Title: Economic Analysis of a Software Partner Relationship

Course:
Year: 1993
Author(s): G. Trace

Report No: P93059

Abstract: This paper is an economic analysis of a program established to increase the capability of a software team's customer support process. The program analyzed will be referred to as the Wipro program. Wipro is a company located in Bangalore, India. Besides producing products, Wipro specializes in developing partner arrangements with companies. As a partner, Wipro supplies software engineers on contract. The problem that this Wipro program attempts to solve is how to best increase the support capability of a software team's customer support process.

The paper presents a discussion of the growing role of software suppliers, detailed economic analysis of one program based on the author's own experience, a comparison with other approaches to increasing a software team's customer support process capability, recommendations for further analysis, and a summary of what he has learned. He quantifies issues in terms of dollar costs, dollar savings, or estimated dollar value, and then applies an economic analysis approach to evaluate the cost effectiveness of the program.

Economic Analysis of a Software Support Partner Relationship

G. Tracy

P9359

Economic Analysis of A Software Support Partner Relationship

Gary Tracy

Emgt 535 Project

November 9, 1993

Introduction

This paper is an economic analysis of a program established to increase the capability of a software team's customer support process. The program analyzed will be referred to as the Wipro program. Wipro is a company located in Bangalore, India. Besides producing products, Wipro specializes in developing partner arrangements with companies. As a partner, Wipro supplies software engineers on contract.

The problem that this Wipro program attempts to solve is how to best increase the support capability of a software team's customer support process.

The paper presents a discussion of the growing role of software suppliers, detailed economic analysis of one program based on my own experience, a comparison with other approaches to increasing a software team's customer support process capability, recommendations for further analysis, and a summary of what I have learned. I will quantify issues in terms of dollar costs, dollar savings, or estimated dollar value, and then apply an economic analysis approach to evaluate the cost effectiveness of the program.

Despite the solid bottom-line appeal of the quantification into dollars, the purpose of the paper is insight, not a simple numerical answer. This notion of insight will be expressed throughout the paper, and forms the basis for the conclusions.

Discussion: Suppliers and Engineering Management Decisions

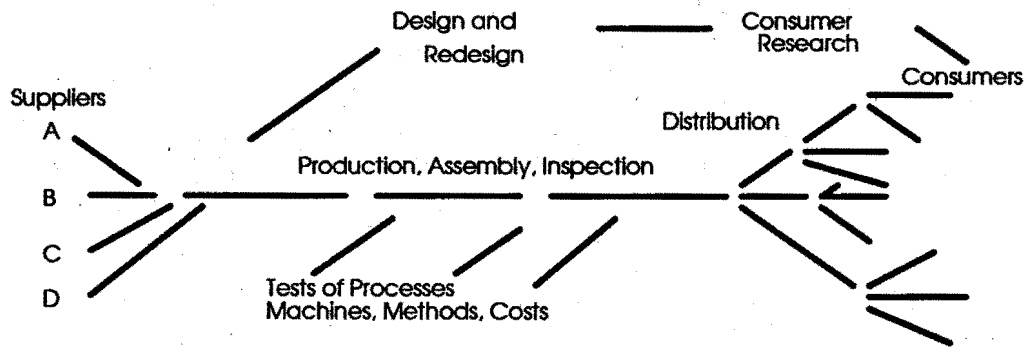
Three methods of increasing a software team's customer support process capability are discussed in this paper. The three methods can be expressed in terms of supplier relationships. Expressing the methods in terms of supplier relationships is useful because suppliers and partnerships play an increasingly important role in software engineering.[11] There is productivity leverage in not making what you can buy, and this leverage enables organizations to reserve research and development capability for strategically important products and capabilities.[1], [12]

Purchasing software and purchasing support for the software forms a direct supplier relationship. Providing the support with an in house engineering team is the make approach, and hiring contractors to provide the support, as in the Wipro program can be

called the contractor approach. Both the make approach and the contractor approach can be thought of as replacing a supplier, or doing the work of a supplier.

The buy approach consists of evaluating software vendors, selecting a partner, and purchasing the product together with a continuing relationship in the form of a maintenance contract which provides for product support from the vendor for a contract fee. The make approach consists of building or buying a software product, and employing a team of software engineers to support the product. The contractor approach consists of building or buying a software product, and hiring contractors to support the product.

The role of suppliers has been extensively described.[3], [4], [11] The quality movement has asserted the value of close working relationships with suppliers to increase quality, and optimize the overall system. W. Edwards Deming claims that the initial concept he taught to the Japanese was to consider their organizations as complete systems, including suppliers and customers.[3], [4] He used the following flow chart to emphasize that suppliers and customers are part of the operational system.



