**DATA ITEM DESCRIPTION**

**Title:** Software Resources Data Reporting: Development, Maintenance and Enterprise Resource Planning Development Reports, and Data Dictionary

**Number**: DI-MGMT-82035A
**AMSC Number:** **9867**
**DTIC Applicable:** No
**Preparing Activity**: CAPE
**Approval Date**: 20171115
**Limitation**:
**GIDEP Applicable:** No
**Project Number:** MGMT-2017-033
**Applicable Forms**: Forms are available to be used to submit required formats as follows:

| Software Data Format | Format Number | Form Number |
|---|---|---|
| Software Development Report | 1 | DD Form 3026-1 (REVISED) |
| Software Maintenance Report | 2 | DD Form 3026-2 (REVISED) |
| Enterprise Resource Planning (ERP) Software Development Report | 3 | DD Form 3026-3 |

1. **USE/RELATIONSHIP**: For background and detailed requirements related to Software Resources Data Reporting (SRDR), refer to DoD 5000.04-M-1 or the latest version of the “Cost and Software Data Reporting (CSDR) Manual.”
    1.1. CSDR is the Department of Defense (DoD) system for collecting actual costs and software data and related business data. The resulting database serves as the primary contract cost and software data (CSD) database for most DoD resource analysis efforts, including cost database development, applied cost estimating, cost research, program reviews, analysis of alternatives (AoA), and life cycle cost estimates. All formats may be used in response to Government solicitations according to Defense Federal Acquisition Regulation Supplement (DFARS) sections 234.7100, 234.7101, 242.503-2, 252.234-7003, and 252.234-7004:
        1.1.1. Format 1, DD Form 3026-1, “Software Development Report”, consists of two parts. Part 1, Software Development Technical Data, reports the software development size, context, and technical information. It consists of Release Level and Computer Software Configuration Item (CSCI) Level sections. CSCI is the lowest level of software development at which configuration management is performed by the developer. It is usually indicated by a separate Software Development Folder (SDF), Software Requirements Specification (SRS) etc. The CSDR plan will serve as the authoritative definition for reporting purposes. The Release Level Data includes all information applicable to the entire software release for the reporting event, defines each of the data elements as required, and describes the methods and rules used to perform the data measurement or estimation. The CSCI Level Data is used to obtain the estimated or actual (as-built) characteristics of a software product and its development process at the CSCI Level. Other terms for CSCI include Software End Item, Software Item (SI), etc., but this document will use CSCI as the primary term throughout. Part 2, Software Development Effort Data, reports the software development efforts associated with each reported release and CSCI. Format 1 uses the term “release” to refer to commonly used terms such as build, product build, and increment.
        1.1.2. Format 2, DD Form 3026-2, “Software Maintenance Report”, consists of two parts. Part 1, Software Maintenance Technical Data, reports the size, context and technical information. It consists of Top Level and Release Level sections. The Top Level Data includes all information applicable to the software maintenance release(s) for the reporting event, defines each of the data elements as required, and describes the methods and rules used to perform the data measurement or estimation. The Release Level Data is used to obtain the actual (as-built) characteristics of the maintenance product and its maintenance process at the Release level. Part 2, Software Maintenance Effort Data, reports the to-date software maintenance efforts for each in-progress and completed release(s) and the annual total software maintenance activities. In Format 2, the words "software release" refer to a set of changes

to the existing baseline software that are delivered to end users and that require formal testing, e.g., a new software configuration has been created, which can include patches, emergency releases, modifications and capability upgrades. Format 2 uses the term "release" to refer to commonly used terms such as release, build, increment or drop.

1.1.3. Format 3, DD Form 3026-3, "ERP Software Development Report", is a special case of the Software Development Report specifically for Enterprise Resource Planning or Defense Business System programs. These programs represent software development for business systems, such as financial, personnel or inventory control systems, either for a specific function or for a complete business enterprise. It consists of two parts. Part 1, System Technical Data, provides the report context, describes both the software product being developed and the development team, lists project requirements for modules, business processes, system functions and interfaces, and captures software product size and complexity at the Release Level. Part 2, Software Development Effort Data, captures project resource and schedule information at the Release Level. Format 3 uses the term "release" to refer to commonly used terms such as build, product build, and increment, but this document will use release as the primary term throughout.

1.2. The SRDR is structured around formats that contain the content and relationships required for the electronic submissions. This Data Item Description (DID) summarizes the Software Development Report, the Software Maintenance Report, and the ERP Software Development Report, and provides instructions to support the data and frequency requirements specified in the contract for CSDR reporting. Unless otherwise stated, instructions for the "Software Development Report" (DD Form 3026-1) in this DID apply also to the "ERP Software Development Report" (DD Form 3026-3). The primary purpose of this data is as follows:

1.2.1. The intent of the SRDR process is to collect objective, measurable data commonly used by industry and DoD cost analysts. These data are used to compile a repository of estimated and actual software product sizes, schedules, effort, and quality that Government analysts can draw upon to build credible size, cost, and schedule estimates of future software-intensive systems.

1.2.2. The Software Development Reports are not management reports. The intent is not to track the progress of software development during contract execution. It does, however, collect the person-hours expended during software development, as well as other software measures.

1.2.3. The Software Maintenance Report collects the person-hours expended during software maintenance and other software measures. It is not a management report. It is not intended for tracking progress of maintenance during contract execution.

1.3. Ground Rules, Terms and Definitions

1.3.1. The minimum level of detail to be reported in each SRDR submission shall be in accordance with the CSDR Plan, as approved by the Office of the Secretary of Defense (OSD) Deputy Director, Cost Analysis (DDCA) or Service Cost Director if not an ACAT I program. Discrete reporting is required for all Work Breakdown Structure (WBS) elements identified in the CSDR Plan.

1.3.2. In the Software Maintenance Report, the maintainer must fill out the Top-Level Data section that defines the data elements. The definitions of the data items must address and provide context to the following categories of data: Context, Project Description, Size, Effort, and Schedule. Definitions as stated in ISO/IEC/IEEE 12207:2008, ISO/IEC 14764: 2006 and other cited standards, e.g. MIL-STD 881, shall be used.

1.3.3. At the annual reporting date, a Software Maintenance Report contains the Common Heading Data (or Metadata), Part 1 Top Level Data, the Part 1 Release Level Data, and the Part 2 data which provides effort by release(s) and annual total effort for associated software maintenance activities. The annual reporting date shall be in accordance with the government fiscal years.

1.3.4. ISO 14764:2006 defines software maintenance as "the totality of activities required to provide cost-effective support to a software system. In this DID, the words software maintenance and software sustainment are synonymous. In this DID, the word “maintainer” refers to the organization that is performing maintenance activities on software. An organization can be either a government institution or a commercial contractor.

1.4. This DID supersedes the following DIDs:
- DI-MGMT-81739B
- DI-MGMT-81740A
- DI-MGMT-82035

## 2. REQUIREMENTS:

2.1. Reference documents. The applicable issue of the documents cited herein, including their approval dates and dates of any applicable amendments, notices, and revisions, shall be as cited in ASSIST at the time of the solicitation; or, for non-ASSIST documents, as stated herein.

2.2. References.

2.2.1. DoD Instruction 5000.02, “Operation of the Defense Acquisition System,” [current version], available at http://www.dtic.mil/whs/directives/. This instruction contains mandatory CSDR requirements.

2.2.2. DoD Instruction 5000.73, “Cost Analysis Guidance and Procedures,” [current version], available at http://www.dtic.mil/whs/directives/.

2.2.3. DoD 5000.04-M-1, “Cost and Software Data Reporting (CSDR) Manual,” [current version], available at http://cade.osd.mil/.

2.2.4. MIL-STD-881, “Work Breakdown Structures for Defense Materiel Items”, [current version], available at https://assist.daps.dla.mil/.

2.2.5. “Operating and Support Cost-Estimating Guide”, [current version], available at http://cade.osd.mil/.

2.2.6. DoD Forms Management Program (i.e. DD Form 2360, Software Description Annotated Outline, and DD Form 250, Material Inspection and Receiving Report), available at http://www.dtic.mil/whs/directives/.

2.2.7. “ISO/IEC 12207, Systems and software engineering – Software life cycle processes,” available at http://www.iso.org/. This document describes standard software development activities. The upcoming update will harmonize this with the ISO/IEC/IEEE 15288.

2.2.8. “ISO/IEC/IEEE 15288, Systems and software engineering – System life cycle processes.” Available at http://www.iso.org/. This document was initially adopted for use by the DoD and also describes the standard software development activities.

2.2.9. “ISO/IEC 20926, Software and systems engineering – Software measurement – IFPUG functional size measurement method 2009.” Available at http://www.iso.org/. This document establishes a standard for function point (FP) counting.

2.2.10. ISO/IEC TR 24748-1, “Systems and software engineering – Life cycle management. Part 1: Guide for life cycle management,” available at http://www.iso.org/. This document outlines the problem classification by priority (see Annex C, Figure C.2).

2.2.11. ISO/IEC 14764:2006, “Software Engineering -- Software Life Cycle Processes – Maintenance," available at http://www.iso.org/iso/. This document describes software maintenance activities.

2.2.12. Unified Code Counter-Government (UCC-G), available at http://cade.osd.mil/. This link provides the approved version of the University of Southern California (USC) Center for Systems and Software Engineering (CSSE) Unified Code Counter, upon which independent verification and validation (IV&V) has been conducted by the Government. The approved version of the tool is also referred to as the UCC-G.

2.2.13. “Defense Agile Acquisition Guide”; The MITRE Corporation, March 2014. This document seeks to adapt proven principles of Agile development specifically to the DoD context.

2.2.14. “Agile and Earned Value Management: A Program Manager’s Desk Guide”; OUSD AT&L (PARCA), March 2016. This document is intended as an informative resource for Department of Defense (DoD) personnel who encounter programs on which Agile philosophies are applied.

2.3. Implementation. The CSDR requirement applies to program contracts and subcontracts regardless of contract type based on the dollar thresholds as specified in DoDI 5000.02.

2.3.1. These reporting requirements also apply to individual Work Breakdown Structure (WBS) elements (or group of WBS elements) within Major Defense Acquisition Programs (MDAPs) and Major Automated Information System (MAIS) programs that are separately managed by other U.S. Government Program Managers.

2.3.1.1. These WBS elements retain the Acquisition Category (ACAT) designation of the parent program and are subject to the same reporting thresholds and requirements as those elements that are directly managed by the parent MDAP.

2.3.1.2. Reporting is required throughout the complete life cycle to include the Operating and Support (O&S) phase of the program.

2.3.2. Contractors are responsible for implementing CSDR requirements on all subcontracts that meet the reporting thresholds as specified in DoDI 5000.02.

2.4. Format. Use the detailed preparation instructions below as required by the DDCA-approved CSDR Plan.

2.4.1. Electronic Submission of Data. The Common Heading (or Metadata) and Part 1 sections of the Software Development Reports  and the Software Maintenance Report shall be submitted electronically in accordance with the CAPE/DCARC-approved Extensible Markup Language (XML) schema to the DCARC's secure website using the CSDR Submit-Review System. There will be no XML reporting required for Part 2 of the Software Development Reports or the Software Maintenance Report; this data will be reported in human readable format.

2.4.1.1. The XML file can be generated automatically from the electronic Excel file (or vice versa) with DCARC's CSDR Planning and Execution Tool (cPET) located at the DCARC website: http://cade.osd.mil. This conversion only works if the file is 100% compatible with the approved Excel formats. The submitting organization retains 100% responsibility for the XML submission.

2.4.1.2. Uploading requires the use of either a DoD Common Access Card (CAC) or a DoD-approved External Certification Authority (ECA) certificate. See the CADE website for cPET and certificate instructions: http://cade.osd.mil/.

2.4.1.3. The purpose of the XML format is to facilitate entry of the data into the Government Cost database being developed from the former Excel- and document-based repository.

2.4.2. Human Readable. The Government may, in the CDRL, require the Part 1 of the Software Development Report and the Software Maintenance Report in human readable format. The submission of the Part 2 of the Software Development Report and the Software Maintenance Report will always be in human readable format.

2.4.3. Representation of Data in XML. The CAPE/DCARC-approved XML Schemas provide a machine-readable representation of the data described in this document. For ease of exposition, some instructions may refer explicitly to the human readable formats.  Any such instructions shall be interpreted with respect to the XML formats so as to render an equivalent representation of the data that complies with the CAPE/DCARC-approved XML Schemas.  For example:

2.4.3.1. Any instruction to enter NA (for "not applicable") in a field or a column shall be interpreted as an instruction to report a NULL value in the XML format using the appropriate mechanism defined in the XML Schema.

2.4.3.2. Any instruction to enter a date in a specific format (such as "YYYYMMDD") in a field or a column shall be interpreted as an instruction to report a date value in the XML format using the appropriate mechanism defined in the XML Schema

2.5. Content.

2.5.1. The Software Development Report shall reflect scope relevant to the reporting event. When the development project is divided into multiple releases, each representing product-level software delivered to the government, information and data shall be submitted for each release. (Other terms for release include build, product build, and increment, but this DID will use release throughout.) SRDR Final submissions for completion of a release shall reflect size, schedule, and (if required) effort, of that release.

2.5.2. The Software Maintenance Report shall contain actual as-built software measurement data. The data shall reflect scope relevant to the reporting event. When the maintenance project is divided into multiple releases (or sets of changes to the existing baseline software that are delivered to end users and that require formal testing) the submission shall reflect each release. Releases associated with Information Assurance Vulnerability Alerts (IAVA) shall be reported as a release. If frequent IAVA releases are completed, the contractor may group the IAVA releases into a single release in the Software Maintenance Report, but this must be agreed upon by the CWIPT and noted in the CSDR plan.

2.5.3. Mandatory data elements for the Software Development Report and the Software Maintenance Report are outlined below.

## 3. PREPARATION INSTRUCTIONS

### 3.1. General Instructions

3.1.1. OSD DDCA-Approved CSDR Plan. All reporting under this DID must be in accordance with the CSDR Plan approved by the OSD DDCA.

3.1.1.1. The reporting level is defined at the WBS level established by the OSD DDCA-approved CSDR Plan. The Software Development Report reporting shall occur at the CSCI level, as described below. The ERP Software Development Report reporting shall occur at the Release level as specified in the CSDR Plan.

3.1.1.2. Software effort data reported in Part 2 must follow WBS parent-child relationship rules (e.g., WBS parent elements must be equal to the sum of their children elements).

3.1.1.3. As lower-level WBS elements are defined, the CSDR Plan will be updated to reflect the changes.

3.1.1.4. Initial, Interim, and Final Reports. This DID specifies the content of Initial, Interim, and Final Reports reporting events which are specified in the DDCA-approved CSDR Plan.

3.1.1.4.1. Software Development Report Types and Definitions

- Initial [Contract] Reports provide initial assessments of and estimates for all software sizing, schedule, effort, and are essential in capturing actual data on the observed growth of these quantities from established baselines. The Initial Report (to include Release-level reporting) shall be submitted after contract award (represented notionally as Event 1 in Figure 1), no later than the beginning of the first release. Release Reporting estimates included in the Initial Report, for not-yet-started releases may be updated, as needed, in Interim Reports for subsequent reporting events. Initial Reports for the Releases contain estimated values.
- An Interim Report is any report other than the Initial Report that is prepared before submission of the Final Report, in accordance with the OSD DDCA-approved CSDR Plan. Interim Reports generally coincide with the beginning or end of a release. In addition, Interim Reports can be required on major contract events and program milestones such as Software Requirements Review (SRR), Preliminary Design Review (PDR), etc. An Interim Report can also be required if a program re-baselines due to user requirements change. Interim Reports contain the Initial Release Reporting for not-yet-started releases (if different from previous reports); Interim Release Reporting for in-progress releases; and Final Release Reporting for completed releases (if different from previous reports). Interim Release

Reporting contains a mix of actuals to date and estimates at completion (EACs).

- Final Reports are intended to capture actual software sizing, schedule, and effort associated with the software development, in accordance with the OSD DDCA-approved CSDR Plan. The Final Report shall be submitted before the end of the contract, no earlier than the end of the last release. Final Release Reporting for previously-completed releases may have been included in earlier Interim Reports. Final Release Reporting contains actual values. The Final Report, containing Final Release Reporting for all releases, must satisfy two conditions:
  - The final end item has been delivered and accepted by the government (e.g., as evidenced by a completed DD 250) or higher-tier contractor in the case of a subcontractor.
  - 95% or more of total contract costs have been incurred
- The figure below helps depict a notional expected reporting process for each software release in the Software Development Reports. The number of releases and the degree of overlap shown is notional. It is expected (but not required) that the Releases reported in the ERP Software Development Report, Format 3, will be sequential with little-to-no overlap.

