

Comparative evaluation of software development strategies based on Net Present Value

Hakan Erdogmus

**Software Engineering Group
National Research Council of Canada**

Montreal Road, Bldg. M-50
Ottawa, Ontario, Canada K1A 0R6
Hakan.Erdogmus@nrc.ca

March 19, 1999¹

Abstract -- Assessment of alternative development strategies in a software project can be difficult due to the interactions among the multiple factors involved. A rigorous, value-based approach allows a more objective comparison of the available alternatives. For example, such an approach can steer the analysis of the economic incentive to choose a strategy that promises rapid development over a strategy that promises high earnings. The corporate finance concept of Net Present Value is used in this context to define a hierarchy of comparison metrics based on six high-level variables. The top-level metric measures relative economic incentive. The analysis focuses on the impact of product risk and rapid development on the decision to select the most valuable strategy.

Keywords – *software engineering economics, software project valuation, software investment analysis, software cost estimation, software development strategies, COTS software, Net Present Value, investment analysis techniques, systematic risk*

Introduction

Decision making in software projects is hard. In evaluating a software development strategy for a new project, designers and managers must consider several interacting factors. These factors include not only cost and schedule, but also flexibility, complexity, skills, existing resources, current practices, risk, market conditions, competitive advantage, compliance to standards, quality control, and so on. In considering these factors, decision makers often rely on insight and experience. In a strictly business context, the governing objective of software development is wealth generation for the company (Favaro and Pfeleeger, 1998; McTaggart et al., 1994). According to this bottom line, the best development strategy for a new software project would be the one that generates the most value. Therefore a facet of decision making must be concerned with economic value, and view the factors involved from the perspective of additional value generated. Although insight is irreplaceable in decision making, it needs to be supported by a more rigorous, value-based approach.

Decision making becomes more complicated in the presence of multiple alternatives. Some of the commonly occurring questions that arise regarding the choice of the development strategy are: Should the system be custom built from scratch, or should it be based on Commercial Off-the-Shelf, or COTS, software components (Carney, 1997)? Is it worth investing in a flexible architecture tailored to accept components (Clements, 1995)? Is it worth investing in formal methods (Raltson and Gerhart, 1991)? In reuse (Favaro et al., 1998)? In the face of such questions, a careful and systematic comparison of the alternatives is crucial (Clemons, 1991). A development strategy that is most economical for a software project in a specific business context and under specific market conditions may not be the best choice for another project, or even for the same project in a different business context or under marginally different market conditions.

Net Present Value (NPV) is the main financial tool for the valuation of capital budgeting projects (Ross et al., 1996). In the context of software engineering, it can be used, in conjunction with other techniques such as cost estimation (Boehm et al., 1995), to predict economic value as a basis of strategic decision making

¹ Revised April 10, 1999

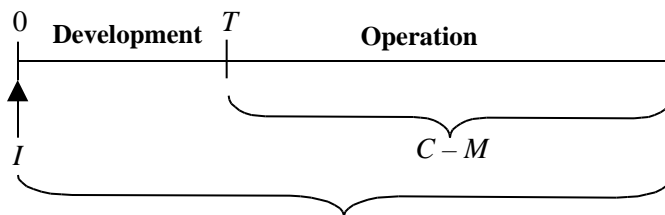
(Favaro et al., 1998; Sullivan et al., 1998; Favaro, 1996; Boehm, 1981). However, in the face of multiple alternatives and uncertainty, the simple “choose the alternative with the highest NPV” rule is not very helpful. The need to understand how the underlying forces affect the NPV decision calls for a deeper analysis.

We address this need through a metrics-oriented approach, where a software project is cast as a capital budgeting decision. Given two alternative development strategies, the NPV of the project is expressed in terms of six high-level variables. The choice of the development strategy affects the estimates of these variables. To evaluate the alternatives, a set of comparison metrics are defined. The comparison metrics are organized in a hierarchical manner, making multi-level, compositional analysis possible. At the same time, a taxonomy is provided for easy and uniform interpretation of these metrics. The metrics are classified into *advantage* metrics, *premium* metrics, and *incentive* metrics. The top-level metric measures the economic *incentive* to favor one of the two strategies over the other. This organization makes it easier to analyze the sensitivity of the higher-level metrics to variations in different variables and lower-level metrics.

