CORADMO Extensions of COCOMO II Schedule

Introduction

The Center for Software Engineering at the University of Southern California is conducting research to update the software development cost estimation model called COCOMO. The project name is COCOMO II and is led by Dr. Barry W. Boehm.

A fundamental requirement for such research is real-world software development project data. This data will be used to test hypotheses and verify the model's postulations. In return the model will be open and made available to the public. The contribution of your data will ensure the final model is useful.

The data that is contributed is important to us. We will safeguard your contribution so as not to compromise company proprietary information. Some Affiliates have an active collection program, and the data from past projects is available for the COCOMO II data collection efforts. This questionnaire can be used to extract relevant CORADMO data.

This questionnaire addresses only a project level of data granularity. The project level of granularity is data that is applicable for the whole project. This includes things like application type and development activity being reported.

This questionnaire has three sections. The first section includes general and project-level COCOMO II related questions. The second section is for summary information of an extension of COCOMO II, COPSEMO (**CO**COMO **P**hase **S**chedule and **E**ffort **Mo**del), the preprocessor for CORADMO (**CO**nstructive **RAD MO**del). The third and last section is for CORADMO itself, another extension of COCOMO II. If you have <u>not</u> submitted regular COCOMO-II data on this project yet, a copy of the form is available from the Points of Contact identified below.

The data collection activity for the COCOMO II research effort started in November 1994. The first calibration was published in 1997 based on 83 datapoints collected. It became popular as COCOMO II.1997 and produced estimates within 30% of the actuals 52% of the time for effort. The second calibration was published in 1998 based on 161 datapoints. It is known as COCOMO II.1998 and produces estimates within 30% of the actuals 71% of the time for effort. The aim of the COCOMO II research team is to continually update the existing COCOMO II database and to publish annual calibrations of the COCOMO II model. Hence by submitting your data to us, you play a significant role in the model calibration.

COCOMO II Points of Contact

For questions on the COCOMO II Model and its extensions, data definitions, or project data collection and management, contact:

A Winsor Brown (Research Scientist)	Voice: (213) 740-6599, Fax: (213) 740-4927
Cyrus Fakharzadeh (Research Assistant)	Voice: (213) 740-5703, Fax: (213) 740-4927
Barry Boehm (Project Leader)	Voice: (213) 740-8163, Fax: (213) 740-4927
Internet Electronic-Mail	*cocomo-info@sunset.usc.edu*

COCOMO II Data Submission Address:

COCOMO II Data Submission
Center for Software Engineering
Department of Computer Science
Henri Salvatori Room 330
University of Southern California
941 W. 37th Place
Los Angeles, CA 90089-0781
U.S.A.

[1] COnstructive RAD schedule and effort MOdel

[2] Constructive Cost Modeling (COCOMO) is defined in <u>Software Engineering Economics</u> by Barry W. Boehm, Prentice Hall, 1981

CORADMO Extensions of COCOMO II Schedule

1. Project Level Information

As described in the Introduction section of this questionnaire, project level information is applicable for the whole project. This includes things like application type and development activity being reported. As this is a questionnaire, fill in the appropriate information in the spaces provided.

1.A. General Information

1.A.1. Affiliate Identification Number Each separate software project contributing data will have a separate file identification number of the form XXX. XXX will be one of a random set of three-digit organization identification numbers, provided by USC Center for Software Engineering to the Affiliate.

____

1.A.2. Project Identification Number The project identification is a three digit number assigned by the organization. Only the Affiliate knows the correspondence between YYY and the actual project. The same project identification must be used with each data submission.

____

1.A.3. Date prepared This is the date the data elements were collected for submission.

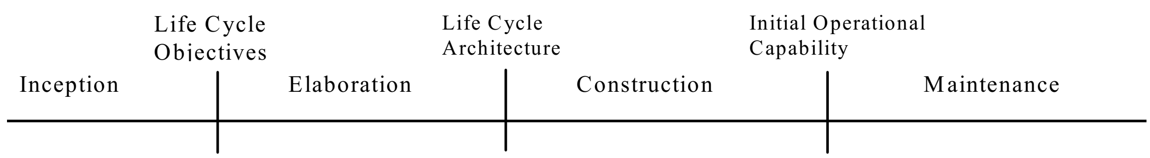
____

1.B. Schedule Year of development. For reporting of historical data, please provide the year in which the software development was completed. For periodic reporting put the year of this submission or leave blank.

