## Introduction and Objectives

Welcome to the Document and Present Estimate lesson. After completing this lesson, you will understand the fifth and final major step of developing a software cost estimate.

### Lesson Objectives

- Appraise software estimate documentation for its sufficiency in areas such as traceability, replicability, and reasonableness.
- Compare two distinct software estimates as part of the reconciliation process and consider the factors driving the differences between the two.
- Critique a software estimate as to whether it has been adjusted to reflect updates, including both changes in the program baseline and the latest actual data.


D

**Long Description**

Graphic illustrates the steps of the Cost Estimating process. The steps from left to right are:
Develop Scope and Approach, Collect and Analyze Data, Develop Estimate Methodology, Consider Risk and Uncertainty, and Document and Present Estimate (highlighted).

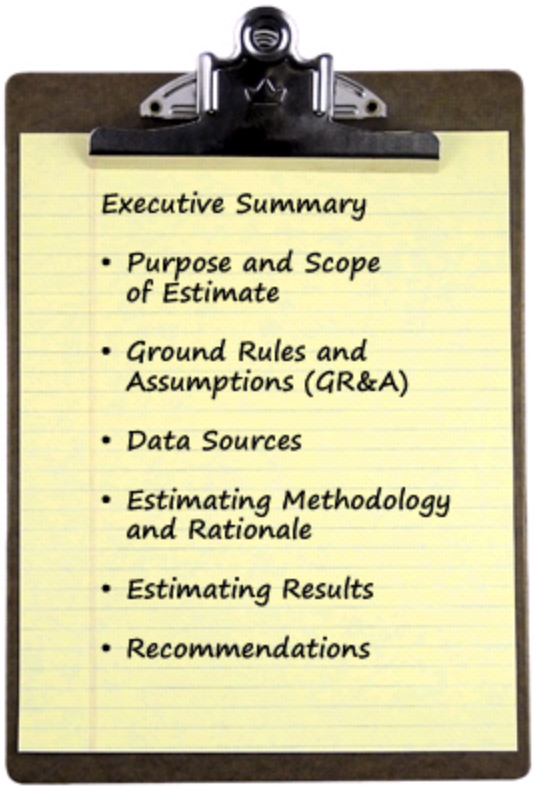
## Software Estimate Documentation

Like any good cost estimate, the software estimate should be documented so that an external reviewer – or even the lead analyst at a later date! – can understand how it was derived.

An Executive Summary should always be provided, so that decision makers do not have to wade through the detailed documentation to find and understand the main points of the estimate.

The summary should be a miniature review of the estimating process and results. Typical sections include:

- Purpose and Scope of Estimate
- Ground Rules and Assumptions (GR&A)
- Data Sources
- Estimating Methodology and Rationale
- Estimating Results
- Recommendations




Questions Managers Should Ask

D

**Popup Text**

**Questions Managers Should Ask**

Are the objectives of the estimate clear and correct?

- Are the objectives of the estimate stated in writing?

Have the factors that affect the estimate been identified and explained?

- Does a written summary of parameter values and their rationales accompany the estimate?

**Long Description**

Clipboard with the components of the Executive Summary in bulleted list.

- Purpose and Scope of Estimate
- Ground Rules and Assumptions (GR&A)
- Data Sources
- Estimating Methodology and Rationale
- Estimating Results
- Recommendations