An analysis of the topmost metric confirms the importance of rapid development as an economic argument for a strategy that promises it. The analysis also reveals that product risk affects the economic incentive to choose one strategy over another in different ways under different circumstances. We discuss some of these findings in the discussion section.

Determinants of Economic Value in a Software Project

The choice of the development strategy may directly or indirectly affect intrinsic determinants of economic value in a software project. We separate a software project into *development* and *operation* phases, as the following timeline diagram illustrates:



The diagram identifies five of the six high-level variables that affect economic value:

Development time (T), or *time to market*, is defined as the elapsed time between the commitment to invest in the project and the time of the first major cash inflow from revenues or cost savings.

Development cost (I) is the total present value at time 0 of all cash outflows from the time the decision to invest is made to the time of the first major cash inflow from revenues or cost savings (time T).

(Future) Asset value (C) is the total present value *at time T* of the cash inflows that the project is expected to generate during its lifetime. Asset value mainly consists of future revenues from sales, licenses, and royalties, and direct cost savings from using the end product.

Operation cost (M) is the total present value *at time T* of all cash outflows of the operation phase. This amount consists mainly of regular maintenance costs and planned future investment outlays.

Flexibility value () measures the contribution, under uncertainty, of implicit strategic flexibility to the project’s total value. For example, the ability to delay the commitment of certain resources, to change the maintenance schedule, to add and replace software components in order to leverage on third-party innovations, as well as growth opportunities cast as follow-on projects are all forms of flexibility that generate additional value for a project. We will disregard this variable in the discussion to follow.

Product risk (d) is the sixth variable considered. It refers to the systematic risk (Ross et al., 1996) inherent in the end software product. This captures effectively the *opportunity cost* of undertaking the project. Since

systematic risk is similar for similar types of assets, the development strategy used does not affect product risk as long as the concept, goals, and requirements of the end product remain the same. This contrasts with the other five variables, which may be affected by the development strategy chosen. Product risk does not cover unsystematic risk associated with the choice of the development strategy.

Net Present Value of a Software Project

Based on the six high-level variables identified, the *Net Present Value*, or NPV, of a software development project can be represented by the equation:

$$NPV = (C - M)/(1 + d)^T - I +$$

Here the quantity $C - M$ will be referred to as the *Net Asset Value (NAV)*, which we assume to be positive.

The interpretation of the NPV equation is straightforward: the present value of the project is calculated by discounting the net asset value back to time 0 using a discount rate d ; the cost of development is deducted from the result; and finally the flexibility value is added to it. The discount rate represents the opportunity cost (or cost of capital) of projects bearing a systematic risk similar to that of the project under evaluation: an investment worth a dollar today is worth $(1+d)^T$ dollars in T periods, the value being compounded at a per-period rate of $d \cdot 100\%$.

Since product risk is implicit in the discount rate d , we use the same value of d as a surrogate for product risk across all strategies for the same project.

A Taxonomy of Comparison Metrics

Given a software project and two alternative development strategies, let V be a variable on which the NPV of the project depends. Let V_A and V_B denote the value of variable V for strategy A and B, respectively. A *comparison metric* is simply a function of V_A and V_B . For a specific value of a comparison metric m , strategy A is said to be *favorable* over B with respect to m , if for that value of the metric the project NPV for strategy A is superior to the project NPV for strategy B when everything else is equal (all the other variables on which NPV depend are equal for both strategies).

To facilitate interpretation, we pick one of the two strategies, say B, as a reference point and call it the *base strategy*. The other strategy, A, which is compared against the base strategy, is called the *test strategy*. The test strategy is usually the one that is under scrutiny. Each metric m is defined such that $m > 0$ implies the test strategy (A) is favorable over the base strategy (B) with respect to m ; $m < 0$ implies the base strategy (B) is favorable over the test strategy (A) with respect to m ; and as m increases the test strategy (A) becomes increasingly more favorable over the base strategy (B) with respect to m .

We define three types of such metrics: an *advantage* is the logarithm of the ratio of two quantities; a *premium* is the absolute difference of two quantities; and an *incentive* is the normalized difference of two quantities. A negative advantage is referred to as a *disadvantage*; a negative premium as a *penalty*; and a negative incentive as a *disincentive*. We use logarithmic scale in advantage metrics for mathematical convenience and ease of interpretation. The favorable regions of the two strategies with respect to such a metric are symmetrically centered at the origin.

