DATA ITEM DESCRIPTION
MAINTENANCE DATA COLLECTION REPORTING

1. **Title:** Software Maintenance Data Reporting

2. **Use/Relationship**: This Data Item Description (DID) identifies and describes the data being collected to build a software operations, maintenance and sustainment cost and quality database. This Software Maintenance Data Reporting form is not a management or measurement report. It is not intended for tracking progress, nor does it intend to collect financial information. Rather, its purpose is to collect empirical data during the software operations and maintenance phase of the weapons systems life cycle (after Milestone C) for use in calibrating models and developing software benchmarks. These data will also be used to first characterize the Department of Defense's software inventory and then substantiate budgets used for future maintenance appropriations.

3. **Timing:** Because we are collecting both estimates and actuals for many of the measures identified, the best time to capture data is at the start and end of a cycle. For example, software size measured in source lines of code would be captured at the beginning of a release with the estimated and the end with a measurement of the actuals when the release is delivered to the field. At the beginning of the cycle, software size estimates will be collected using estimate work sheets as their primary source. At the end of the cycle, a code counter such as the University of Southern California's (USC) Unified Code Counter (UCC) will be run to measure the actual size of the software and the number of source lines of code added, deleted, changed and reused from release to release and version to version (using the USC tool's differential counting capabilities).

Additionally, data will be captured on an annual basis when releases are multi-year because that is how budgets are allocated within the POM (Program Objectives Memorandum). For multi-year projects, the software estimate data must therefore be collected at the start of the cycle, updated with a cost and schedule-to-complete the start of the next fiscal year, and finalized with actual costs and schedule information when the release or version is delivered to the field.

4. **Multi-Level Approach:** Maintenance data will be collected using a three level scheme. At the highest level, software data will be captured at the program level to understand and characterize the program being studied. At the mid-tier, software release information will be collected at the release level to help understand the workload the Department of Defense (DOD) must budget for now and in the future. At the lowest level, data will be collected at the component level to determine what factors drive the cost of software maintenance, when and why. Data at all three levels are needed to satisfy the goals of this Department which are aimed at improving the way software is managed during maintenance.

Data that are being collected as part of this effort will be used for two purposes. First, data collection results will be used to help quantify how large a software inventory the DOD maintains. This information will be used by decision-makers to formulate policy relative to software operations, maintenance and support for weapons systems. Additionally, the data being collected will also be used by the U. S. Army to help estimate and defend software operations, maintenance and sustainment budgets that will become part of the POM. Because the information needs for these two efforts differ, more data than expected may be required to characterize a program and its software deliverables. The team, however, has made every effort to minimize the requirements on programs as they understand that data collection can become an arduous burden for them especially when they are not paid to perform the task.

5. **Program Level Information Needs - Context Data (Mandatory)**. Context data at the program level is needed to identify the following contact, application type and top-level release information using our software maintenance data collection questionnaire. All of these data items are identified as **"Mandatory"** at this level because they represent the minimum set of data that we believe are needed in order for us to realize the goals of this effort.

a. **Contact Information** – the person that we can contact to clarify information by name, organization, address, phone and email information and date the information was submitted.
b. **Submitter Role** – the role of the person submitting the data.
c. **Program Information** – the year the weapons system program was transitioned to maintenance and number of fielded versions.
d. **Application Type** – the type of weapons system application as defined using operating environment and application domain.
e. **Fielded Program Version Information** – location, workforce size and composition, POM budgets, sources of funding, other contracted expenditures and size information for each major software subsystem and/or deliverable software configuration item.

