# Economics of Software Reuse Revisited

Ruth Malan*, Kevin Wentzel
Software Technology Laboratory
HPL-93-31
April, 1993

software reuse, cost-benefit analysis, decision support

The field of software reuse is attracting increasing attention from academics and practitioners who recognize the economies to be gained from reducing duplication of software development effort. However, development for reuse requires additional costs that are difficult to justify under a single-project management view. Since reuse benefits accrue over time and across a number of projects, project management decision making must shift to a multi-project, long-term view and this changes management information requirements. In particular, given scarce resources management needs to be able to assess whether software reuse will have the kind of return that warrants the up-front investment in reusability. This paper evaluates and extends the pioneering contributions of other authors in the area of cost-benefit analysis for software reuse. In addition to incorporating timing and risk inherent in future reuse opportunities, the analysis includes reuse costs and benefits that were not accounted for in previous models.

Internal Accession Date Only
*Stanford University, Stanford, California
To be published in the Proceedings of the *3rd Irvine Software Symposium*, University of California, Irvine, California

# 1. Introduction

Software has played a vital role in improving productivity in almost every business activity, from control and automation of manufacturing operations to improved communications and office automation. Demand for software will continue to escalate as it is incorporated in more and more household and business appliances, and medical and measuring instruments become even more sophisticated. At the same time, the demand for software-intensive systems, like application programs, is expected to increase, while the maintenance and evolution of existing systems will continue to absorb a large share of development resources.

The problem is that this demand for software is growing faster than the ability to supply it. So far no single software development productivity enhancing process or technology has proved to be *the* "silver bullet" (Brooks, 1987) that will put an end to the software crisis. Nevertheless, there is mounting pressure to change software development from an individualistic craft to a production process that exploits reusable and interchangeable software parts (Cox, 1990), avoiding duplication of development effort.

Despite the tantalizing promise of productivity gains through systematic software reuse, it has not yet been adopted on a large scale. Practice has shown that many of the reuse inhibitors lie in the realm influenced by management and the decisions they make (Griss et al., 1993). Decisions based on old information systems do not consider the multi-project, long-term benefits of reuse and inhibit reuse through misallocation of resources. When projects are evaluated individually, project managers do not have any incentive, and more likely have a *dis*incentive, to incur increased costs and project delays through making components reusable. They need to know how their choices impact other projects and the organization as a whole.

This paper develops a framework for assessing the costs and benefits of reuse to facilitate informed choices about reuse options. A survey of the literature revealed a number of deficiencies in the current models, and motivated the development of an enhanced model. This model is presented in a series of stages, where each stage expands the factors considered. Thus, an initial model based on development phase costs is expanded to include maintenance phase costs. Finally, other reuse-related benefits and costs are also incorporated in the model. This approach facilitates comparison with the previous models. More importantly, it provides successively tighter lower bounds on the full opportunity cost of not adopting systematic reuse. The cost saved in the development phase provides a very conservative estimate of reuse benefits. Consideration of life-cycle cost saved together with the increased profits from some of the business opportunities opened up by reuse, provides a more comprehensive estimate of the gains from exploiting reuse.

## 2. Literature Review

A number of seminal papers have contributed useful insights into the costs and benefits of reuse. Bollinger and Pfleeger (1990) emphasize that understanding the benefits of reuse requires a long-term multi-project perspective. They support the notion of a baseline project that facilitates cost estimation by concentrating on the expected differences from the baseline rather than the entire project cost. Barnes and Bollinger (1991) reformulate Bollinger and Pfleeger's reuse benefit model and use it to motivate one of the most comprehensive discussions of reuse cost drivers in the current literature. Mayobre (1991) presents a model for estimating the return on investment for reuse components. Gaffney and Cruickshank's (1992) return on investment (ROI) model is a useful analytical tool, but it relies on a number of simplifying assumptions.

There are a number of deficiencies in the current models. In general, the models do not take into account reuse-specific setup and overhead costs. Only Lim (1992) considers the timing and uncertainty of the future reuse instances. The models of Harris (1992) and Lim (1992) extend the state-of-the-art in economics of reuse models in the direction of incorporating quality and time-to-market, respectively. Still, the benefits of improved quality and lower maintenance phase costs, as well as other, less tangible, benefits of reuse were not fully and explicitly considered in the antecedent analyses. As a result, many of the long term benefits of improved productivity and enhanced business opportunities were overlooked.

## 3. Cost-Benefit Analysis Framework

Since reuse involves multiple products evolving through their respective life-cycles, an assessment of the economic impact of a systematic reuse program must incorporate cost and revenue projections that extend beyond that of a single development project. The basic approach that we advocate consists of systematically identifying the factors that contribute to changes in costs and revenues as a consequence of reuse. Estimates of the resulting changes in the profit position are then incorporated into the cost-benefit assessment. This paper outlines the major reuse-related cost factors associated with the development and maintenance phases, and incorporates them into a long-term, multi-product net benefit model. Strategic business advantages are also identified, and the model is expanded to incorporate revenue enhancements that result from reuse.