## Software Estimate Documentation - Replicability and Traceability

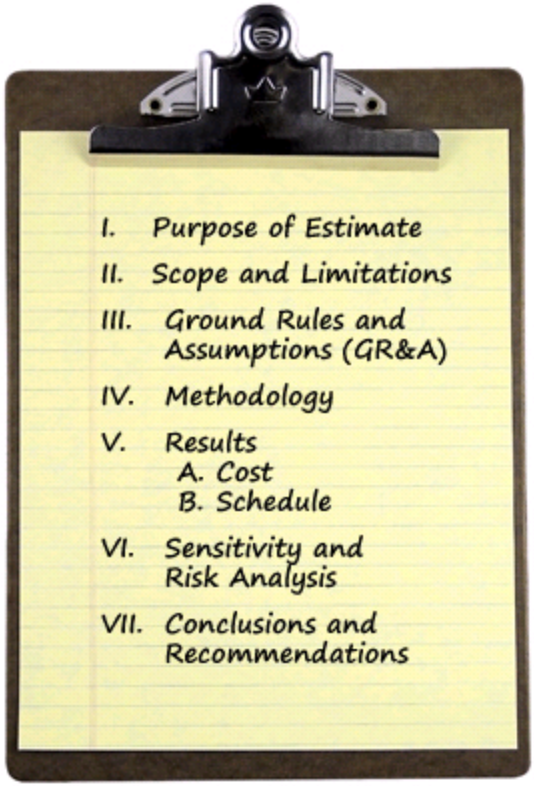
The documentation for a software cost estimate contains the same basic elements as any cost estimate. The structure of the detailed documentation may replicate what has been addressed in this module or something similar to the structure illustrated to the right.

Note that while there is no standard DoD format, there may sometimes be a requirement to follow a certain format, or to include a certain form. Follow local Service or Agency guidelines.

An acid test for documentation is that a competent analyst unfamiliar with an estimate should be able to trace (and even replicate) all calculations, from the original source data to the final answer.

This means that all primary and secondary data used, inputs, model files, etc., need to be included or referenced.

It is important to show key software-related steps like normalization of data to reflect consistent development activities, equivalent source lines of code (ESLOC) calculations, application of code growth, and the like.




D

**Long Description**

Clipboard with basic elements of cost estimate listed. The list is as follows:

- Purpose of Estimate
- Scope and Limitations
- Ground Rules and Assumptions (GR&A)
- Methodology
- Results
    - Cost
    - Schedule
- Sensitivity and Risk Analysis
- Conclusions and Recommendations

## Software Estimate Documentation - Reasonableness

The job of documentation is not just to provide the basis of the estimate, but also the rationale for that basis.

Why were certain data and techniques used? Why were certain adjustments made? The documentation should convince the reviewer that the estimate is reasonable. The Ground Rules and Assumptions (GR&A) should be thoroughly documented and justified.

One way to strengthen the case for reasonableness is to compare certain key inputs and outputs to benchmarks, industry standards, or rules of thumb.

For a software estimate, leading candidates for such a "sanity check" include sizing, code growth, reuse, and productivity, as well as factors that contribute to overall capability and complexity. If an off-the-shelf (OTS) model is being used to generate part or all of the software estimate, it is important to justify the "settings" of the model in terms of inputs and calibration.

While expert judgment should never be the sole basis of a software estimate, it is important to document when and how subject matter experts (SMEs) were consulted and what effect the information they provided had on the estimate, such as normalization of historical data.

## Software Estimate Documentation - ProRad Documentation

The source lines of code (SLOC) estimates for each waveform are summarized in the table below. Also included are the **Type of Requirement** – Key Performance Parameter (KPP), Threshold Requirement, or Objective Requirement; **Difficulty** of the development, on a scale of 1 to 3; whether **Reused** code was available; and whether the software needed to be **Designed for Reuse**.

As an independent reviewer, take a moment to assess what additional information could have (or should have) been provided. Certainly the categorization of requirements and existence of reuse code seems objective, but cited references would help confirm this. The difficulty of development seems to be expert-based, but if you pull the thread, you may find its based on some objective categorization of waveforms. Perhaps most importantly, the SLOC estimates should cite analogy programs, SMEs who were consulted, and any other substantiation.