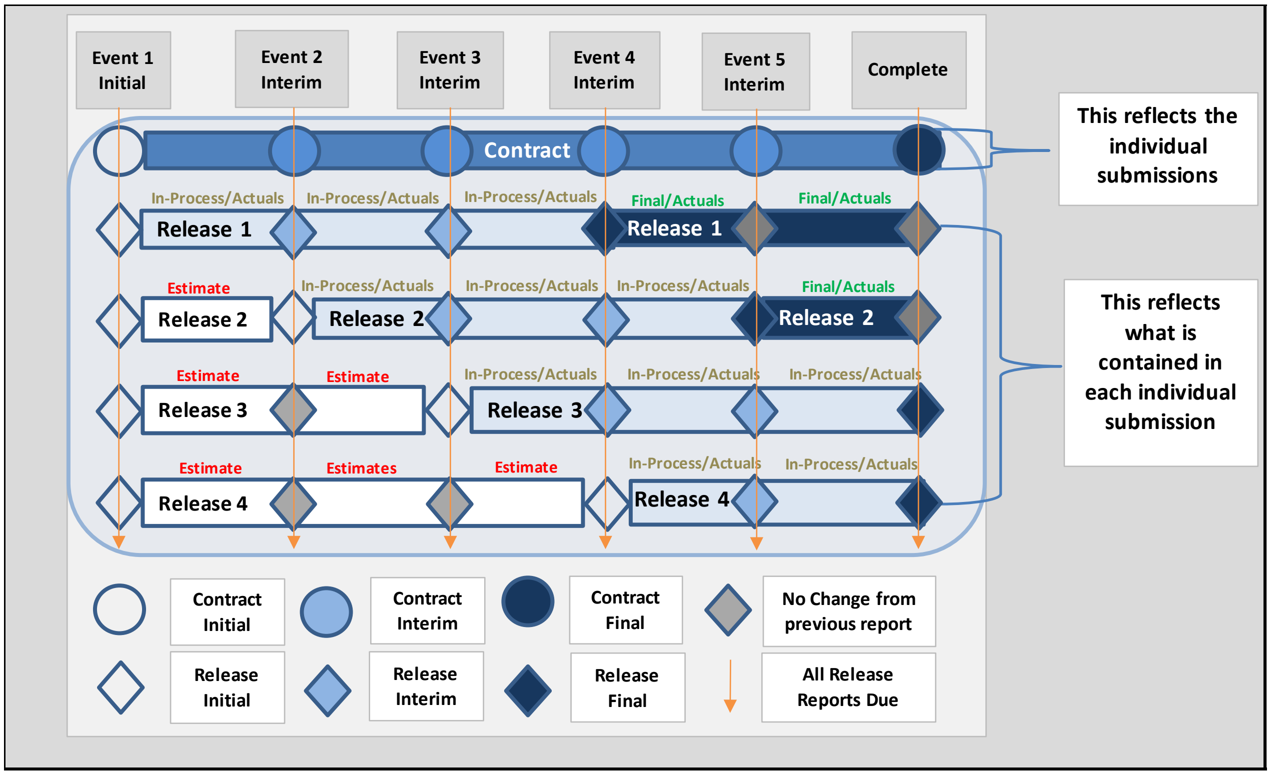

**Figure 1: Software Development Reporting Process**

3.1.1.4.2. Software Maintenance Report Types and Definitions.

- Final reports contain all actual cumulative data for delivered release(s) and are intended to capture all or substantially all costs associated with the annual software maintenance effort. All final reports are prepared in accordance with the OSD DDCA-approved CSDR Plan. A Final Report must satisfy two conditions:
  - The final end item has been delivered and accepted by the government (e.g., as evidenced by a completed DD 250) or higher-

tier contractor in the case of a subcontractor.

- 95% or more of total costs have been incurred. In the case of a support or sustainment contract which has no deliverable end item, the Final Report must capture 95% or more of total costs. If the OSD DDCA-approved CSDR Plan, requires a report at DD 250 and the submitted report has more than 95% of contract costs incurred, the report shall be marked final.
- The figure below helps depict the expected reporting process for the Software Maintenance Report, Format 2

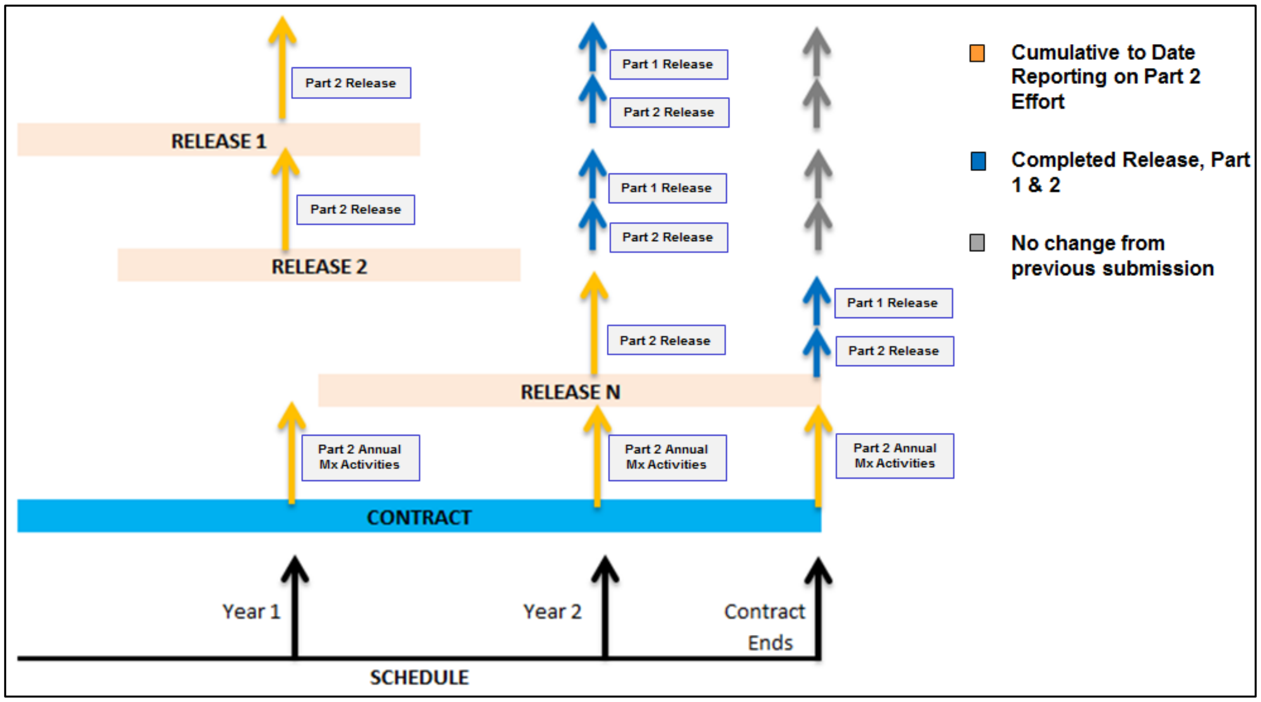

**Figure 2: Software Maintenance Reporting Process**

3.1.1.5. Entries for common data elements (i.e., metadata, quantities, dollars, and hours) used across the DD series of reports for a specific contract must agree as appropriate.

3.1.2. Scope of Reporting

3.1.2.1. Report all currency throughout the forms associated with this DID in thousands of U.S. Dollars, rounded to the nearest tenth. Report all hours in thousands, rounded to the nearest tenth. Enter "0" (zero) for items with null amounts; do not leave items blank.

3.1.2.2. CSDR Reporting on fixed price contracts is not limited to the contract price. All effort associated with the contract must be reported even if the amount exceeds the contract price.

3.1.2.3. Prime Contractors must report on work at cost (i.e., before the summary elements such as Reporting Contractor General & Administrative (G&A), Undistributed Budget (UB), Management Reserve (MR), Facilities Capital Cost of Money (FCCM), and Profit/Loss or Fee). Prime contractors report on work performed by all subcontractors at price (i.e., including subcontractor Profit/Loss or Fee).

3.1.2.4. Effort shall not be omitted based on contract CLIN structure or definition.

3.1.2.5. The Software Development Report shall report EAC's for effort hours for Initial and Interim Reports as specified in the CSDR Plan.

3.1.2.6. Final CSDR reports which contain actual effort less than 100% will require an estimate at completion that includes the remaining effort.

3.1.3. Security Requirements. Mark the security classification of each report as “Unclassified.” However, if the report is classified, contact the DCARC for special processing instructions. Please note: “Proprietary” is not an official DoD security classification. Indicate in the Comments (Section 3.2.1.17) if the use of proprietary disclosure statement is required.

3.2. Specific Instructions: Metadata

3.2.1. Common Heading Information. Preparation instructions for the common heading information apply to all Formats.

3.2.1.1. Program. Enter the name given to the MDAP (ACAT IC or ID) or to the Major Automated Information Systems (MAIS) (ACAT IAC or AM) program as specified on the Defense Acquisition Management Information Retrieval (DAMIR) Program List (e.g., “BLACKHAWK UPGRADE (UH-60M) – Utility Helicopter Upgrade Program”). The name entered must be identical to the name on the DAMIR Program List (http://www.acq.osd.mil/damir/).

3.2.1.2. Phase/Milestone. Check one of the following for the appropriate Phase/Milestone being reported:

- Pre-A (Material Solution Analysis Phase)
- A (Technology Maturation and Risk Reduction Phase)
- B (Engineering and Manufacturing Development Phase)
- C-LRIP (Low-Rate Initial Production)
- C-FDD (Full Deployment Decision for ERP Software Development programs)
- C-FRP (Full-Rate Production)
- C-FD (Full Deployment for ERP Software Development programs)
- O&S (Operations and Support Phase).

3.2.1.3. Prime Mission Product. Enter the most current official military designation for the end item as specified by the appropriate classification standard (e.g., “Military Designation of Military Aerospace Vehicles,” would specify “F-35” for the Joint Strike Fighter).

- For contract (or subcontract) CSDR Plan, the end item being reported may have a different designation than the total program (e.g., the preparer would enter “AN/APG-81 Radar” for the F-35 Radar contract CSDR Plan).
- If the end item does not have a military designation, enter the type of product being developed or procured (e.g., radar, automated information system).

3.2.1.4. Reporting Organization Type. Check the box for the appropriate organization type:

- Prime/Associate Contractor
- Direct-Reporting Subcontractor
- Government

3.2.1.5. Performing Organization. Enter the following information for the organization responsible for reporting the software development or maintenance effort. It is assumed that this is the same as the performing organization unless the effort is outsourced in which case it is separately identified in Section 3.3.2.4, Outsourced Development Organization or Outsourced Maintenance Organization.

3.2.1.5.1. Organization Name. Enter the name and address (city, state and zip code) of the organization actually performing the work

3.2.1.5.2. Division Name. Enter the reporting organization’s division name and address (city, state, and zip code) if different from the performing organization.

3.2.1.6. Approved Plan Number. Enter the Approved Plan Number of the current OSD DDCA-approved contract or subcontract CSDR Plan, including revision that authorized the collection of data for this report.

3.2.1.7. Customer (Direct-Reporting Subcontractor Use Only).

- Enter the name of the prime contractor for whom the work on the subcontract is being performed.
- Otherwise enter NA (for “not applicable”).

3.2.1.8. Type Action.

3.2.1.8.1. Contract Number. Enter the assigned contract number the prime contractor has with the Government customer. This requirement is identical for both reporting contractors and reporting subcontractors.

3.2.1.8.2. Latest Modification Number. Enter the number of the latest contract modification. This requirement is identical to both reporting contractors and reporting subcontractors

3.2.1.8.3. Solicitation Number.

- If the data are in response to a solicitation in accordance with DFARS sections 234.7101, 252.234-7003, and 252.234-7004, enter the solicitation number.
- Otherwise enter NA (for “not applicable”).

3.2.1.8.4. Name. Enter the common reference name for the prime contract.

3.2.1.8.5. Task Order/Delivery Order/Lot Number. If the contract contains task order(s), delivery order(s), and/or lot number(s) being reported on for which the CSDR Plan has reporting requirements, enter one of the following:

- “Task Order” followed by a blank space and the applicable number.
- “Delivery Order” followed by a blank space and the applicable number.
- “Lot Number” followed by a blank space and the applicable number.
- “NA” (for “not applicable”).

3.2.1.9. Period of Performance. Enter the dates for the data being reported (contract, lot, delivery order, or task). Enter the appropriate numeric data for the year, month, and day in YYYYMMDD format. For example, December 31, 2015, would be shown as 20151231.

- Start Date: Enter the actual start date.
- End Date: Enter the projected or actual end date.

3.2.1.10. Report Type. Check the box for the appropriate report type.

- Initial
- Interim
- Final

3.2.1.11. Submission Number. Enter the submission number for the report provided of the current OSD DDCA-approved CSDR Plan.

3.2.1.12. Resubmission Number. A resubmission occurs if prior submission(s) for the submission event were officially rejected with a memo signed by the DCARC Director. Enter “0” (zero) for original submission. If the report is a resubmission, enter the resubmission number, starting with “1” for the first resubmission, “2” for the second resubmission, and so on.

3.2.1.13. Report As Of. Enter the appropriate numeric data for the year, month, and last day of the reporting period. For example, December 31, 2015, would be shown as 20151231.

- For event-driven submissions, the report as of date should be consistent with the event outlined in the OSD DDCA-approved contract or subcontract CSDR Plan.
- If an event date changes due to a programmatic schedule slip, adjustment to the “As of Date” reported in the CSDR Plan must be requested through the CSDR Submit-Review (SR) system for DCARC approval by the Government Program Office prior to the date reflected in the CSDR Plan. A date change request does not require an official CSDR Plan revision.

3.2.1.14. Point of Contact (Name). Enter the following information for the person to contact for answers to any questions about the content of the report:

- Name (Last Name, First Name, and Middle Initial)
- Department
- Telephone Number (Including Area Code)
- E-Mail Address

3.2.1.15. Date Prepared. Enter the appropriate numeric data for the year, month, and day of the date the report was prepared in the appropriate numeric format. For example, December 31, 2015, would be shown as 20151231.

3.2.1.16. Appropriation. Check all the appropriate boxes to identify the type of appropriation used to fund the data reported:

- Research, Development, Test and Evaluation (RDT&E)
- Procurement
- Operations and Maintenance (O&M)

3.2.1.17. Remarks. Note any relevant information that could be used in the interpretation of the data provided in this report. This item must not contain actual data. Include the following (if applicable):

- For contractors that have direct-reporting subcontractors, identify each direct-reporting subcontractor, including any government entity, by name, city, state, and subcontract price.
- Provide analyst context for analyzing the data, such as any unusual circumstances that may have caused the data to diverge from historical norms.
- If the data contained within the submission are proprietary, provide the proper proprietary verbiage.

**<u>Format 1, DD Form 3026-1, "Software Development Report"</u>**

3.3. <u>Software Development Report – Part 1 Software Development Technical Data</u>

3.3.1. Release Level Data. Format 1 includes the following data at the Release Level. Release Level data must be reported for each release included in the SRDR, as specified by the OSD DDCA approved CSDR Plan.

3.3.1.1. Release ID/Name

3.3.1.1.1. Release ID. Enter the software release ID specified in the CSDR Plan.

3.3.1.1.2. Release Name. Enter the software release name specified in the CSDR Plan.

3.3.1.2. Release Schedule Reporting. Format 1 shall contain anticipated or actual schedules at the Release level. On Initial Reports, provide the projected start and end dates of the software release. On Interim Reports, provide the projected start and end dates for not-yet-started releases; the actual start date and projected end date for in-progress releases; and the actual start and end date for complete releases. On the Final Report, provide the actual start and end dates.

