

15th International Forum
on COCOMO and Software Cost Modeling

Software Estimation Experiences at Xerox

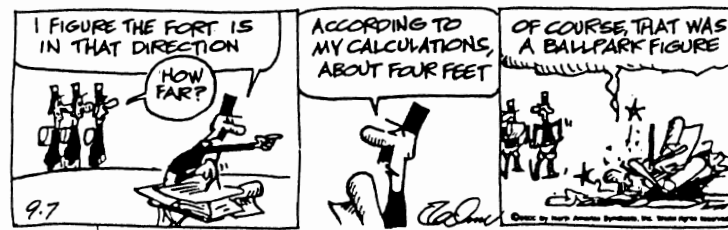
*Dr. Peter Hantos
Office Systems Group, Xerox*

1

Theme

Is making bad estimates a crime?
No, but it is certainly not victimless ...

CROCK By Bill Rechin and Don Wilder



2

Objectives

✓ Focus on recent piloting efforts

- On corporate/business group level
- On project level

✓ Address issues from a wide variety of angles:

- Theoretical
- Practical
- Cultural
- Management
- Other, "soft" issues

3

Agenda

- Objectives
- A Corporate-Level project
- Business Group and Project-Level pilot experiences
- Cultural and Management barriers
- Sizing issues
- Conclusions
- Acknowledgement
- References

4

A Corporate-Level Project

✓ Supporting the Schedule Slip-Rate Metrics

– Slip-Rate Definitions:

- Cycle Time = Actual End Date – Actual Start Date + 1
- Planned Duration = Planned End Date – Actual Start Date + 1
- Schedule Slip Rate =
$$\frac{\text{Actual End Date} - \text{Planned End Date} + 1}{\text{Planned Duration}}$$

5

Slip-Rate Issues

✓ Software Slip-Rate Issues:

- We have to support the overall product benchmarking process, but also have to provide specific guidelines to software project planners
 - Used COCOMO-II as a research tool
 - Promoted COCOMO-II as primary estimation tool

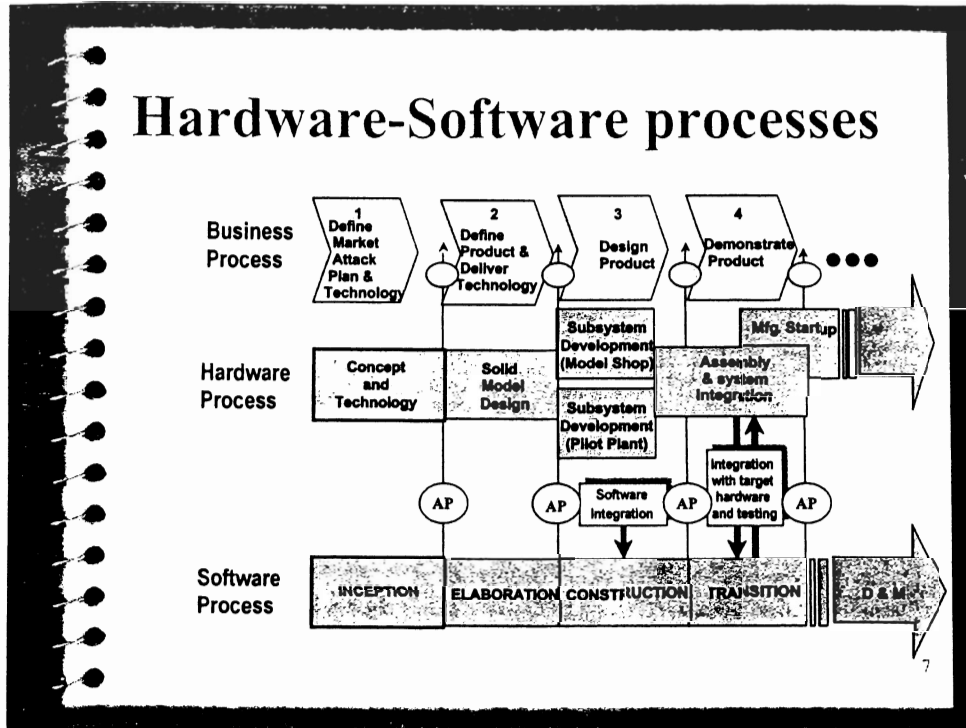
✓ Product* Slip-Rate Issues:

- It is difficult to make the connection between product slip, hardware slip and software slip
 - Used Anchor Points to provide solution

* Dominant Xerox products are viewed as Software-Intensive Systems

6

Hardware-Software processes

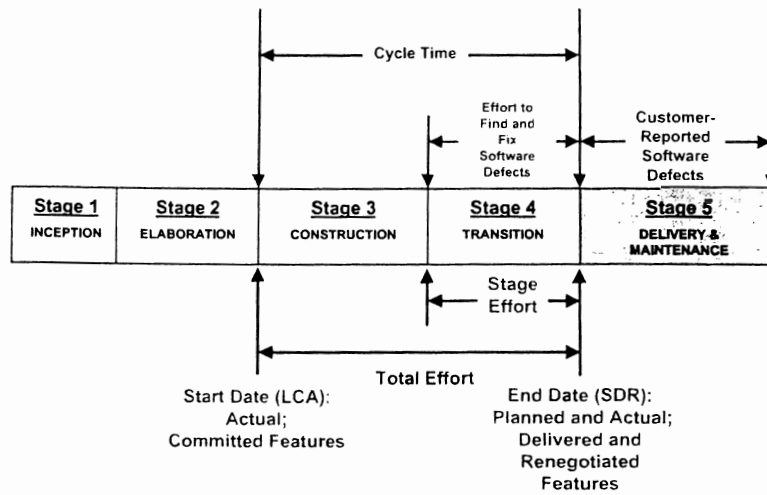


Hardware and Software Development processes are viewed as subprocesses of the overarching Product Development Business Process. The business synchronization of these processes can be carried out via the use of Anchor Points. Note that business synchronization happens much less frequently than the technical synchronization of the subprocesses:

- Business Synchronization takes place via Anchor Points of the Subprocesses
- Technical Synchronization “as needed”, for example h/w-sw integration steps

Please also note that hardware and software development is progressing simultaneously. Consequently to determine and track slip-rate for the product, eventually the **critical path** method has to be used.

Data Elements for S/W Slip-rate



The diagram shows the actual data elements and development stages for software development. Again, due to the systematic and generalized use of anchor points, all stages have equivalents in the various subprocesses, so it is easy to understand the generalization of the slip-rate related metrics.

Benchmark Schedules Directive

✓ The Corporation requires the use of benchmark schedules to determine *planned program duration*:

- Product benchmark schedules are driven by a composite *newness/complexity index**, comprehending technology, hardware and software. Reference Database is confidential.

*Index and Database developed by the PRTM management consulting group

9

A product's composite index is derived from the following H/W, S/W and (Hardware only...)Technology factors:

Newness (Platform – Major – Minor):

Platform or New Product Line

Major: major changes to current product or establish new, single product

Minor: minor changes to current product

Complexity (Low – Medium – High), based on for:

Software: SLOC for new code

Hardware: New items to BOM (Bill Of Materials)

Technology: Percentage of total development effort due to new technology

PRTM: Pittiglio, Rabin, Todd & McGrath – principals

<<< <http://www.prtm.com> >>>

Specific S/W issues

- ✓ “Newness” emphasizes “precedentedness”
- ✓ Higher level of reuse-intent is implicitly equated withprecedentedness
- ✓ SLOC definition deals only with new code, actual reuse is not considered
- ✓ Internal software complexity (as it is defined in COCOMO) is not considered

COCOMO-based Recommendations

- ✓ Instead of simply considering new code, use the COCOMO “*Equivalent KLOC*”*
- ✓ Put emphasis on *complexity* instead of precedentness:
 - See productivity ranges in COCOMO-II**:
 - PREC 1.4 (deviation = 0.046)
 - CPLX 2.4 (deviation = 0.014)

* Page 22, REF [2]

** Page 171, REF [2]

11

Complexity:

Apparently CPLX has a much wider productivity range and at the same time smaller deviation than PREC

Definitions for CPLX still have to be refined and customized for the specific domain.