____

1.C.1. Schedule Months. For reporting of historical data, provide the number of calendar months from the time the development began through the time it completed, i.e. from Life-Cycle Objectives through Initial Operation Capability. For periodic reporting, provide the number of months in this development activity.

Circle the life-cycle phases that the schedule covers:

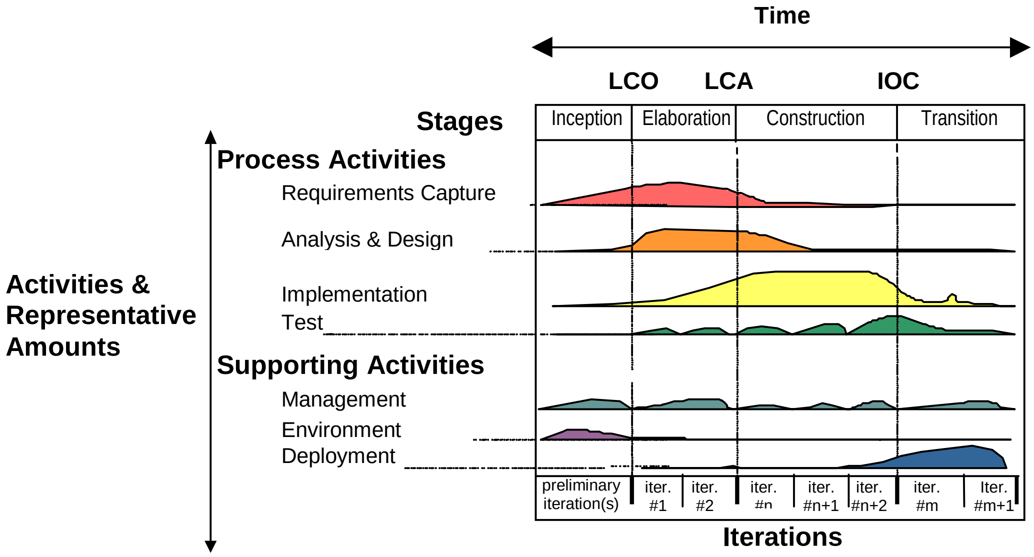


See the Appendix A for definitions of the LCO, LCA, and IOC milestones. The COCOMO II model covers the effort required from the completion of the LCO to IOC. If you are using a waterfall model, the corresponding milestones are the Software Requirements Review, Preliminary Design Review, and Software Acceptance Test.

Schedule in months: ___________________

2. COCOMO Phase Schedule and Effort MODEL (COPSEMO)

COPSEMO is based on the lifecycle anchoring concepts discussed by Boehm[3]. The anchor points are defined as Life Cycle Objectives (LCO), Life Cycle Architecture (LCA), and Initial Operational Capability (IOC). An enhanced version of an illustration from Rational Corporation[4] showing the phases around the anchor points is shown below.



The correspondence between COPSEMO's & CORADMO's "Phases", COCOMOII's submodels and the life cycle anchor points is shown in the following table along with an indication of the relative amounts of the different activities.

COCOMO II Submodel Usage	**Early Design**	**Post-Architecture**		**Maintenance**
	LCO	**LCA**	**IOC**	
Activities \ Phase	**Inception**	**Elaboration**	**Construction**	**Transition**
Requirements Capture	Some usually	Most, peaks here	Minor	None
Analysis & Design	A little	Majority, mostly constant effort	Some	Some, for repair during ODT&E
Implementation	Practically none	Some, usually for risk reduction	Bulk; mostly constant effort	Some, for repair during ODT&E
Test	None	Some, for prototypes	Most for unit, integration and qualification test.	Some, for repaired code.

COCOMOII's effort and schedule estimates are focused on Elaboration and Construction (the phases between LCO and IOC. Inception corresponds to the COCOMO's "Requirements" activity in a waterfall process model. COCOMO's effort for the "Requirements" activity is an additional, fixed percentage of the effort calculated by COCOMO for the development activities. The table also indicates the areas in which the COCOMO II Early Design and Post-Architecture submodels are normally used.

Allocations NEED TO ADD Transition!

