Determining How Much Software Assurance Is Enough?

Tanvir Khan

Concordia Institute of Information Systems Engineering

Ta_k@encs.concordia.ca

Abstract

It has always been an interesting problem for the software development project managers to decide how much quality assurance is necessary. It has too many issues involved which make it hard to come to a conclusion because assurance effort seems to be endless for some of the sophisticated IT projects while it's almost non-existent in some low-profile projects with small budget. So, it's necessary to have an idea to make a baseline when to stop assurance effort.

Key-words: RELY, SCED, COCOMO2, COQUALMO, SLOC, Risk exposure, nominal value

1. Introduction

Software Quality assurance is the function that assures that the standards, processes, and procedures are appropriate for the project and are correctly implemented. SQA includes the process of assuring that standards and procedures are established and are followed throughout the life cycle of the software. Compliance with standards and procedures is evaluated through process monitoring, product evaluation, and audits. Similarly, software quality control is the function that checks that the project follows its standards, processes, and procedures, and that the project produces the required internal and external (deliverable) products. The software assurance consists of these two efforts.

A number of factors can play important role in determining how much assurance the software project requires. Different types of projects require different level of reliability and based on that, the project managers need to decide the effort level for the assurance activity. Even with the same reliability requirement, two projects can have different level of assurance effort based on the schedule time. As a general rule, some projects where human life may be at risk or high financial projects require the maximum level of assurance. Many a times well-developed software projects are released having a very few but vital bugs which are the indication of lack of proper assurance. It's also tough for the IT project managers to decide when to stop the assurance activities and market the product. The success of the project depends on the exact determination of the assurance level the software project team should employ to satisfy the customer requirements. If the project team can have a balanced decision on how to trade off the time, quality and scope – the effort of this project is quite worthwhile.COCOMO and COQUALMO – these two models can be used to find a 'sweet-spot' which will give an educated idea of assurance level for a specific software project.

1.1 Goal

The goal of this article is to find a balanced and reasonable way to estimate the assurance effort needed for the software projects. The project managers of IT development projects and other stakeholders of the projects who do the investment and want to make a decision which project will be attractive to invest on – will be benefited by this article. When the investors have a number of choices for the project and need to be selective, the quantitative approach used in this article can be used as a guideline for them.

1.2 Challenges

The customers of software systems usually don't want to compromise the cost, delivery time and reliability level for the systems. It's hard to have a balanced tradeoff among these three factors to have an optimal solution because the reduction or increase of one factor greatly affects the other two factors.

1.3 Previous Work

Previous researches in this field include value-based perspectives of software assurance, risk-based testing, etc. These researches were dealing with only one side of software assurance requirements and couldn't give a clear guideline when factors like time and/or budget are largely involved. The book "Value-Based Software Engineering" published in 2005 has a detailed set of value-based software engineering theory and practices to be followed. Some other noteworthy works in this field include L. Chung's software quality attributes and University of Southern California's (USC) software quality and cost estimation model which are followed as industry standard for cost estimation. The web-based COCOMO 81 Calculator was developed by Dr. Brad Clark of USC.

1.4 Approach

This article has a quantitative value based approach on the software assurance. We are assigning values on the attributes of the softwares so that they can be combined with the COCOMO2 and its extension COQUALMO to have different estimation having different values of the attributes. This approach should work well when a software system needs added assurance because it may endanger human life or incur high volume financial loss. This approach is best suited for this category of software systems because the project manager can be aware beforehand that added assurance will make delay of the system release and can incur extra budget than the traditional software projects.

2. Methodology

The first step of the project is to classify the software based on its reliability level as the assurance level mostly depends on this factor. Another type of classification is to determine if it losses market share due to late delivery or it doesn't matter if the delivery time is increased. Then I need to calculate the probability of loss and the size of loss for each reliability level. This basically gives the market risk of tolerating a certain level of defects.

The hard task is to determine how many defects on KSLOC I am going to approve when the customer want to have the software virtually defect-free by the deadline and most importantly – with the tight budget. My approach makes the cost and quality estimation by taking the software size in SLOC, a number of parameters for the team, project and organization and the required level of reliability of the resultant software. Each parameter contributes to the final estimation and the cost and/or time can be varied largely depending on the demanding reliability level.

2.1 Alternative Methodology

Some other research work dealt with the value neutral approach which gives the same level of assurance for all kind of software systems. As in real-life, softwares demand a varied level of reliability based on their nature, the assurance level need to be unique as well.

2.2 Implementations

The implementation of this article is done in two phases. The first phase has the simulation of COCOMO model and the second phase finds the 'sweet spot' of assurance effort from the risk exposer, reliability level and the effect of market erosion which indicates the marketplace competitions. I followed this methodology because RELY is the most important parameter in value-based system model and market erosion is a driving factor in determining the optimum assurance for software products. The implementations are done having the non-significant values set as nominal for simplicity. A solid understanding of how COCOMO and COQUALMO models work is essential in order to understand this value-based approach.

2.3 COQUALMO

COQUALMO (COnstructive QUALity MOdel)—is an estimation model that can be used for predicting number of residual defects/KSLOC or number of defects/FP (Function Point) in a software product. It also provides insights into determining ship time, assessment of payoffs for quality investments and understanding of interactions amongst quality strategies. It can be applied in the early stages of the project.

2.4 COCOMO

COCOMO is a software cost estimation model for managers of software projects. It's most widely used and accepted software estimation model. It lets the managers to make trade-offs and experiments with 'what-if' analyses to select the optimal project plan. This model lets the managers to specify the size of the project in four different ways:

1) Function points
2) Adaptation or reuse of an existing system
3) Decomposing the system into subcomponents
4) SLOC (source lines of code). It counts the logical lines excluding comments, blank lines and machine-generated lines.

As SLOC is the most popular and useful method of specifying the size of the project, only this method will be followed in this project.

History of COCOMO

- The original COCOMO model was first published by Dr. Barry Boehm in 1981
- The first USC-CSE implementation of COCOMO II was released to the general public in mid-1997.
- USC COCOMO II.1998.0 beta was released in October 1998. It was developed using a Bayesian statistical approach to blend empirical data with expert opinion to calibrate the model.
- USC COCOMO II.1999.0 was released in 1999 and USC COCOMO II.2000 was released in conjunction with the publication of COCOMO book in 2000.

Parameters

– Quantitative such as size, number of defects, months

– Qualitative such as complexity, required reliability, tool usage, analyst capability

COCOMO model is based on the set of equations. The mostly used models of COCOMO are:

1) COCOMO2. 2000 with traditional phases
2) COCOMO2. 2000 with RUP phases (Rational Unified Process)

Scale drivers

There are five scale drivers to determine the exponent in estimating equations. The COCOMO estimating equations generally has an exponent greater than 1.0. It has an indication that productivity is lower on larger projects. When done by the inexperienced managers or if the estimation is for a unique project, the scale drivers value may be set to Nominal.