COCOMO for Benchmarking

✓ Initial stage: "Average" benchmark

- All drivers are set to nominal, except for
 - *TIME* = H Due to real-time requirements
 - *STOR* = VH Low UMC is always a major constraint
 - *PLEX* = L It is always hard to keep up with the pressure from h/w and OS vendors both on servers and clients

✓ For "Best In Class" benchmark

- Increase/change the following drivers:
 - *RUSE, ACAP, PCAP, PCON, APEX, PLEX, LTEX, TOOL, SCED*
- Increase *PMAT*
 - (Note that CMM Level-2 translates into "nominal" driver rating, and Xerox development teams are expected to operate on at least CMM Level-2)

12

The philosophy behind driver setting:

- Product engineering related drivers have to be understood and set first. Software organizations can not really choose what they are going to work on, and most of the time these engineering parameters are either non-negotiable or have small latitude. Cost-benefit analysis is done in the context of the future market perspectives of the product, in an earlier stage.
- People and process related drivers are more tunable, and can be used in conjunction with cost-benefit analysis during project planning. All project management decisions, resource allocations and improvement efforts will have a primary impact on these drivers.

Still needs more research ...

- ✓ ...on estimating Elaboration effort/schedule
 - Early Design Model is not really applicable for the difficult and prevalent parts of the systems
- ✓ ... on estimating Transition effort/schedule
 - COQUALMO might be used as a starting point...
 - ...but the h/w – s/w interdependencies are not captured, consequently a more complex defect introduction and removal model would have to be developed

13

Early Design Model:

Quoted from page 51 of Reference [2]:

“... This model could be employed in either Application Generator, System Integration, or Infrastructure Development sectors.”

Transition effort estimation:

The most commonly used approaches simply consider the number of features to be tested. Particularly independent test organizations do not have any insight into the regression history of the development organization, so they can not use that parameter during test planning. Similarly, due to lack of proper defect analysis and tracking in the earlier phases there is no solid information on the quality of the code which is provided for testing. (Actually what we are interested is the depth and quality of Unit Testing, proceeding integration.) Finally, -- rephrasing the question of “How Much Testing is Enough?” -- political considerations might impact the definition of “How Many Defects Will Our Customers Tolerate after Launch?”

Business Group and Project Level

✓ Cultural/Management Barriers

- Inability to manage expectations
- Improper positioning of pilot project(s)
- Unresolved size metrics concerns

14

Managing expectations:

Currently at Xerox 10% slip for a software-intensive product is acceptable. Greater than 20% slip at any phase gate triggers a review with the Business Group President, and greater than 50% slip at any phase triggers a review with the CEO.

Since the slip is for the overall product, the affected disciplines (hardware, software, materials, technology) all contribute to the slip. In the case of COCOMO, the effort estimation accuracy is 30% for 75% percent of the data points, assuming that the computations were driven by actual ("100% accurate") size numbers. In practice, size estimation is much more difficult than effort estimation, and the size estimation error itself could be 20-30%. Promoting any approach with this low-level of accuracy is a real challenge!!!

Positioning Pilot Projects:

Essential to position it with the objective of finding local implementation problems and identifying specific support and training needs.

Make it clear that it is not acceptable to mismanage the pilot and then claim that the subject of the pilot is a failed method or tool.

If the top goal is to select methods or tools, run independent pilots to explore them on their own merit to avoid early bias. Make the selection process itself an independent project, building on the results of the earlier pilots.

Unresolved, lingering *sizing* concerns can greatly hinder a *cost estimation* pilot.

