DEFINING CYCLE TIME

Richard D. Stutzke

Science Applications International Corporation
6725 Odyssey Drive
Huntsville, AL 35806-3301

(256) 971-6624 (office)
(256) 971-6550 (facsimile)

Richard.D.Stutzke@saic.com

15 September 1999


© 1999 by Richard D. Stutzke
PRECONDITIONS

- Need a set of equivalent products
  - A series of releases
  - Similar versions of a product line
- The production process must be stable
THE TOTAL SOFTWARE LIFE CYCLE

ANALYZE REQUIREMENTS → DESIGN → CODE → TEST → DEPLOY → OPERATE → DECOMMISSION

New Requirements (SCRs)

Validated Unique Trouble Reports (STRs)
THREE LIFE CYCLE STAGES

- Initial creation
  - Create operational concept, architecture, and product design
  - These tie to Boehm’s process anchors (LCO, LCA, IOC)

- Sustainment
  - Product structure remains unchanged (else return to "creation")
  - Modifications to correct, perfect, adapt, and enhance.

- Decommission
  - Shut off and discard product
  - Replace product (with possible cutover activities)
THE END POINTS

- Start: receive a set of validated, approved change requests (correct, perfect, adapt, enhance)
  ⇒ Build content (size) is defined

- End: all acceptance tests successfully completed
  ⇒ Technical data package is baselined
DEFINING CYCLE TIME

\[ TTBR = TDEV + TNONDEV \] [calendar-days]

where

\[ TTBR = \text{Total Time Between Releases} \]
\[ TDEV = \text{Time to develop the product} \]
\[ TNONDEV = \text{Time when no production is occurring (Pre: analyze change requests & plan. Post: deploy & slack.)} \]

We can decompose TDEV:

\[ TDEV = \text{work days} + \text{weekends} + \text{holidays} + \text{down time} \]

Production occurs on the work days and down time is unplanned. Down time arises from equipment outages, late completion of predecessor tasks, lack of funds (stop work), etc. (These are called "special causes" in Statistical Process Control.)
DEFINING CYCLE TIME (Continued)

Assuming constant staffing, the "duration" is:

\[
TDUR = \frac{\text{Effort Required}}{\text{Effort Delivery Rate}} = \frac{E}{R} \quad \text{[work-days]}
\]

where

\[
E = \text{Effort Required} = E_{\text{build}} + E_{\text{inspect}} + E_{\text{test}} + E_{\text{rework}}
\]

\[
R = \text{Effort Delivery Rate} = (N_{\text{workers}})(AA)(8 \text{ phrs/work-day})
\]

\[
AA = \text{Average Availability} \text{ (ranges from 0 to ~ 2)}
\]

Notes:

1. \(<\text{FTE}> = AA*N_{\text{workers}}
2. There is a coupling between AA and the choice of time units. Some authors define effort delivery per calendar-month. I believe this obscures the effects of different work calendars, seasonal variations, and any outages.
3. Typically,

\[
E = \text{Size/Productivity} = S/P \quad \text{[person-hours]}
\]
DEFINING PRODUCTIVITY

Size = Amount of product produced. For software, this is usually source lines of code or Function Points.

Effort = Direct labor expended by the project staff to build the product. Typically measured in person-months or, better, person-hours.

Productivity = Size divided by effort.

Different definitions of size and effort can yield computed productivity values which differ by factors of ten or more.
FACTORS AFFECTING PRODUCTIVITY VALUES

• Size
  – Product architecture (choice of platform, technology, and decomposition)
  – Amount of reuse
  – The software being counted (reused, prototypes, tooling, breakage)
  – The choice of size unit of size (blank lines, comments, physical or logical lines)

• Effort
  – Project formality ("rigor")
  – Process architecture (life cycle)
  – The activities and phases included (CM & QA direct/indirect; P&R)
  – Degree of automation
  – Staff skill and experience
  – The choice of units (person-hours per person-month)
  – Scope (paid overtime, uncompensated overtime)
FACTORS AFFECTING THE COMPUTED VALUE

- Choice of cycle's end points
- Product size
- Productivity of process (automation, rework)
- Staffing level (number and availability)
- Work calendar (weekends, holidays, planned shutdowns)
- Outages (unplanned shutdowns)
A WORKING DEFINITION

$$TDUR = \frac{S/P}{Nw*AA*8}$$

The units are:

$$\frac{(SLOC)/(SLOC/phr)}{(person)(pure \ number)(phrs/[person*work-day])} = \text{work-day}$$

Observation: We can reduce TDUR by adding staff.

Consequence: We must normalize in order to make meaningful comparisons.
DEFINING NORMALIZED CYCLE TIME

\[ NCT = \text{Normalized Cycle Time} \equiv \frac{TDUR \times Nw \times AA}{S} = \frac{1}{8P} \]

Alternately, using COCOMO’s notation:

\[ NCT \propto \frac{(TDEV)(FTE)}{S} = \frac{MM}{S} = \frac{1}{P} \]

since \( TDEV = k \times TDUR \)

and \( FTE = N_w \times AA = MM/TDEV \)

Completely normalized cycle time is equivalent to productivity!
COMMENTS

• Cycle time is a valid measure for some industries where time-to-market is critical.

• A commonly used definition is:

\[
NCT' = \frac{TDUR}{S} = \frac{1}{8*Nw*AA*P} \quad \text{[work-days/SLOC]}
\]

This measures the time to deliver a "unit of functionality".

• Planners adjust size, productivity, staffing, and work calendars to achieve the desired TDUR. (The next chart lists 8 options to reduce cycle time.)

• The usual goal is to keep \( NCT' \) constant while increasing the size, \( S \), even if productivity decreases.
WAYS TO REDUCE CYCLE TIME*

1. Eliminating tasks (reducing size, automating some operations)

2. Reducing time per task (productivity, overlapping)

3. Avoiding single-point task failures (hardware outages, loss of key people)

4. Reducing rework ("backtracking")

5. Activity network streamlining (reduce task dependencies)

6. Increasing the duration (number of workdays)

7. Better people and incentives

8. Transition to learning organization (process improvement)

*Adapted from [Boehm, 1997]