Table 1. Scale drivers for COCOMO2

Scale Driver	Abbreviation	Values	Consideration
PREC	Precedentedness	ExtraHigh,VeryHigh,High, Nominal,Low,VeryLow	Is the new project comparable to projects your team has done before?
FLEX	Development Flexibility	Do	Are the requirements flexible?
RESL	Architecture/Risk resolution	Do	How complete is the risk management plan?
TEAM	Team Cohesion	Do	How can we describe the relationship among the stakeholders?
PMAT	Process Maturity	Do	What is the SEI Maturity rating?

Table 2. Cost drivers for COCOMO2

Cost Driver	Description	Very Low	Low	Nominal	High	Very High	Extra High
RELY	Required Software reliability	.75	.88	1.0	1.15	1.4	-
DATA	Database Size	-	.94	1.0	1.08	1.16	-
CPLX	Product complexity	.70	.85	1.0	1.15	1.30	1.65
TIME	Execution time constraint	-	-	1.0	1.11	1.30	1.66
STOR	Main storage constraint	-	-	1.0	1.06	1.21	1.56
VIRT	Virtual machine volatility	-	.87	1.0	1.15	1.30	-
TURN	Computer turnaround time	-	.87	1.0	1.07	1.15	-
ACAP	Analyst capability	1.46	1.19	1.0	.86	.71	-

APEX	Applications experience	1.29	1.13	1.0	.91	.82	-
PCAP	Programmer capability	1.42	1.17	1.0	.86	.70	-
VEXP	Virtual machine experience	1.21	1.10	1.0	.90	-	-
LEXP	Language experience	1.14	1.07	1.0	.95	-	-
MODP	Modern programming practices	1.24	1.10	1.0	.91	.82	-
TOOL	Software Tools	1.24	1.10	1.0	.91	.83	-
SCED	Development Schedule	1.23	1.08	1.0	1.04	1.10	-
PLEX	Platform Experience	1.19	1.09	1.0	.91	.85	-
PCON	Personnel Continuity	1.29	1.12	1.0	.90	.81	-

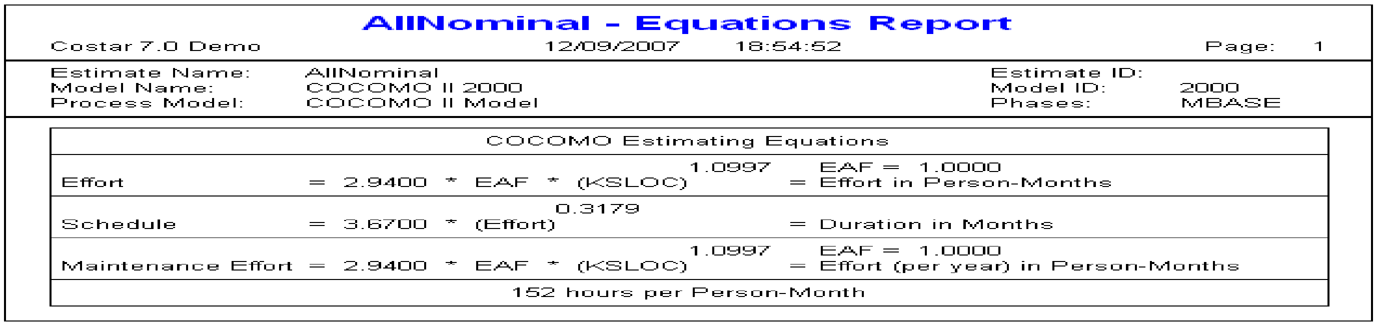
3. Results

3.1 COCOMO 2

The success of the results will be measured on how closely this approach can estimate the assurance effort in a balanced way. I have executed the case of a commercial software which has 5K SLOC. The model is executed having all the nominal value parameters at the first place followed by having reliability and schedule parameters changed from very low to very high in each case. This actually shows how the cost and schedule can be varied based on these two parameter values.

RELY and SCED tradeoff

All the parameters having nominal values (Recommended for novice project managers)

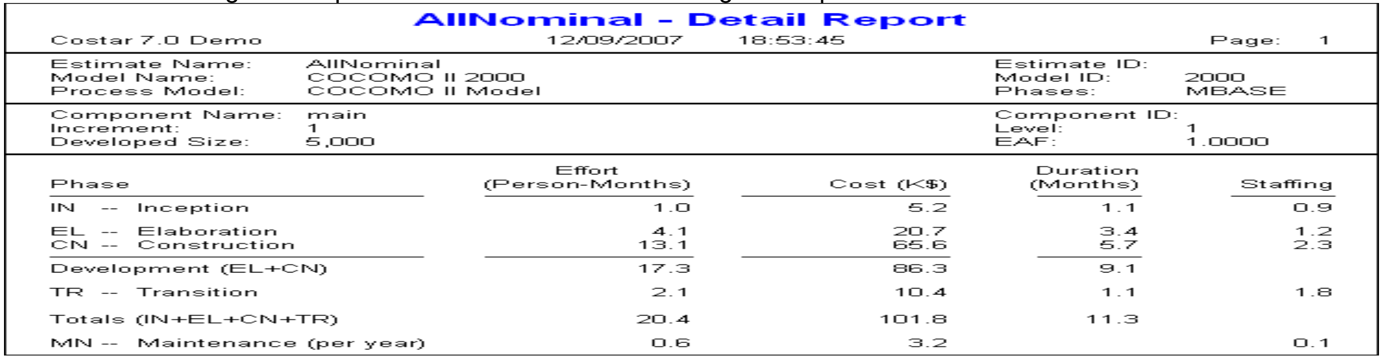


```
                    AllNominal - Equations Report
Costar 7.0 Demo                12/09/2007     18:54:52                     Page:   1

Estimate Name:      AllNominal                                  Estimate ID:
Model Name:         COCOMO II 2000                              Model ID:       2000
Process Model:      COCOMO II Model                             Phases:         MBASE

                        COCOMO Estimating Equations
                                                 1.0997     EAF =  1.0000
Effort             =  2.9400  *  EAF  *  (KSLOC)          =  Effort in Person-Months
                                      0.3179
Schedule           =  3.6700  *  (Effort)                 =  Duration in Months
                                                 1.0997     EAF =  1.0000
Maintenance Effort =  2.9400  *  EAF  *  (KSLOC)          =  Effort (per year) in Person-Months
                        152 hours per Person-Month
```

Figure1. Equations used in simulation having all the paremeters nominal



```
                    AllNominal - Detail Report
Costar 7.0 Demo                12/09/2007     18:53:45                     Page:   1

Estimate Name:      AllNominal                                  Estimate ID:
Model Name:         COCOMO II 2000                              Model ID:       2000
Process Model:      COCOMO II Model                             Phases:         MBASE

Component Name:     main                                        Component ID:
Increment:          1                                           Level:          1
Developed Size:     5,000                                       EAF:            1.0000

                                      Effort                        Duration
Phase                             (Person-Months)     Cost (K$)     (Months)     Staffing
IN  --  Inception                          1.0             5.2          1.1          0.9
EL  --  Elaboration                        4.1            20.7          3.4          1.2
CN  --  Construction                      13.1            65.6          5.7          2.3
Development (EL+CN)                       17.3            86.3          9.1
TR  --  Transition                         2.1            10.4          1.1          1.8
Totals (IN+EL+CN+TR)                      20.4           101.8         11.3
MN --  Maintenance (per year)              0.6             3.2                       0.1
```