| Waveform | Type Requirement | Difficulty | SLOC | New | Reused | Design for Reuse | Comment |
|---|---|---|---|---|---|---|---|
| EPLRS | KPP | 3 | 75000 | No | Yes | No | Based on current software |
| Wideband Digital Waveform | KPP | 3 | 180000 | Yes | | No | All new–possibly based on existing WB waveform |
| SINCGARS ESIP | KPP | 2 | 53000 | No | Yes | No | Based on current software |
| UHF DAMA/DASA SATCOM (181, 182, 183) | KPP | 2 | 88000 | Yes | | Yes | Mil-Std 181 (Basic Waveform), 182 and 183 |
| UHF DAMA/DASA SATCOM (184) | KPP | 2 | 30000 | Yes | | Yes | Mil-Std 184 |
| HAVE QUICK I/II (UHF) | KPP | 1 | +5000 | Yes | Basic AM/FM | Yes | Add frequency hopping |
| Link 16 | Threshold | 3 | 100000 | Yes | | Yes | Based on current SW–but designed for SCA |

Select image to view larger version of the entire table.

## Reconciliation

For most programs, there are almost always at least two cost estimates "on the table," keeping in mind the aforementioned varieties of Budgetary, Comparative, and Planning Estimates.

Focusing on Budgetary Estimates, there is often a Program Office Estimate (POE), a Component Cost Estimate (CCE) led by the cognizant service cost center, and an Independent Cost Estimate (ICE) produced by the CAPE.

It is generally valuable for different estimating teams to gain insight into each other's estimates, methodology, and rationale, and the standard process for this is called reconciliation

## Reconciliation Goal

The goal of reconciliation is not to bring the estimates together – though that may happen if one team finds the other's approach more compelling and adopts it – but rather to understand what is driving the differences between them.

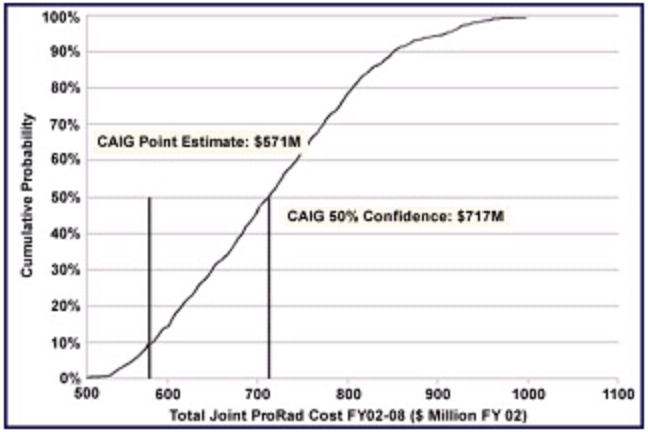
This usually boils down to using different data, different manipulations of that data (normalization and analysis), and different assumptions. Often the two teams will "agree to disagree."

Reconciliation generally focuses first on high-cost WBS elements and key cost-driving inputs and assumptions.

For software, these include core development and supporting activities; sizing, growth, and reuse; development schedule; and OTS model settings or other inputs to capability and complexity.

## Reconciliation - Probabilistic Cost Estimates

As described in the Consider Risk and Uncertainty lesson, the most common depiction of a robust estimate is an S-curve showing the cumulative probability of various outcomes for cost or schedule. The reconciliation process should not be limited to a comparison of point estimates but should also include a comparison of S-curves.




The teams should examine the key points on the S-curve: the mean estimate; and various percentiles, usually the 50th (median) and a higher one like the 80th. They should also compare the relative width or steepness of the S-curve, as captured by the coefficient of variation, or CV. A low CV is the surest indicator that uncertainty is understated.

Causes for this can include failure to incorporate uncertainty associated with estimating techniques; omission of risks; and understatement of ranges for key driving parameters. For example, if reuse is only plus or minus 5% at early stages of development, then overall uncertainty is probably too low.

The Naval Center for Cost Analysis (NCCA) provides an S-Curve Tool that can be used to compare S-curves and benchmark CVs against historical norms. Select the CAIG's S-curve graphic to view the the ProRad program S-curve.