---

[3] Barry W. Boehm, "Anchoring the Software Process," *IEEE Software,* 13, 4, July 1996, pp. 73-82

[4] Rational Corp., "Rational Objectory Process 4.1 – Your UML Process", available at http://www.rational.com/support/techpapers/toratobjprcs/.

CORADMO Extensions of COCOMO II Schedule

2.A.1. <u>Percentage Effort per Phase.</u> Allocate the effort (person months) used in each of the phases as a percentage of the total effort during Elaboration and Construction. The sum of the percentages of Elaboration and Construction should be 100%. The effort during Inception (as a percentage of total Elaboration and Construction) is added to get the Total IE&C which should be greater than 100%.

	LCO	LCA	IOC		
Phase	**Inception**	**Elaboration**	**Construction**	**Total E & C**	**Total I E & C**
%Effort				**100%**	

2.A.2. <u>Percentage Schedule per Phase.</u> Allocate the schedule (calendar months) for each of the phases as a percentage of the total schedule during Elaboration and Construction. The sum of Elaboration and Construction should be 100%. The schedule during Inception (as a percentage of total Elaboration and Construction) is added to get the Total IE&C which should be greater than 100%.

	LCO	LCA	IOC		
Phase	**Inception**	**Elaboration**	**Construction**	**Total E & C**	**Total I E & C**
%Schedule				**100%**	

2.A.3. <u>Person-Power per Phase.</u> Indicate the average number of people actually working during this period of each of the phases. If the loading was not approximately constant during the period except for typical, limited ramp-ups, please indicate the degree of variation by providing the Persons-Max and Persons-Min, and the number of months with that number of people (max and min, respectively). NOTE: summing persons across phases is illogical and incorrect.

		LCO		LCA		IOC		
Phase	**Inception**		**Elaboration**		**Construction**		**Total E & C**	**Total I E & C**
Persons-Ave.							**X**	**X**
	Heads	Mon.	Heads	Mon.	Heads	Mon.	**X**	**X**
Persons-Max							**X**	**X**
Persons-Min							**X**	**X**

3. COCOMO RAD MODEL (CORADMO)

The COCOMO RAD model has its roots in the results of a 1997 CSE Focused Workshop on Rapid Application Development[5]. RAD is taken to mean an application of any of a number of techniques or strategies to reduce software development cycle time. A "RAD Opportunity Tree" presented at the workshop identified five classes of strategies whose degree of implementation can be used to parameterize a schedule estimate given an effort estimate produced by COCOMO II-1998. These strategies are preferable to just adding people to the task. The five classes are: development process re-engineering (DPRS), re-use and very high level languages (RVHL), collaboration efficiency (CLAB), architecture investment and risk resolution (RESL), and pre-positioning of assets (PPOS). RESL corresponds to the COCOMO II scale driver; the other four are new. All have their effects reflected as multipliers on effort (person months, PM), schedule (months, M) and/or number of personnel (P). Person months of effort can actually be increased because certain pro-active strategies, like pre-positioning of assets, are only possible with extra effort.

The COCOMO RAD model utilizes the COCOMO extension which allocates effort and schedule to the phases which are anchored at the LCO/LCA/IOC points in a development life cycle. A phased schedule and effort distribution is needed because the effects of the RAD strategies identified above is different for the different phases. Also, a new mathematical function is used to calculate (predict) the calendar months for a given amount of effort: the function is only radically different in low (under 16) person-month's efforts where it seems more normal have an equal number of people and months to accomplish the task. At the higher (greater than 64) person-month's efforts, the traditional COCOMO II-1998 function is used which is based on the traditional cube-root-like function of effort. A smooth curve is fit within these ranges.

The intent of the COCOMO II RAD model is to calculate/predict the schedule (months, M), personnel (P), and adjusted effort (person-months, PM) based on the distribution of effort and schedule to the various phases, and impacts of the selected schedule driver ratings on the M, P, and PM of each phase.

3.A.1. Reuse and VHLL's (RVHL) The degree to which re-use of other than code and/or very high level languages are utilized. This driver reflects schedule compression in Inception and Elaboration phases due to faster prototyping or option exploration. The rating for this driver depends on the amount of Rapid Prototyping Experience the development team has had in the domain of the project being evaluated. Since the rating applies to the team, it must include the experience of the managers and team leaders and their experience takes precedence over the average of the rest of the team working in the Inception and Elaboration phases.

