MTR090007
MITRE TECHNICAL REPORT

MITRE

Recommendations for Managing Software Reuse

B. S. Woodward
A. E. Taub
Y. M. Perlmutter
J. A. Maurer
L. M. Rosa

23 September 2009

MTR090007
MITRE TECHNICAL REPORT

MITRE

Recommendations for Managing Software Reuse

B. S. Woodward
A. E. Taub
Y. M. Perlmutter
J. A. Maurer
L. M. Rosa

23 September 2009

Sponsor: ESC/EN
Dept. No.: E523, E150, E540
Contract No.: FA8721-09-C-0002
Project No.: 03X93DB0

Abstract

A handbook has been developed for program office (PO) use to manage software reuse and its associated risks. Government policies strongly encourage software reuse in the interests of more rapid fielding, lower life cycle costs, and increased interoperability. However, this approach to product development is fraught with risks, and must be managed properly. The handbook provides a Software Reuse Risk Guide that lists major risk areas, associated risk questions, and a brief tutorial to help a PO identify program risks related to software reuse. Risk templates are included to help a PO assess these risks. The handbook also provides sample wording for a Request for Proposal (RFP) to ensure the PO has consistent information about the offerors' software reuse approaches during a source selection and appropriate levels of insight into the contractor's software reuse approach and design activities after contract award. The handbook includes wording, extensively vetted with subject matter experts, for an RFP's Statement of Objectives, Statement of Work, and Contract Data Requirements List as well as for the Special Contract Requirements (Section H), Representations, Certifications, and Other Statements of Offerors (Section K), Information to Offerors and Instructions for Proposal Preparation (Section L) and Evaluation Factors for Award (Section M). Sample wording is also included for a Request for Information and evaluation criteria for an Award Fee Plan. In addition, the handbook contains (1) detailed worksheets to be completed by the offeror/contractor about a software reuse product's applicability, availability timeline, maturity, modification, and other attributes, (2) a spreadsheet to standardize offerors' presentations of sizing, schedule, and historical information for software reuse products, and (3) an approved Data Item Description for a Reuse Management Report.

Executive Summary

Background

For years, the Government and contractors have eagerly attempted to reuse software in the interests of more rapid fielding, lower life cycle costs, and increased interoperability. Frequently, both parties have been too optimistic: they overstated the amount of code they would be able to reuse and underestimated the effort required to reuse it. As a result, the anticipated benefits of software reuse have been seldom realized. The contractor has had to find an alternate source of software, or unexpectedly develop it from scratch—but with a delayed start. These disruptions undermine program success, becoming a significant cost and schedule driver. Reasons why software reuse can be problematic for programs include: poor assessment of the applicability of the software to the host program, immaturity of the software and its supporting artifacts, slip in the availability timeline, competing requirements for software being used on multiple programs, poor quality of software, and underestimation of the effort required to adapt/modify/integrate the existing software.

Objective

Federal Acquisition Regulation (FAR) policy states:

> "Agencies shall perform acquisition planning and conduct market research for all acquisitions in order to promote and provide for—(1) Acquisition of commercial items or, to the extent that commercial items suitable to meet the agency's needs are not available, nondevelopmental items, to the maximum extent practicable." [*Federal Acquisition Regulation*, Part 7 (7.102), 10 September 2009.]

Recognizing that Government policies encourage software reuse, this handbook was developed for the 653d Electronic Systems Group (653 ELSG), HQ Electronic Systems Center, Air Force Materiel Command and is intended to:

1. Support program office (PO) assessment of risks associated with software reuse
2. Provide recommendations for PO management of software reuse

The handbook was developed in accordance with Air Force acquisition strategy planning regulations as well as Defense Federal Acquisition Supplement (DFARS) requirements for solicitations. It identifies deliverables and activities that a PO can use to manage software reuse and its associated risks on their program. This handbook provides additional words for the Request for Proposal (RFP) package, to ensure the PO has:

1. Consistent information about the offerors' software reuse approaches—during source selection
2. Appropriate levels of insight into the contractor's software reuse approach and design activities—after contract award

Scope

This handbook will help the PO with management of the following types of reuse:

1. Software reuse, including pre-existing software products that will be reused as-is and modified software products (pre-existing software requiring change), for which the offeror/contractor plans to assume responsibility for the performance of the product.
2. Commercial off-the-shelf (COTS) and Government off-the-shelf (GOTS) software products, for which the software provider, either a commercial vendor or the Government, assumes responsibility for the performance and maintenance of the software product. It is assumed that existing COTS/GOTS software products will be reused as-is; COTS/GOTS products requiring modification are considered modified software (i.e., no longer COTS/GOTS, and the responsibility of the contractor). Reuse of COTS/GOTS products that will be modified and maintained by the software provider is not recommended, and is not addressed in this handbook.
3. Issues unique to reuse of open source software are not addressed in the handbook.
4. Issues unique to reuse of firmware are not addressed in the handbook.
5. Issues unique to reuse of services are not addressed in the handbook.

Software reuse is defined as reuse of code. Reuse of software assets (e.g., architectures, algorithms, designs, design patterns, test plans, test cases, interface specs, documentation) without code is not addressed in this handbook.

It also should be noted that the handbook does not address the important topics of security and information assurance, and how these topics affect software reuse. A subsequent version of the handbook will include these topics.

Software Reuse Risk Guide

The Software Reuse Risk Guide, presented in Section 1, helps a PO identify and assess program risks related to software reuse. To understand the challenges associated with reusing software, the first step for a PO should be to identify the major risks that pertain to their specific program. The Guide includes a list of major software reuse risks and a brief tutorial that explains why a PO should be cautious when incorporating software reuse products into their system's software baseline. Risk questions are included for a PO to answer before source selection, to evaluate during a source selection, and to address after contract award (ACA) as the design evolves. The Guide also includes risk templates to help the PO assess software reuse risks.

Recommended RFP Content

It is recommended that the PO use the following new deliverables/activities to manage software reuse: software reuse management, software reuse demonstration(s), and software quality assessment(s). Additionally, it is recommended that the PO augment their standard program deliverables/activities with added focus on software reuse, to include: risk [and opportunity] management, software size, Software Development Plan (SDP), software

metrics, past reuse performance, Integrated Master Schedule (IMS), Integrated Master Plan (IMP), Contract Work Breakdown Structure (CWBS), and Contract Performance Report (CPR). Recommended RFP content is provided for these deliverables/activities in Sections 2 through 13.

This handbook provides words for the Special Contract Requirements (Section H), Representations, Certifications, and Other Statements of Offerors (Section K), Information to Offerors (ITO) and Instructions for Proposal Preparation (Section L) and Evaluation Factors for Award (Section M) to elicit appropriate information in support of the Government evaluation of software reuse during source selection. Deliverables/activities for software reuse management ACA are ensured via the contract, including the Statement of Work (SOW), Contract Data Requirements List (CDRL), Section H, and Section K. The use of these deliverables/activities is at the PO discretion; they should be selected and/or modified based on program characteristics, the expected software reuse, and the associated software reuse risks.

The sample RFP words provided in the following sections may be tailored as needed; notes at the beginning of each section provide additional suggestions for tailoring. Some words, either as part of the notes or each section, are italicized and in brackets, (e.g., [example]). The PO should insert appropriate words, consistent with their program deliverables/activities. For ease of reference, the sample words are often presented in bulleted lists. However, the PO must follow the applicable standards for preparation of all RFP documents.

The RFP additions address only software reuse. Although software reuse is a subset of software engineering, systems engineering, and program management, this handbook does not provide RFP wording for these broader areas. It is the responsibility of the PO to define and describe these engineering and management activities, in the context of their program, in the RFP.

The handbook also includes sample words for a Statement of Objectives (SOO) in Section 14, a Request for Information (RFI) in Section 15, and an Award Fee Plan (AFP) in Section 16.