Deming criticizes the normal company hierarchical organization chart as devoid of information as to what a person's job is. The organization chart tells you who you work for, but not what your job is. He asserts that his flow chart is better. Locate your role on the flow chart, and you learn who depends on your work, and who you depend on. His flow chart brings the customer and suppliers explicitly into the system. Deming says "work with your supplier as a partner on a long-term relationship of loyalty and trust to improve the quality of incoming materials and to decrease costs." Thinking of the three methods for increasing support process capability in terms of supplier relationship provides a paradigm which emphasizes the inclusion of suppliers into the organization's production system. You have a supplier, or else you are doing their work.

Detailed Wipro Project Description

Project Vision

This project was established at the beginning of 1993 to develop a relationship with Wipro to provide Sequent's compiler software team with a continuous stream of improved versions of three of Sequent's own software products: a C compiler, a symbolic debugger, and a C++ compiler. The relationship was established to deliver needed increase to the compiler team's bug fix process capability.

The improvements to the software were not limited to bug fixing. It was envisioned that the relationship would be used to improve compiler products in other ways as well, including but not limited to: implementing new features, implementing new tests, using test coverage analysis to produce faster more effective versions of confidence test suites, and collaborating with the compiler team to implement new bug prevention techniques

The proposed budget for the relationship was ten thousand to fifteen thousand dollars for startup cost. This provided for a Wipro engineer to reside on-site at Sequent for two to three months to scope and specify the workload, and to learn about Sequent's software development processes. The total cost to the end of 1993 was projected to be seventy-two thousand dollars, including five thousand dollars estimated lease fee for prorated machine usage charge.

The compiler project would leverage a company-wide partner relationship that Sequent was developing with Wipro. This larger scale partner relationship would provide the following "free" benefits to the compiler team:

- data communications capability between Sequent and Wipro
- Sequent machine in place at Wipro
- trained operators in place at Wipro
- improvements to Sequent's source code control system

The Sequent compiler team initiated this project in order to provide our customers with more timely fixes to bugs. The incoming bug rate was greater than our current bug fix process capability. The Wipro relationship would allow compiler team members to continue their focus on new product development, and to continue to provide technical backup for Sequent's service department.

The project fit well with Sequent's business strategy by leveraging a partner relationship rather than increasing permanent staff. The project was consistent with corporate values, especially the quality value since the aim was to improve the quality of products already in customer use.

Success would be measured by measuring the increase in capability of the Sequent compiler team's process to close bugs. I did not consider doing economic analysis at that time. During 1991, the compiler team averaged thirty-four bugs closed per month. During 1992 the team averaged nine bugs closed per month due to a reduced team size after Sequent's 1991 reduction in force. The estimate was that Wipro would eventually be capable of closing forty bugs per month. The average monthly incoming bug rate at the time was fifteen, meaning that Wipro would be able to clean up the bug backlog of three-hundred and fifty open bugs over a fourteen month period. The number of normal priority and above bugs in the backlog was one hundred and sixty-eight, so attacking the bug backlog in priority order would deliver substantially improved products within six months. Substantially improved means no critical, serious, or normal bugs in the products.

Project Scope

At the completion of this project, Sequent and Wipro will have a supplier relationship that increases the compiler team's bug-fixing capability by approximately forty bugs per month. The initial three months will cost Sequent ten to fifteen thousand dollars, and a staff-month of time.

The steady-state cost after completion of the initial phase will be approximately sixty-five thousand dollars per year for two off-site engineers, and approximately five thousand dollars per year for machine usage. We had no estimate of the monthly charge back for data communication line usage. We estimated a cost of one to two staff days per week to manage the off site project, review code changes, merge modified source code, and answer technical questions.

Project Milestones

- 11/03/92: policies and processes defined
- 11/10/92: plan approved
- 01/04/93: Wipro engineer on site at Beaverton
- 03/01/93: policies and processes built and tested
- 03/01/93: publish functional specification and project plan

- 03/01/93: Sequent system at Wipro on line and operational
- 06/01/93: Maintenance project delivers products

Cost Benefit Analysis

Summary of actual costs

The plan was developed in November, 1992, presented to decision makers, and approved. A Wipro engineer arrived on site at Beaverton in January 1993, and developed a detailed project plan by March 15. The team and support facilities were in place and operational by April, one month later than planned, and the two Wipro engineers delivered 58 bug fixes by the end of June. Actual cost charge-backs were

- January: $5500
- February: $5500
- March: $5500
- April: $6600
- May: $6600
- June: $6600

Additional direct costs were

- January: $2000 for 20% time Sequent engineer to help develop the plan
- February: $2000 for 20% time Sequent engineer to help develop the plan
- April: $500 for 5% time Sequent engineer consulting
- May $500 for 5% time Sequent engineer consulting
- June $500 for 5% time Sequent engineer consulting

Summary of benefits

Development of the plan included detailed analysis of the base of existing bugs against the product set. This analysis has value in its own right. Execution of the plan resulted in delivery to Sequent of source code changes which repaired 58 bugs, commentary to improve the readability and maintainability of source code in areas where bugs were fixed, test cases for the software regression test suite to ensure that these bugs would never be reintroduced to customers, and reserve

capacity of expertise for the internals of these software products. These elements of added value are examined individually in the following sections.