RVHL		**Very Low**	**Low**	**Nominal**	**High**	**Very High**
Don't Know	**N/A - Not Applic-able**	none	On average, personnel have experience on less than one recent project using Rapid Prototyping	most personnel have worked on more than one project using Rapid Prototyping	on average, personnel have worked on more than two projects using Rapid Prototyping	all personnel have worked on at least three projects using Rapid Prototyping

N/A rationale:____

[5] B. Boehm, S. Chulani, and A. Eyed, "Knowledge Summary: USC-CSE Focused Workshop on Rapid Application Development", USC-CSE Technical Report, June 1997.

3.A.2. Development Process Reengineering and Streamlining (DPRS) The degree to which the project and organization allow and encourage streamlined or re-engineered development processes: the current level of bureaucracy is a clear indicator. The schedule compression or expansion, because of this driver, doesn't alter staff level (P). The following table can be used to make a subjective average to determine the level of bureaucracy.

Level of Bureaucracy Indicators	Very Low	Low	Nominal	High	Very High
Number of approvals required per task	Excessive	Occasionally Reduced	Mature	Actively Reduced	Actively Minimized
Time taken per approval	Excessive	Occasionally Reduced	Mature	Actively Reduced	Actively Minimized
Reduced task dependencies, critical path tasks	None	Little	Mature Tech. Adopted	Advanced Tech. Adopted	Pioneering
Follow-up to expedite task completion	None	Little	Encouraged	Emphasized	Strongly Emphasized
Process measurement & streamlining	None	Little	Mature Tech. Adopted	Advanced Tech. Adopted	Pioneering
Level of Bureaucracy	Heavily Bureaucratic	Bureaucratic	Basic good business practices	Partly streamlined	Fully streamlined

DPRS		Very Low	Low	Nominal	High	Very High
Don't Know	**Not Applic-able**	Heavily Bureaucratic	Bureaucratic	Basic good business practices	Partly streamlined	Fully streamlined

N/A rationale:____

3.A.3. Collaboration Efficiency (CLAB) Teams and team members who can collaborate effectively can reduce both effort and schedule; those that don't collaborate effectively have increased schedule and effort (due to wasted time). With this multiplier, staff level does not change based on collaboration efficiency.

Collaboration efficiency is clearly impacted by TEAM and SITE ratings. Collaboration efficiency is impacted by TOOL, but only for tools that support or enable collaboration. However, the tool technology impact is lessened in the case of a co-located team with high experience ratings (PREX, the combination of application, platform, language and tool experience taken from the early design ratings).

3.A.3.1. Team Cohesion (TEAM). The Team Cohesion cost driver accounts for the sources of project turbulence and extra effort due to difficulties in synchronizing the project's stakeholders: users, customers, developers, maintainers, interfacers, others. See the Model Definition Manual for more details.

Team		Very Low	Low	Nominal	High	Very High	Extra High
Don't Know	**N/A - Not Applicable**	Very difficult interactions	some difficult interactions	basically cooperative interactions	largely cooperative	highly cooperative	seamless interactions

N/A rationale:____

3.A.3.2. Multisite Development (SITE). Given the increasing frequency of multisite developments, and indications that multisite development effects are significant, the SITE cost driver has been added in COCOMO II. Determining its cost driver rating involves the assessment and averaging of two factors: site collocation (from fully collocated to international distribution) and communication support (from surface mail and some phone access to full interactive multimedia). See the COCOMO-II User's Manual. We recommend a 70% and 30% weightings for Collocation and Communications, respectively, when making your subjective average of these two components of SITE.

	Very Low	Low	Nominal	High	Very High	Extra High
SITE: Collocation	Inter-national	Multi-city and Multi-company	Multi-city or Multi-company	Same city or metro area	Same building or complex	Fully collocated
SITE: Communications	Some phone, mail	Individual phone, FAX	Narrowband email	Wideband electronic communication	Wideband elect. comm., occasional video conf.	Interactive multimedia

Don't Know	N/A	Very Low	Low	Nominal	High	Very High	Extra High

N/A rationale:____

3.A.3.3. Applications Experience (AEXP). This rating is dependent on the level of applications experience of the project team developing the software system or subsystem. The ratings are defined in terms of the project team's equivalent level of experience with this type of application. See the COCOMO-II User's Manual.