The following table provides a list of the recommended activities, deliverables, and supporting products included in the handbook. Also presented is the RFP document where they are called out. Templates are available in Microsoft Word or Excel 2007, under separate cover, to permit the input of data for the following products:

- Software Reuse Risk Guide
- Worksheet Questions for Reused As-is/Modified Software (Appendix A)
- Worksheet Questions for COTS/GOTS Software (Appendix B)
- Data Item Description (DID) for the Reuse Management Report (Appendix C)
- Revised Format M-1 (Appendix D)

Recommended Activities, Deliverables, and Products

Sect.	Activity/Deliverable/ Product	Applicable RFP Document		Appendices
		Source Selection	ACA	
2.	Software Reuse Management	H or K L & M	H or K SOW[1] CDRL[2]	A. Worksheet Questions for Reused As-is/Modified Software[3] B. Worksheet Questions for COTS/GOTS Software[3] C. Data Item Description for Reuse Management Report[4]
3.	Risk *[and Opportunity]* Management	L & M	SOW[1] CDRL[2]	C. Data Item Description for Reuse Management Report[4]
4.	Software Reuse Demonstration(s)	L & M	SOW[1] CDRL[2]	
5.	Software Quality Assessment(s)	H or K L & M	H or K SOW[1] CDRL[2]	
6.	Software Size	L & M		D. Format M-1 (Revised)[5]
7.	Software Development Plan	L & M	SOW[1] CDRL[2]	
8.	Software Metrics		SOW[1] CDRL[2]	
9.	Past Reuse Performance	L & M		
10.	Integrated Master Schedule	L & M	SOW[1] CDRL[2]	
11.	Integrated Master Plan	L & M		
12.	Contract Work Breakdown Structure	L & M	CDRL[2]	

Recommended Activities, Deliverables, and Products (Concluded)

13.	Contract Performance Report		CDRL[2]	
14.	Program Objective[1]	SOO		
15.	Pre-RFP Information Request[6]	RFI		
16.	Award Fee Plan Criteria[7]		AFP	

[1] If a PO is not developing a SOW as part of the RFP, a single SOO objective for software reuse is provided in Section 15.

[2] If a program has to limit their CDRL, and cannot incorporate the recommended data items, the PO should strive to obtain from the contractor via other means the software reuse information called out in the CDRL (e.g., in other deliverables, on shared integrated data environment).

[3] Information about product applicability, availability timeline, maturity, modification, and other attributes should be required of the offerors/contractor using the appropriate Worksheet Questions.

[4] This new data item should be used to elicit information about the status of the software reuse approach and a description of alternative strategies. The DID for DI-SESS-81771 is available on the ASSIST database.

[5] Format M-1 (Revised) should be required of the offerors to provide sizing, schedule and historical information for all new, reused as-is, and modified software products.

[6] An RFI should request information about software reuse products being considered by contractors.

[7] Software reuse focus for the AFP is provided via integrated evaluation criteria for each of the Schedule, Technical, and Program Management areas. No software reuse-specific Cost area evaluation criterion has been generated; it is recommended that cost evaluation criteria address the total program.

Use of Handbook

The handbook provides a variety of products that can be used from program start to the end of the contract. Examples of ways a PO can use the handbook include:

- Review the Software Reuse Risk Guide tutorial for a better understanding of the major risk areas and reasons why a PO should be cautious when incorporating software reuse products into their system's software baseline
- Use the Guide's risk questions to identify the major software reuse risk areas applicable to the program; revisit throughout the life of the program
- Use the Guide's risk templates to assess the level of risk; update as the program evolves and more information becomes available
- Incorporate the sample wording into the program's RFI to collect information about software reuse products being considered by the contractors

- Incorporate pertinent Section M sample wording into the program's RFP to ensure the Government will use and evaluate software reuse criteria to make the best value award decision
- Incorporate pertinent Section L sample wording into the program's RFP to make sure the Government will be provided with appropriate information about the offerors' planned software reuse approaches; make certain the Government requests sufficient information, upon which to base the evaluations of the proposals
- Include the Worksheet Questions in the RFP to gather detailed information about key attributes for all software reuse products
- Use the revised M-1 Format to better understand the offerors' methods for estimating software size and obtain a concise computation of effective sizing
- Incorporate the sample words for applicable activities and deliverables into the program's SOW and CDRL so that a contractor's software reuse approach can be monitored ACA
- Use the software reuse objective for a SOO, if a SOW is not being prepared as part of the RFP, to ensure the contractor provides a comprehensive software reuse strategy
- Use the data item of a Reuse Management Report for the contractor to provide the current status, milestones, alternative strategies, and decision points for a the software reuse approach
- Select relevant AFP criteria to incentivize desirable contractor behaviors pertaining to software reuse