The basis for cost savings of the Wipro project is the cost of a Wipro contractor. The charge back to engineering of a single contractor is $5500 per month if the contractor is on site in Beaverton, and $3300 per month if the contractor is working off site at Wipro. This is compared to the $10,000 per month fully burdened average cost of a full time engineer added permanently to the team. For this analysis, however, we want to measure the value of the work done by the Wipro project.

Value of a bug fix

Method 1 is to calculate the cost to Sequent of doing a maintenance release which includes a known number of bug fixes. Maintenance release 91-B is a good example. This release was planned and executed by the compiler team. Maintenance release 91-B delivered 122 bug fixes and cost 4 staff-months of effort. Since maintenance release 91-B leveraged bug fix work which was done over an earlier period of 3 staff months, the most accurate bug fixes per staff month number is 122 divided by 7 or 17. Method 1 analysis indicates that the value of a single bug fix is 10,000 divided by 17 or $588.

Method 2 is to examine the continuous bug fix rate of the compiler team against the amount of staff time budgeted for bug fix work. We budget 20% of the 4 person team to customer support. I estimate that about half of that budget is spent closing bugs, since a lot of time is spent handling interrupts from service to consult with customers and negotiate fast patches. The team is averaging 7 bug fixes per calendar month which translates to 7 divided by 10% of 4 staff months or 17.5 bug fixes per staff month. Method 2 analysis indicates that the value of a single bug fix is $10,000 divided by 17.5, or $571.

Method 3 is to discuss the question with senior engineers to develop an expert consensus opinion of bug fix rate. The result of this was 2 bug fixes per staff week, or about 9 bug fixes per staff month. Method 3 analysis indicates that the value of a single bug fix is $10,000 divided by 9, or $1,111.

Another set of methods to attach value to bug fixes involve examining what Sequent is willing to spend to fix a single bug. Two Sequent processes help to illuminate this.

The first is Sequent's red light process. Any Sequent employee can initiate a red light investigation which has the effect of stopping shipment of the suspect component. This process has been used to stop shipment of compiler products 3 times in 10 years. Once for our FORTRAN compiler, and twice for the COBOL compiler. Red light teams have the highest priority to acquire resources necessary to resume shipment of the product and repair any damage at customer sites. The FORTRAN compiler red-light cost 4 staff months, or $40,000 to turn off. One COBOL red light was due to manufacturing process error, and cost only $3000 to turn off, but one was due to a software bug and cost $20,000 to turn off. Not all bugs can cause red lights, but this is one indicator of what Sequent is willing to spend. This analysis indicates that the value of a single bug fix is $40,000.

The other process of interest is the fast patch process. This process delivers a bug fix directly to a customer in need who will not wait for the next maintenance release. This is much more common than red-light for compiler products. Compiler products have averaged 1 fast patch per month over the past 3 years, with an average cost per fast patch of $5000. A bug that must be fixed by fast patch doesn't need to be especially serious, although it usually is. The rule is that the customer is always right. A fast patch is necessary if the customer says it is necessary. This means that most any known problem in the product is a potential fast patch. This analysis indicates that the value of a single bug fix is $5,000.

A final way to attach a dollar value to a bug fix is to take the amount of revenue that we get from software service contracts and pro-rate a portion of that for the number of bug fixes delivered to customers over the contract period. This method is interesting because it includes the important idea of how much a customer is willing to pay for the work, but unfortunately, the service contract information is not available for this report.

Value of a generalized test case

An important part of the process of fixing bugs is the generation of a test case which is added to our regression test suite. This guarantees that we will never ship a version of the product again which has this particular bug in it, but equally important, the test case usually can be generalized to cover more than just the failure mode exhibited by the bug. Full generalization means that the entire subsystem of the product which contained the defect is exercised in the manner that demonstrated the defect. Experience has shown that this often exposes more bugs. An area for future research is to perform experiments and gather statistics to quantify this.

Value of continuously improving documentation

When an engineer analyzes the area of the product which is demonstrated to be broken due to the bug, the engineer must rediscover the technical details about how this subsystem works in order to find out what is wrong. The process of rediscovery usually leads to insight about how the subsystem can be better explained in the internal commentary so the engineer adds to or clarifies this commentary as part of the bug fix process. This means future problems in this area will be easier to diagnose. Quantifying this value is an area for future research.