3.3.1.2.1. Release Start Date: Enter actual or projected start date in YYYYMMDD format.

3.3.1.2.2. Release End Date: Enter actual or projected end date in YYYYMMDD format.

3.3.1.3. Software Requirements Count Definition. Provide the contractor's specific rules and tools used to count requirements reported in Section 3.3.2.6. The definition shall address what types of requirements are included in the count, such as functional, non-functional, security, safety, privacy, certification and accreditation, and other derived requirements; the units, such as "shalls," "sections," or paragraphs; and counting methods used. The definition must also identify the source document used for tallying requirements and must map and track to said source documents such as (primarily) the Systems Requirements Specification (SRS), and (secondarily) the Software Development Plan (SDP), and Software Architecture Design Document (SADD).

3.3.1.4. External Interface Requirements Count Definition. Provide the contractor's specific rules and tools used to count external interface requirements reported in Section 3.3.2.6.2. The definition shall address what types of requirements are included in the count, such as functional, non-functional, security, safety, privacy, certification and accreditation, and other derived requirements; the units, such as "shalls," "sections," or paragraphs; and counting methods used. The definition must also identify the source document used for tallying requirements and must map and track to said source documents such as (primarily) the Interface Requirements Specification (IRS), Systems Viewpoint 6 (SV-6), and (secondarily) the Software Development Plan (SDP), and Software Architecture Design Document (SADD).

3.3.1.5. Hours per Staff-Month. Provide the number of hours per staff-month used to convert between labor hours and full-time equivalent (FTE) staff reported in Section 3.3.2.5.12.

3.3.1.6. Hours per Staff-Month Type. Indicate whether the reported hours per staff-month reflect an accounting standard or a computation. If a computation, check the box and provide the total labor hours and staff used, and indicate which efforts are included in those totals.

3.3.1.7. Product Quality Reporting Definition. Provide the contractor's internal definitions for product quality metrics being reported and specific rules and tools used to produce the metrics. If a metric is based on computations involving two or more metrics, clear definitions of all metrics used must be provided along with a description of the formula used.

3.3.1.8. Software Process Maturity. Format 1 shall report the characterization of the developer's software process maturity using a methodology such as the Software

Engineering Institute (SEI) software Capability Maturity Model (CMM), the Capability Maturity Model Integration (CMMI)-SW, or an alternative equivalent rating. The reported software process maturity shall reflect the rating that the primary development organization has formally certified as of the date of the reporting event. If no formal certification has been conducted, leave these items below blank. If a single submission is used to represent the work of multiple organizations, enter the level of the organization that will be expending the most effort on the development project (not necessarily the prime contractor) and note which organization the rating reflects. If the Government has accepted an alternative assessment mechanism, such as the Air Force's Software Development Capability Evaluation (SDCE), enter those results and explain the meaning of the assessment.

- Software Process Maturity. Identify the developer's software process maturity rating.
- Lead Evaluator. Identify the name of the person that performed the assessment.
- Evaluator Affiliation. Identify the evaluator's affiliation.
- Certification Date. Identify the date of certification.

3.3.1.9. Software Development Activities. Format 1 shall report schedule and effort data related to all the SW development activities listed below. These activities are taken directly from the ISO 12207:2008 software implementation and support processes (reference 2.2.7). An update to this international standard is scheduled to be released and will be aligned with the ISO/IEC 15288. This alignment takes the first step toward harmonization of the structures and contents of the two international standards and provides the foundation to facilitate evolution to an integrated and fully harmonized treatment of life cycle processes. This DID requires the contractors to only report activities from Requirements Analysis to Qualification Testing, as well as, activities under Software Support Processes.

- Software Requirements Analysis (RA)
- Software Architectural Design (AD)
- Software Detailed Design (DD)
- Software Construction (Con)
- Software Integration (I)
- Software Qualification Testing (QT)
- Software Support Processes (SP)

3.3.1.9.1. Contractor-Defined Software Development Activities. If the contractor accounting system collects effort by SW development activities below the CSCI level, Format 1 shall be used to provide a definition of those activities. Format 1 shall identify the name of each SW development activity, and indicate all of the ISO/IEC 12207 software implementation and support processes included in that activity.

3.3.1.9.2. SW-specific Common WBS Elements. Consistent with the CSDR Plan, Format 1 shall provide the common WBS elements such as Systems Engineering and Program Management, Systems Test and Evaluation, etc., that can be attributed to software development efforts. In order to understand the effort reported in relation to the ISO/IEC 12207, Format 1 shall indicate all of the ISO/IEC 12207 software implementation and support processes included in each of the identified common elements.

3.3.1.10. System Description. Provide a top-level system description of the product that this software supports, what the software does, and a description of the software release reported in the SRDR.

3.3.1.11. Precedents. List the full names of at least three similar systems and briefly describe how those systems are analogous to the current release. These systems should be comparable in software size, scope, and complexity. Include programs developed by the same software organization or development team whenever possible. If precedents list changed from previous reports, the report shall provide explanation.

3.3.1.12. Code Counter Version. Format 1 shall use the Government-approved version of the University of Southern California (USC) Center for Systems and Software Engineering (CSSE) Unified Code Count (UCC) tool for the below required Source Lines of Code (SLOC) counts. At the time of submission of actual code counts (not planned counts), the most recent version of the code counter must be used. If subsequent versions of the code counter do not change original code counts by more than plus or minus one percent (1%), they are permitted for later submissions. If a contractor-specific code counter is also being used, provide name, version number, description, and results of any benchmarking studies (e.g., how counts compare to those generated by UCC-G). See list outlined below.

- UCC-Government (UCC-G) version
- Alternate Code Counter Name
- Alternate Code Counter Version
- Alternate Code Counter Description
- Alternate Code Counter Comparison to UCC-G

3.3.1.13. Release-Level Comments. Note any relevant information that could be used in the interpretation of the Release-Level Data provided in this report. This item must not contain actual data.

3.3.2. CSCI Level Data. Format 1 includes the following data at the CSCI level. CSCI Level data must be reported for each CSCI for each release included in the SRDR as specified by the OSD DDCA approved CSDR Plan.

3.3.2.1. Release ID/Name

3.3.2.1.1. Release ID. Enter the software release ID specified in the CSDR Plan.

3.3.2.1.2. Release Name. Enter the software release name specified in the CSDR Plan.

3.3.2.2. CSCI ID/Name

3.3.2.2.1. CSCI ID. Enter the CSCI ID specified in the CSDR Plan.

3.3.2.2.2. CSCI Name. Enter the CSCI Name specified in the CSDR Plan.

3.3.2.3. WBS Element Code/WBS Element Name.

3.3.2.3.1. WBS Element Code. Enter the WBS Element specified in the CSDR Plan.

3.3.2.3.2. WBS Element Name. Enter the WBS Element Name specified in the CSDR Plan.

3.3.2.4. Outsourced Development Organizations.

3.3.2.4.1. Name. List the names of the companies or organizations that took part in the development of the CSCI if different than the reporting organization. If outsourced development organizations are reported, an explanation must be placed in the Outsourced Development Organizations Comments below (Section 3.3.2.4.4).

3.3.2.4.2. Location. List the corresponding locations of the companies or organizations that took part in the development of the CSCI if different than the reporting organization.

3.3.2.4.3. Primary or Secondary Developer. Indicate whether each company or organization was a primary or secondary developer of the software product(s). Primary developer is defined as performing the largest portion of the software development for the CSCI, and may be the prime contractor or one of the subcontractors. Secondary developer is defined as not performing the largest portion of the software development for the CSCI.

3.3.2.4.4. Outsourced Development Organizations Comments. If outsourced development organizations are reported, provide comments about the reporting organization's relationship to the development organizations and the development organization's effort on the software development.

3.3.2.5. Product and Development Description.

3.3.2.5.1. Functional Description. For each element reported, provide a description of its function. The Functional Description addresses what the CSCI is, what it does, and how it interfaces with other elements (both internal and external to the effort). If available, describe precedent systems at the CSCI level. Reference precedent systems cited in Section 3.3.1.11.

3.3.2.5.2. Software Development Characterization. Provide a brief description for each element reported that characterizes the software development work undertaken or to be undertaken on that element. Examples might include completely new development, rehosting of software to different processor/operating system, reengineering of legacy code into open architecture, and translation of legacy code from Ada to C.

3.3.2.5.3. Software State of Development. Indicate whether the software is Prototype, Production-Ready, or a Mix of the two.

3.3.2.5.4. Operating Environment(s). Identify the operating environment or environments in which the developed software system operates.

- Surface Fixed – software is embedded in a system at a fixed site
- Surface Mobile – software is embedded in a system that is either moved and set up independently or in a platform
- Surface Portable – software is embedded in a handheld or portable device
- Surface Vehicle – software is embedded as part of a moving vehicle
- Air Vehicle – software is embedded as part of an aircraft
- Sea Systems – software is embedded as part of a surface or underwater boat/ship
- Ordnance Systems – software is embedded as part of a rocket or other ordnance systems
- Missile Systems – software is embedded as part of a missile
- Space Systems – software is embedded as part of a spacecraft
- Other – provide explanation if other

3.3.2.5.5. Manned vs. Unmanned. For the operating environment above, indicate if it is a manned or unmanned environment.

3.3.2.5.6. Application Domain. Identify one primary application domain (i.e., the end-user mission) developed or to be developed as part of the given CSCI from those listed below. Definitions and examples may be found on the CADE website: http://cade.osd.mil/ .

- Primary Application Domain
  - – Microcode and Firmware
  - – Signal Processing
  - – Vehicle Payload
  - – Vehicle Control
  - – Other Real-Time Embedded
  - – Command and Control
  - – Communication
  - – System Software
  - – Process Control
  - – Scientific and Simulation
  - – Test, Measurement, and Diagnostic Equipment

– Training
– Software Tools
– Mission Planning
– Custom AIS Software
– Enterprise Service System
– Enterprise Information System

- Application Domain Comments. Provide any additional information that will aid the analyst in interpreting the application domain data.

3.3.2.5.7. Development Process. Enter the name of the development process followed for the development of the software. Typical development processes include waterfall, spiral, Rapid Application Development (RAD), and Agile development. If the contractor uses Agile, indicate whether it is part of an Agile Acquisition approach, and whether it would be considered Hybrid Agile (e.g., Waterfall for Architecture and Requirements, followed by Agile for design, code, and unit test). If the contractor uses or used an atypical internal process, provide a description of the development process in Product and Development Description Comments (Section 3.3.2.5.13).

3.3.2.5.8. Software Development Method(s). Identify the software development method or methods used to design and develop the software product, such as Structured Analysis, Object Oriented, or Vienna Development Method.

3.3.2.5.9. Upgrade or New Development. Indicate whether the primary development will be or was new software or an upgrade. If the primary product is a new development check the box, otherwise leave the box unchecked indicating an upgrade. A software system is considered new if no existing system currently performs its function or if the development completely replaces an existing system. A software system that replaces part of an existing system, such as the replacement of a database, is considered an upgrade. An existing software system that was ported to a new platform or reengineered to execute as a Web or distributed application, for example, would be considered an upgrade unless it was also completely redeveloped from scratch (new requirements, architecture, design, process, and code).

3.3.2.5.10. Software Reuse.

3.3.2.5.10.1 Name. Identify by name the software products reused from prior development efforts, such as source code, software designs, or requirements documentation.

3.3.2.5.10.2 Description. Briefly describe software products reused from prior development efforts.

3.3.2.5.11. COTS/GOTS/Open-Source Applications Used.

- Name. List the name of each application or product that constitutes part of the final delivered product, whether Commercial Off-The-Shelf (COTS), Government Off-The-Shelf (GOTS), or Open-Source products. If a proprietary application or product that is not generally commercially available will be included, identify it here and include any necessary explanation in COTS/GOTS Comments.
- Glue Code. Format 1 shall contain an estimated or actual count of new/modified glue code required to accommodate each package. This count shall be included in the total reported in Section 3.3.2.6.3.
- Configuration Effort. Format 1 shall contain an estimated or actual amount of effort required to configure COTS/GOTS applications identified. This effort is not included in the effort and schedule sections below.
- COTS/GOTS Comments. Provide any additional information that will

aid the analyst in interpreting COTS/GOTS data.

3.3.2.5.12. Staffing (Initial and Interim Release Reporting only).

- Peak Staff. For the element reported, enter the estimated peak team size, measured in full-time equivalent (FTE) staff. Include only direct labor. FTE reported should be consistent with the conversion factor given in Section 3.3.1.5.
- Peak Staff Date. Enter the date when the estimated peak staffing is expected to occur. The appropriate numeric data for the year, month, and day is in YYYYMMDD format.

3.3.2.5.13. Product and Development Description Comments. Provide any additional comments about the product and development description.

3.3.2.6. Product Size Reporting.

3.3.2.6.1 Software Requirements Count. Provide the estimated or actual number of software requirements. The method of counting actual number of requirements implemented by the development software must be the same as that used for counting estimated requirements (as reported in the SRDR Initial Development Report). Do not count requirements concerning external interfaces not under project control. Section 3.3.1.3 under Release Level Data shall provide a definition of what types of requirements are included in the count and the counting methods used.

- Total Requirements. Enter the estimated or actual number of total requirements satisfied by the developed software product at the completion of the release or contract. This count must be consistent with the total size of the delivered software (i.e., it must not solely focus on new development, but must reflect the entire software product). If the final total software requirements count differs from the SRDR Initial Development Report by more than 25% (higher or lower), provide explanation(s) (e.g., scope increase) in Software Requirements Comments.
- Inherited Requirements. Of the total estimated or actual number of requirements reported in sub-bullet one in Section 3.3.2.6, enter how many were inherited requirements.
- Added/New Requirements. Of the total estimated or actual number of requirements reported in sub-bullet one in Section 3.3.2.6, enter how many were added or new requirements.
- Modified Requirements. Of the total estimated or actual number of requirements reported in sub-bullet one in Section 3.3.2.6, enter how many were modified requirements.
- Deleted Requirements. Enter the estimated or actual number of requirements deleted, and thus not included in the total reported in sub-bullet one in Section 3.3.2.6.
- Deferred Requirements. Enter the estimated or actual number of requirements deferred, and thus not included in the total reported in sub-bullet one in Section 3.3.2.6.
- Requirements Volatility (Initial and Interim Release Reporting only). Indicate the amount of requirements volatility expected during development as a percentage of requirements at the Software Requirements Review (SRR) that will change or be added thereafter.
- Security, Safety, and Privacy Requirements. Of the total estimated or actual number of requirements reported in sub-bullet one in Section 3.3.2.6, enter how many were Security, Safety, and Privacy requirements.

- Certification and Accreditation Requirements.  Of the total estimated or actual number of requirements reported in sub-bullet one in Section 3.3.2.6., enter the estimated or actual number of requirements addressing Cybersecurity, Information Assurance Vulnerability Management (IAVM), Airworthiness, Safety, and Networthiness. Cybersecurity, formerly Information Assurance (IA), and the Risk Management Framework (RMF) for DoD Information Technology, formerly the DoD Information Assurance Certification and Accreditation Process (DIACAP), are processes that verify the software system against externally defined domain performance criteria.
- Software Requirements Comments. Provide any additional information that will aid the analyst in interpreting requirements data.

3.3.2.6.2 Number of External Interface Requirements. Provide the estimated or actual number of external interface requirements, as specified below, not under project control that the developed system satisfies. External interfaces include interfaces to computer systems, databases, files, or hardware devices with which the developed system must interact but which are defined externally to the subject system. If the developed system interfaces with an external system in multiple ways (such as for reading data and also for writing data), then each unique requirement for interaction should be counted as an interface requirement. The method of counting actual number of external interface requirements handled by the development software must be the same as that used for counting estimated interface requirements (as reported in the SRDR Initial Development Report). Section 3.3.1.4 under Release Level Data shall provide a definition of what types of requirements are included in the count and the counting methods used.

- Total External Interface Requirements. Enter the estimated or actual number of total external interface requirements satisfied by the developed software product at the completion of the release or contract. This count must be consistent with the total size of the anticipated or delivered software (i.e., it must not solely focus on new development, but must reflect the entire software product). If the final total external interface requirements count differs from the SRDR Initial Development report by more than 25% (higher or lower), provide explanation(s) (e.g., scope increase) in External Interface Requirements Comments.
- Inherited External Interface Requirements. Of the total estimated or actual number of external interface requirements reported in sub-bullet one in Section 3.3.2.6.2, enter how many were inherited requirements.
- Added/New External Interface Requirements. Of the total estimated or actual number of external interface requirements reported in sub-bullet one in Section 3.3.2.6.2, enter how many were added or new requirements.
- Modified External Interface Requirements. Of the total estimated or actual number of external interface requirements reported in sub-bullet one in Section 3.3.2.6.2, enter how many were modified requirements.
- Deleted External Interface Requirements. Enter the estimated or actual number of external interface requirements deleted, and thus not included in the total reported in sub-bullet one in Section 3.3.2.6.2.
- Deferred External Interface Requirements. Enter the estimated or actual number of external interface requirements deferred, and thus not included in the total reported in sub-bullet one in Section 3.3.2.6.2.
- External Interface Requirements Volatility (Initial and Interim Release Reporting only). Indicate the amount of external interface requirements

volatility expected during development as a percentage of requirements at the Software Requirements Review (SRR) that will change or be added thereafter.

