surveys tell us about the performance of software projects in meeting their budget and deadlines?
7. What is the difference between a "productivity model" and an "estimation process"?
8. If you know the accuracy of a productivity model, what is the expected accuracy of its use in an *estimation* context?
9. How can you design a productivity model?
10. How do you evaluate the performance of a productivity model?
11. What are the benefits of mathematical productivity models?
12. For estimation, how would you handle *cost drivers* that are not included in the productivity model?
13. For estimation, how would you handle *risk factors* that are not included in the productivity model?

14. How can an organization take into account *potential scope changes* when using its *productivity model* in an *estimation* context?

15. Discuss the key differences between providing an estimate for a project and taking a decision on a project budget. Discuss roles and responsibilities in estimation.

16. What are some of the key characteristics of estimation? Taking into account these key characteristics, what can you deliver when an organization expects accurate estimates from you? Provide your management with a better definition of "accuracy" in this context.

17. When a manager selects a project budget from a range of estimates, what other major decision should he/she take concurrently?

18. How can an organization take into account *actual scope changes* in its *estimation process*?

19. Why should an organization have not only a plain estimation model, but also a re-estimation model?

TERM ASSIGNMENTS

1. Document the estimation process in your organization.

2. Compare the performance of your projects with that documented in industry surveys, such as the Standish Group Chaos Report.

3. Compare the estimation process in your organization with those illustrated in Figures 1.2 and 1.15. Identify improvement priorities for your organization's estimation process.

4. Propose an action plan to address the top three priorities for improving an organization's software estimation process.

5. Compare the estimation process in Figure 1.15 with an estimation model *proposed in a book*. Comment on the similarities and differences. Identify strengths and weaknesses in the productivity model analyzed.

6. Take an estimation model *proposed by a vendor* and compare it to the estimation process in Figure 1.15. Comment on the similarities and differences. Identify strengths and weaknesses in the productivity model analyzed.

7. Take an estimation model that is *available free on the Web* and compare it to the estimation process in Figure 1.15. Comment on the similarities and differences. Identify strengths and weaknesses in the productivity model analyzed.