Premiums are often normalized with respect to a (positive) quantity that measures *scale*, giving rise to an incentive metric. Incentive metrics are preferable to premium metrics since they allow comparison across projects of variable scale. The base strategy (B) is used to define the measure of scale.

Comparison metrics are always defined *relative to* the test strategy. Hence, an advantage (premium, or incentive) is implicitly interpreted as an advantage (premium, or incentive) *for* the test strategy.

Measuring Economic Incentive through Net Present Value

For a given project with two alternative strategies, the economic incentive to choose one strategy over the other increases as the gap between the two NPVs widens. Thus the most obvious way to measure the economic incentive would be through the NPV premium, or the difference between the NPVs of the two strategies being compared. This approach works well when a project is considered in isolation, however, it breaks down when comparing incentives across projects of different scale.

To see why, consider two projects P and Q. Suppose NPV premium for choosing the test strategy is expected to be \$100K for project P, and \$25K for project Q. Also, project P has a much larger scale at an estimated volume of \$1,000K than project Q whose estimated volume is only \$50K. Is the incentive to choose the test strategy higher in project P or Q? It is tempting to think that the incentive in project P is higher since choosing the test strategy is expected to increase the wealth of the firm by a larger net amount. However, when project scale is taken into account, at an equal margin of error, the test strategy is a more likely winner over the base strategy in project Q than it is in project P: project Q's NPV premium is a whopping 50% of the project scale; whereas project P's is a mere 10%.

For this reason, we normalize NPV premium using a measure of project scale. The resulting metric, *Net Present Value Incentive* or *NPVI*, is defined as follows:

$$NPVI = \frac{\text{NPV Premium}}{\text{Total Project Scale}} = \frac{NPV_A - NPV_B}{NAV_B + I_B} = \frac{NPV_A - NPV_B}{C_B - M_B + I_B}$$

NPVI effectively measures the economic incentive to choose the test strategy (A) over the base strategy (B) for a given project. The positive quantity *Total Project Scale (TPS)* is the sum of the net asset value (NAV) and the development cost (*I*) of the base strategy.

The breakdown of *NPVI* into less complex metrics results in the hierarchy illustrated in the Appendix. Lower levels of the hierarchy are occupied by *advantage* metrics, each of which is based on a variable of the NPV equation. Of particular interest is *Development Time Advantage (DTA)*, defined as the logarithm of the ratio of T_A to T_B . The others, *Net Asset Value Advantage (NAVA)*, *Operation Cost Advantage (OCA)*, and *Development Cost Advantage (DCA)* are defined in a similar fashion based on the variables, *C*, *M*, and *I*, respectively.

At the bottommost layer, the *PCA* (Product Cost Advantage) metric captures the expected relative contribution of direct product cost savings to the asset value of the test strategy. For example, if the test strategy uses a higher ratio of COTS components than the based strategy, this metric is likely to be negative, representing a disadvantage, due to royalties and licensing fees to be paid for these components. The *QFA* (Quality/Functionality Advantage) metric captures the expected relative contribution to the asset value of the test strategy of the ability to control the quality and functionality of the end system. Similarly, if the test strategy uses a higher ratio of COTS components than the based strategy, this metric is likely to be negative, representing a disadvantage due to the potential negative impact on product acceptance of extra, missing, or undesirable functionality, as well as of inferior performance or reliability. The *TVA* (Termination Value Advantage) metric captures the expected relative contribution of termination value to the asset value of the test strategy. Termination value includes the value of reusable software salvaged upon project termination. Thus if the test strategy uses a higher ratio of COTS components than the based strategy, *TVA* is likely to be positive. The *EEA* (Early Entry Advantage) metric is discussed later.

Impact of Development Time Advantage and Product Risk

An analysis of NPV incentive with respect to development time advantage (*DTA*) confirms the importance of rapid development. It also provides valuable insight into the impact of product risk on the comparison.