- Security, Safety, and Privacy Requirements. Enter the estimated or actual number of Security, Safety, and Privacy requirements.
- Certification and Accreditation Requirements. Of the total requirements reported in sub-bullet one in Section 3.3.2.6.2, enter the estimated or actual number of requirements addressing Cybersecurity, Information Assurance Vulnerability Management (IAVM), Airworthiness, Safety, and Networthiness. Cybersecurity, formerly Information Assurance (IA), and the Risk Management Framework (RMF) for DoD Information Technology, formerly DoD Information Assurance Certification and Accreditation Process (DIACAP), are processes that verify the software system against externally defined domain performance criteria.
- External Interface Requirements Comments. Provide any additional information that will aid the analyst in interpreting external interface requirements data.

3.3.2.6.3 SLOC-Based Software Size. Format 1 shall capture the estimated or actual delivered size of the product developed, not including any code that might be needed to assist development but that will not be delivered (such as temporary stubs, test scaffoldings, or debug statements). Additionally, the code shall be partitioned (exhaustive with no overlaps) into the below development categories. The code shall also be segregated into two groups: developed by the prime contractor, or developed by one or more subcontractors. The latest Government approved version of the UCC-G tool (reference 2.2.12), shall be used for Interim and Final SRDRs to obtain a set of standardized code counts that reflect logical size, as opposed to physical, non-commented, etc. For Interim SRDRs, the reporting organization will count the code that has undergone verification and validation (V&V) to date, and report the estimate at complete for the whole software development effort. Provide SLOC by software language in descending order of percentage of overall size. Specify the top two languages used (L1, L2 as outlined on the form) and combine the rest in "Other." These can be a compiled language, such as FORTRAN, Ada, or C; an interpreted language, such as BASIC; or a graphical or model-based language, such as Rhapsody/UML or Simulink. If the UCC-G tool is not available for the given language, specify how the code counts were derived (i.e., with UCC-G tool for a similar language, or using an Alternative Code Counter). The intent is to capture not just Delivered SLOC (DSLOC) but also information that will enable the calculation of Equivalent SLOC (ESLOC).

- New Code. New code shall be partitioned into human-generated and auto-generated code.
- Reused Code With Modification. When code from external sources (government-furnished or otherwise) is included that is or was reused with modification, Format 1 shall provide an assessment of the amount of redesign, recode, and retest required to implement the modified or reused code. Modified code is defined as pre-developed code that can be incorporated in the software component with a significant amount of effort, but less effort than required for newly developed code. In addition to the size of Modified Code, Format 1 shall provide either individual Design Modification (DM), Code Modification (CM), and Integration Modification (IM) percentages, *or* the Adaptation

Adjustment Factor (AAF) used for each component's Modification calculation. DM is the fraction of design that must be modified in order to adapt the component to meet the functional and interface requirements of the new system. CM is the fraction of code that must be modified in order to implement the new functions and interfaces. IM is the fraction of integration effort required to integrate the adapted software into the overall product, and to test the resulting product compared to the normal amount of integration effort for software of comparable size. AAF is defined as the top-level percentage of effort of adapting the software to the new system relative to a line of new code (e.g., a Modified line of code might require 50% of the effort of its equivalent New line of code when considering the design, code, and testing work that has already been performed). While individual DM, CM, and IM percentages are preferred, a single AAF percentage will be accepted when these are not available. Any deviation from a standard AAF percentage for Modified code shall be explained. The below items may be taken into consideration in assessing DM, CM, and IM factors.

Redesign
• Required an architectural design change
• Required a detailed design change
• Required reverse engineering
• Required modification to existing documentation
• Required revalidation of the new design
Recode
• Required code changes
• Required code reviews
• Required unit testing
Retest
• Required test plans to be rewritten
• Required test procedures to be identified and written
• Required new test reports
• Required test drivers and simulators to be rewritten
• Required integration testing
• Required formal demonstration testing

- Reused Code Without Modification. Code reused without modification is code that has no design or code modifications. However, there may be an amount of retest required. In addition to the size of Reused Code, Format 1 shall provide either an individual IM percentage, or the AAF used for each component's Reuse calculation. (For Reused Code Without Modification, DM and CM are defined to be 0%, with effort existing in the IM portion of adaptation.) While an individual IM percentage is preferred, a single AAF percentage will be accepted when it is not available. Any deviation from a standard AAF percentage for Reused code shall be explained.
- Carryover Code. Do not count the same code as new in more than one SRDR incremental report. Format 1 shall distinguish between code developed in previous releases that is carried forward into the current

release and code added as part of the effort on the current release. Examples of such carryover code include code developed in Spiral 1 that is included in Spiral 2 or code that is developed for Version 3.0 software that is included in Version 3.1 software. Table 1 provides one possible example of reporting code from previous releases for the Final Developer Reports associated with releases. Release 1 Total Delivered Code is used as a Carryover Without Modification at the start of Release 2. Release 2 Total Delivered Code is used as Carryover With and Without Modification for Release 3.

Table 1: Example of Reporting Carryover Code from Previous Builds

| | | **Release 1 Finish** | **Release 2 Finish** | **Release 3 Finish** | **Contract Finish** |
|---|---|---|---|---|---|
| | | | | | |
| New Code | Human Generated | 1,000 | 0 | 2,500 | 3,500 |
| | Auto Generated | 0 | 500 | 2,500 | 3,000 |
| External Reused | With Modification | 5,000 | 15,000 | 500 | 20,500 |
| | Without Modification | 3,000 | 0 | 2,000 | 5,000 |
| Carryover Code from Previous Release | With Modification | 0 | 0 | 12,250 | N/A |
| | Without Modification | 0 | 9,800 | 13,250 | N/A |
| Government Furnished Code | With Modification | 800 | 200 | 0 | 1,000 |
| | Without Modification | 0 | 0 | 0 | 0 |
| Total Delivered Code | | 9,800 | 25,500 | 33,000 | 33,000 |
| Deleted Code | | 0 | 0 | 100 | 100 |

- Auto-generated Code. If the developed software contains auto-generated source code, Format 1 shall include an auto-generated code sizing partition as part of the set of development categories.
- Government Furnished Code. Indicate the amount of Government Furnished Code delivered to the contractor. If the Government did not provide the contractor any code, enter 0.
- Deleted Code. Format 1 shall include the amount of code that was created and subsequently deleted from the final delivered code and hence not included in the reported total.

3.3.2.6.4 Agile Measures. If an Agile approach is used for development of software, enter data here to describe the technical scenario in accordance with the approved CSDR plan.

The following definitions and instruction apply to the Agile Software Development Efforts:

- Each Software Development SRDR reports Agile Development data for each Release. For reporting agile development metrics, a "Release" is a defined compilation of software features delivered within a specific time frame. In this report, a CSCI is represented by high level requirements referred to as Epics or Capabilities. Data shall be reported for each CSCI for each agile development Release as specified by the OSD DDCA approved CSDR plan. Specific software end items are explained in paragraph 3.3.2.6.5 below as "Features" within a Release. The software development reporting process shown in Figure 1 for sequential Releases also applies to Agile Releases within a contract.

- The Release is divided into "Sprint" time boxes, which are used to schedule specific daily workload to complete from a Backlog of defined software development tasks. Each Sprint is a recurring, non-overlapping cadence for development, usually 1 - 6 weeks in duration.
- Work within a Sprint is assigned to Scrum teams, who meet daily to plan activities, assign Backlog, and assess progress and impediments.

Include the vendor's specific "Days per Release" and "Days per Sprint" or other productivity (time duration) metrics that apply to the Release.

3.3.2.6.4.1 Release Map. The workload to assign at the Sprint level represents the high level requirements (Epics/Capabilities referred to as CSCIs in this report) that are then decomposed into the requirements into well-defined Feature Stories or cases. Each Feature Story must then be decomposed into smaller Stories that define the workload to assign at the Sprint level (Backlog). Output from each Sprint (completed software) shall be integrated at the Feature level into working software that the user can evaluate. One or more Features roll-up comprise completed Epics/Capabilities or CSCIs. One or more Epics/Capabilities/CSCIs comprise the Release.

Use the "Release Map" to identify the relationship of Features to Epics/Capabilities/CSCIs for the Release; one or more Features per Epic/Capability/CSCI. Add additional lines to accommodate the full Feature/Epic map. The Feature or Epic/Capability "Identifier" is assigned by the developer using their own conventions for naming or tracking workload. If terms other than "Feature" or "Epic" are used (for example: Acquisition Requirements Documents – or ARDs; Software Requirements Specifications – or SRSs), then provide the details in the Feature Description column.

Enter a short Feature Description for each Epic/Capability/CSCI reported to provide a brief description of its function.

3.3.2.6.4.2 Planned and Achieved Development (by Feature per Epic/Capability/CSCI). Development tasks in the Backlog are sized using a sizing convention, such as number of Feature Stories, Story Points, Function Points or SLOC. Planned and Actual Stories describe the relative complexity of the Feature. Story Points represent a relative scale that assigns a size to each task in the Backlog, so that the tasks can be "fit" into Sprints. Planned Story Points represent an estimate for all of the Backlog tasks needed to complete each Feature. Actual Story Points represent the known size of the Backlog tasks actually completed per Feature. If known, Total Hours required to complete the Backlog tasks for each Feature should also be included. The example row shows that each row includes a measure of the Planned and Actual Story Points along with Hours by Feature (one Feature per row).

Note: "Hours" are a measure of effort. Effort is captured specifically in Part 2 Software Development SRDR, as mapped to a breakout of functions, such as: Planning, Design, Build, etc … The Agile development hours included here are not mapped to any specific Part 2 function, and will overlap the hours provided in Part 2. The Feature hours included here will be used to help correlate specific Story Point sizing to the effort required per Feature, rather than correlating the Story Point sizing with the overall effort for the Release.

3.3.2.6.4.3 Summary Total for this Release. This table summarizes the entries for all Release Map rows, and is not aligned with any specific Release Map row. The total number of Features, Epics/Capabilities, Story Points and Feature Hours recorded as "Planned and Achieved Development" are summed here, along with the total number of Sprint time periods scheduled within the Release. Planned values apply to the Initial and Interim Reports. Actual values apply to Interim and Final reports.

3.3.2.6.5 Non-SLOC-Based Software. SLOC-Based is required unless reporting alternative sizing measures are agreed to by the CWIPT and documented in the CSDR Plan. If the following sizing measures are used instead of or in addition to SLOC-based measures, the associated information shall be provided.

- FP Measure. If Function Points (FP) are used for sizing, then the IFPUG standard (reference 2.2.9) shall be used. The Initial Report shall provide the FP count. Interim and Final Reports shall include updated FP counts, if available, but they shall also include actual SLOC counts per the guidelines above. The SRDR shall indicate the count type and include counts of both Data Functions – Internal Logical Files (ILF) and External Interface Files (EIF) – and Transactional Functions – External Inquiries (EQ), External Inputs (EI), and External Outputs (EO) – all broken out by Low, Average, and High. The Total Unadjusted FP count shall be multiplied by a Value Adjustment Factor (VAF) to produce and Adjusted FP count.
- RICE-FW Measure. If RICE-FW are used, counts shall be provided for Initial, Interim, and Final Reports, broken out by Low, Medium, and High Complexity. Any standards or guidelines applied in developing counts shall be fully explained.

3.3.2.6.6 Other Measures. If other sizing measures are used, other than Function Points, SLOC or Agile measures, they shall be named, and the associated counts provided. Any standards or guidelines applied in developing counts shall be fully explained.

3.3.2.6.7 Product Size Reporting Comments. Provide any additional comments about product size reporting.

3.3.2.7. Product Quality Reporting. Quality shall be quantified operationally (through failure rate and defect discovery rate). However, other methods may be used with DCARC approval. If different, explain the differences in the Section 3.3.1.7 under Release Level Data.

3.3.2.7.1 Number of Defects Discovered. Report the actual total number of software and firmware defects that were discovered. Format 1 shall list the actual defect software and firmware discovery counts for both Priority 1 and Priority 2, as described in the following table as referenced in ISO/IEC TR 24748-1 (reference 2.2.10).

Table 2: Software Defect Priority

| Priority | Criteria |
|---|---|
| 1 | a) Prevent the accomplishment of an operational or mission essential capability |
| | b) Jeopardize safety, security, or other requirement designated “critical” |
| 2 | a) Adversely affect the accomplishment of an operational or mission essential capability and no workaround solution is known |
| | b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, and no work-around solution is known |
| 3 | a) Adversely affect the accomplishment of an operational or mission essential capability but a workaround solution is known |
| | b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, but a work-around solution is known |
| 4 | a) Result in user or operator inconvenience or annoyance but does not affect a required operational or mission essential capability |
| | b) Result in inconvenience or annoyance for development or support personnel, but does not prevent the accomplishment of those responsibilities |
| 5 | Have any other effect |

3.3.2.7.2 Number of Defects Removed. Report the actual total number of software and firmware defects resolved in order to pass qualification testing. Of the total defects resolved, Format 1 shall list the total resolved defect software and firmware counts for Priority 1 and Priority 2, as defined in the above table.

3.3.2.7.3 Number of Defects Deferred. Report the total number of software and firmware defects deferred to post deployment or other software releases. Of the total defects deferred, Format 1 shall list the total deferred defect software and firmware counts for Priority 1 and Priority 2, as defined in the above table.

3.3.2.7.4 Product Quality Reporting Comments. Provide any comments about product quality for the given CSCI. Include detailed definitions and general explanations in the Section 3.3.1.7 under Release Level Data section.

3.3.2.7.5 CSCI Schedule Reporting. Format 1 shall contain anticipated or actual schedules at the CSCI level, broken out by contractor-specific activities as defined in Section 3.3.1.9 where available. Provide the projected start and end dates for the total element or the lowest-level software development activity that the contractor tracks. If there were multiple start and end dates for the same activity, as would be the case for iterative or spiral development, then report the earliest start date and latest end date for each activity, to the extent that is sensible for the approach used. On Initial Reports, provide the projected start and end dates. On Interim Reports, provide the projected start and end dates for not-yet-started CSCIs/activities; the actual start dates and projected end dates for in-progress CSCIs/activities; and the actual start and end dates for complete CSCIs/activities. On the Final Report, provide the actual start and end dates.

- CSCI Start Date: Enter actual or projected start date in YYYYMMDD format.
- CSCI End Date: Enter actual or projected end date in YYYYMMDD format.

3.3.2.7.6 Schedule Comments. Provide any comments about schedule reporting.

## 3.4. <u>Software Development Report – Part 2 Software Development Effort Data</u>

3.4.1. Resource Reporting. The Initial and Interim Reports shall contain actual-to-date and EAC total effort. The Final SRDRs shall contain the final actual effort hours expended by month for all levels reported (i.e., CSCI level, contractor-defined SW development activities, SW-specific Common WBS elements). Contractor accounting pulls for effort should follow standard accounting months; no additional mapping of time periods is required for Format 1. Provide the end date for each accounting month provided, as well as the end date of the month immediately preceding the first month reported. Monthly reporting should be included for completed releases in Interim Reports, where possible.

3.4.1.1. Prime Contractor Effort. Enter the staff hours (estimated or actual) for software development effort performed by the Prime Contractor as defined below.

- CSCI level, broken out by contractor-defined SW development activities as identified in Section 3.3.1.9.1. All WBS elements must be reported as specified in the CSDR Plan and WBS parent elements must be equal to the sum of their children elements.
- SW-specific Common WBS Elements as identified in Section 3.3.1.9.2 for each release within Format 1 as specified in the CSDR Plan. Format 1 shall provide staff hours for the common WBS elements such as System Engineering and Program Management, Systems Test and Evaluation, etc. that can be attributed to software development efforts if already tracked by the prime contractor. The Final Report for each release shall provide cumulative actual effort hours expended by month.