Value of added skill and expertise

Every software engineering team has the problem of how to cover a product set with appropriate expertise. The more specialized the engineering staff becomes, the better the product is understood which enables innovation and breakthroughs in product improvement, and fast reliable defect repair. The higher the degree of specialization, however, the greater the exposure to problems when an engineer leaves the team. The engineer's deep understanding leaves the team too, since other team members have their own area of specialization. To combat this, software engineering teams develop more general expertise to provide redundancy. A proven method for accomplishing this cross training is to assign bugs from one specialist area of expertise to another specialist. The assignee consults with the specialist, but is responsible for fixing the bug, and in doing so, develops expertise in a new area. This is an example of how a bug fix adds value in the form of training material for the team. Quantifying this value is an area for future research.

Value as pilot project

Every project should teach the project team more about how to do projects. The Wipro project is especially valuable in this area since it serves as a pilot project for the Wipro partnership which Sequent is developing. The extra leverage is that lessons learned during the pilot project will result in cost savings to other project teams as the number of Wipro partnership projects grows.

Intangibles

Sequent's commitment to quality

Casey Powell, the CEO of Sequent, likes to remind us how important our reputation is to us. He points out that it takes a lot of time and strong execution to build a good reputation, but a good reputation is easily lost. Sequent commissions an independent survey agency to talk to our customers annually to provide us with feedback about our perceived quality, and in this area, perception is reality. How do you use this fact to attach value to a bug fix?

Synergy with existing team

A special benefit of the Wipro project is synergy with the existing product team. The Wipro engineers are able to focus without interrupt on their work with our product set. As they work and study, the questions that they ask serve to sharpen our team, and inspire innovation and breakthroughs. You cannot plan innovation and breakthrough, but this synergistic effect helps condition the environment. [2]

Flexibility of work force

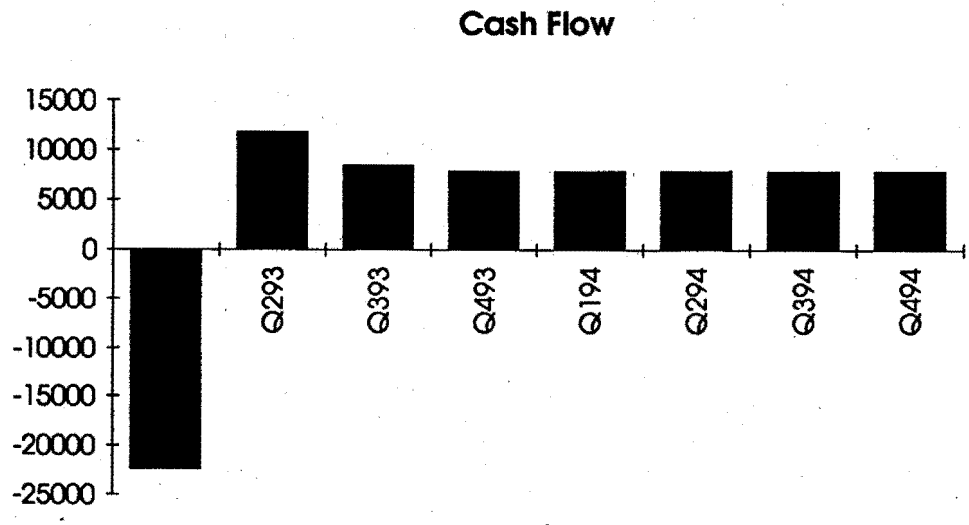
If Sequent encounters a difficult financial situation, as in 1991, Sequent would be forced to reduce expenses. This usually means a reduction in force. By forming partnerships like the one with Wipro, Sequent builds flexibility against required reductions. The partnership could be down sized with less disruption and less loss of morale than a reduction in force would cause.

Economic Analysis

The economic analysis of this project is summarized in the spreadsheet in the appendix. The charts and values used in this discussion were generated from that spreadsheet. The spreadsheet models this project from its start in January of 1993 until the end of 1994. At various points in the discussion, the economic performance of the project in 1993 is considered, as well as the projected economic performance over the entire two year, or eight quarter, period. Note that the number of bugs fixed is an actual result through August of 1993. The number of bugs fixed each month from September 1993 until December 1994 is an estimated number.

Cash Flow Diagram

The next step in the economic analysis is to develop a cash flow diagram for the project. Two study periods are considered, start up through December 1993, and start up through December 1994. It is more interesting to examine the project from start up, even though over ten months of the project have already occurred. Economic analysis was not used to get approval for the project, so there might be lessons to learn about future projects. The following bar graph is a cash flow diagram for the entire eight quarter period.