Figure2. Detalied report of simulation having all the paremeters nominal

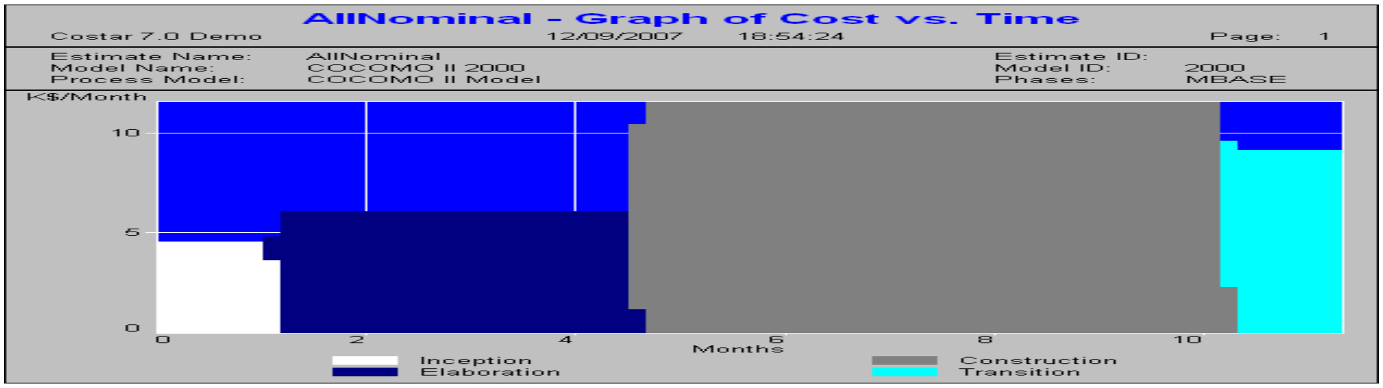
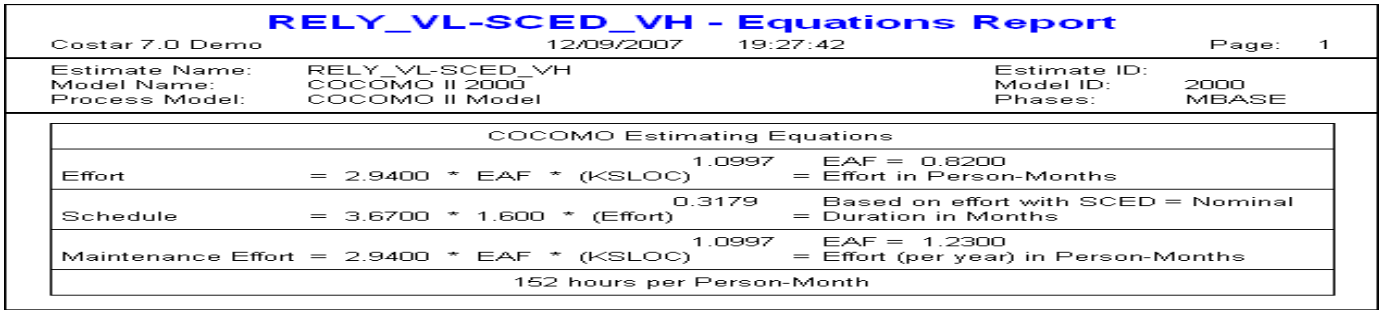


Figure3. Cost vs schedule graph having all the paremeters nominal

Scenario 1: RELY – very low and SCED – very high



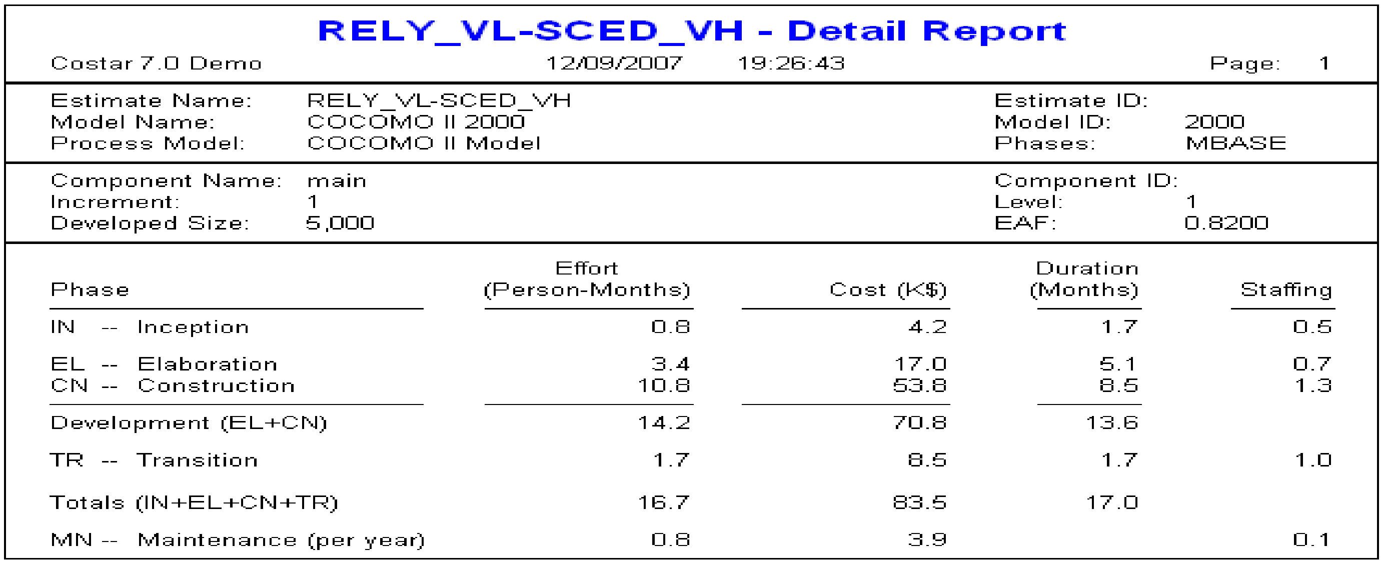
RELY_VL-SCED_VH - Equations Report

```
Costar 7.0 Demo                     12/09/2007    19:27:42                        Page:   1

Estimate Name:    RELY_VL-SCED_VH                                    Estimate ID:
Model Name:       COCOMO II 2000                                     Model ID:      2000
Process Model:    COCOMO II Model                                    Phases:        MBASE

                         COCOMO Estimating Equations
                                             1.0997     EAF =  0.8200
Effort             =  2.9400  *  EAF  *  (KSLOC)      =  Effort in Person-Months
                                          0.3179       Based on effort with SCED = Nominal
Schedule           =  3.6700  *  1.600  *  (Effort)    =  Duration in Months
                                             1.0997     EAF =  1.2300
Maintenance Effort =  2.9400  *  EAF  *  (KSLOC)      =  Effort (per year) in Person-Months
                         152 hours per Person-Month
```