3.4.1.2. Subcontractor Effort. For the CSCI-level WBS elements, Format 1 shall contain the total subcontracted effort by either hours or cost, whichever is available. For each reported subcontractor hours or dollars, indicate which outsourced development contractors are included in those totals.

3.4.1.2.1 Enter the total staff hours (estimated or actual) for all software development effort performed by Subcontractors for this element.

3.4.1.2.2 Enter the total cost (estimated or actual) for all software development effort performed by subcontractors for this element.

3.4.2. The contractor is not required to create Control Accounts for the sole purpose of reporting EACs for effort hours on the SRDR and should be consistent with the OSD DDCA-approved CSDR Plan. Format 1 shall indicate in Effort Comments whether or not the element being reported is below a pre-established control account.

3.4.3. Effort Comments. Provide any comments about resource reporting.

**Format 2, DD Form 3026-2, "Software Maintenance Report"**

3.3. Software Maintenance Report – Part 1, Software Maintenance Technical Data

3.3.1. Format 2 includes the following top-level data and data at the Release Level, in addition to the Common Heading Information above.

3.3.1.1. System Description. Provide a top-level system description of the final overall product maintained or to be maintained and include the MIL-STD-881 system type.

3.3.1.2. Number of Unique Baselines Maintained. Provide a count of the number of unique baselines of the software that are concurrently maintained. This includes the total of those currently fielded (that are being maintained) and those that are actively being modified. If able, explain the type of commonality with the system being maintained and the amount of commonality by percentage. Multiple baselines may exist to support different platform configurations.

3.3.1.3. Approximate Number of Users. Provide the approximate number of users of the Information Technology (IT) system. IT users are the number of accounts or people who interact or access the system. If unknown, list NA.

3.3.1.4. Number of Total Hardware Platforms this Software Operates On. Provide a count of the total number of hardware platforms (operating system types) where the software is resident.

3.3.1.5. Operation Tempo. Describe the operation tempo for the field units using this software as either:

- Extensive (full-time use of the system; system is central to mission accomplishment)
- Regular (less than full-time use of the system)
- Event-driven (occasional use of the system)
- Other (provide explanation)

3.3.1.6. Software Process Maturity. Format 2 shall report the characterization of the maintainer's software process maturity using a methodology such as the Software Engineering Institute (SEI) software Capability Maturity Model (CMM), the Capability Maturity Model Integration (CMMI)-SW, or an alternative equivalent rating. The reported software process maturity shall reflect the rating that the primary maintenance organization has formally certified as of the date of the reporting event. If no formal certification has been conducted, leave these items below blank. If a single submission is used to represent the work of multiple organizations, enter the level of the organization that will be expending the most effort on the maintenance project (not necessarily the prime contractor) and note which organization the rating reflects. If the Government has accepted an alternative assessment mechanism, such as the Air Force's Software Development Capability Evaluation (SDCE), enter those results and explain the meaning of the assessment.

- Software Process Maturity. Identify the developer's software process maturity rating.
- Lead Evaluator. Identify the name of the person that performed the assessment.
- Evaluator Affiliation. Identify the evaluator's affiliation.
- Certification Date. Identify the date of certification. The date format is year, month, and day (YYYYMMDD).

3.3.1.7. Lead Government Organization. Enter the following information for the lead government maintenance organization actually performing or managing the maintenance work.

- Lead government organization name
- Location. Enter the lead government maintenance location

3.3.1.8. Precedents. List the full names of at least three similar systems and briefly describe how those systems are analogous to the current system being maintained. These systems should be comparable in software size, scope, and complexity. Include programs maintained by the same software organization or development team whenever possible. Also provide a description as to how each analogous system identified is similar to the system being maintained.

3.3.1.9. Software Requirements Count Definition. Provide the maintainer's specific rules and tools used to count requirements reported in 3.3.2.6.3. The definition addresses what types of requirements are included in the count, such as functional, non-functional, security, safety, privacy, Cybersecurity, and other derived requirements; the units, such as "shalls," "sections," or paragraphs; and counting methods used. The definition must also identify the source document used for tallying requirements and must map and track to said source documents such as (primarily) the Systems Requirements Specification (SRS), and (secondarily) the Software Development Plan (SDP), and Software Architecture Design Document (SADD).

3.3.1.10. External Interface Requirements Count Definition. Provide the maintainer's specific rules and tools used to count external interface requirements reported in 3.3.2.6.2. The definition shall address what types of requirements are included in the count, such as functional, non-functional, security, safety, privacy, Cybersecurity, and other derived requirements; the units, such as "shalls," "sections," or paragraphs; and counting methods used. The definition must also identify the source document used for tallying requirements and must map and track to said source documents such as (primarily) the Interface Requirements Specification (IRS), Systems Viewpoint 6 (SV-6), and (secondarily) the Software Development Plan (SDP), and Software Architecture Design Document (SADD).

3.3.1.11. Software Size Definitions.

3.3.1.11.1. SLOC-Based Software Size. Describe the rules and the tools used to count the Source Lines of Code (SLOC).

3.3.1.11.1.1 Unified Code Counter – Government (UCC-G). If software size is being reported using SLOC counts, the Government approved version of the Unified Code Count (UCC-G) tool shall be used to obtain a set of standardized code counts that reflect logical size, as opposed to physical, non-commented, etc. These results shall be included in the report unless other sizing metrics are used. The most recent version of the UCC-G at the time of first submission shall be used. If subsequent versions of the code counter do not change original code counts by more than plus or minus one percent (1%), they are permitted for later submissions.

- Provide the UCC-G version used for the code counts.

3.3.1.11.1.2 Contractor-specific Code Counter. If a contractor-specific code counter is also being used, provide name, version number, description, and results of any benchmarking studies (e.g., how counts compare to those generated by UCC-G). See list outlined below.

- Alternate Code Counter Name
- Alternate Code Counter Version
- Alternate Code Counter Description
- Alternate Code Counter Comparison to the UCC-G

3.3.1.11.2. Non-SLOC Based Software Size. If another sizing measure is used, provide the type (e.g. function points, story points, RICE objects) and version/definition (e.g. Common Software Measurement International Consortium (COSMIC), Full Function Points (FFP), MKII function points). Any standards or guidelines applied in developing counts should be explicitly referenced, or fully explained.

3.3.1.12. Software Change Count Definition. If software changes are tracked, explain how software changes are tracked and what tools are used to do the tracking. Software changes may be called Change Requests, Problem Reports, Issues, Defects, or other organizational / program specific names. Provide the internal definitions for change count measures being reported and specific rules used to count the measures. If a derived measure is based on a combination of two or more base or derived measures, clear definitions must be provided along with a description of the formula used to aggregate the numbers resulting in the derived measure reported. Include an explanation of the specific software changes that result from Information Assurance Vulnerability Alerts (IAVAs), which are separate from other software change requests, problem reports, issues, or defects. The counts for IAVA changes should be mutually exclusive from Priority 1-5 changes and not double counted (i.e. Total number of software changes should be the sum of IAVAs and P1-P5 changes). Provide separate rules for counting IAVA changes.

3.3.1.13. Release Schedule Information. Indicate what event defines a release start date and a release end date. For example, a release start date might be predicated on the completion of:

- System Requirements Review (SYS RR)
- Software Requirements Review (SW RR)
- Configuration Steering Board (CSB)
- Change Board Review (CBR)
- Some other event (provide explanation)

A release end date might be predicated on the completion of:

- End of Software Integration and Test (SW I&T)
- End of Acceptance Test
- End of System I&T
- Delivery to Field
- Some other event (provide explanation)

3.3.1.14. Comments. Note any relevant information that could be used in the interpretation of the data provided in this report. This item must not contain actual data. Include the following (if applicable):

- Provide context for analyzing the data such as any unusual circumstances that may have caused the data to diverge from historical norms.

3.3.2. The SRDR for Maintenance also calls for reporting Release-Level Data. For a delivered release, cumulative actual values shall be provided in this section.

3.3.2.1. Report Context and Maintenance Organization.

3.3.2.1.1. Release ID. Provide the ID number for the Release as reported on the DDCA-approved CSDR Plan.

3.3.2.1.2. Release Name. Provide the name (or number) of the release.

3.3.2.1.3. WBS Element Code. Enter the WBS Element Code that corresponds to the Release and as reported in the DDCA-approved CSDR Plan.

3.3.2.1.4. WBS Element Name. Enter the WBS Element Name that corresponds to the Release and as specified in the CSDR Plan.\

3.3.2.1.5. Release Type. Describe this release as either:

- Regular (changes to the existing baseline software that enhance and correct problems with the software that proceed through the maintenance process described above)

- Patch/Emergency (change to the existing baseline software that provides a work-around repair and proceeds through an abbreviated maintenance process as part of a quick response)
- IAVA (changes to the existing baseline software that are a result of Information Assurance and Vulnerability Alerts)
    - If approved by the CWIPT and included on the DDCA-approved CSDR Plan, frequent IAVAs may be reported as a single release. In this case, check the box for a cumulative IAVA release. If the release represents a cumulative total, check if the IAVA releases are completed weekly, monthly or quarterly.
- Other (provide description)

3.3.2.2. Outsourced Maintenance Organizations.

3.3.2.2.1. Name. List the names of the companies or organizations that took part in the maintenance of the software product(s) contained within the system if different than the reporting organization. If outsourced maintenance organizations are reported, an explanation must be placed in the Outsourced Maintenance Organizations Comments.

3.3.2.2.2. Location. List the corresponding locations of the companies or organizations that took part in the maintenance of the software product(s) if different than the reporting organization.

3.3.2.2.3. Primary or Secondary Maintainer. Indicate whether each company or organization was the primary or a secondary maintainer of the software product(s). There should always be one primary maintainer and is defined as the one performing the largest portion of the software maintenance. There may be multiple secondary maintainers who perform a smaller portion of the software maintenance.

3.3.2.2.4. Outsourced Maintenance Organizations Comments. If outsourced maintenance organizations are reported, provide comments about the reporting organization's relationship to the maintenance organizations. The maintenance organization should describe their level of responsibility and scope of work on the system maintained.

3.3.2.3. Schedule Reporting

3.3.2.3.1. Release Start Dates. Provide planned and actual start dates for each release that either starts or ends in this reporting period. The date format is year, month, and day (YYYYMMDD).

3.3.2.3.2. Release End Dates. Provide the actual end dates for each release in this reporting period. If a release is still active (has not finished), enter the planned end date. The date format is year, month, and day (YYYYMMDD).

3.3.2.3.3. Schedule Comments. Provide any comments about schedule reporting.

3.3.2.4. Product and Maintenance Description

3.3.2.4.1. Functional Description. Provide a description that characterizes the software function. This addresses the functionality that needs to be changed - what is it, what does it do, how does it interface with other system elements (both internal and external to the effort)?

3.3.2.4.2. Software Maintenance Characterization. Describe the maintenance performed in this release. Include a description of any unusual circumstances that impact the size, effort or duration of the performed maintenance, e.g., maintenance program funding, staffing, major changes, unplanned technical refresh, etc. Describe the software maintenance work undertaken or to be undertaken on that system element.

3.3.2.4.3. Software Maintenance Process. Describe the different phases in the maintenance process for this system as described in ISO/IEC 14764:2006: software requirements review, change board authorization, implementation, configuration item test, software system integration, acceptance testing, and deployment.

3.3.2.4.4. Operating Environment(s). Identify the operating environment(s) in which the maintained software system operates.

- Surface Fixed – software is embedded in a system at a fixed site
- Surface Mobile – software is embedded in a system that is either moved and setup independently or in a platform.
- Surface Portable – software is embedded in a handheld or portable device
- Surface Vehicle – software is embedded as part of a moving vehicle
- Air Vehicle – software is embedded as part of an aircraft
- Sea Systems – software is embedded as part of a surface or underwater boat/ship
- Ordnance Systems – software is embedded as part of a rocket or other ordnance systems
- Missile Systems – software is embedded as part of a missile
- Space Systems – software is embedded as part of a spacecraft
- Other – provide explanation if other

3.3.2.4.5. Manned vs. Unmanned. For the operating environment above, indicate if it is a manned or unmanned environment.

3.3.2.4.6. Application Domain.  Identify at least one application domain maintained or to be maintained from those listed below. Definitions and examples may be found on the DCARC website: http://cade.osd.mil/.

3.3.2.4.6.1. Application Domain and Percent of Overall Product Size.  For each of the applicable application domains, enter the overall approximate percentage (up to 100%) of the product that is associated with that domain.  At least one application domain must be identified.

– Microcode and Firmware
– Signal Processing
– Vehicle Payload
– Vehicle Control
– Other Real-Time Embedded
– Command and Control
– Communication
– System Software
– Process Control
– Scientific and Simulation
– Test, Measurement, and Diagnostic Equipment
– Training
– Software Tools
– Mission Planning
– Custom AIS Software
– Enterprise Service System
– Enterprise Information System

3.3.2.4.6.2. Application Domain Comments. Provide any additional information that will aid the analyst in interpreting the application domain data.

3.3.2.4.7. Software License Information.  Provide cost and non-cost (i.e. open source licenses) information on all software licenses for the software being maintained. Include the cost of End User License Agreements (EULA) and Support/Maintenance agreements.

3.3.2.4.7.1. Software License Name. List the names of the applications or products that constitute each part of the final delivered product, whether they are Commercial-Off-The-Shelf (COTS), Government-Off-The-Shelf (GOTS), or open-source products. If a proprietary application or product that is not generally commercially available will be included, identify it here and include any necessary explanation in the Software License Comments.

3.3.2.4.7.2. Quantity. The number of licenses for each license name.

3.3.2.4.7.3. Coverage. Indicate whether the license is for one "seat" or computer, one site, enterprise-wide, or unlimited.

3.3.2.4.7.4. Total Cost. The total cost for all licenses under this license name. Specify costs using a dollar format.

3.3.2.4.7.5. Type. Whether it is EULA or Support.

3.3.2.4.7.6. Duration. The duration of the license as either annual renewal (renewed every year) or indefinite (free or one-time cost for unlimited license time).

3.3.2.4.7.7. Integration Effort. The amount of effort in hours required to integrate all COTS, GOTS, or open-source application identified Software License Name.

3.3.2.4.7.8. Software License Comments. Provide any additional information that will aid the analyst in interpreting licensing data.

3.3.2.4.8. Product and Maintenance Description Comments. Provide any additional comments about the product and maintenance description that are necessary to understand the data provided.

3.3.2.5. Labor Hours Reporting

3.3.2.5.1. Hours per Staff-Month. Provide the number of hours per staff-month used to convert between labor hours and full-time equivalent (FTE) staff.

3.3.2.5.2. Hours per Staff-Month Type. Indicate whether the reported hours per staff-month reflect an accounting standard or a computation.

- If staff-month, provide the selection.
- If a computation, provide the selection and the total labor hours and staff used, and indicate which efforts are included in those totals.

3.3.2.5.3. Minimum Labor Hours Required. Specify the minimum number of labor hours required to sustain the capability to perform basic maintenance functions, e.g., maintain maintenance assets, update maintenance software, test beds, and support equipment. These are the hours required to maintain a warm industrial base including keeping the facilities open and retaining essential personnel available for software maintenance, when full funding is not available.

3.3.2.5.4. Unique Job Skills. Describe, if any, special job skills that are required for this work, e.g., ERP system architect, leading specialist in a language or domain that is either new or becoming obsolete.

3.3.2.6. Product Size Reporting

3.3.2.6.1. Software Requirements

3.3.2.6.1.1 Number of Software Requirements. Provide the total number of delivered software requirements as of the beginning of each software release, plus the number of new software requirements added with this maintenance release.

3.3.2.6.1.2 Number of Software Requirements Implemented. Provide the total number of software requirements changed in this release.

3.3.2.6.1.3 Software Requirements Comments. Provide any additional information that will aid the analyst in interpreting requirements data. Indicate if the counts include Security, Safety, and Privacy requirements.

3.3.2.6.2. External Interface Requirements Counts

3.3.2.6.2.1. Number of External Interface Counts at Release Start. Provide the total number of software external interfaces as of the beginning of each software release.

3.3.2.6.2.2. Number of External Interface Counts Implemented. Provide the total number of software external interfaces that were changed and added in this release.

3.3.2.6.2.3. External Interface Requirements Comments. Provide any additional information that will aid the analyst in interpreting requirements data. Indicate if the counts include Security, Safety, and Privacy requirements.