AEXP		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	≤ 2 months	6 months	1 year	3 years	≥6 years

N/A rationale:____

3.A.3.4. Platform Experience (PEXP). The Post-Architecture model broadens the productivity influence of PEXP, recognizing the importance of understanding the use of more powerful platforms, including more graphic user interface, database, networking, and distributed middleware capabilities. See the COCOMO-II User's Manual.

PEXP		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	≤ 2 months	6 months	1 year	3 years	≥6 years

N/A rationale:____

3.A.3.5. Language and Tool Experience (LTEX). This is a measure of the level of programming language and software tool experience of the project team developing the software system or subsystem. See the COCOMO-II User's Manual.

LTEX		Very Low	Low	Nominal	High	Very High
Don't Know	N/A - Not Applicable	≤ 2 months	6 months	1 year	3 years	≥6 years

N/A rationale:____

3.A.3.6. Personnel Experience (PREX) This Early Design cost driver combines the three Post-Architecture cost drivers application experience (AEXP), platform experience (PEXP), and language and tool experience (LTEX). While these three Post-Architecture ratings normally apply to a module, for CoRADMo they are applied across the entire project. Their individual rating information is given above.

The approach for mapping the Post-Architecture cost drivers and rating scales onto their Early Design model counterparts involves the use and combination of numerical equivalents of the rating levels. Specifically, a Very Low Post-Architecture cost driver rating corresponds to a numerical rating of 1, Low is 2, Nominal is 3, High is 4, Very High is 5, and Extra High is 6. For the combined Early Design cost drivers, the numerical values of the contributing Post-Architecture cost drivers are summed, and the resulting totals are allocated to an expanded Early Design model rating scale going from Extra Low to Extra High. The Early Design model rating scales always have a Nominal total equal to the sum of the Nominal ratings of its contributing Post-Architecture elements.

The table below assigns PREX ratings across this range, and associates appropriate effort multipliers and rating scales to each of the rating levels.

PREX		Extra Low	Very Low	Low	Nominal	High	Very High	Extra High
Sum of AEXP, PEXP, and LTEX ratings		3, 4	5, 6	7, 8	9	10, 11	12, 13	14, 15
Applications, Platform, Language and Tool Experience		≤ 3 mo.	5 months	9 months	1 year	2 years	4 years	≥ 6 years
Don't Know	**N/A - Not Applicable**							

To determine the CLAB rating, take the subjective/fuzzy average of TEAM and SITE ratings from COCOMO II's post-architecture definitions and the PREX ratings using COCOMO II's Early Design definitions.

	Very Low	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
SITE	<== COCOMO II Post-Arch. Ratings ==>				High plus negotiation/tradeoff tools basic	advanced
TEAM	<=== <=== <=== COCOMO II Scale Factor Ratings ===> ===> ===>					
PREX	(EL & VL) <=== <=== <=== COCOMO II Early Design Ratings ===> ===> ===>					
Fuzzy Average						

CLAB		**Very Low**	**Low**	**Nominal**	**High**	**Very High**	**Extra High**
Don't Know	**Not Applic-able**	<== Pick most appropriate rating level based on fuzzy average ==>					

N/A rationale:____

3.A.4. Architecture & Risk Resolution (RESL) This rating is exactly the same as the COCOMO II RESL rating. The architecture portion enables parallel construction, thus reducing schedule during the construction phase assuming that staff level increases during construction while applying the same effort. Good risk resolution in a schedule driven development effort applying RAD strategies increases the probability of the strategies success.

RESL		Very Low	Low	Nominal	High	Very High	Extra High
Don't Know	**N/A - Not Applicable**	<== Use COCOMO II's RESL Rating Level ==>					

N/A rationale:____

3.A.5. Prepositioning Assets (PPOS) This driver assesses the degree to which assets are pre-tailored to a project and furnished to the project for use on demand. This clearly has impacts from people skills and team building. The assets that are being pre-positioned include processes and tools, and architecture and componentry.

In order to take advantage of PPOS, the organization must either be taking a product-line approach or have made a 3, 6 or 10% pre-Inception effort investment!

PPOS		Nominal	High	Very High	Extra High
Don't Know	**N/A - Not Applicable**	Basic project legacy, no tailoring	Some prepositioning & tailoring	Key items prepositioned & tailored	All items prepositioned & tailored

N/A rationale:____