Product risk and rapid development work towards the test strategy when this strategy has operation cost advantage, and against it when it has asset value advantage over the base strategy. In the latter case, the

benefit of a higher asset value is partially cancelled by the *time value of money* (as measured by d) associated with the product risk. An interesting case to note is where the test strategy has marginal to significant net asset value disadvantage ($-1.39 < NAVA < -0.01$; or test strategy's NAV at 25% to 99% of that of the base strategy), as shown in Figure 1. With significant net asset value disadvantage, increasing product risk consistently increases the incentive for the test strategy. This effect becomes exceedingly more pronounced as the test strategy's development time advantage increases. With marginal net asset value disadvantage, the same observation holds only up to a certain product risk level. This is most noticeable when the test strategy has moderate development time advantage. After this critical level, product risk starts working against the test strategy, although $NPVI$ may remain considerably high.

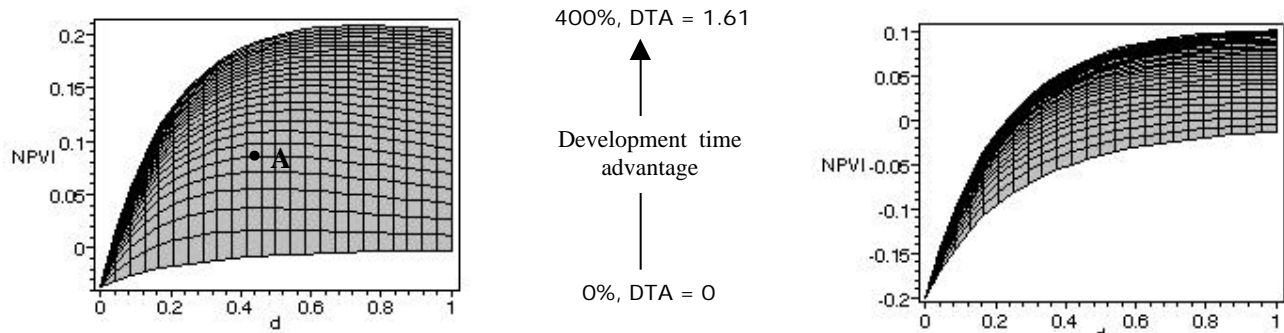


Figure 1 Impact of product risk and development time on $NPVI$ when the test strategy has marginal net asset value disadvantage (left; at $NAVA = -0.11$, or 90%) and significant net asset value disadvantage (right; at $NAVA = -0.69$, or 50%). Development costs are equal. Each curve represents a fixed value of development time advantage. Point A represents maximum $NPVI$ for a moderate development time advantage. For higher values of product risk, $NPVI$ gradually drops for that value of development time advantage.

Factoring in Early Market Entry

The impact of rapid development on $NPVI$ increases under a *market-entry reward model*. The simplest model assumes an already ripe market for the end product, and maximum reward is achieved through immediate entry. The *Early Entry Advantage (EEA)* metric is an expression of this reward for the test strategy in terms of its contribution to the test strategy's asset value advantage. For example, if as a result of early market entry, the asset value of the test strategy (C_A) is expected to be 20% higher had the test strategy not have this advantage relative to the base strategy, then the value of the EEA metric equals $-\log[1-0.2] = 0.22$. The reward (value of EEA) increases as the development time advantage of the test strategy (DTA) increases, as shown in Figure 2, approaching to its maximum value ($MEEA$) as DTA approaches infinity. Under this model, increasing the value of the DTA metric increases $NPVI$ at a faster rate.

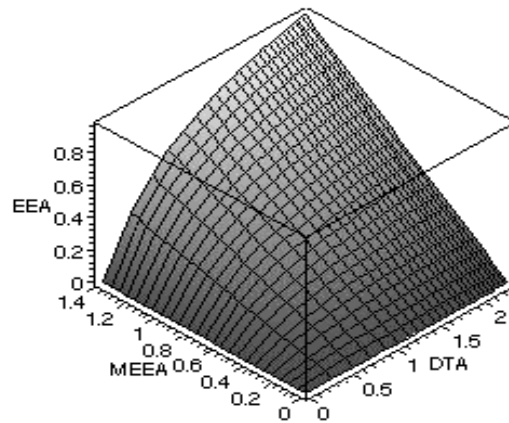


Figure 2 Early Entry Advantage: combined effect of Maximum EEA ($MEEA$) and Development Time Advantage (DTA) on EEA , as modeled by the equation $EEA = MEEA \cdot (1 - 1/(1 + DTA))$.