## Reconciliation - ProRad Reconciliation

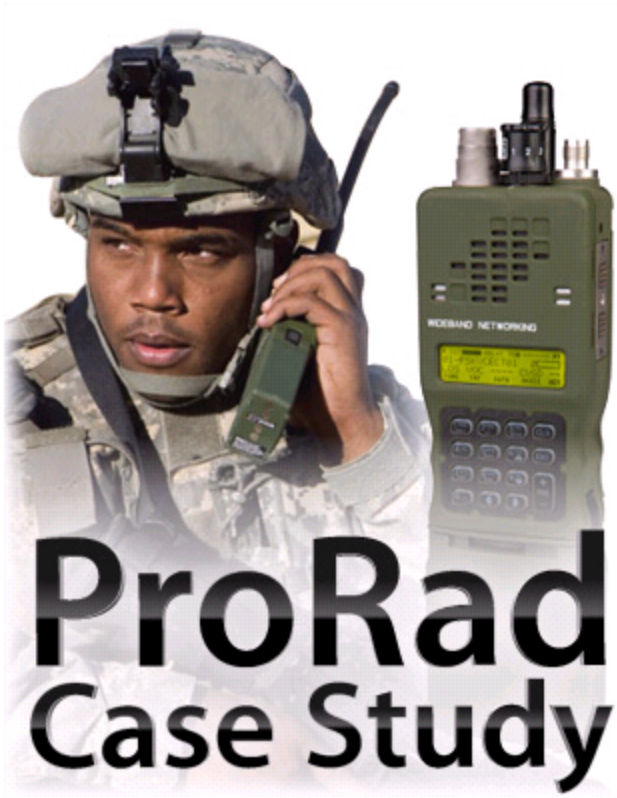
The CAIG prepared the ICE for the ProRad program. At an Overarching Integrated Product Team (OIPT) meeting prior to the Defense Acquisition Board (DAB), they presented the comparison of their ICE to the Joint ProRad POE (only cost elements with differences were shown).

Costs shown are for the period FY 2002-2008, in FY 02 base year dollars in millions.

Select the links below to view the program cost charts.

- ProRad JPO Costs (02-09)
- Summary - CAIG Risk Analysis

Click here to view the reasons for key differences between CAIG and POE costs.



**Popup Text**

**Overarching Integrated Product Team (OIPT)**

An Integrated Product Team (IPT) led by the appropriate Office of the Secretary of Defense (OSD) director, and composed of the Program Manager (PM), program executive officer (PEO), component staff, user/user representative, and OSD and Joint Staff (JS) members involved in the oversight and review of a particular Acquisition Category (ACAT) ID or ACAT IAM program.

**Popup Text**

**Defense Acquisition Board (DAB)**

The DAB is the Department's senior-level forum for advising the Under Secretary of Defense for Acquisition, Technology, and Logistics (USD(AT&L)) on critical decisions concerning Acquisition Category (ACAT) ID programs, and selected ACAT IA programs. The DAB is composed of the DoD's senior executives. The DAB is chaired by the USD(AT&L). Other executive members of the DAB include:

- Vice Chairman, Joint Chiefs of Staff (VCJCS)
- Under Secretary of Defense (Comptroller) (USD(C))
- Under Secretary of Defense (Policy) (USD(P))
- Under Secretary of Defense (Personnel and Readiness) (USD(P&R))
- Chief Information Officer of DoD Director of Operational Test and Evaluation (DOT&E)
- Director, Cost Assessment and Program Evaluation (CAPE) Director, Acquisition Resources and Analysis (ARA) (also executive secretary of the DAB)
- Secretaries of the Military Departments