Figure 4. Equations report for RELY very low and SCED very high



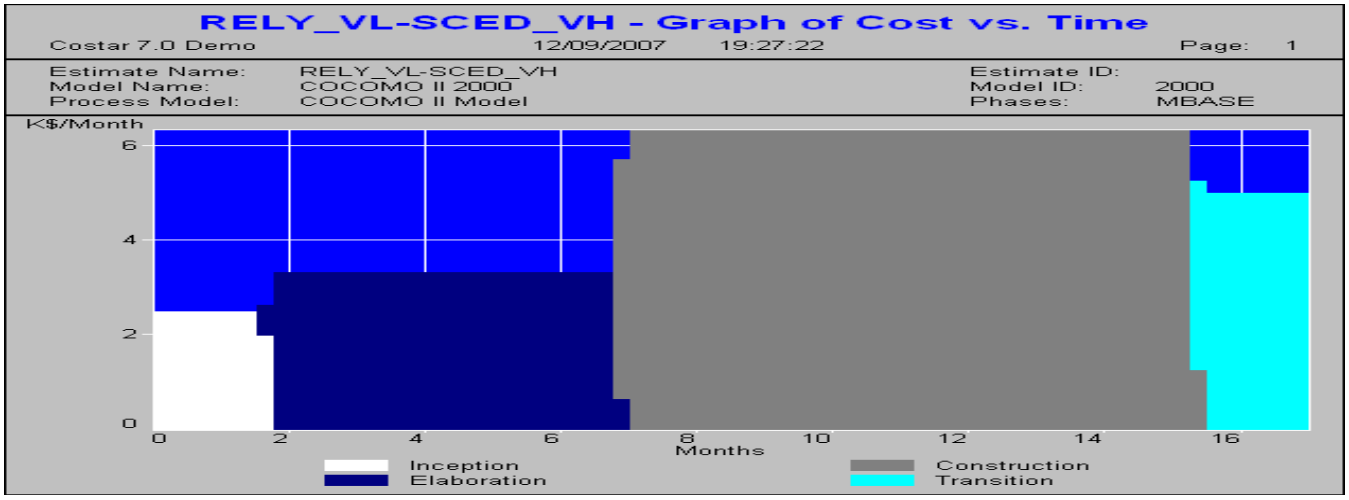
RELY_VL-SCED_VH - Detail Report

Costar 7.0 Demo 12/09/2007 19:26:43 Page: 1

Estimate Name:	RELY_VL-SCED_VH	Estimate ID:	
Model Name:	COCOMO II 2000	Model ID:	2000
Process Model:	COCOMO II Model	Phases:	MBASE
Component Name:	main	Component ID:	
Increment:	1	Level:	1
Developed Size:	5,000	EAF:	0.8200

Phase	Effort (Person-Months)	Cost (K$)	Duration (Months)	Staffing
IN -- Inception	0.8	4.2	1.7	0.5
EL -- Elaboration	3.4	17.0	5.1	0.7
CN -- Construction	10.8	53.8	8.5	1.3
Development (EL+CN)	14.2	70.8	13.6	
TR -- Transition	1.7	8.5	1.7	1.0
Totals (IN+EL+CN+TR)	16.7	83.5	17.0	
MN -- Maintenance (per year)	0.8	3.9		0.1

Figure 5. Detailed report for RELY very low and SCED very high



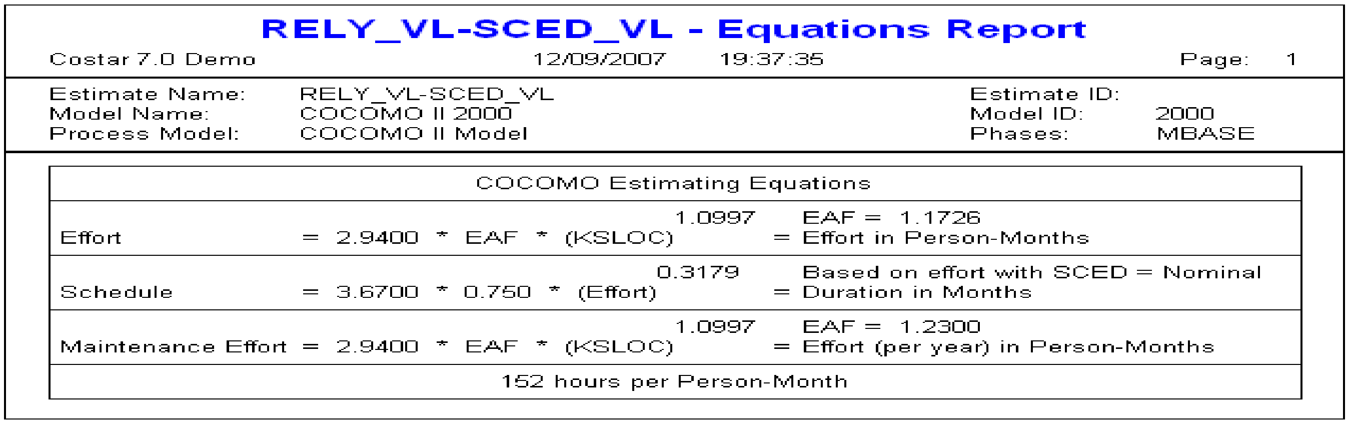
RELY_VL-SCED_VH - Graph of Cost vs. Time

Costar 7.0 Demo 12/09/2007 19:27:22 Page: 1

Estimate Name:	RELY_VL-SCED_VH	Estimate ID:	
Model Name:	COCOMO II 2000	Model ID:	2000
Process Model:	COCOMO II Model	Phases:	MBASE