3.3.2.6.3. SLOC-Based Software Size. Format 2 shall capture the actual delivered size of the product maintained by the prime and subcontractor and also by software language, not including any code that might be needed to assist maintenance but that will not be delivered (such as temporary stubs, test scaffoldings, or debug statements). Additionally, the code shall be partitioned (exhaustive with no overlaps) into appropriate source code categories: new (human-generated), new auto-generated, reused with modification, reused without modification, government-furnished, carry-over, and deleted. Open source code will be captured in reused code. The latest version of the UCC-G tool shall be used to obtain a set of standardized code counts that reflect logical size, as opposed to physical, non-commented, etc. These results shall be included in the report unless other sizing metrics are used. SLOC counts shall be reported using single units (as opposed to thousands of units). If UCC is not available for the given language, specify how the code counts were derived (i.e., with UCC-G for a similar language, or using the Alternative Code Counter). The intent is to capture not just Delivered SLOC (DSLOC) but also information that will enable the calculation of Equivalent SLOC (ESLOC).

3.3.2.6.3.1. Software Language. Format 2 shall indicate the programing language of the code that is being sized. Provide SLOC by software language in descending order of percentage of overall size. Specify the top two languages used (L1, L2 on the form) and combine the rest in "Other." These can be a compiled language, such as FORTRAN, Ada, or C; an interpreted language, such as BASIC; or a graphical or model-based language, such as Rhapsody/UML or Simulink. If UCC-G is not available for the given language, specify how the code counts were derived.

3.3.2.6.3.2. New Code. New code shall be partitioned into human-generated which is source code that is created manually by a programmer and auto-generated code.

3.3.2.6.3.3. Reused Code With Modification. When code from external sources (government-furnished or otherwise) is included that is or was reused with modification, provide an assessment of the amount of redesign, recode, and retest required to implement the modified or reused code. Modified code is defined as pre-developed code that can be incorporated in the software component with a significant amount of effort, but less effort than required for newly developed code. In addition to the size of Modified Code, provide the Adaptation Adjustment Factor (AAF) used for each component's Modification calculation if available. AAF is defined as the top-level percentage of effort of adapting the software to the new system relative to a line of new code (e.g., a Modified line of code might require 50% of the effort of its equivalent New line of code when considering the design, code, and testing work that has already been performed). Any deviation from a standard AAF percentage for Modified code should be explained.

3.3.2.6.3.4. Reused Code Without Modification. Code reused without modification is code that has no design or code modifications. However, there may be an amount of retest required. In addition to the size of Reused Code, Format 2 will provide the AAF used for each component's Reuse calculation if available. Any deviation from a standard AAF percentage for Reused code (if provided) shall be explained.

3.3.2.6.3.5. Carryover Code. Code developed in previous releases that is carried forward into the current release and delivered (also called Internal Reuse). Do not count the same code as new in more than one SRDR for Maintenance report. An SRDR for Maintenance shall distinguish between code developed in previous releases that is carried forward in the current release and code added as part of the effort on the current release. Table 1 provides one possible example of reporting code from previous releases that is carried forward in the final report.

Table 1: Example of Reporting Carryover Code from Previous Releases

| | | **Release 1 Finish** | **Release 2 Finish** | **Release 3 Finish** | **Contract Finish** |
|---|---|---|---|---|---|
| | | | | | |
| New Code | Human Generated | 1,000 | 0 | 2,500 | 3,500 |
| | Auto Generated | 0 | 500 | 2,500 | 3,000 |
| External Reused | With Modification | 5,000 | 15,000 | 500 | 20,500 |
| | Without Modification | 3,000 | 0 | 2,000 | 5,000 |
| Carryover Code from Previous Build | With Modification | 0 | 0 | 12,250 | N/A |
| | Without Modification | 0 | 9,800 | 13,250 | N/A |
| Government Furnished Code | With Modification | 800 | 200 | 0 | 1,000 |
| | Without Modification | 0 | 0 | 0 | 0 |
| Total Delivered Code | | 9,800 | 25,500 | 34,000 | 34,000 |
| Deleted Code | | 0 | 0 | 100 | 100 |

3.3.2.6.3.6. Auto-generated Code. If the developed software contains auto-generated source code, Format 2 shall include an auto-generated code sizing partition as part of the set of development categories.

3.3.2.6.3.7. Government Furnished Code. Indicate the amount of Government Furnished Code delivered to the contractor. If the Government did not provide the contractor any code, enter 0.

3.3.2.6.3.8. Deleted Code. Format 2 shall include the amount of code that was created and subsequently deleted from the final delivered code and hence not included in the reported total.

3.3.2.6.4. Agile Measures. If an Agile approach is used for maintenance of software, enter data here to describe the technical scenario.

The following definitions apply to the Agile Measures section of the Maintenance Report:

- Each measure is for a single Release. For Agile developments, a "Release" is a defined compilation of software features delivered within a specific time frame. Specific software end items are explained in paragraph 3.3.2.6.4.1 below as "Features" within a Release. The software maintenance reporting process shown in Figure 2 for sequential Releases also applies to Agile Releases within a contract
- The Release is divided into "Sprint" time boxes, which are used to schedule specific daily workload to complete from a Backlog of defined software development tasks. Each Sprint is a recurring, non-overlapping cadence for development, usually 1 - 6 weeks in duration.
- Work within a Sprint is assigned to Scrum teams, who meet daily to plan activities, assign Backlog, and assess progress and impediments.

Include the vendor's specific "Days per Sprint" or other productivity metrics that apply to the Release.

3.3.2.6.4.1 Release Map. The workload to assign at the Sprint level represents high level requirements (Epics/Capabilities) that are then decomposed into well-defined Feature Stories or cases. Each Feature Story is then decomposed into smaller Stories that define the workload to assign at the Sprint level (Backlog). Output from each Sprint (completed software) is integrated at the Feature level into working software that the user can evaluate. One or more Features roll-up to comprise completed Epics/Capabilities. One or more Epics/Capabilities comprise the Release.

Use the "Release Map" to identify the relationship of Features to Epics/Capabilities for the Release defined by each Maintenance Activity (one or more Epics/Capabilities for the Release; one or more Features per Epic/Capability). Additional lines may need to be inserted to accommodate the full Feature/Epic map. The Feature or Epic/Capability "Identifier" is assigned by the developer using their own conventions for naming or tracking workload. If terms other than "Feature" or "Epic" are used (for example: Acquisition Requirements Documents – or ARDs; Software Requirements Specifications – or SRSs), then provide an explanation in the comments.

3.3.2.6.4.2 Planned and Achieved Development (by Feature per Epic/Capability). Development tasks in the Backlog are sized using a sizing convention, such as number of Feature Stories, Story Points, Function Points or SLOC. Planned and Actual Stories describe the relative complexity of the Feature. Story Points represent a relative scale that assigns a size to each task in the Backlog, so that the tasks can be "fit" into Sprints. Planned Story Points represent an estimate for all of the Backlog tasks needed to complete each Feature. Actual Story Points represent the known size of the Backlog tasks actually completed per Feature. If known, Total Hours required to complete the Backlog tasks for each Feature should also be included. The example row shows that each row includes a measure of the Planned and Actual Story Points along with Hours by Feature (one Feature per row).

Note: The Feature hours included here will be used to help correlate specific Story Point sizing to the effort required per Feature, rather than correlating the Story Point sizing with the overall effort for the Release.

3.3.2.6.4.3 Summary Total for This Release. This table summarizes the entries for all Release Map rows, and is not aligned with any specific Release Map row. The total number of Features, Epics/Capabilities, Story Points and Feature Hours recorded as "Planned and Achieved Development" are summed here, along with the total number of Sprint time periods scheduled within the Release. Planned values apply to the Initial and Interim Reports. Actual values apply to Interim and Final reports.

3.3.2.6.5. Non-SLOC-Based Software Size. SLOC-Based Sizing is required unless reporting alternative sizing measures only is agreed to by the CWIPT and documented in the OSD DDCA-approved CSDR Plan. The exception is RICE-FW, which is the preferred sizing measure for enterprise resource planning (ERP) systems.

3.3.2.6.5.1 FP Measure. If Function Points (FP) are used for sizing, then the IFPUG standard (reference 2.2.9) shall be used. They shall also include actual SLOC counts per the guidelines above. The SRDR shall indicate the count type and include counts of both Data Functions – Internal Logical Files (ILF) and External Interface Files (EIF) – and Transactional Functions – External Inquiries (EQ), External Inputs (EI), and External Outputs (EO) – all broken out by Low, Average, and High. The Total Unadjusted FP count shall be multiplied by a Value Adjustment Factor (VAF) to produce and Adjusted FP count.

3.3.2.6.5.2 RICE-FW Measure. For enterprise resource planning (ERP) implementation, Reports, Interfaces, Conversions, Extensions, Forms, and Workflow (RICE-FW) are common size measures. If RICE-FW are used, counts shall be provided, broken out by Low, Medium, and High Complexity. Any standards or guidelines applied in developing counts shall be fully explained.

3.3.2.6.5.3 Other Measures. If other size measures are used, such as use cases, story points, feature points, (predictive) object points, rung ladder, etc., they shall be named, and the associated counts provided. Any standards or guidelines applied in developing counts shall be fully explained. If the contractor is using Agile development, provide a complete set of sizing metrics (story points, user stories, features, epics, etc.).

3.3.2.6.5.4 Product Size Reporting Comments. Provide any additional comments about product size reporting.

3.3.2.6.6 Software Change Counts. Report the number of software changes as defined in 3.3.1.12.

3.3.2.6.6.1 Number of Software Changes Implemented. For each release and priority, provide a count of the software changes implemented. Table 2, may be used for the definitions of priority.

3.3.2.6.6.2 Number of Software Changes Deferred. For each release and priority, provide a count of the software changes that were deferred to later releases. Table 2, may be used for the definitions of priority.

3.3.2.6.6.3 Software Change Volatility. For each release and priority, provide a count of the number of unplanned software changes that were added/changed/deleted from the release after the release began.

3.3.2.6.6.4 Number of software changes associated with IAVA. For each release, provide a count of the number of IAVAs implemented, deferred, and closed out/required no action.

Table 2: Software Change Priority

| Priority | Criteria |
|---|---|
| 1 | a) Prevent the accomplishment of an operational or mission essential capability |
| | b) Jeopardize safety, security, or other requirement designated "critical" |
| 2 | a) Adversely affect the accomplishment of an operational or mission essential capability and no workaround solution is known |
| | b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, and no work-around solution is known |
| 3 | a) Adversely affect the accomplishment of an operational or mission essential capability but a workaround solution is known |
| | b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, but a work-around solution is known |
| 4 | a) Result in user or operator inconvenience or annoyance but does not affect a required operational or mission essential capability |
| | b) Result in inconvenience or annoyance for development or support personnel, but does not prevent the accomplishment of those responsibilities |
| 5 | Have any other effect |

## 3.4. Software Resources Data Report – Part 2, Software Maintenance Effort Data

### 3.4.1. Resource Reporting

3.4.1.1. WBS Element Code. Enter the WBS Element Code that corresponds to each Release and as it is reported on the DDCA-approved CSDR Plan.

3.4.1.2. WBS Element Name. Enter the WBS Element Name that corresponds to each Release and as it is reported on the DDCA-approved CSDR Plan.

3.4.1.3. Cumulative Software Change Hours by Release or Sub-Release (i.e. reporting at lower level of detail under release). Hours associated with defining, allocating, generating, integrating, and testing software changes for an operational software product or system. Includes effort associated with changing code to address defects, enhancements and IAVAs, as well as effort associated with integration baseline and test, Airworthiness/Safety/Networthiness certification, and Independent Verification and Validation for the software. Specify the actuals to date (ATD) of prime and subcontractor labor hours spent working software changes for each planned and completed release or sub-release (a release is a software product that passed acceptance testing).This data shall be provided for each organization working on the maintenance of this system.  If subcontractor hours are not available, the total cost of subcontracted labor can be provided.  If included in the DDCA-approved CSDR plan, reporting of software change hours at the sub-release level is required.

3.4.1.4. Annual Prime and Subcontractor Labor Hours by Software Maintenance Activity.  Specify the actuals to date of labor hours worked for the activities listed below by prime contractor and subcontractor. This data shall be provided for each organization working on the maintenance of this system.  This data shall be provided annually and should only include the software specific effort for each activity.  If subcontractor hours are not available, subcontractor dollars in thousands may be included

3.4.1.4.1. Sustaining/Systems Engineering Hours. Hours associated with software specific sustaining engineering activities such as studies/investigations for SW specific issues.  Sustaining Engineering does not include any effort or cost for either maintenance (corrections) or capability enhancements: these are included in the release data. User support should not include Field Software Engineering, nor data in other sub-categories

3.4.1.4.2.  Cybersecurity Hours. Hours associated with activities such as software Cybersecurity and Information Assurance Vulnerability Management (IAVM). Cybersecurity, formerly Information Assurance (IA), and the Risk Management Framework (RMF) for DoD Information Technology, formerly DoD Information Assurance Certification and Accreditation (C&A) Process (DIACAP), are processes that verify the software system against externally defined domain performance criteria.

3.4.1.4.3. Project Management Hours. Hours associated with system specific software maintenance project and technical management, and hours associated with system specific license management to include procurement and renewal of software licenses for operational software. The license management activities include managing licenses for the maintenance facility as well as deployed systems.

3.4.1.4.4. Help Desk Hours.  Hours associated with providing software specific help desk support for end users. For MAIS the following applies: This includes Levels I through III. This support will include user account management. The Help Desk/Operations Support Team (OST) will provide Tier I level support for problems related to systems administration and monitoring, event management, and database administration including restart, recovery, backups, and restorations. The help desk support staff is the initial focal point for answering questions and providing status information for the hosted site. The typical support hours are 24 X 7 X 365.

3.4.1.4.5. System Facilities Hours. Hours associated with establishing and operating software maintenance related development including development assets / workstations, integration, and test facilities, and support equipment and tools. Only report hours that are direct charge to the Government.

3.4.1.4.6. Field Software Engineering (FSE) Hours. Hours associated with the on-site support of a deployed software product or system in its operational environment. FSE duties include on- site technical assistance, problem troubleshooting, software installation, operational assistance, on-site training

3.4.1.4.7. Operational Management Hours (for Government-Performed Maintenance Organizations). The system specific charge for hours associated with establishing and operating the organizational infrastructure required to implement common software maintenance business and technical processes across multiple software systems. These activities include operations, organization management, personnel management, financial management, information management, process management and change management. This category is for the use of government organizations providing the SRDR. If a contractor has reportable hours in this category, please specify in the comments which activities are covered by these hours.

3.4.1.4.8. Follow-on User Training Hours. Hours associated with follow-on user training. This includes new release training/periodic training events driven by a software change.

3.4.1.5. Effort Comments. Provide any comments about effort reporting. Also, provide a list of the C&A/Cybersecurity related activities that the software maintenance program is subject to and provide the frequency of each activity.

**Format 3, DD Form 3026-3, "Enterprise Resource Planning (ERP) Software Development Report"**

3.3 ERP Software Development Report – Part 1 Software Development Technical Data. Release Level data must be reported for each release included in the SRDR, as specified by the OSD DDCA approved CSDR Plan.

3.3.1 Subsection A - Report Context. Format 3 (DD Form 3026-3) includes the following data at the Release Level.

3.3.1.1 Release Name, the name used to refer to the software being developed, including any applicable version, build, or other identifier, as specified in the CSDR Plan.

3.3.1.2 Release ID, the associated Work Breakdown Structure Element from the OSD DCCA approved CSDR Plan.

3.3.1.3 Certified CMM Level (or equivalent). This is the SEI Capability Maturity Model (CMM) number of the level (1 through 5) at which the primary development organization has been formally appraised, and which they will implement for the program. If no formal appraisal has been conducted, leave the item blank. If a single submission is used to represent the work of multiple organizations, enter the level of the organization that will be expending the most amount of effort on the development project (not necessarily the prime contractor) and note this in the in "Comments on Subsection A" (ref para 3.3.1.8).

3.3.1.4 Certification Date. This is the date when the formal assessment associated with the indicated level was conducted.

3.3.1.5 Lead Evaluator of CMM Certification. This is the name of the person that led the formal SEI CMM assessment and determined the maturity level indicated.