Points of Contact

Questions regarding this handbook, requests for subject matter expert support, and feedback on use of these materials should be directed to:

Yvonne Perlmutter (ymp@mitre.org)
Beverly Woodward (bsw@mitre.org)
Audrey Taub (ataub@mitre.org)
John Maurer (johnm@mitre.org)
Lynda Rosa (lmrosa@mitre.org)

Acknowledgments

The authors would like to thank Colonel Cordell DeLaPena and Mr. Michael Therrien of the 653 ELSG for their support and the funding to develop this handbook. We also appreciate Mr. Jeffrey Mayer's (ESC/EN) support in making the handbook accessible and available to AF programs, and within the DOD community.

The authors wish to acknowledge the valuable insights provided by Ms. Susan Angell (ESC/AQ) and Mr. Paul Commeau (ESC/AQ). We would also like to thank Mr. Rick Andreoli (ESC/AQ), Mr. Richard Bean (ESC/JA), Ms. Carol Hoffses (ESC/AQ), Mr. Robert Klauzinski (ESC/JA), Ms. Marla Levenson (ESC/AQ), and Mr. Richard Stillman (ESC/AQ) for their thorough technical review of the handbook, constructive recommendations, and comprehensive comments.

The authors are grateful to many MITRE subject matter experts, who contributed to this document. We would like to thank Wayne Addy, C. Wayne Chitwood, Sandra Cole, Paul Funch, Andrew King, Robert Martin, and Jim Moore for sharing their knowledge and influencing the handbook content.

We would like to convey our appreciation to Patricia Jack for her patience, professionalism, and extensive effort to produce the MTR.

Table of Contents

1 Software Reuse Risk Guide

The Software Reuse Risk Guide helps a program office (PO) identify and assess program risks related to software reuse. The Guide includes a list of major risk areas and associated questions to identify the risks, and risk templates to assess the risks.

The major risk areas were selected based on discussions with program managers and software engineers, who had extensive experience with software reuse and its risks. These risks have the potential for significant impact to a program's performance, cost and/or schedule. They should serve as a starting point for identification of program risks. The PO should determine which set of risk areas are most applicable to their program and delete, add, or modify them, as necessary. Some additional examples of software reuse risks are included as well; these risks are more program-specific. The PO should not be constrained to select from these lists. Note that some risks are pertinent to reused as-is/modified software, while others are more applicable to commercial off-the-shelf/Government off-the-shelf (COTS/GOTS) software.

The risk questions explore software reuse risks to be considered before source selection and after contact award as the design evolves. To identify the risks during a source selection, the Guide provides a mapping of the major risk areas to the Worksheet Questions (denoted Q. #) that are presented in Appendices A and B. The Guide also maps the risk areas, if applicable, to Format M-1 in Appendix D as well as to specific deliverables and activities in Section L. This mapping helps identify where information pertaining to the risk areas may be found in the offerors' proposals. However, it should be noted that the Source Selection Evaluation Team (SSET) must consider all applicable information provided by the offerors when evaluating risks.

As shown, there are many questions that the PO should be asking about the risks of software reuse. These questions could be answered by the PO, by the offeror/contractor, or both, as appropriate. Although examples of questions are provided in the Guide, the PO should add to or tailor these risk questions to reflect their program's characteristics.

The risk templates provide criteria for assessing the major risks areas for software reuse. If the PO wants to use the templates as part of a source selection, the PO must convey this intent to the offerors and provide them with the specific risk criteria, against which their software reuse products will be assessed. Alternatively, the PO may want to incorporate a few specific risk criteria (e.g., for the product's availability timeline or maturity) in their Request for Proposal (RFP). These criteria could be included directly in the risk or software development sections in the Information to Offerors and Instructions for Proposal Preparation, and the Evaluation Factors for Award.

Criteria are provided for low, medium, and high risk; however, it is important not to confuse these risk criteria with proposal risk during a source selection. These criteria may be used as-is or reworded by the PO to better reflect program specifics. At the time of the risk

assessment, there may be insufficient information, upon which to form a judgment for a particular risk area. The PO is able to indicate this lack of information in the template, and then update the assessment, as information becomes available. Once the level of risk has been selected, it is recommended that a brief description for the basis of assessment be included.