## 3.1 Development Costs

### 3.1.1 Setup and Overhead Costs

New systems will be needed to support a full-fledged systematic reuse program. For instance, a library classification system and a library registration, retrieval, and update mechanism may be critical to some reuse programs and may need to be developed or purchased. Other setup costs would include designing the organization and software development process to support systematic reuse. Since these are generally non-trivial activities, the level of investment in these support systems should be evaluated in conjunction with the expected benefits over the anticipated life of the system[1].

The ongoing costs of expanding and maintaining the library system, a management support structure to ensure systematic reuse, and training programs, should be assessed as indirect overhead. Additional reuse-specific activities include domain and reusability analyses to identify the components that will most effectively support the consumers'[2] development projects. If the related costs are *not* reflected in the analysis of the net benefits for the reuse program, the investment decision is biased in favor of reuse.

Sunk costs of unsuccessful library searches and other failed reuse attempts are also part of the labor overhead. While generally incurred by the consumer, these costs are attributable to the producer's level of investment. Therefore, if these costs are significant, they should be included in the producer-consumer network-wide costs so that they can be reflected in investment decisions and in the determination of the appropriate library stocking level and library support services.

### 3.1.2 Producer Costs

The reusability of software components depends on a number of factors such as the degree of generality, complexity, and fit to expected use, as well as the quality of the component, and the extent and utility of documentation and accompanying test suites. Further, the component has to be available, and hence must be certified and entered in a library, or broadcast by some other means. Therefore, component producers face additional costs over and above the usual development-cycle costs, and these are estimated to be anywhere from 30% to 200% (Balda and

---

[1] If the producer-consumer network is in a very unstable environment, the risk of not recouping the investment may be too high to justify the up-front outlay.

[2] We adopt the convention of referring to development groups that reuse components to create their products as consumers, and those that create or re-engineer components as producers. A component is any packaged set of work products that is made available for more than one use, and architectures, interface specifications, requirements specifications, high and low level designs, code modules, documentation, and test data are all considered to be potentially reusable work products.

Gustafson, 1990) higher than the cost of producing a component not intended for reuse. When a component is re-engineered, the producer still has to expend effort in understanding the existing component, modifying it to include the desired reusability features, and ensuring that it is of high quality.

### 3.1.3 Consumer Costs

In order to use an existing component, the consumer must first select the component. Selection entails articulation of the component requirements in a suitable form, search and retrieval of the component, understanding of what the component does, and verification that it does indeed fit the purpose. The component may need to be specialized to fit the consumer's current needs. This involves adaptation (with corequisite program understanding and subsequent testing). Lastly, the component must be integrated into the system under development, and tested.

The consumer still has to develop the code that is unique to the project, but saves by not having to re-invent and re-implement the reused work products — as long as the selection, specialization and integration efforts are less than the consumer would have to expend to create the required functionality. In general there is a tradeoff between the producer investment in reusability, and the consumers' cost to incorporate the component. For instance, the producer's cost to design and implement black box[3] components may be higher than that for white box[3] components, but consumers save by not having to understand, modify and test the component in order to reuse it.

### 3.1.4 Development Phase Model

#### Model 1: Basic Development Phase Model

For any software development organization, independent of its orientation with respect to reuse, the development costs are the overhead and labor costs incurred in the various stages of the development cycle. Component producers face additional costs to make component(s) reusable. Consumers of reusable components incur costs of selecting, adapting and integrating the component(s), but save by not having to develop the component(s) anew in each product. The net benefit from reuse is not simply the difference between the total consumer's saving and the producer's cost. Reuse-specific overhead and setup costs, such as the cost of installing and

[3] In black box reuse, the component is used as-is. On the other hand, in white box reuse the component is modified to obtain the desired variant, with the consequence that upgrades to the component also have to be modified if they are to be incorporated in the product, reducing the potential maintenance phase saving.

managing the library, and conducting a domain analysis, should also be subtracted from the consumer savings to give a more accurate representation of the net development cost saving, S. The basic model for S is:

$$S = \left[ \sum_{i=1}^{n} (C_{N_i} - C_{CR_i}) \right] - [C_{PR} + A]$$

where

| | | |
|---|---|---|
| $n$ | = | number of products sharing the reusable components |
| $C_{N_i}$ | = | cost to develop product *i* without reuse |
| $C_{CR_i}$ | = | cost of creating product *i* with reuse |
| $C_{N_i} - C_{CR_i}$ | = | expected consumer cost saved for product *i* |
| $C_{PR}$ | = | expected cost that the producer incurs in producing the reusable components |
| $A$ | = | reuse-specific overhead and setup costs incurred by the family of products |