6. **Release Level (Mandatory).** Release data for the identified weapons system program is needed to identify the program's software's characteristics and history. Again, all of the data items are identified as **"Mandatory"** at this level because they represent what we need to understand and improve how maintenance efforts are managed and budgeted.

a. **Program Name** – the name of program.
b. **Project Information** – basic information about the project, its host/target platforms, programming languages, number of distinct end users, and number of user locations.
c. **Release Type** – whether the software release includes major new capabilities, upgraded capabilities or patches or some combination of all the above.
d. **Release Process** – the process paradigm used to generate the software release being described.
e. **Release History**- a summary of the number and types of software releases fielded by the weapons system program during the past few years.
f. **System Integration Environment** – information about the integration and test laboratory and associated test equipment employed to generate the software release.
g. **Activities Included** – the activities performed as part of the software maintenance, sustaining engineering, acquisition management and independent testing activities performed during the software release cycle.
h. **Activity Percentages** – the percentage of the total effort that each of these activities and their subtasks consume (should add up to 100 percent).
i. **Labor Categories Included in Maintenance Effort** – the labor categories for the team that performs the software maintenance activities identified as part of the software release cycle.
j. **Labor Categories Included in Sustaining Engineering Effort** – the labor categories for the team that performs the sustaining engineering activities.
k. **Stage of Maintenance** – the stage of maintenance cycle that the weapons system program is currently in.
l. **In-House or Contracted** – whether software maintenance is performed by in-house, contractor or a combined team.
m. **Size of Maintenance Team** – the average and peak size of the software maintenance team in either FTE (Full Time Equivalent heads) or contract dollars ($).
n. **Metrics and Measures** – the metrics that the project collects and uses to provide insights into their progress, quality and productivity.
o. **Software Defect Backlog** – the size of the software problem report backlog in terms of open trouble reports by defect category. Defects are cataloged in terms of five defect categories each of which is defined in the most current version of the "*Glossary of Software Operations, Maintenance and Sustainment Terms"* which is available from the authors and is available on our web site.

7. **Component Level:** Component data for the identified software release(s) is needed to identify those factors which drive the cost of maintenance. Again, those data items identified as **"Mandatory"** represent the minimum data set to be collected using our questionnaires. Such data includes both estimated and actual values. It represented the minimum set of data that we need to perform our analyses and generate our findings relative to software maintenance cost, productivity and quality.

a. **Year of Development** – start and stop dates for the software release being described.

b. **Size – Requirements (Mandatory)** - Requirements are established for the project, release or version from approved Software Change Requests (SCRs). Software requirements are defined at a detailed level assuming the DOORS tool by IBM/Rational is used for their elicitation and management. Such requirements are typically expressed in a complete sentence containing both a subject and predicate. These sentences need to consistently use the verb "shall" or "will" or "must" to show the requirement's mandatory nature. The whole requirement specifies a desired end goal or result and contains success criterion or other measurable indication of quality. The specification should be consistent in the manner in which it states requirements and complete. In addition, the source of each requirement should be identified (i.e., by mapping to system specification or other source document) and its testability shall be verified.

This set of data is being collected to substantiate budgets for software enhancements including funds needed for software maintenance, sustaining engineering and product support during operations. The data to be reported in this category includes:

- **Name** – the name of the project, release or version being described.
- **Added** – The number of new requirements added to the current version or release.
- **Deleted** – The number of existing requirements deleted from the previous version or release.
- **Changed** – The number of existing requirements modified for the current version or release.
- **Deferred** – The number of new and/or deleted requirements deferred from the new version or release solely due to funding constraints.
- **Total Number of Requirements** – The actual number of requirements in the new version or release when it is delivered for operational use.

Estimate data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

c. **Size – Source Lines of Code (Mandatory)**

The size of the software counted in non-blank, non-comment logical source lines of code (SLOC). Counting conventions for logical source lines vary by language. However, counters exist and should be used when available to count source lines for the language in question using conventions established by the Software Engineering Institute (SEI) in the following referenced standard as modified to count deleted source lines of code:

- Robert E. Park, ***Software Size Measurement: A Framework for Counting Source Statements***, Technical Report CMU/SEI-92-TR-020, 1992.

Such counters include the USC Unified Code Counting (UCC) tool which can be downloaded free from the following web site: http://sunset.usc.edu.

Should other measures of size like function points or object points be used, they must be converted to source lines of code using industry accepted standards like the Capers Jones function point to source lines of code conversion tables in:

- Software Productivity Research, ***SPR Programming Language Table***, Version PLT2007c, 28 December 2007.