The Guide includes a brief tutorial to provide a better understanding of each risk area. The tutorial is not intended to present a highly detailed discussion of each risk, but rather include helpful information that supplements an assessment of software reuse risks. The Excel format of the Guide provides these descriptions as pop-up comments. The descriptions are also included at the end of this section for easy reference.

Major Risk Areas and Risk Questions

Major Risk Areas	Risk Questions for the PO to Answer Before Source Selection	Risk Questions to Evaluate During Source Selection		Risk Questions As the Design Evolves
		Reused As-is/ Modified Software	COTS/GOTS Software	
1. Applicability	What program functional requirements can be satisfied by software reuse? How well is the functionality of the reuse products known or understood? What are the mismatches or gaps, if any?	Appendix A: Q. 7-8	Appendix B: Q. 5-6	Is the software still applicable? Does the TRD requirements flowdown support the planned use of this product? Does the information obtained from the demos continue to support the use of these products? Have new gaps been identified? What are the plans to fill them?
2. Hardware/Software Platform	Does your program have a hardware or software platform that will present challenges when hosting the potential reuse software?	Appendix A: Q.11	Appendix B: Q.9	Do the selected platforms pose any problems for the software reuse products? If so, what are the implications of these problems?
3. Architecture	Is there a mismatch between your program architecture and the architectures of the potential software reuse candidates?	Appendix A: Q. 10	Appendix B: Q. 8	Do you still have architectural compatibility? If not, what is your solution?
4. Interfaces	N/A	Appendix A: Q. 9	Appendix B: Q. 7	Are there interface mismatches between the software reuse product and the system? If so, what are the implications of the mismatches?
5. Modification	Do the potential software reuse candidates require modification? If so, how much?	Appendix A: Q. 12-15	N/A	Has the extent of the modifications to the software reuse products changed? If so, how much? Will the COTS and GOTS software products remain unmodified? If not, what is the plan for assuming responsibility for them?
6. Maturity	Are the potential software reuse candidates mature (e.g., formal qualification tested, system level tested, fielded)?	Appendix A: Q. 16-17, 19, & 21	Appendix B: Q. 15-16	Has the maturity profile of the software reuse products changed?
7. Availability Timeline	Are the potential software reuse candidates currently available? If not, will they be available at contract award? If not, when will they be available? Is the availability of the software dependent upon another source, e.g., another program, the Government, or a COTS vendor?	Appendix A: Q. 22-24	Appendix B: Q. 18-19	Has there been any change in the availability timeline for the software reuse products?
8. Reuse History	Have other Government programs reused as-is or modified this software? If yes, what programs? Do you know whether the reuse was successful?	Appendix A: Q. 20	Appendix B: Q. 17	N/A
9. Developer's Experience with Software	N/A	Appendix A: Q. 26-28	Appendix B: Q. 20-22	Does the contractor have staff who are knowledgeable about the software reuse products? Does the contractor have access to the originating developers of the reuse products?
10. Documentation	N/A	Appendix A: Q. 29-32	Appendix B: Q. 23	Is the software reuse product documented? Is the documentation available? Has the quality of the documentation been assessed? Does it provide the information needed? Does the documentation contain proprietary information?

Major Risk Areas and Risk Questions (Concluded)