**Model 2: Taking the Time Value of Money into Account**

When the reuse instances are expected to occur over a longer time horizon, the timing of the cash flows should be taken into account. This is done by incorporating a standard present value analysis into the model. The nomenclature is extended further so that:

| | | |
|---|---|---|
| $i$ | = | interest rate per period |
| $M$ | = | number of periods in the time-horizon under consideration |
| $m_p$ | = | expected number of periods taken to produce the reusable component |
| $s_k$ | = | expected period at which a consumer starts to create product *k* |
| $f_k$ | = | expected period at which a consumer finishes creating product *k* with reuse |
| $f_k^N$ | = | expected period at which a consumer finishes creating product *k* without reuse |
| $A_j$ | = | expected reuse-specific overhead cost in period *j* |
| $C_{N_{k,j}}$ | = | expected Consumer cost to create product *k* in period *j* without reuse |
| $C_{CR_{k,j}}$ | = | expected Consumer cost to use the Reusable component in product *k* in period *j* |
| $C_{PR_j}$ | = | expected cost to produce component incurred in period *j* |

$$\overline{C}_{PR} = \sum_{j=1}^{m_p} \frac{C_{PR_j}}{(1-i)^j}$$ = discounted expected Producer's cost to create the Reusable component(s)

$$\overline{C}_{CR_k} = \sum_{j=s_k}^{f_k} \frac{C_{CR_{k,j}}}{(1-i)^j}$$ = discounted Consumer cost to use the Reusable component(s) in product *k*

$$\overline{C}_{N_k} = \sum_{j=s_k}^{f_k^N} \frac{C_{N_{k,j}}}{(1-i)^j}$$ = discounted total cost of producing product *k* with No reusability

$$\overline{A} = \sum_{j=1}^{M} \frac{A_j}{(1-i)^j}$$ = discounted overhead and setup costs

The discounted net benefit (or net present value) is:

$$S = \left[ \sum_{i=1}^{n} \left( \overline{C}_{N_i} - \overline{C}_{CR_i} \right) \right] - \left[ \overline{C}_{PR} + \overline{A} \right].$$

**Model 3: Taking Uncertainty in Reuse Instances into Account**

The degree of uncertainty about the evolution of a product family tends to increase as the time horizon is stretched. Thus, anticipated reuse opportunities arising from products or upgrades planned for, say, the fifth year of the planning horizon are likely to be much more uncertain than those in the current one-year business plan. To incorporate the uncertainty as to whether the component will indeed be reused, the probability of each reuse instance should be estimated, and the expected consumer savings computed. The model then becomes:

$$S = \left[ \sum_{i=1}^{n} \left( \overline{C}_{N_i} - \overline{C}_{CR_i} \right) p_i \right] - \left[ \overline{C}_{PR} + \overline{A} \right]$$

where $p_i$ = probability that reuse instance *i* will be actualized.

Other aspects of uncertainty may also have a significant impact on the expected net saving. For example, the consumer cost saved depends on whether the consumer takes advantage of black box reuse opportunities (when they exist) or modifies the component. In that case, the probability that a consumer will opt for black box or for white box reuse would need to be assessed in order to estimate the expected consumer cost saving. Considerable uncertainty also surrounds the producer cost and consumer savings estimates. Techniques such as the Delphi method could be used to establish credible probabilities and expected values. Sensitivity analyses should also be conducted to determine the impact of varying input parameters like the expected costs.

## 3.2 Life-cycle Costs

A decade ago, Wegner (1983) pointed out that:

> "The view of maintainability as a form of reusability is novel and important. It captures the idea of reusability in time within a dynamically evolving system. Evolutionary dynamic systems require reusability in time of unchanging parts of the system while other parts of the system evolve."

More recently, researchers have considered the impact of reusability on maintenance but have not explicitly considered maintenance costs in any of the cost-benefit models. This deficiency was noted by Lim (1992), but he did not present an explicit formulation of maintenance cost savings.

By centrally maintaining the reuse components, managing their evolution, and propagating upgrades to new products as well as updated versions of older products, the organization can exploit further opportunities to reduce duplication of effort. Moreover, centralized enhancements to black box components enable a whole platform of derivative products to be produced more quickly and at lower cost.

The maintenance and management of evolving components increases the cost to the producer/maintenance group. Consumers benefit from not having to duplicate corrective and evolutionary maintenance activities, though they do have to incur some cost to incorporate upgraded component(s) into their products. The net cost saving, S, is the net cost saved in the development cycle, plus the net cost saved for each product upgrade. Thus,

$$\dot{S} = \left[ \sum_{i=1}^{n} \left\{ \left[ \overline{C}_{N_i} - \overline{C}_{CR_i} \right] + \sum_{j=1}^{r} \left( \overline{C}_{NU_{i,j}} - \overline{C}_{CRU_{i,j}} \right) \right\} \right] - \left[ \overline{C}_{PR} + \sum_{j=1}^{r} \overline{C}_{PRU_j} + \overline{A} \right]$$

where $\overline{C}_{N_i} - \overline{C}_{CR_i}$ is, as before, the discounted expected consumer cost saved in the development of product *i*, and $\overline{C}_{PR}$ is the discounted expected producer's development phase cost. For each of the *r* upgrades:

$\overline{C}_{NU_{i,j}} - \overline{C}_{CRU_{i,j}}$ = cost to produce upgrade *j* to product *i* without reuse less the consumers cost to integrate the upgraded component(s) into product *i*

= consumer saving on product *i* upgrade *j*

$\overline{C}_{PRU_j}$ = discounted expected producer's cost to create upgrade *j*

The discounted aggregated life-cycle consumer costs saved is

$$\overline{S}_A = \sum_{i=1}^{n} \left\{ \left[ \overline{C}_{N_i} - \overline{C}_{CR_i} \right] + \sum_{j=1}^{r} \left( \overline{C}_{NU_{i,j}} - \overline{C}_{CRU_{i,j}} \right) \right\} \qquad (1)$$

and the discounted life-cycle producer and overhead cost is

$$\overline{C_P} = \overline{C_{PR}} + \sum_{j=1}^{r} \overline{C_{PRU_j}} + \overline{A} \qquad (2)$$

In addition to avoiding duplication of engineering effort, reuse of high quality components reduces the life-cycle cost of quality. Balda and Gustafson (1990) warn that the "cost of software failure can be enormous, and even greater when a failing component is in several products." Reuse does not magically improve the quality of components. However, the extra cost of designing and building a more robust component, and thoroughly testing it, can be amortized over the reuse instances (Malan, 1993b). Moreover, if a component defect is uncovered during test or use of one of the consumers' products, the repair can be propagated to upgrades of all of the products that use the component, and all future uses of the component also benefit. As a result, the number of defects in a reusable component generally decreases the more the component is reused (Gaffney and Durek, 1989, and Biggerstaff, 1991).

Malan (1993b) shows that reuse of high quality work products reduces the expected time to complete development, and enhances the predictability of the process, by reducing the amount of rework and retest. As a result, reuse reduces internal failure and appraisal costs. Therefore, reuse may be viewed as an up-front investment in defect prevention that reduces the total cost of quality in a series of consumer development projects.

## 3.3 Other Reuse Benefits

Thus far, the potential savings in development and maintenance costs from reusing common components in a number of product variants and across product lines, have been considered. While these savings may be sufficient to justify the investment in systematic reuse, a number of managers at Hewlett-Packard (HP), and elsewhere, have initiated reuse programs for strategic business reasons (Wentzel, 1992). Indeed, gains from higher productivity may translate into earlier time-to-market, higher quality, and expanded business opportunities, enhancing profits not just by reducing costs, but also by increasing revenues. Many of these effects are hard to quantify, but if reuse is expected to significantly enhance the company's position then it would be wrong to ignore them. As Kaplan (1986) notes:

> "Although intangible benefits may be difficult to quantify, there is no reason to value them at zero in a capital expenditure analysis. Zero is, after all, no less arbitrary than any other number. Conservative accountants who assign zero values to many intangible benefits prefer being precisely wrong to being vaguely right."

### 3.3.1 Accelerated Time to Market

The ability to get the right product to market quickly without sacrificing quality is becoming the hallmark of competitive advantage in a growing number of industries (Wheelwright and Clark, 1992). Rapid product development is of strategic value because it allows an agile response to competitors' actions (Stalk, 1988), and a timely response to shifts in demand or changes in technology. Increasingly short product life cycles are also adding to the pressure to shorten the time it takes to get from a new product idea to market introduction (Rosenau, 1990). In many industries the price that can be obtained for the product diminishes with time (Ulrich et al., 1991) as competitive products enter the market. On the other hand, being among the first to introduce an innovative product in many cases enables the company to capture some of the more sluggish competitors' market share. Furthermore, earlier introduction brings the revenue stream forward, and receipt of earlier cash flows contributes additional interest revenue (Ulrich et al., 1991).

As a result of these competitive advantages of shortened time to market, one of the common reasons for considering reuse is its potential to reduce development time (Wentzel, 1992). It is particularly attractive if development time can be accelerated without increasing development costs. However, the ability to reduce time to market *and* development costs is a fairly rare feature since acceleration of a project will normally increase development costs (Graves, 1989). Not only do the diseconomies of scale associated with software development team size (immortalized in Brook's "Mythical Man-Month", 1975) mean that project acceleration costs tend to grow non-linearly, but software frequently remains on the critical path for product development. In the case of software reuse, however, the effort reduction in consumer projects may be applied to reduce development time, as long as the reused software is on the critical path for the product development, or the resources freed by reuse can be applied to critical path activities. Moreover, it is quite possible that, over the span of the reuse program, development costs will indeed decrease because the up-front reuse costs are spread across a number of reuse instances.