3.3.1.6 Affiliation to Development Organization. This is the affiliation of the Lead Evaluator in the previous item.

3.3.1.7 Precedents. List up to ten analogous software development projects that have been completed by the same performing organization or development team.

3.3.1.8 Comments on Subsection A responses. To be used as needed.

3.3.2 Subsection B – Product and Team Description. Format 3 (DD Form3026-3) includes the following data at the Release Level.

3.3.2.1 Project Functional Description. Using one or more Primary Application Domain names from the list below (para 3.3.2.2), describe the primary application type or types being developed for this Release. If none of the examples shown in paragraph 3.3.2.2 are appropriate, enter a phrase to describe the application domain and define the domain in the "Primary Application Domain Comments" section as described by paragraph 3.3.2.3.

3.3.2.2 Primary Application Domains for ERP Programs. Select the domains(s) that most closely reflect(s) the domain(s) developed or to be developed as part of the Release from those listed below. Definitions and examples may be found on the CADE website: http://cade.osd.mil/.

– Microcode and Firmware
– Signal Processing
– Vehicle Payload
– Vehicle Control
– Other Real-Time Embedded
– Command and Control
– Communication
– System Software
– Process Control
– Scientific and Simulation
– Test, Measurement, and Diagnostic Equipment

– Training
– Software Tools
– Mission Planning
– Custom AIS Software
– Enterprise Service System

3.3.2.3 Primary Application Domain Comments. Provide any additional information that will aid the analyst in interpreting the application domain data.

3.3.2.4 Comments on Subsection B responses. To be used as needed.

3.3.3 Subsection C – Project Requirements. Enter the number of requirements for the system as defined by formal system specification documents. Format 3 (DD Form3026-3) includes the following data at the Release Level.

3.3.3.1 Business Modules. Number of business modules required by the customer. This section documents a top-level business process that must be supported by the system, such as Bill and Collect Revenue. It captures information regarding the benefits to the client realized through this process, a high-level process schematic, and key performance indicators that can be used to measure the process.

3.3.3.2 ERP Modules. Number of standard business modules required by the customer. These standard modules are those captured in the ERP product (SAP/Oracle/PeopleSoft Suite) and implemented as part of the ERP project. Example modules might be:

- Finance (FI)
- Controlling (CO)
- Funds Management (FM)
- Materials Management (MM)
- Real Estate Management (RE)
- Sales and Distribution (SD)
- Business Warehouse (BW)
- Human Resources (HR)
- Project Systems (PS)

3.3.3.3 Business Process. A Business Process is typically defined as a set of logically related tasks performed to achieve a defined business outcome. A process is the way of organizing a company's activities and resources that establish cross-functional coordination throughout the company with the purpose of creating value for customers, employees, shareholders and all other stakeholders. This field contains the number of processes required by the customer.

3.3.3.4 Sub-Business Process. A Sub-Process is simply a lower-level process of a higher-level process. Business processes occur at multiple levels within organizations and all high-level processes (e.g." Fulfill Demand") are likely to include lower level of processes, or sub-processes, within the same organization. These fields will have the number of sub-processes required by the customer. This section describes the lowest level of detail that is captured within a process. An activity typically captures a process step that is performed by a single person and that contributes to the completion of a sub-process. This form documents information such as who performs the activity, when and where it is performed, what is involved in completing it, and how many times it is performed in a particular time period. For example, an activity in the Process Payments sub-process, which is part of the Create Sales Order process, might be Create Customer Master.

3.3.3.5 Functional Requirements. This field identifies the number of functional requirements needed by the customer. Functional requirements are defined as a detailed breakdown of business solution outcomes in terms of the intended functional behaviors of the application.

3.3.3.6 Non-Functional Requirements. Number of system technical requirements needed by the customer (hosting, performance, response time, user connectivity, etc.). Cybersecurity requirements should be specifically addressed in this section.

3.3.3.7 Legacy System Interfaces. Number of external legacy systems (i.e., not under project control) expected to interface with the delivered ERP system. This number accounts for both inbound and outbound interfaces.

3.3.3.8 Legacy System Phase Out. Legacy System Phase Out would generally refer to legacy systems which will be gradually discontinued. To distinguish between "Legacy System Phase Out" and "Legacy System Migration," treat all legacy systems subsumed by the ERP application as "Legacy System Migration" requirements, The "Legacy System Phase Out" requirements are the number of legacy systems being replaced that are not being subsumed.

3.3.3.9 Legacy System Migration. Legacy System Migration generally refers to the migration of a legacy or existing application to a new operating environment. It could also mean to replace a legacy system, moving the data from the legacy application to the new application while preserving data integrity. The migration process helps consolidate legacy systems eliminating duplication by moving data and functionality from various systems to a smaller number of systems.

3.3.3.10 Comments on Subsection C. To be used as needed.

3.3.4 Subsection D.1 – Product Size Reporting. The following information provides the proposed software resource data for ERP Product Size Reporting, sorted by the type of objects being developed and their associated complexity. The numbers used to complete the information are based on the developer's tracking data for configuration objects (included with the baseline ERP product), and newly developed objects identified by the users during gap analysis.

Sizing data must be included with each ERP Software Development SRDR. This section describes the preferred approach for sizing: counting development objects by type and complexity. Section D.2 (paragraph 3.3.5) provides instructions for alternate sizing measures, such as Function Points, Software Lines of Code (SLOC) or Agile development measures that can be provided along with the object counts, or in place of the object counts if objects have not yet been defined.

The information for Product Size Reporting by object type and complexity is divided into 3 categories, depending upon the maturity of the program:

- Functionally Designed: objects that have been designed and are ready to be developed.
- Technically Built: objects that have been, or are being coded, but have not been tested or implemented.
- Tested/Implemented: completed code or objects.

This division may lead to different counts as objects transition between design, build and test categories. For example: counts are likely to all be Estimates at Complete (EAC) for an initial ERP Software Development SRDR. Interim reports will include a mix of Actual counts and EACs for designed, built and tested objects, with the trend being more towards Actuals for designed and built, and more towards EACs for tested and implemented. The final report will likely be all Actual counts for tested/ implemented objects. Based on the amount of re-work and re-design required, the final Actual count of tested/implemented may not equal the original EAC count in design.

Format 3 (DD Form 3026-3) includes the following data at the Release Level.

3.3.4.1 Configuration Objects included with the commercial ERP product (out-of-the-box objects). Includes counts by complexity as defined below:

3.3.4.1.1 Simple Complexity. An already existing functional module or enterprise element is available within the ERP system, and can be implemented with only minor modifications/set-up for the required tables, panels, screens/reports, or business rules.

3.3.4.1.2 Medium Complexity. An already existing functional module or enterprise element is available within the ERP system, and can be implemented with moderate modifications/set-up for the required tables, panels, screens/reports, or business rules.

3.3.4.1.3 High Complexity. An already existing functional module or enterprise element is available within the ERP system, and can be implemented with significant modifications/set-up for the required tables, panels, screens/reports, or business rules.

3.3.4.2 Report Objects. Includes counts by complexity as defined below, to include Within Reports and Business Warehouse Reports:

3.3.4.2.1 Simple Complexity. Less than 5 standard application tables. As many as 1 external file. Straight forward data retrieval. Logic: Basic, single-level report. Little aggregation or sorting. No use of external subroutines. One version suits all requirements.

3.3.4.2.2 Medium Complexity. 5 to 8 standard application tables. As many as 3 external files. Some cross-checking. Logic: Multiple-level drill down capability. Moderate calculation, sorting. Some customization (ex: company-wide). Field translations required.

3.3.4.2.3 High Complexity. 9 or more standard application tables. 3 or more external files. Data from multiple functional areas. Logic: Use of sub-screens, pop-ups, etc. Significant authorization checking. Complicated data retrieval. Some customization (ex: plant-wide). Field translations required.

3.3.4.3 Interfaces (Inbound and Outbound). Includes counts by complexity as defined below:

3.3.4.3.1 Simple Complexity. Inbound: 1 external file, with fewer than 3 different record types. Logic: Up to 2 transactions in upload. No retry logic (errors to report to log). No reconciliation. Batch. Outbound: 1 external file, Fewer than 3 different record types. Logic: No translations of codes. Batch. Data read from less than 5 tables.

3.3.4.3.2 Medium Complexity. Inbound: 2 to 4 external files, 3 to 5 more different record types. Logic: 2 to 5 transactions in upload. Moderate coding (some validation). Some retry logic and error processing. Minimal reconciliation. Outbound: 2 to 4 external files, 3 to 5 difficult record types. Logic: Batch. Moderate translations of codes. Data read from 5 to 9 tables.

3.3.4.3.3 High Complexity. Inbound: 5 or more external files, 6 or more different record types. Logic: More than 6 transactions in upload. Complex coding (complex validation). Significant retry logic and error handling. Heavy reconciliation. Outbound: 5 or more external files, 6 or more different record types. Logic: heavy translations. Near real-time/Real-time. Triggering via user exits. Data read from 9 or more tables.

3.3.4.4 Conversions. Includes counts by complexity as defined below:

3.3.4.4.1 Simple Complexity. Data is pre-extracted & formatted. Up to 2 input files/record types. Logic: Use of standard application load programs. Loading basic master data. Single load program. Assume zero, until identified.

3.3.4.4.2 Medium Complexity. Some reformatting of data is required. 3 or 4 input files/record types. Logic: Baseline coding (some validation) Single load program.

3.3.4.4.3 High Complexity. – Significant reformatting is required. 5 or more input files/record types. Logic: Moderate coding (moderate validation). Loading lowest level master data. Single load program.

3.3.4.5 Extensions. Includes counts by complexity as defined below:

3.3.4.5.1 Simple Complexity. Manipulation of 1 standard table. Logic: Does not require user exits. Initial & detail screen. Menu extensions. No database updates. One version suits all requirements.

3.3.4.5.2 Medium Complexity. Manipulation of 2 standard tables. Logic: User exits to capture data only. Initial & detail screen. Function exit. Update database. Some customization (ex. company-wide).

3.3.4.5.3 High Complexity. Manipulation of 2 or more standard tables. Logic: User exits with substitution logic. Step-loop to maintain header & detail. Initial screen with sub-screens. Dynapro extension. Some customization (ex. plant-wide).

3.3.4.6 Security Patches. Includes counts by complexity as defined below:

3.3.4.6.1 Simple Complexity. Number of security patches requiring zero to minor tailoring effort by the system integrator.

3.3.4.6.2 Medium Complexity. Number of security patches requiring moderate tailoring effort by the system integrator.

3.3.4.6.3 High Complexity. Number of security patches requiring significant tailoring effort by the system integrator.

3.3.4.7 Bolt-Ons. This is the number of COTS applications requiring tailoring effort by the system integrator other than the ERP application. Includes counts by complexity as defined below:

3.3.4.7.1 Simple Complexity. Number of COTS applications requiring zero to minor tailoring effort by the system integrator.

3.3.4.7.2 Medium Complexity. Number of COTS applications requiring moderate tailoring effort by the system integrator.

3.3.4.7.3 High Complexity. Number of COTS applications requiring significant tailoring effort by the system integrator.

3.3.4.8 Forms. Includes counts by complexity as defined below:

3.3.4.8.1 Simple Complexity. Standard forms (i.e. invoice, quotation, etc). No custom database access is required. Logic: Minor modifications to the SAP/Oracle/PeopleSoft standard forms. Printing of forms is configured into SAP/Oracle/PeopleSoft, no custom programming required.

3.3.4.8.2 Medium Complexity. Non-standard forms (i.e. new invoice form). Accesses one or more logical databases. Logic: Creating a form from scratch, and printing it on plain printed paper. No need to create cosmetics such as grids or boxes. Printing of forms may require custom work.

3.3.4.8.3 High Complexity. Non-standard forms (i.e. new invoice form). Accesses one or more logical databases. Logic: Creating a form from scratch, but printing it on plain paper. Will need to create cosmetics such as grids or boxes. Printing forms may require custom work.

3.3.4.9 Workflows. Includes counts by complexity as defined below:

3.3.4.9.1 Simple Complexity. Standard Workflow. No customization is required. May have minor modification to standard workflow.

3.3.4.9.2 Medium Complexity. Non-standard workflow (i.e. new workflow). Standard custom work, moderate modification to standard workflows.

3.3.4.9.3 High Complexity. – Non-standard workflow. Creating workflow from scratch. Will need to significant customization.

3.3.4.10 Other Program Defined Objects. If other objects are used, such as Info Cube, Data Store Object, Utility, etc., they shall be named, and the associated counts provided. Any standards or guidelines applied in developing counts shall be fully explained.

3.3.4.11 Comments on Product Sizing. To be used as needed.

3.3.5 Subsection D.2 – Alternative Product Size Reporting. Provide SLOC or Non-SLOC based measures of software and object sizing for ERP. Users must provide sizing data. The preferred approach is to provide object counts as described in paragraph 3.3.4. The alternate approaches described below can be provided in place of object counts, when objects have not yet been defined, or in addition to object counts. Format 3 (DD Form3026-3) includes the following data at the Release Level:

3.3.5.1 Function Point Measure, based on the International Function Point User Group (IFPUG) standard for determining relative size and counts. The Initial Report shall provide the FP count. Interim and Final Reports shall include updated FP counts. The “ISO/IEC 20926, Software and systems engineering – Software measurement – IFPUG functional size measurement method 2009” (ref para 2.2.9) provides complete details on the terms used below.

3.3.5.1.1 Count Type. The ERP Software Development SRDR shall indicate the count type: Enhancement Project FP; Application Project FP; or Development Project FP.

3.3.5.1.2 Function Types. Include counts for:

- Data Functions, stated as Internal Logical Files and External Logical Files
- Transactional Functions, stated as External Inquiries, External Inputs, and External Outputs
- All counts are further divided into Low, Medium and High values as defined by the IFPUG standard.

3.3.5.1.3 The Total Unadjusted FP count (sum of the Data Functions and Transactional Functions) may be multiplied by a Value Adjustment Factor (VAF) to produce an Adjusted FP count.

3.3.5.2 Software Lines of Code (SLOC). Logical SLOC counts for the ERP Software Development SRDR shall capture the estimated or actual delivered size of the product developed, not including any code that might be needed to assist development but that will not be delivered (such as temporary stubs, test scaffoldings, or debug statements).
Use for measuring software size when numbers of requirements or object counts/complexity are not available.

3.3.5.2.1 Code Counter Version. For development, use the Government-approved version of the University of Southern California (USC) Center for Systems and Software Engineering (CSSE) Unified Code Count (UCC) tool, or UCC-G, for the below required Source Lines of Code (SLOC) counts. The most recent version at the time of first submission shall be used. If subsequent versions of the code counter do not change original code counts by more than plus or minus one percent (1%), they are permitted for later submissions. If a contractor-specific code counter is also being used, provide name, version number, description, and results of any benchmarking studies (e.g., how counts compare to those generated by UCC-G).

3.3.5.2.2 Count by Code Category. For Interim ERP Software Development SRDRs, the reporting organization will count the code that has undergone verification and validation (V&V) to date, and report the estimate at complete for the whole software development effort. The intent is to capture not just Delivered SLOC (DSLOC) but also information that will enable the calculation of Equivalent SLOC (ESLOC). Categories include:

- New Code/Auto Generated Code. New code shall be partitioned into human-generated and auto-generated code. If the developed software contains auto-generated source code, the ERP Software Development SRDR shall include an auto generated code sizing partition as part of the set of development categories. Include the percentage of the new code that is provided as a COTS or GOTS product, or the percentage of code provided from an Open Source product.
- Modified/Adapted Code. When code from external sources (government-furnished or otherwise) is included that is or was reused with modification, the ERP Software Development SRDR shall provide an assessment of the amount of redesign, recode, and retest required to implement the modified or reused code, along with an assessment of the percentage of the modified code being reused.
  Modified code is defined as pre-developed code that can be incorporated in the software component with a significant amount of effort, but less effort than required for newly developed code.
  In addition to the size of Modified Code, the ERP Software Development SRDR shall either provide individual Design Modification (DM), Code Modification (CM), and Integration Modification (IM) percentages, or provide the Adaptation Adjustment Factor (AAF) used for each component's Modification calculation
  - DM is the fraction of design that must be modified in order to adapt the component to meet the functional and interface requirements of the new system.
  - CM is the fraction of code that must be modified in order to implement the new functions and interfaces.
  - IM is the fraction of integration effort required to integrate the adapted software into the overall product, and to test the resulting product compared to the normal amount of integration effort for software of comparable size.
  - AAF is defined as the top-level percentage of effort of adapting the software to the new system relative to a line of new code (e.g., a Modified line of code might require 50% of the effort of its equivalent New line of code when considering the design, code, and testing work that has already been performed).