Major Risk Areas	Risk Questions for the PO to Answer Before Source Selection	Risk Questions to Evaluate During Source Selection		Risk Questions As the Design Evolves
		Reused As-is/ Modified Software	**COTS/GOTS Software**	
11. Software Quality	N/A	Software Quality (Section L)	Software Quality (Section L)	What is the quality of the software? Has the contractor conducted an assessment of the quality of the software? What are the results of the assessment? Has the Government SQAE been conducted? What are the results of the SQAE?
		What are the results of the Offeror's quality assessment of the software?		
12. Software Defects	N/A	Appendix A: Q. 37-38	N/A	What is the current defect profile (e.g., number of open/closed defects, closure rate, priority of defects) of the software reuse products?
13. Relationships with Sources of Software Reuse Products	N/A	Appendix A: Q. 24	Appendix B: Q. 22	Has the contractor established a good working relationship with the entities or sources responsible for all software reuse products?
14. Software Reuse Schedule	N/A	Appendix D SDP (Section L) IMS (Section L) Offeror BOEs	Appendix D SDP (Section L) IMS (Section L) Offeror BOEs	Are the durations of the modification (if needed), integration and test schedules realistic given the amount of work to be performed? Do the schedules reflect sufficient up-front tasks and decision points to maximize the likelihood of reuse success?
		Are the durations of the modification (if needed), integration and test schedules realistic given the amount of work to be performed? Do the schedules reflect sufficient up-front tasks and decision points to maximize the likelihood of reuse success?		
15. Software Reuse Sizing	N/A	Appendix A: Q. 15 Appendix D Offeror BOEs	Appendix B: Q.14 Appendix D Offeror BOEs	Have the effective sizing estimates changed? What is the impact on the program?
		Are the effective sizing estimates realistic given the amount of work to be performed?		

Program-specific Risk Areas and Risk Questions

Program-specific Risk Areas	Questions for the PO to Answer Before Source Selection	Risk Questions to Evaluate During Source Selection		Risk Questions As the Design Evolves
		Reused As-is/ Modified Software	COTS/GOTS Software	
Future Releases	N/A	Appendix A: Q. 40	Appendix B: Q. 32	If the program is planning to incorporate future releases into the system's software baseline, is there a plan to assess the impacts to the system, address any changes in performance or functionality, interoperability with other systems, etc., and reintegrate the releases into the system? If the program is not planning to incorporate future releases into the system's software baseline, is there a plan to address critical fixes and vendor support when the product becomes obsolete?
Dead and Unused Code	Could dead or unused code in potential reuse software pose problems related to security or testing? What are the potential problems?	Appendix A: Q. 41	N/A	Does the dead or unused code in the software reuse products pose any problems related to security or testing? What are the problems?
Certifications and Accreditations	Could the certification and accreditation (C&A) of the new software be affected by the reuse as-is or modification of the potential software reuse candidates? What are the potential problems? Should NSA (or other C&A agencies) be involved with the RFP preparation and source selection? Does the program schedule reflect enough time for the C&A process?	Appendix A: Q. 18	Appendix B: Q. 17	How is the C&A of the new software affected by the reuse as is or modification of the software reuse products?
Designed for Reuse	N/A	Appendix A: Q. 25	N/A	Has the software reuse product been designed for reuse? Identify the attributes (e.g., standards, design patterns, architecture paradigms) that support reuse.
Data and Software Rights	Are there any potential problems with the data and software rights for the software reuse candidates such that they are inconsistent with the program's maintenance philosophy?	Appendix A: Q. 34-36	Appendix B: Q. 24-26	What data and software rights will the Government have to the software reuse products? Are these rights consistent with the program's maintenance philosophy?
Maintenance & Support Strategy	Are there any potential problems with the planned maintenance of the software reuse candidates such that they do not fit with the maintenance philosophy for the program?	Appendix A: Q. 39	Appendix B: Q. 30-31	How does the maintenance of the software reuse products fit with the maintenance philosophy for the program?
Integration into Build Plan	N/A	Appendix D SDP (Section L)	Appendix D SDP (Section L)	When is the reused as-is/modified software integrated into the build plan? Does the build plan reflect early integration activities to maximize the likelihood of reuse success?
		When is the reused as-is/modified software integrated into the build plan? Does the build plan reflect early integration activities to maximize the likelihood of reuse success?		
Vendor Viability	Is the long-term viability of each vendor of each software reuse candidate sound?	N/A	Appendix B: Q. 33-36	Does the long-term viability of each vendor of each software reuse product continue to be sound?
Standards	N/A	Appendix A: Q. 33	N/A	What development standards (e.g., IEEE/EIA Std 12207.0) were followed during the development of the software intended to be reused as-is/modified? Do the standards followed during the development of this software pose any potential problems for the software system that will be delivered for this program? Are these standards consistent with all standards to which the software system must adhere?
Licensing	N/A	N/A	Appendix B: Q. 27-29	How will the COTS/GOTS software be licensed (e.g., per seat, per site, per host) for both development and run-time for this program? Does the licensing arrangement pose any potential problems during the acquisition or maintenance of the system?