Figure 6. Cost vs. time graph for RELY very low and SCED very high

Scenario 2: RELY- very low and SCED – very low



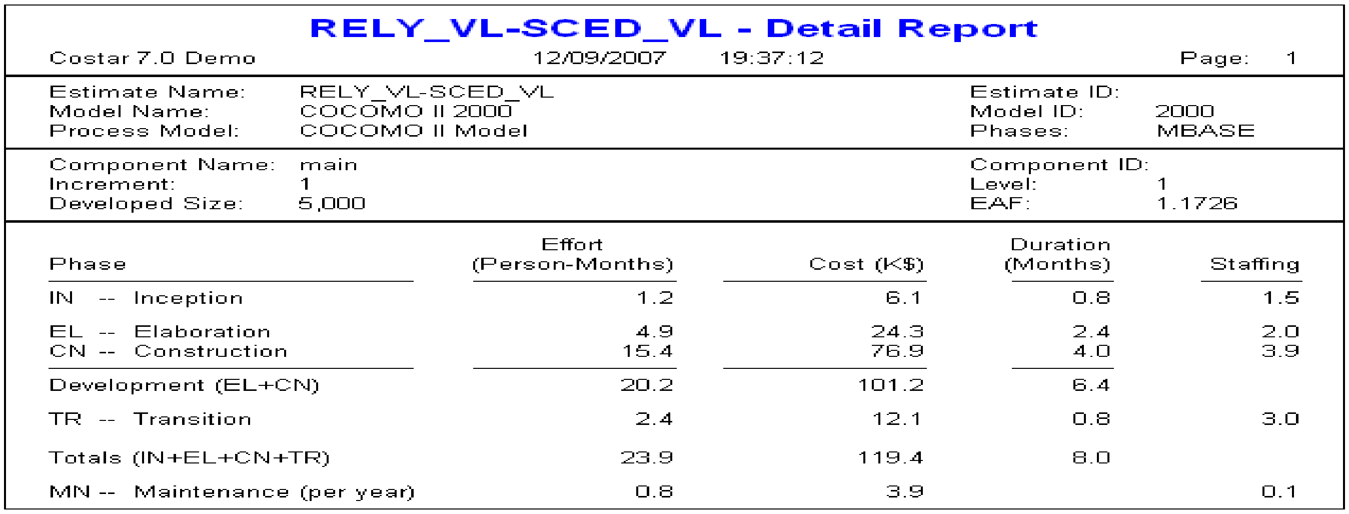
RELY_VL-SCED_VL - Equations Report

Costar 7.0 Demo 12/09/2007 19:37:35 Page: 1

Estimate Name:	RELY_VL-SCED_VL	Estimate ID:	
Model Name:	COCOMO II 2000	Model ID:	2000
Process Model:	COCOMO II Model	Phases:	MBASE

```
                      COCOMO Estimating Equations
                                          1.0997     EAF =  1.1726
Effort              =  2.9400  *  EAF  *  (KSLOC)     =  Effort in Person-Months
                                       0.3179        Based on effort with SCED = Nominal
Schedule            =  3.6700  *  0.750  *  (Effort)  =  Duration in Months
                                          1.0997     EAF =  1.2300
Maintenance Effort  =  2.9400  *  EAF  *  (KSLOC)     =  Effort (per year) in Person-Months
                        152 hours per Person-Month
```

Figure 7. Equations report for RELY very low and SCED very low



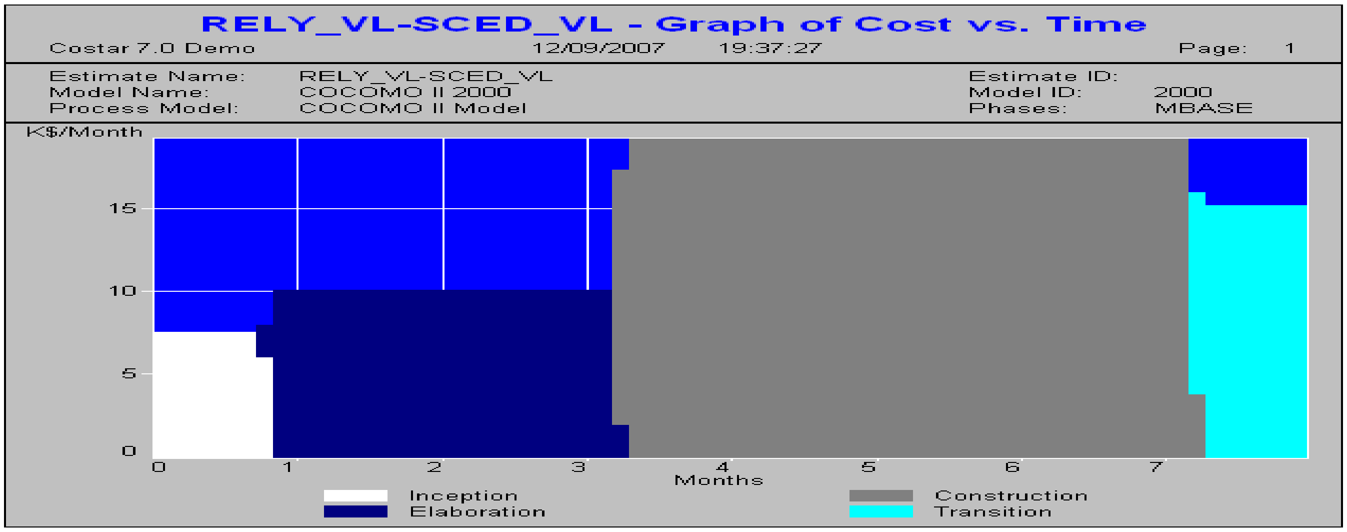
RELY_VL-SCED_VL - Detail Report

Costar 7.0 Demo 12/09/2007 19:37:12 Page: 1

Estimate Name:	RELY_VL-SCED_VL	Estimate ID:	
Model Name:	COCOMO II 2000	Model ID:	2000
Process Model:	COCOMO II Model	Phases:	MBASE
Component Name:	main	Component ID:	
Increment:	1	Level:	1
Developed Size:	5,000	EAF:	1.1726

Phase	Effort (Person-Months)	Cost (K$)	Duration (Months)	Staffing
IN -- Inception	1.2	6.1	0.8	1.5
EL -- Elaboration	4.9	24.3	2.4	2.0
CN -- Construction	15.4	76.9	4.0	3.9
Development (EL+CN)	20.2	101.2	6.4	
TR -- Transition	2.4	12.1	0.8	3.0
Totals (IN+EL+CN+TR)	23.9	119.4	8.0	
MN -- Maintenance (per year)	0.8	3.9		0.1

Figure 8. Detailed report for RELY very low and SCED very low

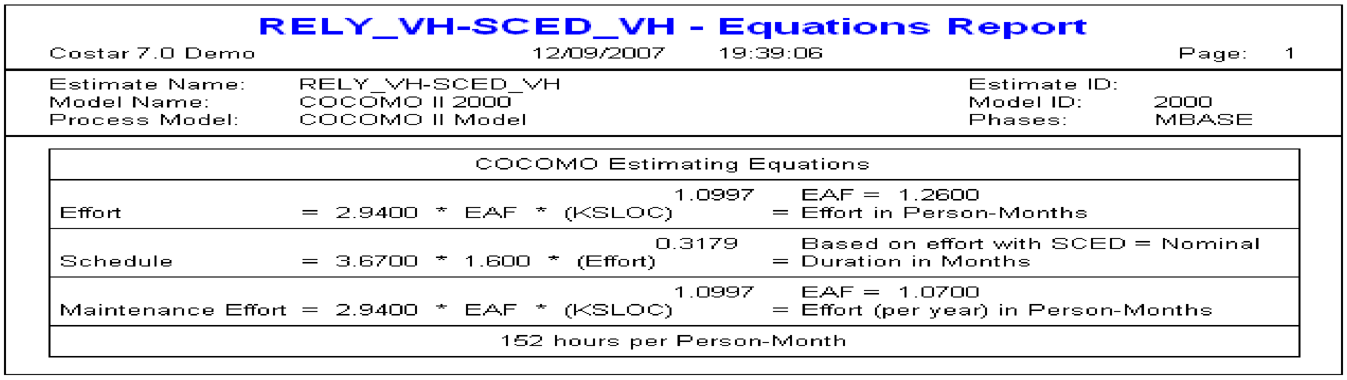


RELY_VL-SCED_VL - Graph of Cost vs. Time

Costar 7.0 Demo	12/09/2007 19:37:27	Page:	1
Estimate Name: RELY_VL-SCED_VL		Estimate ID:	
Model Name: COCOMO II 2000		Model ID:	2000
Process Model: COCOMO II Model		Phases:	MBASE