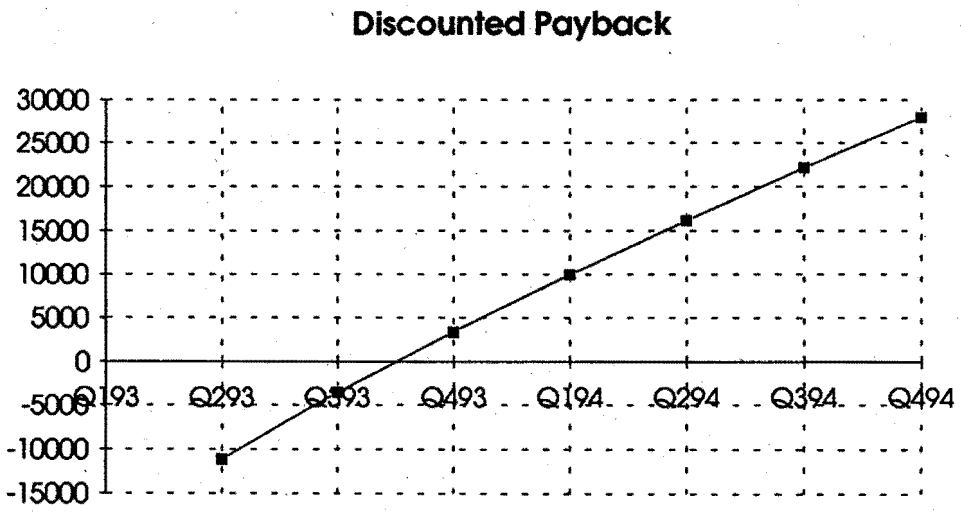


The assumptions used to produce this diagram are that a bug fix is worth $571, and that the Wipro team will be able to fix 17 bugs per month. These are

reasonable expected values of these variables. The $571 bug fix value is reasonable because it is the smallest of the values quantified in the analysis above. The 17 per month fix rate is reasonable because the Wipro team has averaged more than that over the four months of actual results, and 17 is the rate established for our own engineering team in the bug value analysis above. The original estimate of 40 bug fixes a month for the Wipro team was unrealistic.

Discounted Payback Period

The spreadsheet contains a calculation of the present value of each of the quarterly cash flows. From this we can build a chart showing the discounted pay back period for the project.



The above chart shows pay back happens during the third quarter of 1993, using $571 as the value of a bug fix, and 17 as the estimated bug fix rate. Present values were calculated using a 20% discount rate. Sequent currently uses 20% as minimum acceptable rate of return. We will analyze sensitivity to this discount rate below.

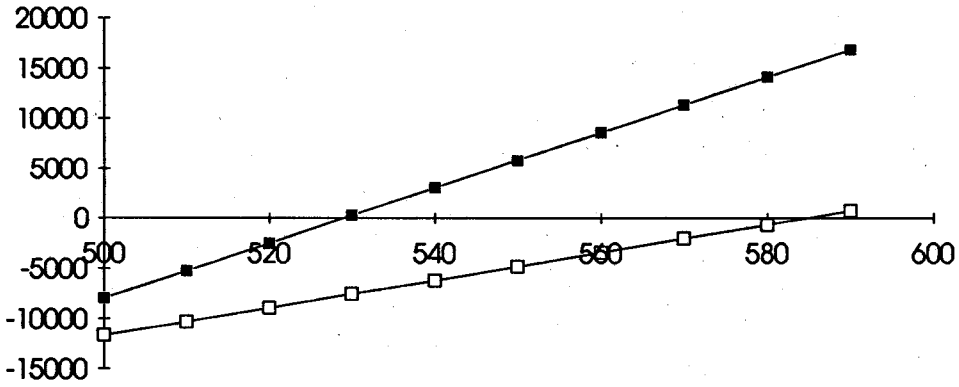
Net Present Value

The spreadsheet contains two net present values for this project. The first is net present value for a two year run of the project. The second is net present

value if the project were to conclude at the end of 1993. In either case, the net present value is positive. This value is calculated with some interesting assumptions. The bug fix value is assumed to be $571. The Wipro team is assumed to be capable of sustaining a bug fix rate of 17 per month, and the discount rate is 20%.

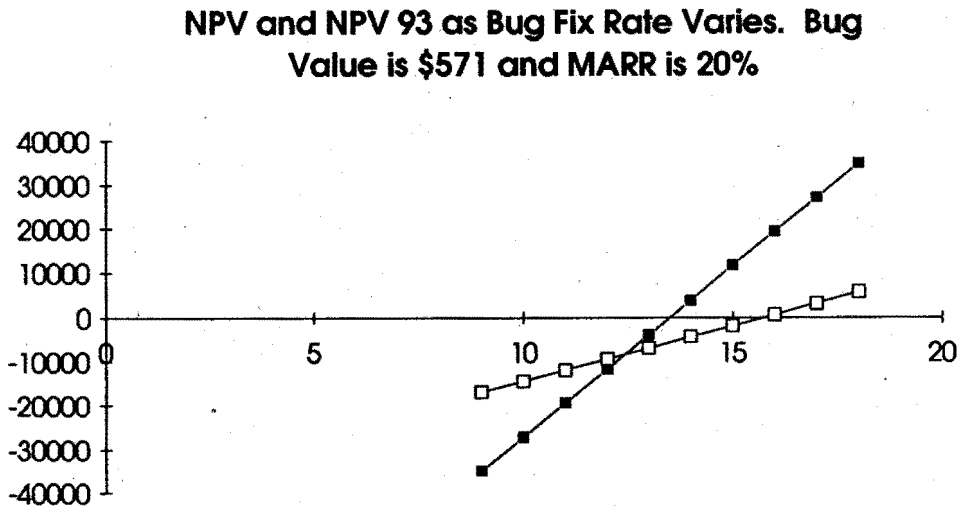
How sensitive is the net present value to these assumptions? The following is a chart of net present values with the fix rate dropped slightly to 15, and the value of a bug fix varied.