### 3.3.2 Alternative Use of Freed Scarce Resources

The resources that are freed up when redundancy in development and maintenance is eliminated may be used to improve the product (by adding features or enhancing the quality), shorten development time of the current project or be reallocated to concurrent projects. In both of the latter cases, the scarce software engineering resources become available earlier than they would if reuse was not employed. Thus, the benefit of reuse can extend beyond the current reuse project to concurrent or subsequent projects.

### 3.3.3 New Product Opportunities

Because product development is more nimble and less costly with reuse, developers can consider attacking niche markets with greater product variety, and extending their market by providing custom solutions that may not have been practical without reuse (Wentzel, 1992). Indeed, the Japanese "software factories" arose because there were insufficient software engineering resources in Japan to supply the demand for customized systems (Cusumano, 1991). In the context of modularity of physical products, Ulrich and Tung (1991) make a point that is very germane to software components: in a make-to-order environment, the order lead time for customized systems can be reduced by having standard components on hand, so that only the customized parts of the system have to be developed once the order comes in. Thus, not only can customization be achieved more cost-effectively, but in a more timely manner as well.

Unless the components provide unreplicable competitive advantage over competitors, the components or even libraries (or kits) could themselves be sold as products, extending the number of reuse instances beyond the bounds of an internal producer-consumer network.

### 3.3.4 Product Line Consistency

A number of very large organizations, like HP, are faced with the problem of having to divide themselves up into operating divisions. In the process, members of product families are sometimes produced in different divisions. With this in mind, it is perhaps not surprising to find that one of the reasons for considering reuse in HP was reported as:

> "Standardize the product line — A collection of divisions developing products which fit in the same product line heard it's customers asking for standardized interfaces" (Wentzel, 1992).

Standardized user interfaces give a product family an edge over competitors' products when the end users' switching costs are high. For instance, if the end users have had to invest in learning how to use one member of the product family, they may prefer to use a product from the same family so that they are not required to learn a new interface. Standardized device interfaces may also factor into a customer's purchase decision, because it allows them to hook up the same devices that they use for the other members of the product family. If standardized interfaces are not in place, then reuse provides these benefits. On the other hand, if the products already have standardized interfaces, then this may be a good opportunity to employ reuse to centralize maintenance of the common interface.

### 3.3.5 Specialization and Centralization of Expertise

Software reuse can be applied to achieve enhanced productivity through division of labor and specialization. With more up-front planning in the domain analysis phase, components can be specified in a such a way as to make component production independent of consumption. Producers can then develop greater expertise and experience in their more focused portfolio of components, making development and maintenance of related components more efficient.

Differences between people account for the largest variation in software quality (Boehm and Papaccio, 1988), and productivity differences between software engineers also vary widely (Boehm, 1987). In view of these differences, Boehm (1987) remarks that "one of the biggest leverage actions you have at your disposal is to get the best people working for your project or organization." Software reuse introduces the beguiling possibility of simultaneously using the best people on multiple projects. Thus, by reusing their work products, their high productivity and high quality output benefits a range of products rather than just one project at a time. Furthermore, recognizing that outstanding problem solving ability in specialized areas is scarce, organizations can leverage their experts through reuse to benefit multiple projects simultaneously.

Thus, specialization or centralization of expertise can have two effects. First, features not formerly feasible for projects that lacked the expertise become viable. Second, enhancing productivity through specialization, or capitalizing on the most productive, talented workers, will further reduce development time.

### 3.3.6 Additional Opportunities

Reuse might give rise to other opportunities to increase profits. For instance, one HP division, under regulatory pressure to certify its embedded software, is considering reusing previously certified components in order to reduce its cost of compliance (Wentzel, 1992). This would be an additional cost saving beyond the development and labor cost saving, and should be included in an assessment of the reuse benefits in that environment. Such case-specific opportunities should not be overlooked.

## 3.4 Other Costs

Although software reuse has great potential to enhance a software development organization's competitive position, it would be a mistake to approach reuse with unguarded optimism. For

instance, while downstream product development is accelerated by well-managed reuse, the first product to be released with reused components could be delayed by component production. The penalty for this delay should be included in the evaluation, and alternative options, like retrofitting the component for reuse, should be considered if the penalty is too high.

It should also be recognized that the amount of commonality among products is a decision variable that may entail tradeoffs between fine-tuning components to meet the specialized needs of the target product niche, and achieving economies of scope through standardization of components. Of course, the consumer could modify the component so that its performance and features meet the customer's requirements exactly — but this will be at the expense of maintenance savings through black box reuse.

## 3.5 Estimating Increased Profits from Reuse

In contrast to the development and maintenance costs which are generally estimated by applying a labor rate to project management's estimates of effort reduction, many of the benefits discussed are manifest in product sales. Consequently the marketing department should be involved in estimating expected increases in profits that arise from exploiting reuse.