Risk Templates

Major Risk Areas	The following conditions must be met for a software reuse product to be considered low risk	The following conditions result in a software reuse product being considered medium risk	The following conditions result in a software reuse product being considered high risk	Insufficient Information	Basis of Assessment
1. Applicability	*Both conditions must be met for low risk:* 1) The contractor has performed a thorough analysis to assess the applicability of the software reuse product to this program. This analysis has been discussed with and shown to the Government. 2) The software reuse product's functionality meets all of the applicable requirements.	*Presence of both conditions results in a medium risk:* 1) The contractor has performed at least a marginally acceptable analysis to assess the applicability of the software reuse product to this program. This analysis has been discussed with and shown to the Government. 2) The software reuse product's functionality meets all of the applicable critical requirements and most of the non-critical requirements.	*Presence of any of the four conditions results in a high risk:* 1) The contractor has not performed an acceptable analysis to assess the applicability of the software reuse product to this program. 2) The contractor has not shared the analysis with the Government. 3) The software reuse product's functionality does not meet some of the applicable critical requirements. 4) The software reuse product's functionality does not meet many of the non-critical requirements.	There is insufficient information to form a judgment of risk.	
2. Hardware/Software Platform	The software reuse product has been used on a hardware/software platform identical to the intended platform.	*Presence of either condition results in a medium risk:* 1) The software reuse product has been used on a hardware/software platforms similar (e.g., same operating system/different version, standard hardware from a different vendor) to the intended platform. 2) The contractor has staff on their team who has experience porting between the specific platforms.	*Presence of both conditions results in a high risk:* 1) The hardware/software platforms, on which the software reuse product currently resides, is dissimilar (e.g., different operating system or operating system from a different vendor) from the intended platform. 2) The contractor has no previous experience porting between these specific platforms.	There is insufficient information to form a judgment of risk.	
3. Architecture	*Both conditions must be met for low risk:* 1) The architecture is consistent throughout the system (i.e., for both software reuse products and new software.) 2) There are no apparent mismatches between the architecture of the software reuse product and the architecture of the system. There may be a need for a minimal amount of simple glue code.	*Presence of either condition results in a medium risk:* 1) The software reuse product's architecture is based on well-known standards, but is not consistent with standards of the system. 2) There are some mismatches between the architecture of the software reuse product and the architecture of the system. Some glue code needs to be developed.	*Presence of either condition results in a high risk:* 1) The software reuse product's architecture is based on immature, ad hoc or no standards. 2) There are significant mismatches between the architecture of the software reuse product and the architecture of the system. There is a need to develop substantial glue code.	There is insufficient information to form a judgment of risk.	
4. Interfaces	The interfaces between the software reuse product and the system are well-defined, well-controlled, and highly compatible. There may be a need for a minimal amount of simple glue code.	The interfaces between the software reuse product and the system are well-defined and well-controlled; however, there are some compatibility problems. Some glue code needs to be developed.	*Presence of either condition results in a high risk:* 1) The interfaces between the software reuse product and the system are not well-defined or not well-controlled. 2) There are significant interface mismatches between the software reuse product and the system. There is a need to develop substantial glue code.	There is insufficient information to form a judgment of risk.	

Risk Templates (Continued)