Defense Acquisition Board advisors include the Assistant Secretary of Defense (Acquisition); Assistant Secretary of Defense (Logistics & Materiel Readiness); Deputy Under Secretary of Defense (Installations and Environment); DoD Deputy General Counsel (Acquisition & Logistics); Director, Defense Research & Engineering; DoD Component Acquisition Executives; the relevant OIPT Leader(s); Director, National Geospatial-Intelligence Agency; Deputy Director, Cost Assessment; Director, Defense Procurement & Acquisition Policy; Director, Systems Engineering; Director, Developmental Test & Evaluation; Director, Industrial Policy; Director International Cooperation; Assistant Secretary of Defense (Legislative Affairs); Chair, Functional Capabilities Board(s); Cognizant Program Executive Officer(s) and Program Manager(s). The USD(AT&L) may ask other department officials to participate in reviews, as required.

**Popup Text**

Click here to view the reasons for key differences between CAIG and POE costs

Reasons for key differences between CAIG and POE costs are:

- CAIG software reuse methodology gives much less credit for reuse
  - Program Office claim is 60% reuse
  - CAIG methodology gives credit for 28% reuse (72% new)
- CAIG considered Wideband Networking Waveform (WNW) as all new effort
- CAIG does not add risk to estimate (CAIG 50% risk would add $146M)
- Most of the other cost factors are proportional to software deltas

## Update for Changes

One of the biggest challenges in a software estimate is keeping it up to date, to reflect both the latest program baseline and actual data.

Key baseline changes for software include:

- Updated requirements
- Restructured development
- Reuse changes
- COTS changes
- Staffing changes
- Schedule changes
- Interface changes

These baseline changes should be documented in a Cost Analysis Requirements Description (CARD) or a CARD-like document, even if only a working version.

### Software Changes

| Y | N | Change Type |
|---|---|---|
| | | Requirements |
| | | Restructure |
| | | Reuse |
| | | COTS |
| | | Staffing |
| | | Schedule |
| | | Interface |

Questions Managers Should Ask

D

**Popup Text**

**Updated requirements**

Depending on Life Cycle Development Methodology, these could be evolving requirements or new requirements via engineering change proposals (ECPs).

**Restructured development**

New increments or requirements deferred to future increments.

**Reuse changes**

Shortfalls in utility of reused code, requiring additional new development.

**COTS changes**

Shortfalls in functionality of COTS, requiring additional new development.

**Staffing changes**

Shortfalls in staffing levels or capability of developers.

**Schedule changes**

Delivery date slipped.

**Interface changes**

External interfaces late or more difficult than anticipated, requiring additional development and testing.

**Popup Text**

**Questions Managers Should Ask**

Have steps been taken to ensure the integrity of the estimating process?

- Do the groups that will be doing the work accept the estimate as an achievable target?

Has the situation changed since the estimate was prepared?

- Does the estimate remain valid in view of recent events, changing requirements, and management action (or inaction)?
- Is the estimate is being used as the basis for assigning resources, deploying schedules, and making commitments?
- Is the estimate the current baseline for project tracking and oversight?

**Long Description**

Matrix document labeled Software Changes with the following change types listed:

- Requirements
- Restructure
- Reuse
- COTS
- Staffing
- Schedule
- Interface

## Update for Changes - Updated Data

Not only should the cost estimate reflect the latest baseline, but it should make use of the most recent available data.

As CSCIs, development phases, or increments are completed, actual effort (hours) and other data can be used to update parametric and analogy estimates.

If the development effort is using Earned Value Management (EVM), then the primary or secondary estimating method can shift to Actuals, using an Estimate At Completion (EAC). As early as 10% or 15% complete, EACs can serve as useful estimates of the total effort.

Software management measures, even if not directly used in estimates, can be useful indications of program progress (or lack thereof), which may necessitate an estimate update.

These may including defect rates, defect densities, test results, and the like.