This set of data is being collected to define the size of the release so it can be estimated using calibrated software cost models. The data to be reported in this category includes:

- **Name** – the name of the project, release or version being described.

- **New (added)** – The number of new source lines of code added to the new version or release.
- **Reused (old)** – The number of existing source lines of code that were included in the new version or release. These lines are not changed in any way.
- **Modified (changed)** – The number of existing source lines of code that were changed and included in the new version or release. These lines can include design modified, code modified and/or integration modified elements.
- **Deleted** – The number of existing source lines of code that were deleted from the previous version or release.
- **Auto-generated** – The number of auto-generated source lines of code added to the new version or release. Auto-generated code is produced using specialized tools at a pace far exceeding manual development.

Estimate data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

If other size measures like function or feature points are being used, identify how they convert into SLOCs in terms of conversion ratios.

If a counting tool is being used, identify it by name and version.

Provide the information requested on the number of COTS/GOTS packages being used in the deliverable and their characteristics.

d. Schedule (Mandatory) - The schedule represents the calendar time spent to generate the software version or release from its start to its actual delivery date. The set of data being collected is being collected to calibrate software cost models so that their predictions are as accurate as possible. The software effort starts when allocated software requirements are provided to the software team by the systems engineering organization. The software effort ends when the software is delivered to systems engineering for integration and test typically in some System Integration Lab or facility. The data to be reported in this category includes:

- **Name** – the name of the project, release or version being described.
- **Estimated Begin Date** – The estimated calendar date that work on the new version or release should have begun.
- **Estimated End Date** – The estimated calendar date that the new version or release should have been delivered to systems engineering for integration and test.
- **Actual Begin Date** – The actual calendar date that work on the new version or release began. This may differ from the estimated date due to any number of reasons.
- **Actual End Date** – The actual calendar date that the new version or release was delivered to systems engineering for integration and test.

Define the event that starts and ends the life cycle encompassed by the schedule.

Estimate data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

e. Effort (Mandatory) [OPS-29 Category: Organic Labor System Infrastructure] - The effort captured is that required to manage and maintain operational capability of systems in the field. The effort represents the number of staff-hours spent during the time from when allocated software requirements are provided for the version to when the software completes integration and test. The number of hours includes all directly chargeable hours to the software project including all of those expended by management, development, test and support personnel involved in getting the software product delivered. The data to be reported in this category includes:

- **Name** – the name of the project, release or version being described.

- **Estimated Effort (staff-hours or number FTE (Full Time Equivalents)** – The estimated effort in staff-hours or number of Full Time Equivalents (FTE) for the new version or release provided prior to the work begins.
- **Actual Effort (staff-hours or number FTE)** – The actual effort expended in staff-hours or FTE for the new version or release provided when the work was completed.

  Provide the conventions used for effort in terms of staff months, hours or FTE.

  Estimate data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

f. Cost (Mandatory) - The cost represents the actual year dollars ($) spent during the time from when allocated software requirements are provided to when the software release is replaced by a new release or the system is retired. The number of dollars ($) differs from effort as it includes all those expended on the project including those spent on licenses, travel, and other applicable costs like Certification and Accreditations (C&As), IAVA (Information Assurance & Vulnerability Assessment), sustaining engineering and field support. The data to be reported in this category includes:

- **Name** – the name of the project, release or version being described.
- **Labor Costs [OPS-29 Category: Organic Labor System Infrastructure]**
  - **Estimated Labor Costs ($)** – The estimated labor costs in $ for the new version or release prior to the work on it being started. Reflects the cost of the estimated effort identified above in either labor hours or FTE.
  - **Actual Labor Costs ($)** – The actual labor costs expended in $ for the new version or release when the work on it was completed. Reflects the cost of the actual effort identified above in either labor hours or FTE.
- **License Costs [OPS-29 Category: Licenses]**
  - **Estimated License Costs ($)** – The estimated license costs in $ for the new version or release prior to the work on the new version it being started. Includes both developmental and run-time licenses, as appropriate.
  - **Actual License Costs ($)** – The actual license costs expended in $ for the new version or release when the work on it was completed. Includes both developmental and run-time licenses, as appropriate.
- **Other Direct Costs (including Travel)**
  - **Estimated Other Direct Costs ($)** – The estimated other direct costs (including travel) in $ for the new version or release prior to the work on it being started. Includes all travel including field support and expendables.
  - **Actual Other Direct Costs ($)** – The actual other direct costs (including travel) expended in $ for the new version or release when the work on it was completed. Includes all travel including field support and expendables.
- **Facility Costs [OPS-29 Category: Sys Open Door]**
  - **Estimated Facility Costs ($)** – The estimated costs in $ for equipment, facilities and their maintenance in $ needed for sustain, test and support of the new version or release prior to the work on it being started.
  - **Actual Facility Costs ($)** – The actual costs in $ for equipment, facilities and their maintenance in $ needed to sustain, test and support the new version or release when the work on it was completed.
- **Certification and Accreditation Costs [OPS-29 Category: C&A's]**
  - **Estimated C&A's ($)** – The estimated costs in $ for conducting Defense Information Assurance Certification and Accreditation Process (DIACAP) or for flight safety

certifications. Includes the costs for analysis and updates/fixes needed to comply with standards.

- **Actual C&A's ($)** – The actual costs in $ for conducting Defense Information Assurance Certification and Accreditation Process (DIACAP) or for flight safety certifications. Includes the costs for analysis and updates/fixes needed to comply with standards.

- **Information Assurance Vulnerability Assessment Costs [OPS-29 Category: IAVA's]**
  - **Estimated IAVA ($)** – The estimated costs in $ for conducting IAVA. Includes the costs to patch software to address emerging Information Assurance (IA) and Vulnerability Alerts/Assessment (VA) requirements.
  - **Actual IAVA ($)** – The actual costs in $ for conducting IAVA. Includes the costs to patch software to address emerging Information Assurance (IA) and Vulnerability Alerts/Assessment (VA) requirements.
- **Sustaining Engineering Costs (SE)**
  - **Estimated SE ($)** – The estimated effort in $ for providing sustaining engineering support on a release by release basis. Besides performing tasks like network administration, security, quality assurance, independent testing, and configuration and distribution management, sustaining engineering provides user handholding, training and support.
  - **Actual SE ($)** – The actual effort in $ for providing sustaining engineering support on a release by release basis. Besides performing tasks like network administration, security, quality assurance, independent testing, and configuration and distribution management, sustaining engineering provides user handholding, training and support.
- **Field Software Engineers Costs [OPS-29 Category: FSEs]**
  - **Estimated FSE ($)** – The estimated effort in $ for supporting the system in the field including expenditures for contracts and contractors required to keep the system operational (hardware maintenance, etc.). Field support personnel troubleshoot and patch the system. They are not users or operators of the system.
  - **Actual FSE ($)** – The estimated effort in $ for supporting the system in the field including expenditures for contracts and contractors required to keep the system operational (hardware maintenance, etc.). Field support personnel troubleshoot and patch the system. They are not users or operators of the system.
- **Contractual Capability Sets [OPS-29 Category: Cap Sets FY (XX/XX)]**
  - **Estimated Cap Sets ($)** – The estimated costs in $ for acquiring a capability set for the fiscal years (XX/XX) that cover the following software releases (Release XXX to XXX).
  - **Actual Cap Sets ($)** – The actual costs expended in $ for acquiring a capability set for the fiscal years (XX/XX) that cover the following software releases (Release XXX to XXX).

  The contractual capability set includes the following costs:
  - Troubleshoot/correct issues
  - Cyclic release of new/revised versions
  - Respond to new threats or requirements
  - Maintain interoperability with other changing systems
  - Accommodate new weapons, systems, and/or munitions
  - Increase efficiency/effectiveness
  - Support new doctrine/tactics
  - Ensure compatibility with replacement COTS packages
  - Satisfy policy mandates

- Address affordability concerns

Each new capability set must be mapped to requirements, both functional and performance. External interfaces must be specified via an Interface Control Document (ICD). In order to assess reasonableness of cost, there also must be a detailed size estimate in SLOC provided along with effort estimates mapped to WBS work packages. Should there be an affordability goal; capabilities should be mapped to Total Ownership Costs (TOC).