NPV 93 and 8 Quarter NPV as Bug Value varies. Fix Rate is 15 and MARR is 20%



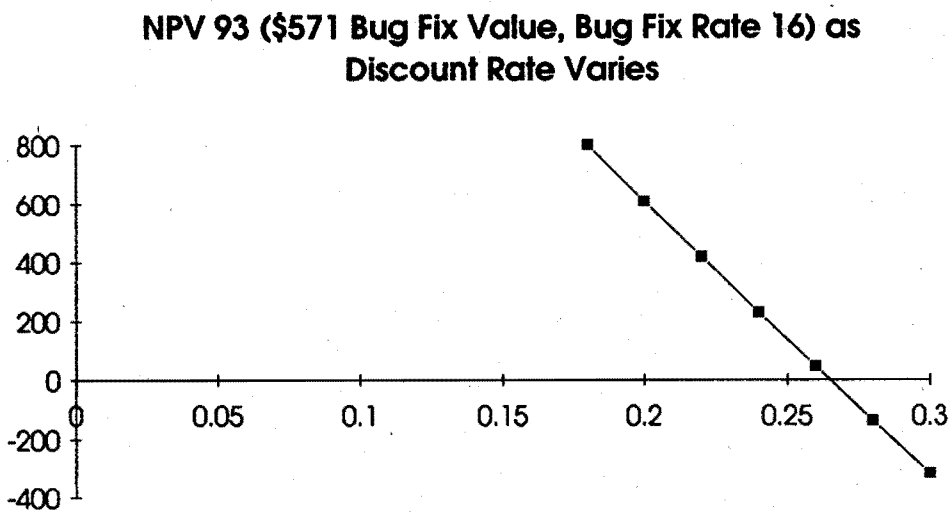
The upper line with dark boxes represents the net present value of the full eight quarter project, while the lower line represents the project economic performance in 1993. The full program net present value turns positive when the bug value is between $520 and $530, while the 1993 project requires a bug value somewhere between $580 and $590. This means that if the Wipro program is canceled at the end of 1993, and the bug fix rate is 15 per month during the months after August, then the program fails to clear the 20% minimum acceptable rate of return if the $571 value for a bug fix is accurate.

What if the fix rate of 17 is unreasonable? The following chart shows net present value change as bug fix rate varies.



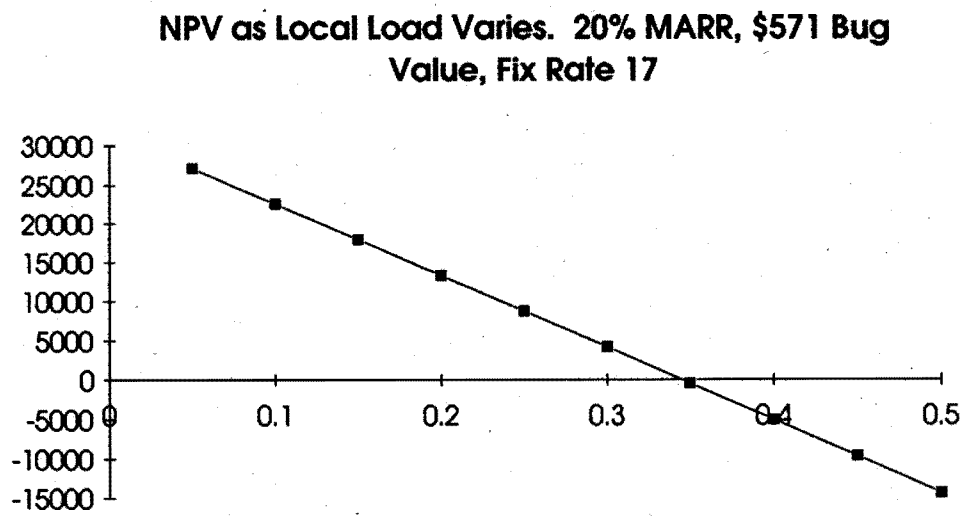
The steeper slope line with dark boxes represents net present value for the full eight quarter project. Bug fix rate is an important multiplier to the bug fix value, and so it is a major component of project value. Under this set of assumptions, the 1993 project requires a bug fix rate of between 15 and 16 per month to clear the minimum acceptable rate of return, and the eight quarter project requires a fix rate of between 13 and 14.