It should be noted that when estimating the benefits from these various categories, it is important not to double count benefits (or costs). Thus, for instance, if the resources freed by earlier completion are to be used to create new products, then the opportunity cost should not be estimated for *both* new product and freed resource categories.

The nomenclature is extended, so that

$\overline{\nabla\Pi_A}$ = discounted aggregated incremental profit from new opportunities afforded by reuse

The incremental profit from additional business opportunities, the life-cycle consumer costs saved, (equation 1), and the life-cycle producer costs and overhead (equation 2), are the primary components of the net benefit model. Putting them together, the net benefit, B, from reuse is:

$$B = \overline{S_A} + \overline{\nabla\Pi_A} - \overline{C_P}$$

## 4. Illustrative Example

Although the following example is hypothetical, the assumptions are consistent with real project data. Based on measured data obtained from many software development projects at Hewlett-

Packard, Harris (1992) estimated that with a 50% reuse level and a 5x quality improvement in the reused component over new code, producer effort was increased by 108% and consumer effort reduced by 40% during the development phase. During the maintenance phase Harris estimated that producer effort was increased by 25%, and consumer effort reduced by 42%. These effort factors (assuming a 50% reuse level) are used together with the following assumptions to estimate reuse benefits.

| | |
|---|---|
| Hourly rate for software engineers (including basic salary and administration overhead) | $75 |
| Project team size | 20 |
| Development cycle time without reuse (months) | 12 |
| Annual inflation in labor rate | 5% |

The simplifying assumption that all of the products are comprised of the same amount of new and reused code is made to better demonstrate a number of points. The model results are shown in the table below. Considering first the results from the basic model (Model 1) it is clear that increasing the number of reuse opportunities increases the total net benefit. The break-even point is reached by the third reuse.

In Model 2, annual cash flows are discounted back to the present to reflect the fact that a dollar earned in future periods is worth less than a dollar invested today. Although the net saving is still positive by the third reuse, the discounted cash flow results show that models that do not take the time value of money into account give unduly optimistic estimates of net benefit when the reuse instances are spread out over time.

Model 3 incorporates uncertainty as to whether future reuse instances will actually take place. Since the third reuse is considered highly likely, though not certain, the expected (discounted) value of the consumer savings is less than in the previous deterministic model, and now the break-even point is only reached after the third reuse. In Model 4, the savings from one upgrade to each product are incorporated. The break-even point is reached after the first two products and their upgrades, but the total savings over five reuses plus upgrades is increased by 27%. This indicates that incorporating (corrective and evolutionary) maintenance savings into the model better reflects the reuse opportunity.

| **Model 1: Basic Development Phase Costs** | | | | | | |
|---|---|---|---|---|---|---|
| | Producer Cost | Product 1 | Product 2 | Product 3 | Product 4 | Product 5 |
| Year of Release | 0 | 1 | 1 | 2 | 2 | 3 |
| Without Reuse | 300,000 | 630,000 | 630,000 | 661,500 | 661,500 | 694,575 |
| With Reuse | 624,000 | 378,000 | 378,000 | 396,900 | 396,900 | 416,745 |
| Reuse-specific Overhead | | 35,000 | | 25,000 | | 25,000 |
| Consumer Saving | | 252,000 | 252,000 | 264,600 | 264,600 | 277,830 |
| Cumulative Net Saving | -624,000 | -407,000 | -155,000 | 84,600 | 349,200 | 602,030 |
| **Model 2: Taking the Time Value of Money into Account** | | | | | | |
| Consumer Saving after Discounting (i = 7.5%[4]) | | 234,419 | 234,419 | 228,967 | 228,967 | 223,642 |
| Cumulative Discounted Net Saving | -624,000 | -422,140 | -187,721 | 19,613 | 248,580 | 452,098 |
| **Model 3: Taking Uncertainty in Reuse Instances into Account** | | | | | | |
| Probability of Reuse | | 1 | 1 | 0.90 | 0.75 | 0.50 |
| Consumer Saving with Discounting & Uncertainty | | 234,419 | 234,419 | 206,070 | 171,725 | 111,821 |
| Cumulative Discounted Expected Net Saving | -624,000 | -422,140 | -187,721 | -3,284 | 168,441 | 260,138 |
| **Model 4: Including a Future Upgrade** | | | | | | |
| Year of Release | 1 | 2 | 2 | 3 | 3 | 4 |
| Probability of Reuse | | 1 | 1 | 0.68[5] | 0.38 | 0.25 |
| Upgrade without Reuse | 75,000 | 157,500 | 157,500 | 165,375 | 165,375 | 173,644 |
| Upgrade with Reuse | 93,750 | 91,350 | 91,350 | 95,918 | 95,918 | 100,713 |
| Reuse-specific Overhead | | 10,000 | | 10,000 | | 10,000 |
| Additional Consumer Saving | | 57,242 | 57,242 | 38,019 | 21,246 | 13,653 |
| Cumulative Discounted Expected Net Saving | -717,750 | -467,301[6] | -175,641 | 38,766 | 231,737 | 329,599 |

[4] The interest rate may be the prevailing bank rate, reflecting the interest that the investment would earn if it was deposited instead of invested in reuse, or the company's hurdle rate, reflecting what the investment would earn in some alternative use within the company.