- **System Mission Capability [OPS-29 Category: System Mission Cap]**
  - **Estimated System Mission Cap ($)** – The estimated costs in $ for acquiring system mission capabilities that support evolving system requirements.
  - **Actual System Mission Cap ($)** – The actual costs expended in $ for acquiring system mission capabilities that support evolving mission requirements.

  The acquired capabilities include the following costs:
  - Troubleshoot/correct issues
  - Cyclic release of new/revised versions
  - Respond to new threats or requirements
  - Maintain interoperability with other changing systems
  - Accommodate new weapons, systems, and/or munitions
  - Increase efficiency/effectiveness
  - Support new doctrine/tactics
  - Ensure compatibility with replacement COTS (Commercial Off-The-Shelf) packages
  - Satisfy policy mandates
  - Address affordability concerns

  Each set of mission capabilities must be mapped to requirements, both functional and performance. External interfaces must be specified via an Interface Control Document (ICD). In order to assess reasonableness of cost, there also must be a detailed size estimate in SLOC provided along with effort estimates mapped to Work Breakdown Structure (WBS) software work packages. Should there be an affordability goal; capabilities should be mapped to TOC at least across the POM life cycle for the project.

g. **Sources of Funds (Mandatory)** – This entry is used to identify which types of budgets cover the costs identified in the previous section. For each cost item, identify the type of budget (R&D (Research and Development), O&M (Operations and Maintenance), Production, FMS (Foreign Military Sales), other) and allocation amount in $ that you have been authorized by Fiscal Year (FY). Budgets should be expressed in current budget year dollars ($). Whenever possible, they should correlate directly to actual expenditures (previous year), current year and out-year POM budgets, as applicable.

- **Name** – the name of the project, release or version being described.
- **Source of Funds** – For the project, release or version, identify the type of funds (R&D, O&M, Production, FMS and other) and amount of $ allocated for each of the following cost categories for the previous, current and next two fiscal years (FY):
  - Labor costs
  - License costs
  - Other direct costs (including travel)
  - Facilities costs
  - C&A costs
  - IAVA costs

- Field software engineers cost
- Contractual capability sets
- Contractual system mission capabilities

h. **Budgets (Mandatory)** – Provide information about how your budgets are allocated and whether or not they are sufficient to satisfy expectations.

i. **Defects (Mandatory)** - The number of defects is determined by the tallying the number of Software Problem Reports (SPR) as they are entered into the problem reporting system. A defect is an error, flaw, mistake or fault in a software program that causes it to produce either incorrect or unexpected results, or causes it to behave in untended ways. Defects are sometimes separated by phase in which they are discovered in an attempt to determine how many escape detection in-phase and out-of-phase.

This set of data is being collected to define the relative quality of the release so it can be estimated using calibrated defect models. The data to be reported in this category includes:

- **Name** – the name of the project, release or version being described.
- **Number of Defects** – The actual number of defects in this version or release separated into the following five categories:
  - **Category 1 Defects (Catastrophic)** – the number of catastrophic defects found and fixed in this release. Catastrophic defects are those that prevent the accomplishment of an operational or mission-essential capability and for which no work-around solution is known. In addition, catastrophic defects include all system/software lockups and those defects that jeopardize safety, security, or other requirement designated "critical."
  - **Category 2 Defects (Critical)** – the number of critical defects found and fixed in this release. Critical defects are those that adversely affect the accomplishment of an operational or mission-essential capability and for which a work-around solution is not known. In addition, such defects include those that adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system and for which no work-around solution is known.
  - **Category 3 Defects (Serious)** – the number of serious defects found and fixed in this release. Serious defects are those that adversely affect the accomplishment of an operational and/or mission-essential capability, but for which a work-around solution is known.
  - **Category 4 Defects (Annoyance)** – the number of annoyance defects found and fixed in the release. Annoyance defects are those that typically result in user/operator inconvenience, but do not affect any required operational or mission-essential capability.
  - **Category 5 Defects (Minimal)** – the number of defects that both have minimal impacts and do not appear in any other category found and fixed in this release. They may be provided for informational purposes.
- **Defect Information** - Information supplied for each defect in each of these categories via a spreadsheet includes:
  - Number of latent defects; i.e., those existing prior to this release
  - Number of latent defects planned to be fixed in this release
  - Number of latent defects actually fixed in this release
  - Number of new defects found in this release
  - Number of new defects fixed in this release
  - Number of latent and new defects deferred to the next release

Provide information about whether you keep track of age of defects, defect backlogs and whether you use a software reliability model to predict number of defects expected and, if so, which one.