What if the discount rate changes? The following is a chart of the net present value of the project in 1993 as the discount rate varies.



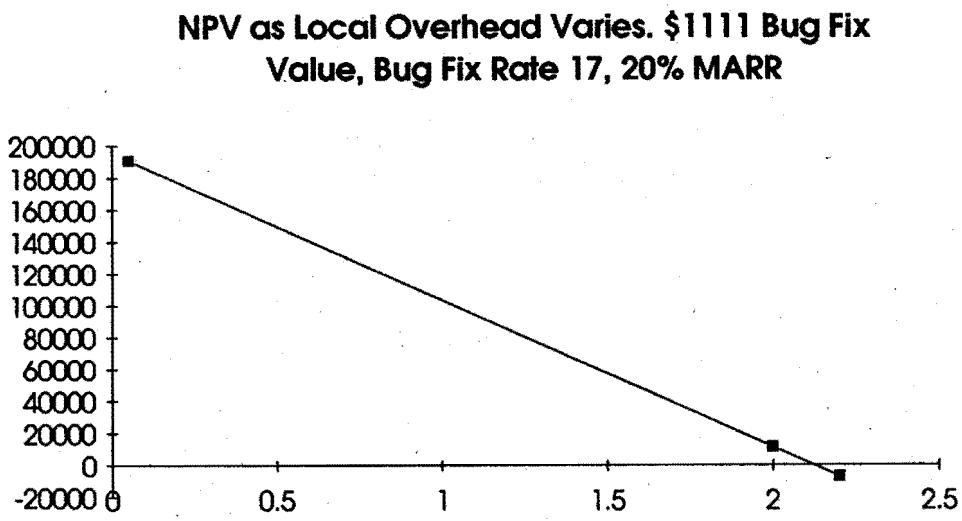
The 1993 project is the only one charted, since the eight quarter project showed no sensitivity to discount rate under these assumptions. The 1993 project is not only relatively insensitive, it is also nearly over, so changes to the discount rate are not likely to affect this project. Sequent's standard 20% minimum acceptable rate of return is a valid hurdle for this project since this program is a pilot for future expansion of the partnership. This is not a high risk program requiring special caution, and there is leverage beyond compiler support.

Finally, what if more than 5% of a Sequent engineer's time is required to support the Wipro team? The following chart shows net present value as local overhead varies from the assumed half day every other week (5%)to two and a half days per week (50%).



This chart shows that net present value is sensitive to local overhead. Recall from the Wipro project scope section above, that the original project estimate was for local overhead to range from 20% to 40%. Anything over 35% will cause the eight quarter project to fail to meet the minimum acceptable rate of return given the other assumptions.

The following chart also shows net present value change as local overhead varies. In this chart, the value of a bug fix is increased to $1111. Net present value remains positive even if local support increases to over two full time engineers. It is anticipated that local support might reach half time effort on an interrupt basis, but overall, 5% is a reasonable estimate.



Future Research

The value of a bug fix provides basis for operational actions and decisions. More research can refine and clarify the issue. The next step I recommend is a determination of what customers are willing to pay through research into what customers actually do pay, and statistical analysis of what that payment buys from software support organizations.

The Wipro program introduces a mix of permanent and contract software engineers. A study of the team effects with the aim of measuring productivity and quantifying synergy as compared to a standard software team could be valuable. Engineering managers are increasingly encouraged to look for leverage through partnerships. Gagnon and Krasner have applied mixed integer linear programming to provide analytic solution for problems like this. [6] Case study material could build from their work.

The Wipro program introduces another element into the team. The fear of exporting jobs becomes real. The challenge is to demonstrate an abundance mentality. There is more engineering to do than there are engineers, and a partner arrangement like the Wipro program can provide a cost effective, flexible, increase to a team's capabilities. This frees the local team to do more, like spend more time with customers, and develop better understanding of customer requirements. This customer knowledge is one way to

keep in touch with product capability which is strategically important to the organization. [5], [15]

More work is needed to quantify aspects of software support such as the value of improved commentary, or the value of a generalized test case. Statistical research into the value of process steps in software support can provide insight which sharpens decision making.