[5] Conditional on Product 3 being produced, the probability of its upgrade being produced is assessed as 0.75. Thus, the probability of producing the upgrade is 0.9*0.75 = 0.68. Similarly, the conditional probability of upgrades to products 4 and 5 being produced is 0.5.

[6] This is the cumulative saving for the first product plus its upgrade. Overhead for the year is assumed to be incurred regardless of whether the next reuse instance is actualized (i.e. a semi-fixed cost, such as salaries).

### Estimating the Opportunity Cost of Shortened Time to Market

In the hypothetical scenario, the consumer effort is reduced by 40% as a result of reuse and it is assumed that this corresponds to reducing the development cycle by an equivalent amount, so that the product reaches the market in seven months rather than twelve. Working with marketing's profit forecasts, an estimate of the increased revenue can be obtained by simply shifting the product's entire revenue stream forward to the introduction date that reuse is expected to make possible. The estimate of additional revenue is the interest on the earlier cash flows after any cannibalization effects are taken into account. A somewhat less conservative approach assumes that the earlier cash flows are over and above those that marketing had forecast. Then, after any cannibalization effects are accounted for, not only is there increased profits from extra sales, but also from interest on earlier sales as well.

It is further assumed that the release dates for products 1 and 2 are not delayed by component production, and that product 3 does not compete with any of the company's existing products, the sales forecast for the next year is 24,000 units and the selling price is $1,000. Now, if sales increase by 5,000, and the profit margin is 15%, then profit increases by $750,000 plus interest. This example demonstrates that the profit impact of earlier time to market might dominate the cost savings from reuse. Therefore, if the competitive environment dictates that time to market is an important determinant of life-cycle profits, then the contribution that reuse makes to profits by hastening product release should be incorporated in the assessment of reuse benefits. This example shows how important it is to include additional profits from any business opportunities that reuse gives rise to, even when they are not easy to quantify.

## 5. Conclusion and Directions for Future Research

Systematic software reuse differs from traditional software development in important ways that impact the choices and opportunities that business decision makers face. As a result, management faces new information needs. Like manufacturing facilities, systematic reuse entails a significant up-front investment and the return is only received over an extended period of time. However, unlike plant and equipment, "knowledge assets" like reusable software components are not reflected on business balance sheets. Because they are not valued, it is more difficult to justify the investment in them. An analysis framework that appropriately reflects the factors relevant to reuse has been developed to support the reuse investment decision. The cost-benefit models are easily extended to reflect return on investment or to determine the break-even number of reuse instances, or modified to support component-level investment decisions (Malan, 1993a). The decision criteria that the framework supports ranges from the return on reuse investments, to the impact of

various resource allocation alternatives. As a tool for determining the value of components, it also provides a basis for a mechanism to redistribute some of the reuse benefits to producers so that they have the incentive to produce components whose expected return justifies the investment.

It should be noted that this model requires cost and benefit estimates as input parameters, and the complex issue of how best to estimate these costs and benefits is beyond the scope of this paper[7]. Nevertheless, the model is intended for practical application, and therefore cost estimation is of vital importance. Pfleeger and Bollinger (1990) survey a number of software cost estimation models, and point out the deficiencies of many of these models with respect to reuse programs. Balda and Gustafson (1990) develop a cost estimation model specifically for reuse. Scacchi (1991) analyzes a number of studies of software productivity, and identifies variables that influence productivity. Since productivity drivers may be viewed as the inverse of cost drivers, this approach is very relevant to cost estimation for reuse. Current research is focused on identifying metrics suitable for reuse cost estimation, reusability assessment, and project control.

Implications for activity based costing techniques for software development in a systematic reuse environment also need to be investigated. By identifying the various activities in a reuse program, better estimates of the effort, and hence costs, can be obtained, and the relevant tradeoffs can be made explicit. Producer cost drivers include choice of technology (programming language and other tools, for instance), reuse mode (create afresh or re-engineer) and degree of generality of the components. Consumer cost drivers include the technology chosen by the producer, and the modularity or independence, functionality, correctness, and complexity of the components. While Barnes and Bollinger (1991) present a useful discussion of a number of factors that influence reuse costs, much remains to be done in this area.

## Acknowledgments

This study was made possible by the generous support of Hewlett-Packard. In particular, we would like to thank Corey Billington and Martin Griss for arranging funding for this project. Without their enthusiastic support for academic-industry cooperation and field research this project would not have been possible. Also, the shared insights and thoughtful comments of the members of HP's Software Reuse Department and Corporate Software Initiative are gratefully acknowledged.