Defect data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

j. **Latent Defect Information (Mandatory)** – Latent defects are hidden flaws, weaknesses or imperfections that may cause failure or malfunction that is not discoverable by reasonable inspection until after delivery of a software release to the field. Information requested about latent defects by year include the follow:
   - Number of latent defects existing prior to the release found during maintenance.
   - Number of latent defects fixed in this release.
   - Number of new defects (those inserted by maintenance actions) found in this release.
   - Number of these defects that were fixed in this release.
   - Total number of latent and new defects found during maintenance in this release.
   - Total number of latent and new defects deferred to the next release.

   Defect data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

k. **Earned Value** - Earned value is a project management technique used to measure progress in an objective manner. It combines measurement of scope, schedule and cost into an integrated framework for determining status and assessing progress. The data to be reported in this category includes:
   - **Name** – the name of the project, release or version being described.
   - **Budgeted Cost of Work Performed (BCWP)** – the budgeted cost of the work actually completed.
   - **Actual Cost of Work Performed (ACWP)** – the actual cost of the work completed taken from the financial records.
   - **Budgeted Cost of Work Scheduled (BCWS)** – the budgeted cost of the work scheduled but not performed as of yet.
   - **To Completion Performance Index (TCPI)** - TCPI indicates the future required cost efficiency needed to achieve either a target BAC (Budget at Complete) or EAC (Estimate at Completion).
   - **TCPI (EAC)** – TCPI calculated based on EAC.
   - **Budget At Completion (BAC)** – the current budget allocated to complete the work.
   - **Estimate At Completion (EAC)** – the current estimated cost to complete the work.

   If you do not use earned value to track progress, identify the approach you use in its stead.

l. **Test Effort** - The effort represents the number of staff-hours spent to FQT the software. It does not include staff-hours for unit testing. However, it does include staff-hours needed to conduct dry runs and prepare automation scripts. The number of hours includes all directly chargeable hours to the software project including all of those expended by management, test and support personnel involved in getting the software product delivered. The data to be reported in this category includes:
   - **Name** – the name of the project, release or version being described.
   - **Number of Test Cases** – The actual number of test cases developed for the new version or release separated into the following categories:
     - **Dry Run** – the actual number of test cases that were developed for the dry-run of the new version or release.

- **Dry-Run Regression** – the actual number of automation scripts that were developed for the dry-run regression tests of the new version or release.
- **Formal Qualification Test (FQT)** – the actual number of test cases that were developed for the FQT of the new version or release.
- **FQT Regression** – the actual number of automation scripts that were developed for the FQT regression tests of the new version or release.