Business Group and Project Level

- ✓ Misinterpretation of the CMM
 - Core issue: The CMM Goals can be satisfied without assuring performance quality targets for process execution
 - It is enough to define, document and follow processes, however weak they are
 - Shooting for the Grade – “Why do more if less is enough to be assessed on Level-x”

15

In organizations simply aspiring to get the Level-x CMM Rating, the following issues are typically mishandled:

- Historical data collection. Consistency of data and easy access to data are usually missing
- Milestones' and general project data consistency. It was particularly a major problem before the introduction of Anchor Points
- Life-cycle model inconsistencies
- Relying on the Wideband Delphi method, instead of putting effort into the calibration of parametric estimation methods
- To avoid SLOC or FP controversies, experimenting with “off-beat” size metrics. These bogus metrics prevent the use of industry-wide methods or benchmarks, or even comparison between business groups of the same company. This kind of experimentation is o.k. for Academia or Research, but highly inappropriate for production environments.

Sizing Issues

✓ Conceptual

- Almost impossible to get consensus on what size metrics to use
- Backfiring FP into SLOC is highly inaccurate
- High programmer sensitivity due to potential misuse of SLOC
- *Collectors* of data and *users* of data are different
- Definition of DM in COCOMO is vague and inconsistent
- Definition of CM in COCOMO is inconsistent
- Deleted LOC (and design?) are intentionally not accounted for in COCOMO

✓ Practical

- Counting LOC, particularly measuring Modified Code Size (CM) for COCOMO is not as simple as we were led to believe by metrics gurus

16

It is almost impossible to get consensus on what size metrics to use:

- Of course all sizing methods make the disclaimer that they only work in certain domains. Nevertheless, even in a particular domain the emotions and arguments can flare up dramatically when it comes to selection
- In view of recent trends in programming languages, development methodologies and environments backfiring is less and less reliable.

Definition of DM is vague:

Frankly, most people can not even agree on what a “Design” is. Document? UML diagrams? Non-UML diagrams and various architectural views? In reality in most cases the basis of the development is a low-quality unstructured design document. What is the percentage we are looking for? How do we baseline the method of determining this percentage, so subsequent determinations would be consistent, at least in a company/project context?

Definition of DM and CM is inconsistent

On page 136 of [1] it is stated that “There is no reason why the values of DM, CM, and IM can not be greater than 100”. On the other hand on page 26 of [2] it is clearly indicated that CM range is 0-100, and in fact the COCOMO-II tool will not accept greater than 100 % as an input.

Measuring Modified Code Size

✓ Factors to consider

- (1) Directories of old and new code have to be accessed simultaneously
- (2) Tagging mechanism is needed to identify Adapted Code Modules
- (3) A "DIF"-type tool is needed (Caveat: OS dependent) to compare files
- (4) Default tool capability: showing modifications visually, on screen, and on the physical lines only
- (5) Results have to be segmented to "untouched", and "modified+new"
- (6) A CodeCount - type analyzer has to be applied to both segments to determine the equivalent Logical Lines of Code
- (7) DIF and CodeCount have to be integrated with the SCM system

✓ Issues

- (1) Having CodeCount is a very small fraction of the needed solution
- (2) Integration of all of these tools with the SCM tools is a big undertaking
- (3) Due to tools and vendor diversity all possible solutions have to be highly customized, and a general solution is not really feasible.

17

Conclusions

- ✓ Metrics problems in general are more political than technical in nature
- ✓ There are substantial cultural and management barriers to the use of sophisticated Cost Estimation Methods, or even to start a small-scale pilot
- ✓ Success of a Cost Estimation pilot is highly dependent on having an accepted metric and stable method for sizing

18

Acknowledgement

We would like to acknowledge Mr. Don Reifer's dedicated support on behalf of CSE/USC to the piloting teams and also thank him for the workshops he led in Rochester.

19

References

- [1] Boehm, B. "*Software Engineering Economics*", Prentice Hall, 1981
- [2] Boehm, B. et al. "*Software Cost Estimation with COCOMO II*", Prentice Hall, 2000

20