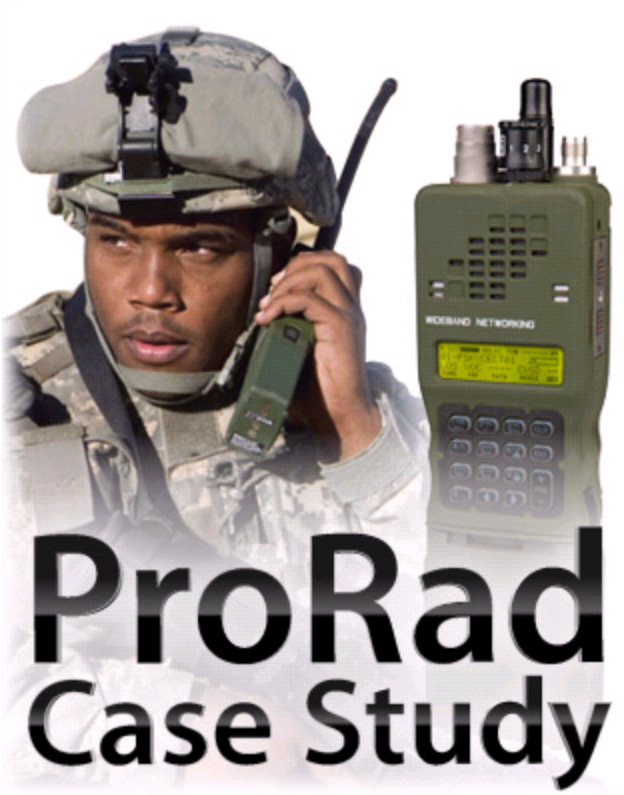
## Update for Changes - ProRad Estimate Updates

After the development of the first few waveforms is complete, the ProRad estimate is updated.

First, the estimates for these waveforms are replaced with actual costs, since sunk cost should always be included in the life cycle cost estimate (LCCE).

Second, the cost estimating team should look for any "return data" from these waveforms that may inform our estimates of yet-to-be-completed waveforms.

This may include code growth experiences, amount of reuse achieved, development productivity achieved, finishing ahead of or behind schedule, etc.

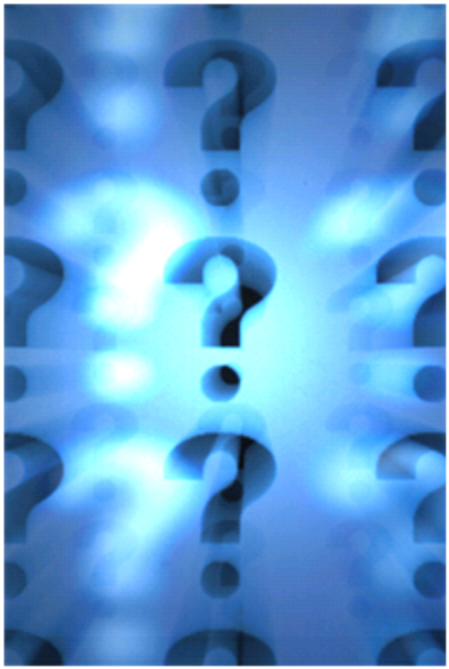


## Knowledge Review

Which of the following does **not** represent a change in the program baseline for a software-intensive system?

- [ ] UAV flight control software required additional modes for aircraft carrier landing
- [ ] ERP system pilot that served 10,000 users now expanded to 100,000 users
- [x] Cost Performance Index (CPI) for first six months of software development was 0.90
- [ ] Due to delays in follow-on system, currently-fielded system will need to extend its service life by an additional five years