- **Test Case Effort (staff-hours)** – The actual effort expended in staff-hours for developing test cases for the new version or release separated into the following categories:
  - **Dry Run** – the actual effort expended in staff-hours to develop test cases for the dry-run of the new version or release.
  - **Dry-Run Regression** – the actual effort expended in staff-hours to develop scripts for the dry-run regression tests of the new version or release.
  - **Formal Qualification Test (FQT)** – the actual effort expended in staff-hours to develop test cases for the FQT of the new version or release.
  - **FQT Regression** – the actual effort expended in staff-hours to develop scripts for the FQT regression tests of the new version or release.
- **Number of Tests Run** – The actual number of tests run for the new version or release separated into the following categories:
  - **Dry Run** – the actual number of test cases that were run during the dry-run of the new version or release.
  - **Dry-Run Regression** – the actual number of automation scripts that were run during the dry-run regression tests of the new version or release.
  - **Formal Qualification Test (FQT)** – the actual number of test cases that were run during the FQT of the new version or release.
  - **FQT Regression** – the actual number of automation scripts that were run during the FQT regression tests of the new version or release.
- **Test Conduct Effort (staff-hours)** – The actual effort expended in staff-hours for conducting the testing of the new version or release separated into the following categories:
  - **Dry Run** – the actual effort expended in staff-hours to run the test cases developed during the dry-run. of the new version or release
  - **Dry-Run Regression** – the actual effort expended in staff-hours to run the scripts developed during the dry-run regression tests of the new version or release.
  - **Formal Qualification Test (FQT)** – the actual effort expended in staff-hours to run the test cases developed for the FQT of the new version or release.
  - **FQT Regression** – the actual effort expended in staff-hours to run the scripts developed for the FQT regression tests of the new version or release.
  - **Procedures** – the actual effort expended in staff-hours to develop and run specialized test procedures developed during the FQT of the new version or release. For example, the effort needed to run test procedures developed for flight safety certification might be included in this category.
- **Actual Test Cost ($)** – The actual test cost expended in $ for the new version or release separated into the following categories:
  - **Dry Run** – the actual effort expended in $ to run the test cases developed during the dry-run of the new version or release.
  - **Dry-Run Regression** – the actual effort expended in $ to run the scripts developed during the dry-run regression tests of the new version or release.
  - **Formal Qualification Test (FQT)** – the actual effort expended in $ to run the test cases developed for the FQT of the new version or release.

- **FQT Regression** – the actual effort expended in $ to run the scripts developed for the FQT regression tests of the new version or release.
- **Procedures** – the actual effort expended in $ to develop and run specialized test procedures developed during the FQT of the new version or release.

Test data need to be collected at the start of the cycle, updated at the start of the fiscal year (if multi-year) and finalized with actuals at the end of the cycle.

5. **Software Cost Model Information**

If a software cost model (COCOMO II, SLIM, SEER-SEM, True S, etc.) was used to develop effort and duration estimates, please provide a copy of the estimate file and basis for estimate for each software project, version or release. Multiple files are needed, i.e., that containing the initial estimate and another that updates the drivers to reflect the estimated cost- and schedule-to complete at the end the fiscal year for multi-year projects and actuals at the end of the effort. As an example, you may have planned to use experienced people for the job. But, they may have had difficulties finding them because the technology involved was so old that it has not been taught for years (e.g., developing software using the JOVIAL programming language and its integral COMPOOL features). The result is that the initial estimate assumed applications experience ("APEX" for the COCOMO II cost model) was "High" when in actuality it was "Low" for the updates. The values for experience should be captured along with an explanation in each updated file (cost-to-complete and actual). If you do not have these files, please provide cost driver information like that appearing the following two tables using the guidance on the model's web site (http://sunset.usc.edu for COCOMO or http://www.galorath.com for SEER).

a. **Cost Model** – Identify the cost model used to estimate resources for the release from the list provided in the questionnaire.

The materials provided in the remainder of this section are for the COCOMO II model. The following example cost model data is provided for the COCOMO II because its data model is public domain. Similar data should be provided when other cost models are used along with the size and additional information called out in this DID. When actuals are provided, please update the estimate ratings to reflect the most current and not the initial ratings for these parameters.

b. **COCOMO II Scale Factors** - Rate the following five COCOMO II scale drivers. These are the factors that influence the exponent of the estimating equation. When in doubt use the nominal setting. Please provide the two versions of this table that were requested.

	Very Low	Low	Nominal	High	Very High	Extra High	Estimate Rating
Precedentedness	Thoroughly un-precedented	Largely un-precedented	Somewhat un-precedented	Generally familiar	Largely familiar	Largely familiar	
Development Flexibility	Rigorous	Occasional relaxation	Some relaxation	General conformity	Some conformity	Some conformity	
Architecture/ Risk Resolution	Little 20%	Some 40%	Often 60%	Generally 75%	Mostly 90%	Mostly 90%	
Team Cohesion	Strongly adversarial	Occasionally cooperative	Moderately cooperative	Largely cooperative	Highly cooperative	Highly cooperative	
Process Maturity	CMM Level 1 (lower half)	CMM Level 1 (upper half)	CMM Level 2	CMM Level 3	CMM Level 4	CMM Level 5	

c. **COCOMO II Cost Drivers** - Rate the following seventeen COCOMO II cost drivers. These factors are multiplied together to adjust the project cost to factors that have been found to influence over the effort and duration estimates. When in doubt use the nominal setting. Again, please provide the two versions of this table that were requested.