How well does economic analysis do? A feedback system to find improved methods of developing assumptions would be useful. House has proposed a return map to guide research and development and build in a feedback system for continuous improvement. [7]

Conclusions

W. Edwards Deming says that the important figures are unknown and unknowable.[3], [4] He has a point. Quantifying intangibles can clearly be carried too far. But others say to measure the unmeasurable, and imperfect estimates of productivity are better than no estimates at all.[2]

The Wipro program provides good value to Sequent if the assumed dollar values are valid. It is interesting to note that the lower values for a single bug fix provide sufficient justification. The economic analysis provides basis for asking and answering questions about the variables of the operation. The economic analysis provides a different way to look at operational actions and decisions. Operational actions define the value of a bug fix.

When I proposed the Wipro program a year ago, I did no economic analysis to make my case. I had a problem, and Sequent's Wipro partnership provided a solution in terms of pure bug fixing capability. Economic analysis has shown areas of sensitivity in the program such as the sensitivity to local overhead. The value that the economic analysis adds is to make this common sense concern explicit. Working with the numbers makes the potential problem more real and manageable. Prior to the analysis, I would have had an uneasy reluctance to apply more local resources to help Wipro if necessary. The analysis provides an economic basis for this uneasiness and clarifies the issues. Economic analysis provides a framework or mental model for what decisions means in terms of dollars.

Bibliography

[1] Bruce Buchowicz, "A process model of make-vs-buy decision making; the case of manufacturing software," *IEEE Trans. Eng. Manag.*, vol. EM-84, no. 1, Feb. 1991

[2] W. B. Chew, "No-nonsense guide to measuring productivity," *Harvard Bus. Rev.*, pp. 110-118, Jan.-Feb. 1988

[3] W. Edwards Deming, *Out of the Crisis*, Cambridge, Mass: MIT Center for Advanced Engineering Study, 1986

[4] W. Edwards Deming, *The New Economics*, Cambridge, Mass: MIT Center for Advanced Engineering Study, 1993

[5] D. A. Garvin, "Competing on the eight dimensions of quality," *Harvard Bus. Rev.*, pp. 101-109, Nov.-Dec. 1987

[6] Roger Gagnon and James Krasner, "The optimal mix of internal/external engineering staff and equipment with different levels of capability and performance," *IEEE Trans. Eng. Manag.*, vol. EM-37, no. 1, pp. 37-47, Feb. 1990

[7] Charles H. House and Raymond L. Price, "The return map: tracking product teams," *Harvard Bus. Rev.*, pp. 92-100, Jan.-Feb. 1991

[8] David I. Levine, "Participation, productivity, and the firm's environment", *California Management Review*, Vol 332, Summer 1990, pp 86-100

[9] Jordan D. Lewis, "Competitive Alliances redefine companies", *Management Review*, pp 14-18, Apr 1991

[10] M. J. Liberatore, "An extension of the analytic hierarchy process for industrial R&D project selection and resource allocation," *IEEE Trans. Eng. Manag.*, vol. EM-34, no. 1, pp. 12-18, 1987

[11] Dennis H. O'Brien, "Supplier Relationships: Software quality starts with the customer", *Quality*, pp 22-24, June 1991

[12] Steven E. Prokesch, "Mastering chaos at the high-tech frontier: an interview with Silicon Graphic's Ed McCracken," *Harvard Bus. Rev.*, pp. 135-144, Nov.-Dec. 1993

[13] R. L. Schmidt, "A model for R&D project selection with combined benefit, outcome, and resource interactions," *IEEE Trans. Eng. Manag.*, vol. EM-40, no. 4, pp. 403-410, Nov. 1993

[14] Daniel A. Thomann and Donald E. Strickland, "Managing collaborative organizations in the 90s", *IM*, pp 26-29, July/August 1992

[15] M. Tribus, "Applying quality management principles," *Res. Manag.*, pp. 11-21, Nov.-Dec. 1987

[16] Ravi Venkatesan, "Sourcing: to make or not to make," *Harvard Bus. Rev.*, pp. 98-107, Nov.-Dec. 1992

Appendix 1: Spread Sheet

This appendix contains the spread sheet used in the economic analysis. I have also included a page of displayed formulas which show how the spread sheet is set up. The following list is a key for reading the spread sheet.

- ppl cost is the cost of the Wipro contractors.
- ovrhd is the cost of local Sequent engineers to support the Wipro contractors
- bugs fixed is the number of bugs fixed in a given month. This is an estimate after August, 1993. The estimate is taken from the Bug Fix Rate variable at the bottom of the page.
- Cash Flow is bugs fixed times bug fix value from variables for sensitivity analysis minus the sum of ppl cost and overhead.
- Cash Flow by quarter is the sum of the appropriate 3 month Cash Flows
- PV is the cash flow from that quarter times (1+discount rate/4) raised to the appropriate power
- NPV is the sum of PV through December 1994
- NPV 93 is the sum of PV through December 1993
- Discounted Payback is the sum of PV from January 1993 to the current cell
- The next sections contain data for charts. The data was collected by varying the appropriate variables for Sensitivity Analysis, and copying the resulting NPV and NPV 93 into the appropriate cell