  While individual DM, CM, and IM percentages are preferred, a single AAF percentage will be accepted when these are not available. Any deviation from a standard AAF percentage for Modified code shall be explained. The below items may be taken into consideration in assessing DM, CM, and IM factors.

  Redesign:

  - Required an architectural design change
  - Required a detailed design change
  - Required reverse engineering
  - Required modification to existing documentation
  - Required revalidation of the new design

Recode:

- Required code changes
- Required code reviews
- Required unit testing

Retest:

- Required test plans to be rewritten
- Required test procedures to be identified and written
- Required new test reports
- Required test drivers and simulators to be rewritten
- Required integration testing
- Required formal demonstration testing

- Deleted Code. The ERP Software Development SRDR shall include the amount of code that was created and subsequently deleted from the final delivered code and hence not reported in the final total. Include a count for the deletion of COTS, GOTS and Open Source code.

3.3.5.3 Agile Measures. If an Agile approach is used for development of software, enter data here to describe the technical scenario. If "Function Points" or "SLOC" are measured instead of "Agile" measures, then use paragraph 3.3.5.1 (Function Point) or 3.3.5.2 (SLOC) to capture sizing data.

The following definitions apply to the ERP Software Development SRDR:

- Each ERP Software Development SRDR reports Development data for a single Release. For Agile developments, a "Release" is a defined compilation of software features delivered within a specific time frame. Specific software end items are explained in paragraph 3.3.5.3.1 below as "Features" within a Release. The software development reporting process shown in Figure 1 for sequential Releases also applies to Agile Releases within a contract; however, it is possible that each ERP Software Development SRDR may be a Final Report only for smaller Releases.
- The Release is divided into "Sprint" time boxes, which are used to schedule specific daily workload to complete from a Backlog of defined software development tasks. Each Sprint is a recurring, non-overlapping cadence for development, usually 1 - 6 weeks in duration.
- Work within a Sprint is assigned to Scrum teams, who meet daily to plan activities, assign Backlog, and assess progress and impediments.

Include the vendor's specific "Days per Sprint" or other productivity metrics that apply to the Release described for each ERP Software Development SRDR form.

3.3.5.3.1 Release Map. The workload to assign at the Sprint level represents high level requirements (Epics/Capabilities) that are then decomposed into well-defined Feature Stories or cases. Each Feature Story is then decomposed into smaller Stories that define the workload to assign at the Sprint level (Backlog). Output from each Sprint (completed software) is integrated at the Feature level into working software that the user can evaluate. One or more Features roll-up to comprise completed Epics/Capabilities. One or more Epics/Capabilities comprise the Release.

Use the "Release Map" to identify the relationship of Features to

Epics/Capabilities for the Release defined by each ERP Software Development SRDR (one or more Epics/Capabilities for the Release; one or more Features per Epic/Capability). Additional lines may need to be inserted to accommodate the full Feature/Epic map. The Feature or Epic/Capability "Identifier" is assigned by the developer using their own conventions for naming or tracking workload. If terms other than "Feature" or "Epic" are used (for example: Acquisition Requirements Documents – or ARDs; Software Requirements Specifications – or SRSs), then provide an explanation in the comments.

3.3.5.3.2 Planned and Achieved Development (by Feature per Epic/Capability). Development tasks in the Backlog are sized using a sizing convention, such as number of Feature Stories, Story Points, Function Points or SLOC. Planned and Actual Stories describe the relative complexity of the Feature. Story Points represent a relative scale that assigns a size to each task in the Backlog, so that the tasks can be "fit" into Sprints. Planned Story Points represent an estimate for all of the Backlog tasks needed to complete each Feature. Actual Story Points represent the known size of the Backlog tasks actually completed per Feature. If known, Total Hours required to complete the Backlog tasks for each Feature should also be included.

Note: "Hours" are a measure of effort. Effort is captured specifically in Part 2 of the ERP Software Development SRDR, as mapped to a breakout of functions, such as: Planning, Design, Build, etc … (ref para 3.4) The Agile development hours included here are not mapped to any specific Part 2 function, and will overlap the hours provided in Part 2. The Feature hours included here will be used to help correlate specific Story Point sizing to the effort required per Feature, rather than correlating the Story Point sizing with the overall effort for the Release.

3.3.5.3.3 Summary Total for This Release. This table summarizes the entries for all Release Map rows, and is not aligned with any specific Release Map row. The total number of Features, Epics/Capabilities, Story Points and Feature Hours recorded as "Planned and Achieved Development" are summed here, along with the total number of Sprint time periods scheduled within the Release. Planned values apply to the Initial and Interim Reports. Actual values apply to Interim and Final reports.

3.3.5.4 Other Measures. If other sizing measures are used, other than Function Points, SLOC or Agile measures, they shall be named, and the associated counts provided. Any standards or guidelines applied in developing counts shall be fully explained.

3.3.5.5 Developmental Defects. Discovering and resolving defects in the software being delivered is included here as a sizing measure and quality measure for the overall software development associated with this Release. If other measures of development quality are used instead, describe them under Comments (ref para 3.3.5.6.)

3.3.5.5.1 Number of Defects Discovered. Report the actual total number of software and firmware defects that were discovered. The ERP Software Development SRDR shall list the actual defect software and firmware discovery counts for both Priority 1 and Priority 2, as described in ISO/IEC TR 24748-1 (reference 2.2.10), and shown in Table 2 below. Defects for all other priorities will be summed into a single value.

**Table 3: Software Change Priority**

| Priority | Criteria |
|---|---|
| 1 | a) Prevent the accomplishment of an operational or mission essential capability |
| | b) Jeopardize safety, security, or other requirement designated "critical" |
| 2 | a) Adversely affect the accomplishment of an operational or mission essential capability and no workaround solution is known |
| | b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, and no work-around solution is known |
| 3 | a) Adversely affect the accomplishment of an operational or mission essential capability but a workaround solution is known |
| | b) Adversely affect technical, cost, or schedule risks to the project or to life cycle support of the system, but a work-around solution is known |
| 4 | a) Result in user or operator inconvenience or annoyance but does not affect a required operational or mission essential capability |
| | b) Result in inconvenience or annoyance for development or support personnel, but does not prevent the accomplishment of those responsibilities |
| 5 | Have any other effect |

3.3.5.5.2 Number of Defects Removed or Corrected. Report the actual total number of software and firmware defects resolved in order to pass qualification testing. Of the total defects resolved, the ERP Software Development SRDR shall list the total resolved defect software and firmware counts for Priority 1 and Priority 2, as defined in the above table. Defects for all other priorities will be summed into a single value.

3.3.5.5.3 Number of Defects Deferred. Report the total number of software and firmware defects deferred to post deployment or other software releases. Of the total defects deferred, the SRDR for Development shall list the total deferred defect software and firmware counts for Priority 1 and Priority 2, as defined in the above table. Defects for all other priorities will be summed into a single value.

3.3.5.6 Comments on Subsection D2 responses. To be used as needed.

3.3.6 Subsection E - COTS Procurement Reporting. Enter the appropriate information for each commercial product procured to support the ERP system. Provide actual values only for the program in the Final version of the SRDR. Subsection E does not need to be included in Initial or Interim reports. Format 3 (DD Form 3026-3) includes the following data at the Release Level.

3.3.6.1 ERP Software Product Purchases. This section will provide a list of mission applications (ERP, Bolt-Ons). Includes: Product Name, Release ID, and Quantity Procured. Add rows as needed to accommodate the Primary ERP product, and any Secondary or additional ERP products.

3.3.6.2 Other Software Products (e.g., Bolt-Ons). As indicated in DOD 5000.4-M-2, other support software products that perform functions such as decision support, management information support operating system support, etc. This section will provide the software Product Name, Release, and Quantity of Licenses procured. Add rows as needed to accommodate all software products procured to support the program

3.3.7 Subsection F – Project Implementation Reporting. This subsection contains data to describe the fielding and implementation of the ERP System, to include a description of the implementation sites, a count of system users by site type and a narrative description of training classes developed in support of the system. Format 3 (DD Form 3026-3) includes the following data at the Release Level.

3.3.7.1 Implementation Sites. Enter a count by type for all places or locations the system is being deployed. A “site” is an installation for the purposes of system deployment. Provide separate counts by Site Type for sites located in the Continental United States (CONUS) and for sites outside of the Continental United States (OCONUS). Site types include:

3.3.7.1.1 Development and Test. Government or vendor owned and managed facility and equipment used to develop and test the ERP system.

3.3.7.1.2 System Hosting/Operations. Government or vendor owned and managed facility and equipment hosting the operational ERP system.

3.3.7.1.3 System Back-up/Continuity of Operations (COOP). Government or vendor owned and managed facility and equipment hosting a back-up version of the primary system, to ensure continuity of operations for system users and data.

3.3.7.1.4 User Locations. Number of locations where users can access the primary or back-up systems for daily operations; for example: military installations; other government installations; commercial locations.

3.3.7.1.5 Other locations where the system may be installed or accessed as defined by the system integrator.

3.3.7.2 Users (by Site Type). Enter the number of users associated with each ERP increment/release. Enter counts separately for sites by type in the CONUS and OCONUS.

3.3.7.2.1 Developer User. Physical count of actual individuals who use the development and administration tools provided with the software for the purpose of modifying, deploying and managing SAP/ORACLE/ PeopleSoft or 3rd party applications or for the purpose of creating, modifying, deploying and managing custom developed applications.

3.3.7.2.2 Professional User. Physical count of actual individuals who perform operational related roles supported by the Software. The Professional User license includes the rights granted under the Limited Professional User license.

3.3.7.2.3 Limited Professional User. Physical count of actual individuals who perform limited operational related roles supported by the Software. In particular employees of Business Third Parties are to be licensed as Limited Professional Users.

3.3.7.2.4 Employee User. Physical count of actual individuals who perform employee self-service related (non-job specific) roles supported by the Software. Each Employee User shall access the Software solely for such individual's own purposes and not for or on behalf of other individuals.

3.3.7.3 Initial Training Courses by site type shown above. Describe the training courses in the training inventory maintained by the ERP Training Team. List and describe all Training courses (computer-based courses and instructor led courses) developed for the Major Program shown in the Common Heading (Ref para 3.2.1.1).

3.3.7.3.1 Instructor Led Training. Describe each course designed to be presented in a classroom by an instructor. Add rows as needed for each course to be described.

3.3.7.3.2 Computer Based Training (CBT). Describe each course designed to be presented on-line or with the aid of a computer. Add rows as needed for each course to be described.

The following are examples of both, Computer-Based Training (CBT) and Instructor-Led Training (ILT) courses for an ERP Project:

| CBT | ILT |
|---|---|
| 1. ERP Navigation and Reports | 1. ERP Basic SAP Navigation |
| 2. Journal Entries Processing and Approval | 2. ERP Basic Query and Reporting |
| 3. Budget Distribution | 3. Introduction to Financials |

| | |
|---|---|
| 4. Purchase Requisitioning and Approval Workflow<br>5. Project Systems Process Overview<br>6. Plant Maintenance Process Overview<br>7. Purchase Orders and Contacts | 4. Financials Master Data Maintenance<br>5. Journal Entries Processing and Approval<br>6. Financial Reporting<br>7. Period End Closing |

3.3.7.4 Comments on Project Implementation Reporting. To be used as needed.

3.4 Part 2. Software Development Effort Data. Part 2 of the ERP Software Development SRDR provides a means to collect data on labor effort by development activity for an ERP system development.

The Initial and Interim Reports shall contain Actual-to-Date and EAC total effort. The Final SRDRs shall contain the Total Actual effort hours expended by month for all activities reported. Contractor accounting pulls for effort should follow standard accounting months; no additional mapping of time periods is required for Format 3. Provide the end date for each accounting month provided, as well as the end date of the month immediately preceding the first month reported. Monthly reporting should be included for completed releases in Interim Reports, where possible.

- Prime Contractor Effort. Enter the staff hours (estimated or actual) for each development activity performed by the Prime Contractor as defined below in paragraphs 3.4.1 through 3.4.6.
- Subcontractor Effort. Enter the total staff hours (estimated or actual) for all software development activities performed by Subcontractors for each activity.
- Common WBS Elements. Provide staff hours for the common activities such as System Engineering and Program Management, Systems Test and Evaluation, etc. that can be attributed to software development efforts if already tracked by the prime contractor. The Final Report for each release shall provide cumulative actual effort hours expended by month.

The ERP Development WBS published in Mil-Std-881C, Appendix K does not include ERP specific activities below Level 4 (Release Level Data) for each release; therefore, the following lower level activities may not be included in each SW release. However, the contractor is not required to create Control Accounts for the sole purpose of reporting EACs for effort hours on the SRDR. Hours should be consistent with the OSD DDCA-approved CSDR Plan. Format 3 (DD Form 3026-3) shall indicate in “Comments on Part 2 Responses” (para 3.4.7) whether or not any activity being reported is below a pre-established control account.

Format 3 (DD Form 3026-3) includes the following data at the Release Level. Para 3.4.1 through 3.4.3 below generally describe the effort to design, code and unit test ERP software. Para 3.4.4 through 3.4.5 describe the general effort to deploy the system software to user sites, go-live with the new ERP system and provide post-support to the newly activated sites.

3.4.1. Plan and Analyze. The Plan/Analyze phase is where the project scope is reviewed, and the solution blueprint defined. Activities include (for example):

- Release Planning. Report total hours for Release Planning activities.
- Blueprinting/Gap Analysis. Report total hours for Blueprinting and Gap Analysis activities.
- Other (specify). Report other hours as defined by the system integrator.

3.4.2. Design / Build. The Design Phase is where all design takes place that includes configurations, reports, extensions, and conversions. Upon design completion the build phase begins which includes building the master configuration, developing detailed technical designs, planning component test, and building and testing the application components. Activities include (for example):

- Functional Integration. Report total hours completed for Functional Integration.
- Technical Integration. Report total hours completed for Technical Integration
- Object Development. Report total hours completed for Object Development.
- Conversion Development. Report total hours completed for Conversions.
- Build Testing. Report total hours completed for component testing.
- Enterprise Architecture. Report total hours completed for Enterprise Architecture activities.

- Other (specify). Report other hours as defined by the system integrator.

3.4.3. Test. The test phase is where the assembly and product tests occur (Development Test and Evaluation). The assembly test ensures that components work correctly when integrated into a complete application. The product test is performed to ensure each application meets the product requirements and that all applications work together. At the end of the phase is where User Acceptance testing occurs (Operational Test and Evaluation). Include other test activities as defined by the System Integrator.

- Development Level Testing. Report total hours completed all development testing events, to include application component testing.
- Other (specify). Report other hours as defined by the system integrator.

3.4.4. Deployment. The deployment phase is defined to complete the tasks and deliverables needed to deploy the application to the users and transition the management responsibilities to the supporting unit. Activities include (for example):

- Hardware and Software Installation. Report hours completed for hardware and software installation at all sites.
- User Documentation. . Report hours completed for developing user level documentation to support the fielded system.
- Site Activation. Report hours completed to activate the ERP system at each site and verify/validate the system performance.
- User Training. Report hours completed to train users of the ERP system.
- Data Migration. Report hours completed or migrating data from a legacy system to the ERP system.
- Other. Report other hours as defined by the system integrator.

3.4.5. System Support. Includes activities provided by the development vendor to support the system as it is being developed and deployed.

- System Administration. Report hours completed for system administration activities.
- Help Desk. Report hours completed for all help desk activities.
- Post Go Live Support. Report all hours completed for post go live support.
- Other. Report other hours as specified by the system integrator.

3.4.6. Other Direct Labor. Includes vendor program management costs and government program office costs (when the government is the system integrator) for:

- System Engineering. Report hours completed for SE.
- Program Management. Report hours completed for PM.
- Change Management. Report hours completed for Change Management
- System Level Test and Evaluation. Report hours completed for System Test and Evaluation (Operational Test and Evaluation).
- Develop and Manage Training. Report hours completed for Training.
- Engineering Data. Report hours completed for Engineering Data.
- Other (specify). Report other hours as specified by the system integrator.

3.4.7. Comments on Part 2 Responses. Provide any comments about resource reporting.

**END OF DI-MGMT-82035 A**