	Very Low	Low	Nominal	High	Very High	Extra High	Estimate Rating
Required Software Reliability	Slight in-convenience	Low, easily recoverable losses	Moderate, easily recoverable losses	High financial loss	Risk to human life		
Data Base Size		D/P < 10	10 ≤ D/P < 100	100 ≤ D/P < 1000	D/P ≥ 1000		
Product Complexity	Simple	Straight-forward	Routine, some math, multi-file	Processing intense	Interrupt-driven	Complex real-time	
Required Reusability		None	Across project	Across Program	Across Product Line	Across Multiple Product Lines	
Documentation Match to Life Cycle Needs	Many life cycle needs uncovered	Some needs uncovered	Right-sized to life cycle needs	Excessive for life cycle needs	Very excessive for life cycle needs		
Execution Time Constraints			≥ 50% use of available exec. time	70% use	85% use	95% use	
Main Storage Constraints			≥ 50% use of available storage	70% use	85% use	95% use	
Platform Volatility		Major - 12 months Minor - 1 month	Major - 6 months Minor – 2 weeks	Major - 2 months Minor - 1 week	Major - 2 weeks Minor – - 2 days		
Analyst Capability	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile		
Programmer Capability	15th percentile	35th percentile	55th percentile	75th percentile	90th percentile		
Personnel Continuity	48%/year	24%/year	12%/year	6%/year	3%/year		
Application Experience	≤ 2 months	6 months	1 year	3 years	6 years		
Platform Experience	≤ 2 months	6 months	1 year	3 years	6 years		
Language/Tool Experience	≤ 2 months	6 months	1 year	3 years	6 years		
Use of Software Tools	Edit, code, debug	Simple front-end, backend CASE, little integration	Basic life cycle tools, moderate integration	Strong, mature tools, moderate integration	Strong, mature tools, well integrated with processes		
Site – Collocation	International	Multi-city and multi-company	Multi-city and multi-company	Same city or metro area	Same building or complex	Fully co-located	
Site – Communica-tions	Some phone, mail	Individual phone, FAX	Narrow-band email	Wide-band electronic comm.	Wideband electronic comm., some video conf.	Inter-active multi-media	
Required Development Schedule	75% of nominal	85% of nominal	100% of nominal	130% of nominal	160% of nominal		

d. **Comments** – Attach any additional explanatory materials to the questionnaire and submit it when completed to the identified sources.

Acronyms

The following acronyms have been used within this document:

ACWP	Actual Cost of Work Performed
APEX	Applications Experience (cost driver in COCOMO II model)
BAC	Budget at Completion
BCWP	Budgeted Cost of Work Performed
BCWS	Budgeted Cost of Work Scheduled
C&A	Certification and Accreditation
CM	Configuration Management
COTS	Commercial Off-The-Shelf
DID	Data Item Description
DOD	Department of Defense
EAC	Estimate at Completion
FMS	Foreign Military Sales
FQT	Formal Qualification Test
FSE	Field Software Engineers
FTE	Full Time Equivalent
FY	Fiscal Year
IAVA	Information Assurance Vulnerability Assessment
IBM	International Business Machines
ICD	Interface Control Document
O&M	Operations & Maintenance
PC	Personal Computer
POM	Program Objectives Memorandum
QA	Quality Assurance
R&D	Research and Development
SCR	Software Change Request
SE	Sustaining Engineering
SLOC	Source Lines of Code
SPR	Software Problem Report
TCPI	To-Completion Performance Index
TOC	Total Ownership Costs
UCC	Unified Code Counter
USC	University of Southern California
WBS	Work Breakdown Structure