Figure 9. Cost vs. time graph for RELY very low and SCED very low

Scenario 3: RELY – very high and SCED – very high



RELY_VH-SCED_VH - Equations Report

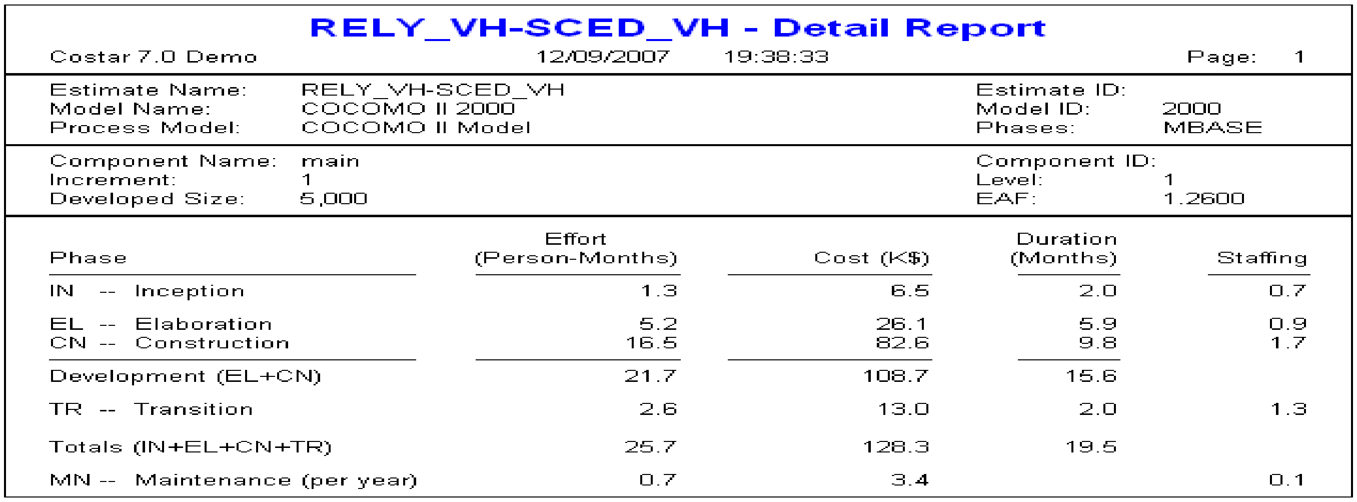
Costar 7.0 Demo	12/09/2007 19:39:06	Page:	1
Estimate Name: RELY_VH-SCED_VH		Estimate ID:	
Model Name: COCOMO II 2000		Model ID:	2000
Process Model: COCOMO II Model		Phases:	MBASE

COCOMO Estimating Equations

Effort	$= 2.9400 * \text{EAF} * (\text{KSLOC})^{1.0997}$	EAF = 1.2600 = Effort in Person-Months
Schedule	$= 3.6700 * 1.600 * (\text{Effort})^{0.3179}$	Based on effort with SCED = Nominal = Duration in Months
Maintenance Effort	$= 2.9400 * \text{EAF} * (\text{KSLOC})^{1.0997}$	EAF = 1.0700 = Effort (per year) in Person-Months

152 hours per Person-Month

Figure 10. Equations report for RELY very high and SCED very high

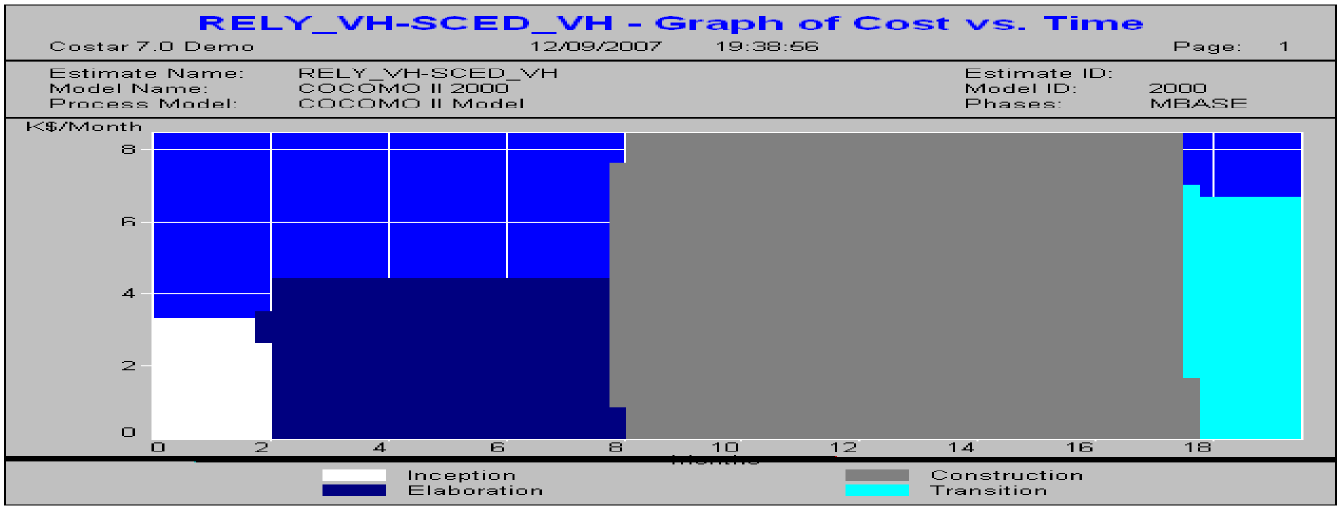


RELY_VH-SCED_VH - Detail Report

Costar 7.0 Demo	12/09/2007 19:38:33	Page:	1
Estimate Name: RELY_VH-SCED_VH		Estimate ID:	
Model Name: COCOMO II 2000		Model ID:	2000
Process Model: COCOMO II Model		Phases:	MBASE
Component Name: main		Component ID:	
Increment: 1		Level:	1
Developed Size: 5,000		EAF:	1.2600

Phase	Effort (Person-Months)	Cost (K$)	Duration (Months)	Staffing
IN -- Inception	1.3	6.5	2.0	0.7
EL -- Elaboration	5.2	26.1	5.9	0.9
CN -- Construction	16.5	82.6	9.8	1.7
Development (EL+CN)	21.7	108.7	15.6	
TR -- Transition	2.6	13.0	2.0	1.3
Totals (IN+EL+CN+TR)	25.7	128.3	19.5	
MN -- Maintenance (per year)	0.7	3.4		0.1