7 In the case of code level reuse, a simple approach is to apply a unit cost to some estimate of code size to obtain the cost estimates (as Gaffney and Cruickshank (1992) do in their reuse economics models). However, we prefer to defer placing such restrictions on the model so that the environment in which the model is used can dictate any simplifying assumptions that are appropriate.

## References


Balda, D,. M. and D. A. Gustafson, "Cost Estimation Models for the Reuse and Prototype Software Development Life-Cycles", *ACM Sigsoft Software Engineering Notes*, **15** (3), pp. 42-50, 1990.

Banker, R. D. and C. F. Kemerer, "Scale Economies in New Software Development", *IEEE Transactions on Software Engineering*, **15** (10), pp. 1199-1205, October 1989.

Barnes, B. H. and T. B. Bollinger, "Making Reuse Cost-Effective", *IEEE Software*, **8**(1), pp. 13-24, January, 1991.

Biggerstaff, T. J., "An Assessment and Analysis of Software Reuse", STP-MT-119-91, 1991.

Boehm, B. W. "Improving Software Productivity", *IEEE Computer*, pp. 43-57, September 1987.

Boehm, B. W. and Papaccio, P. N., "Understanding and Controlling Software Costs", *IEEE Transactions on Software Engineering*, **14**(10), pp. 1462-1477, October 1988.

Bollinger, T. B. and S. L. Pfleeger, "Economics of Software Reuse: Issues and Alternatives", *Information and Software Technology*, **32**(10), pp.643-652, December 1990.

Brooks, F. P., *The Mythical Man Month*, Addison-Wesley, Reading, MA, 1975.

Brooks, F. P., "No Silver Bullet: Essence and Accidents in Software Engineering", *COMPUTER*, April 1987.

Cox, B. J., "There Is a Silver Bullet", *BYTE*, October 1990.

Cusumano, M. A., *Japan's Software Factories: A Challenge to U.S. Management*, Oxford Univ. Press, 1991.

Gaffney, J. E. Jr. and R. D. Cruickshank, "A General Economics Model of Software Reuse", *Proceedings of the 14th International Conference on Software Engineering*, pp. 327-337, May 1992.

Gaffney, J. E. Jr. and Durek, T. A. "Software Reuse — Key to Enhanced Productivity: Some Quantitative Models", *Information and Software Technology*, **31**(5), pp. 258-267, June 1989.

Graves, S. B., "The Time-Cost Tradeoff in Research and Development: A Review", *Engineering Costs and Production Economics*, **16**, pp. 1-9, Elsevier Science Publishers, 1989.

Griss, M. L., Favaro, J., and P. Walton, "Managerial and Organizational Issues - Starting and Running a Software Reuse Program", *Software*, Shaefer, March 1993.

Harris, K., "Using an Economic Model to Tune Reuse Strategies", *Proceedings of the 5th Annual Workshop on Software Reuse*, 1992.

Kaplan, R. S., "Must CIM be Justified by Faith Alone?", *Harvard Business Review*, pp. 87 - 93, March-April, 1986.

Lim, W. C., "A Cost Justification Model for Software Reuse", *Proceedings of the 5th Annual Workshop on Software Reuse*, 1992.

Malan, R. A., "Software Reuse: A Business Perspective", *Hewlett-Packard Technical Report*, February 1993a.

Malan, R. A., "Evaluating the Impact of Software Reuse on the Cost of Quality", *Hewlett-Packard Technical Report*, March 1993b.

Mayobre, G., "Using Code Reusability Analysis to Identify Reusable Components from the Software Related to an Application Domain", *Proceedings of the 4th Annual Workshop on Software Reuse*, pp. 1-14, November 1991.

Rosenau, M. D., *Faster New Product Development*, American Management Association, New York, 1990.

Scacchi, W., "Understanding Software Productivity: Towards a Knowledge-Based Approach", *International Journal of Software Engineering and Knowledge Engineering*, Vol. 1, No. 3, 1991.

Stalk, G. Jr., "Time — The Next Source of Competitive Advantage", *Harvard Business Review*, pp. 41-51, July-August, 1988.

Ulrich, K. T., D. Sartorius, S. Pearson and M. Jakiela, "Including the Value of Time in Design-for-Manufacturing Decision Making", MIT Internal Paper WP #3243-91-MSA, 1991.

Ulrich, K. T., and K. Tung, "Fundamentals of Product Modularity", Proceedings of the 1991 ASME Winter Annual Meeting Symposium on Issues in Design/Manufacturing Integration, Atlanta, 1991.

Wegner, P., "Varieties of Reusability", *Workshop on Reusability in Programming*, ITT Programming, Newport, RI, pp. 30-44, September 1983.

Wentzel, K. D., "Software Reuse — It's a Business", *Proceedings of the 5th Annual Workshop on Software Reuse*, 1992.

Wheelwright, S. C. and K. B. Clark, *Revolutionizing Product Development*, The Free Press, New York, 1992.