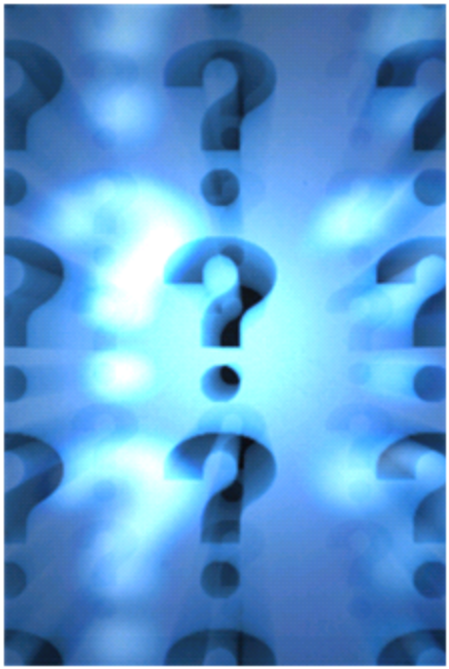
Check Answer



**Cost Performance Index (CPI) for first six months of software development was 0.90** does not represent a change in the program baseline but rather a change in program performance.

## Knowledge Review

Which of the following would tend to make Team B's estimate **higher** than Team A's?



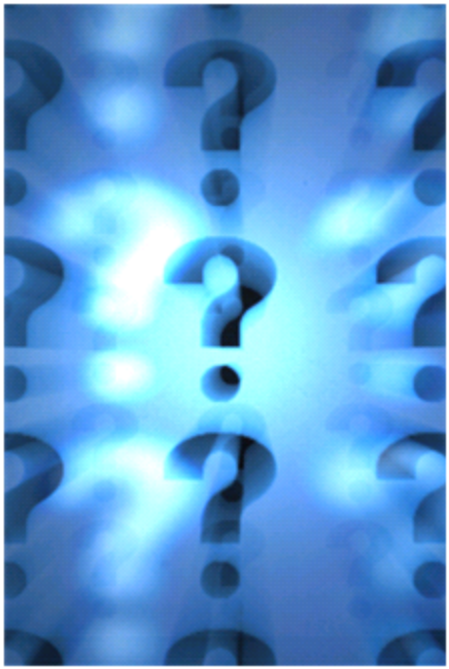
- [x] Team A omitted COTS license costs
- [ ] Team A applied a higher code growth factor
- [ ] Team A estimated 150 function points compared to Team B's 120
- [ ] Team A applied real growth to software developer labor rates, whereas Team B applied a cost-of-living annual increase only
- [ ] Team A assumed 24 x 7 help desk support instead of 8 x 5

Check Answer

**Team A's omission of COTS license costs** would tend to make Team B's estimate higher than Team A's. All the other differences would tend to make Team A's estimate higher.

## Knowledge Review

A software estimate uses a Productivity factor of 1.1 SLOC/labor hour. Which of the following statements is not required in the documentation for purposes of traceability?



- [ ] Productivity was determined by Analogy to Program A
- [ ] Program A produced 5.5K SLOC of new code
- [ ] Program A had a total effort of 5,000 labor hours for software development
- [x] Program A software development took 12 calendar months

Check Answer

**Program A software development took 12 calendar months** is not required in the documentation for purposes of traceability. Development time may be relevant to compute staffing level or assess degree of schedule compression, but it is not needed for the traceability of the Productivity calculation.

**Summary**

This completes the Document and Present Estimate lesson. In this lesson, you learned:

- Software estimates should be supported by documentation that provides a clear trace from data and assumptions to the estimate itself.
- Software estimates should be developed independently by at least two organizations and then compared during the reconciliation process.
- Software estimates require frequent updates to reflect both changes in the program baseline and the latest actual data.


D

**Long Description**

Graphic illustrates the steps of the Cost Estimating process. The steps from left to right are:
Develop Scope and Approach, Collect and Analyze Data, Develop Estimate Methodology, Consider Risk and Uncertainty, and Document and Present Estimate (highlighted).

## Module Completion

You have completed the content for this lesson.

At this point you should have completed all of the lessons in this module.

Please take the Module Exam and complete the Module Survey so you may receive credit for this course.

If you have closed or hidden the Table of Contents, click the Show TOC button at the top in the Atlas navigation bar.