Figure 11. Detailed report for RELY very high and SCED very high



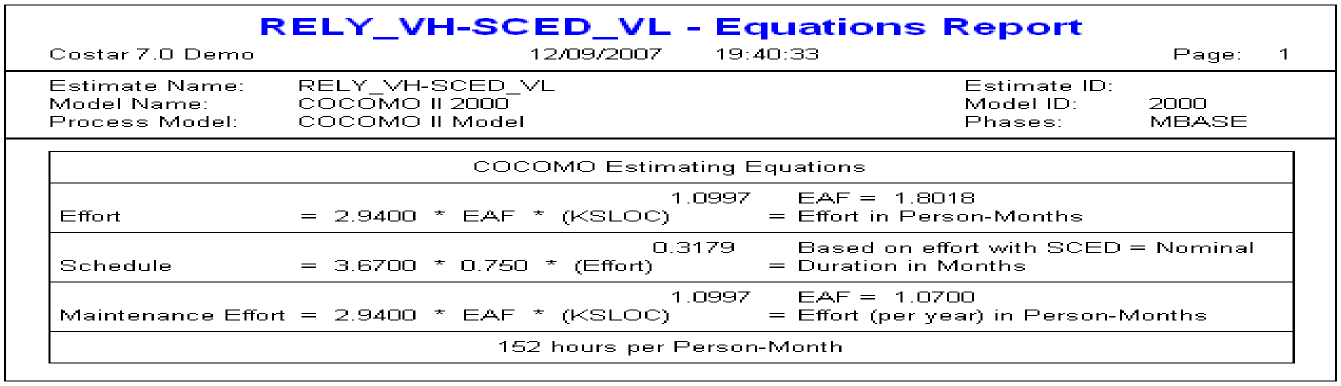
RELY_VH-SCED_VH - Graph of Cost vs. Time

```
Costar 7.0 Demo                    12/09/2007      19:38:56                     Page:    1

Estimate Name:      RELY_VH-SCED_VH                              Estimate ID:
Model Name:         COCOMO II 2000                               Model ID:      2000
Process Model:      COCOMO II Model                              Phases:        MBASE
```

Figure 12. Cost vs. time graph for RELY very high and SCED very high

Scenario 4: RELY – very high and SCED – very low



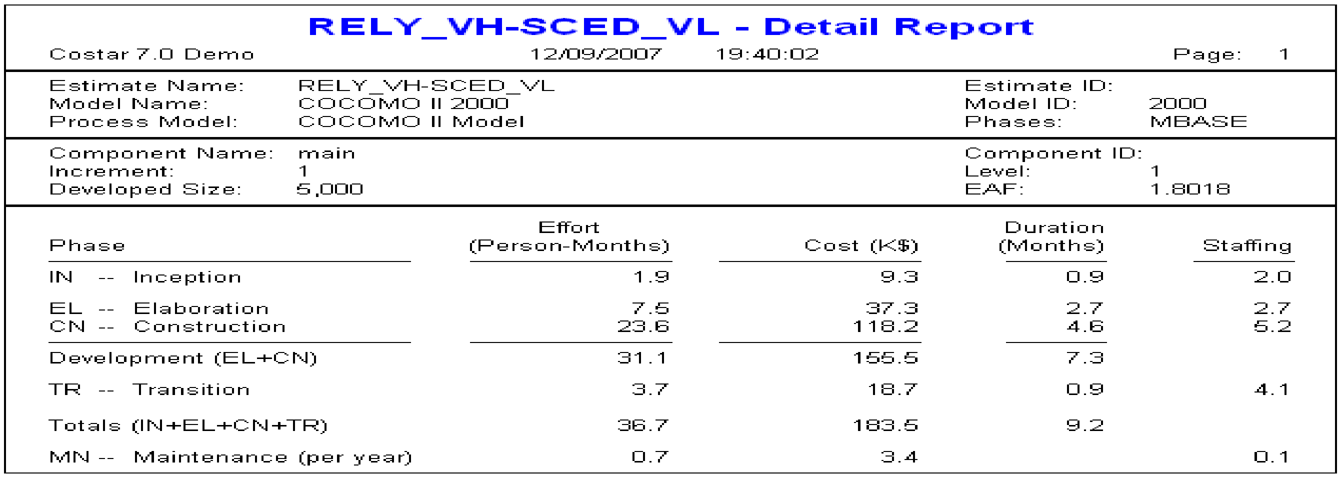
RELY_VH-SCED_VL - Equations Report

```
Costar 7.0 Demo                    12/09/2007      19:40:33                     Page:    1

Estimate Name:      RELY_VH-SCED_VL                              Estimate ID:
Model Name:         COCOMO II 2000                               Model ID:      2000
Process Model:      COCOMO II Model                              Phases:        MBASE
```

COCOMO Estimating Equations	
Effort = 2.9400 * EAF * (KSLOC)$^{1.0997}$	EAF = 1.8018 = Effort in Person-Months
Schedule = 3.6700 * 0.750 * (Effort)$^{0.3179}$	Based on effort with SCED = Nominal = Duration in Months
Maintenance Effort = 2.9400 * EAF * (KSLOC)$^{1.0997}$	EAF = 1.0700 = Effort (per year) in Person-Months
152 hours per Person-Month	

Figure 13. Equations report for RELY very high and SCED very low



RELY_VH-SCED_VL - Detail Report

```
Costar 7.0 Demo                    12/09/2007      19:40:02                     Page:    1

Estimate Name:      RELY_VH-SCED_VL                              Estimate ID:
Model Name:         COCOMO II 2000                               Model ID:      2000
Process Model:      COCOMO II Model                              Phases:        MBASE

Component Name:     main                                         Component ID:
Increment:          1                                            Level:         1
Developed Size:     5,000                                        EAF:           1.8018
```

Phase	Effort (Person-Months)	Cost (K$)	Duration (Months)	Staffing
IN -- Inception	1.9	9.3	0.9	2.0
EL -- Elaboration	7.5	37.3	2.7	2.7
CN -- Construction	23.6	118.2	4.6	5.2
Development (EL+CN)	31.1	155.5	7.3	
TR -- Transition	3.7	18.7	0.9	4.1
Totals (IN+EL+CN+TR)	36.7	183.5	9.2	
MN -- Maintenance (per year)	0.7	3.4		0.1

Figure 14. Detailed report for RELY very high and SCED very low

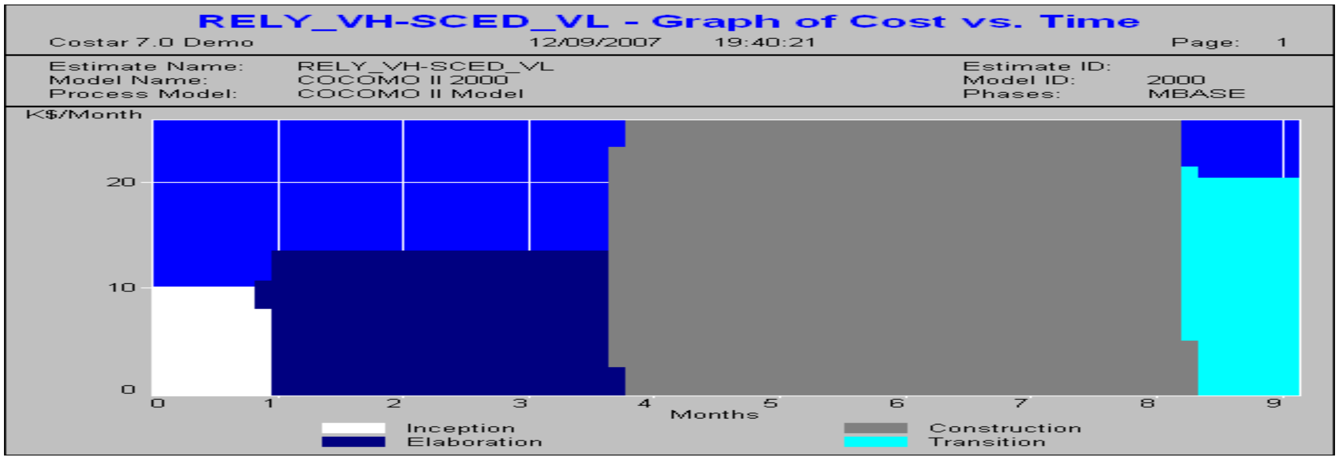


Figure 15. Cost vs. time graph for RELY very high and SCED very low

Comparison Chart:

Table 3. Summary of cost and time for different RELY and SCED levels

Cost(K$)/Time(Month)		RELY		
		VL	VH	N
SCED	VL	119.4/8	183.5/9.2	
	VH	83.5/17	128.3/19.5	
	N			101.8/11.3

3.2 Determining the sweet-spot

COQUALMO gets the probability of loss with is multiplied by size of loss due to quality factors to get the Risk exposure.

RE =Probability (Loss) x Size (Loss)

RE is added with market share erosion factor to get the combined risk exposure. RE and combined RE then used to determine the optimum point of assurance effort.

Table 4. Risk Exposure for different RELY levels for commercial software project

RELY	RE = P(L)xS(L)	P(Loss)	S(Loss)	Market Share Erosion	Combined RE =RE+MSE
			Commercial		
VL	1	1	1	0.008	1.008
L	0.191	0.360	0.53	0.027	0.218
N	0.055	0.190	0.29	0.09	0.145
H	0.017	0.081	0.21	0.3	0.317
VH	0.009	0.072	0.12	1	1.009
XH	0	0	0		

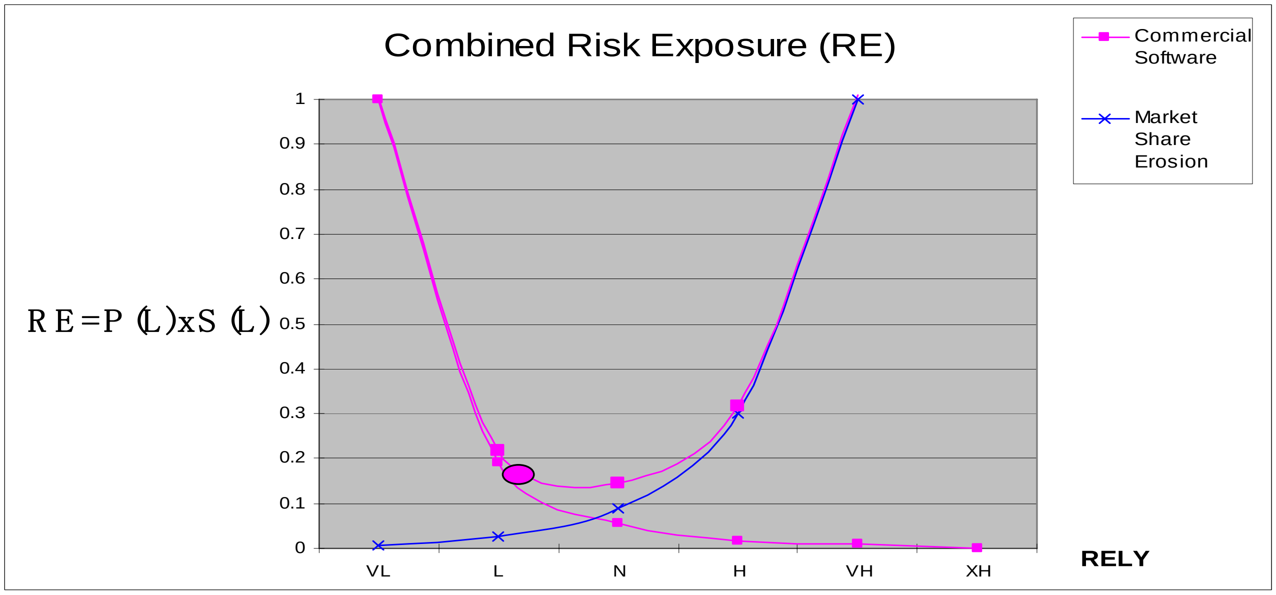


Figure16. Finding optimum level of assurance effort for commercial software product

3.3 Indication

The results above indicate that cost and money resource needed for software assurance may be varied to a large extent based on the reliability the customers are looking for on the resultant product. Not a single solution should be followed in every situation.

4. Discussion

Table 3 shows the cost and time needed for the commercial software with different RELY and SCED levels. The cost of the same project can be varied up to 183.5- 83.5=100K dollars and the time can be up to 11.5 months. If the customers are looking for a software with extremely high reliability within a tight schedule, the cost is the highest. Intuitively, low RELY-rating softwares demand less cost. If the required reliability is very high but the project doesn't have a strict deadline, the project manager can make it 19.5 months in duration and save 183.5-128.3= 55.2 K dollars. In the same manner, if the required reliability is very low and schedule is not an issue, the project can save 119.4=83.5= 35.9 K but the duration will be 17 months.

If the customers of the software are looking for the release as soon as possible but are not picky about the reliability inherent on this, it can be finished within 8 months but it will cost 119.4-83.5 =35.9 K dollar more than that of the relaxed-schedule estimate.

Table 4 shows the probability of loss and the size of loss for every RELY level. The higher the RELY level is, the lower is the probability and size of loss are. It's due to the risk balanced approach for assurance. Higher RELY level basically means the lower number of defects present after doing the defect removal operation on the software. On the contrary, the lower the RELY level is, the lower the market share erosion is. This factor follows a specific trend consistent with the present economy and product value and very important to make decision about the release time of the product.

5. Conclusion

At this article, I have tried to balance the risk exposure of doing too much with risk exposure of doing too little. As the test case is a commercial software, I have shown the relative risk exposure with market erosion which are combined with reliability level to get the optimum software assurance. Same methodologies apply for other types of softwares such as high finance or low-profile softwares which are not discussed here. Not the single solution is applicable for all the types of softwares but the assurance can be determined by using the same methodology having different values for the parameters.

This article is the combination of concepts of quality and cost estimation models, business value estimating relationship and trade-off analysis to have a quantitative approach on optimal software assurance investment level in terms of time, money and human resource.

6. References

[1] Boehm, B., *Software Engineering Economics*, Prentice-Hall, Inc., 1981.

[2] Wikipedia – the free encyclopedia. (2007). Retrieved December 12, 2007, from http://en.wikipedia.org/wiki/Software_Assurance.

[3]B. Boehm and L. Huang, "Value-Based Software Engineering: A Case Study", IEEE Computer, vol. 36, no. 3, March 2003, pp. 33-41.