Discussion

A metrics-oriented, comparative valuation approach supports decision making in the conception phase of a software development project. With such an approach, it is possible to weigh the costs and benefits of alternative development strategies in a more systematic way. A metric based on Net Present Value measures the economic incentive to favor one strategy over another in terms of six underlying variables. Although it is possible to define economic incentive using other investment analysis measures such as *return on investment* (Violino, 1997) or *internal rate of return* (Favaro, 1996), NPV remains the most acceptable and objective criterion.

The analysis of the incentive metric reveals the economic basis for advocating rapid development, high product risk, and early-market-entry reward as a winning combination for a strategy under evaluation even when the net asset value comparison is unfavorable. This argument can be used in favor of COTS-centric development, since one of the main drivers for using COTS components in software development is rapid development (Carney, 1997). Under such circumstances, underestimating product risk is safer than overestimating it, as it is likely to make the strategy under evaluation look less attractive than it actually is relative to its alternative. This observation confirms, from a different perspective, Clemons' (1991) warning against using a high discount rate as a surrogate for dealing with uncertainty.

We have not considered the close relationship between development time and development cost. The COCOMO schedule estimator identifies this relationship as $T = k I^c$ (Boehm et al, 1995; Boehm, 1981), where k is constant, and the value of c depends on the attributes of the software being developed. However, it is not clear whether this model applies to more recent paradigms, for example, when the development is centered on COTS components². In general, that a strategy with development time advantage is also likely to cost less to develop magnifies the impact of rapid development on the incentive metric.

We have deliberately omitted flexibility value in this discussion due to lack of space. Embedded strategic flexibility can be formulated and valued as a portfolio of *real options* (Luehrman, 1998). Sullivan et al. (1998) and Favaro, et al. (1998) illustrate some applications of this approach to software engineering.

Finally, the use of economic valuation techniques in software decision making has its limitations. Cost, cash flow, schedule, and market estimation remains a difficult and imprecise task. Nevertheless, economic valuation is still a worthy intellectual tool. In particular, rigorous comparative analysis can help justify the choice of the proposed development strategy in a sound manner, and as such, support business case development for the project.

References

- Boehm, B., *Software Engineering Economics*, Prentice Hall, 1981.
- Boehm, B., Horowitz, C.W., Madachy, R. and Selby, R., "Cost Models for Future Software Lifecycle Processes: COCOMO 2.0," *Annals of Software Engineering*, 1995.
- Carney, D., "Assembling large scale systems from COTS components: opportunities, cautions, and complexities," SEI Monographs on Use of Commercial Software in Government Systems, Software Engineering Institute, Pittsburgh, PA, June 1997.
- Clements, P., "From subroutines to subsystems: component-based system development," *The American Programmer*, November 1995.
- Clemons, E.K., "Evaluation of strategic investments in information technology," *Communications of the ACM*, January 1991.

² Work on COCOTS, a specialized submodel of COCOMO 2, is under way. COCOTS has not yet defined a schedule equation that predicts the development time of a COTS-centric system based on the estimate of development effort or cost.. For information on COCOTS, see <http://sunset.usc.edu/COCOMOII/suite.html>.

Favaro, J., "A comparison of approaches to reuse investment analysis," in *Proceedings of the 4th International Conference on Software Reuse*, 1996.

Favaro, J.M., Favaro, K.R., and Favaro, P.F., "Value based software reuse investment," *Annals of Software Engineering*, vol. 5, 1998.

Favaro, J.M. and Pfleeger, S.L., "Making software development investment decisions," *Software Engineering Notes*, 23(5), 1998.

Luehrman, T.A., "Strategy as a portfolio of real options," *Harvard Business Review*, September-October 1998.

MacTaggart, J.M., P.W. Kontes, and M.C. Mankins, *The Value Imperative*, The Free Press, 1994.

Ralston, T.J. and S.L. Gerhart, "Formal methods: history, practice, and prognosis," *American Programmer*, May 1991.

Ross, S.A. et al., *Fundamentals of Corporate Finance*, Irwin, 1996.

Sullivan, K.J. et al., "Software design as an investment activity: a real options perspective," submitted for publication, 1998.

Violino, B., "Measuring value: return on investment," *Information Week*, June 1997.

Appendix: Breakdown of NPV Incentive into lower-level metrics

A: Test strategy
B: Base strategy