Major Risk Areas		The following conditions must be met for a software reuse product to be considered low risk	The following conditions result in a software reuse product being considered medium risk	The following conditions result in a software reuse product being considered high risk	Insufficient Information	Basis of Assessment
5. Modification	**Reused As-is/ Modified**	The software reuse product requires no modification to meet the program's requirements.	The software reuse product requires moderate modification to meet the program's requirements, and the contractor has performed an analysis and understands the required changes.	*Presence of either condition results in a high risk:* 1) The software reuse product requires substantial modification to meet the program's requirements. 2) The software reuse product requires moderate modification to meet the program's requirements, but those changes are not well understood.	There is insufficient information to form a judgment of risk.	
	COTS/GOTS	N/A (If the OTS software reuse product requires modification, it is considered to be "Reused As-is/Modified" software.)				
6. Maturity		*Conditions 1 & 2, or 3 must be met for low risk:* 1) The software reuse product has successfully completed a Government-witnessed formal qualification test on another DoD program and is operational in the field. 2) This maturity has been confirmed with the DoD PO. 3) *For COTS products only:* The contractor has provided information showing the COTS product is mature, widely used, and has marketplace acceptance.	*Presence of conditions 1 & 3, 2 & 3, or 4 results in a medium risk:* 1) The software reuse product has successfully completed a Government-witnessed formal qualification test on another DoD program, but is not yet operational. 2) The software reuse product has successfully completed a Government-witnessed formal qualification test on another Government (non-DoD) program and may or may not be operational in the field. 3) This maturity has been confirmed with the DoD or non-DoD PO. 4) *For COTS products only:* The contractor has provided information about the COTS product's maturity, extent of usage and marketplace acceptance; the information indicates limited product maturity.	*For Reused As-is, Modified or GOTS software products only:* The software reuse product has not completed a Government-witnessed formal qualification test. *For COTS products only:* The contractor has provided information about the COTS product; the information indicates that the product lacks maturity.	There is insufficient information to form a judgment of risk.	
7. Availability Timeline		*Both conditions must be met for low risk:* 1) The software reuse product is currently available. 2) The availability has been confirmed with the source.	*Presence of both conditions results in a medium risk:* 1) The software reuse product is not currently available, but is anticipated to be available prior to contract award. 2) The contractor and the source have provided information supporting availability prior to contract award.	*Presence of either condition results in a high risk:* 1) The software reuse product will not be available at contract award. 2) The contractor and the source have not provided adequate information supporting availability prior to contract award.	There is insufficient information to form a judgment of risk.	

Risk Templates (Continued)

Major Risk Areas	The following conditions must be met for a software reuse product to be considered low risk	The following conditions result in a software reuse product being considered medium risk	The following conditions result in a software reuse product being considered high risk	Insufficient Information	Basis of Assessment
8. Reuse History	*Both conditions must be met for low risk:* 1) The software reuse product has been successfully reused as-is or modified for another Government (DoD or non-DoD) program and is operational in the field. 2) Successful product reuse has been confirmed with the Government PO.	*Presence of conditions 1 & 3 or 2 & 3 results in a medium risk:* 1) The software reuse product has been reused as-is or modified for another Government (DoD or non-DoD) program and is operational in the field, but there were moderate technical, cost or schedule problems with the product. 2) The software reuse product has been reused as-is or modified for another Government (DoD or non-DoD) program and has successfully completed a Government-witnessed formal qualification test, but is not yet operational. 3) Status of product reuse has been confirmed with the Government PO.	*Presence of either condition results in a high risk:* 1) The software reuse product has been reused as-is or modified for another Government (DoD or non-DoD) program, but there were significant technical, cost or schedule problems with the product. 2) The software reuse product has never been reused as-is or modified for another Government (DoD or non-DoD) program, or is being reused as-is or modified for another Government program, but has not yet completed a formal qualification test.	There is insufficient information to form a judgment of risk.	
9. Developer's Experience with Software	*Both conditions must be met for low risk:* 1) The contractor has staff who have experience developing or reusing the software reuse product. 2) The PO has a commitment from the contractor that the experienced staff will be available to support this program.	*Presence of both conditions results in a medium risk:* 1) The contractor has staff, who have experience developing or reusing a similar software reuse product. 2) The PO has a commitment from the contractor that the experienced staff will be available to support this program.	The contractor does not have staff who have experience developing or reusing this or a similar software reuse product.	There is insufficient information to form a judgment of risk.	
10. Documentation	There exists sufficient, up-to-date documentation for the software reuse product to support developers, end users, and maintainers.	There exists sufficient documentation for the software reuse product to support developers, end users, and maintainers; however, it is not up-to-date.	Documentation for the software reuse product to support developers, end users, and maintainers is limited.	There is insufficient information to form a judgment of risk.	
11. Software Quality	The contractor and/or PO has conducted an assessment of the quality of the software reuse product. There are no concerns about the quality of the software.	The contractor and/or the PO has conducted an assessment of the quality of the software reuse product. There are moderate concerns about the quality of the software and there is a strategy for mitigating them.	*Presence of either condition results in a high risk:* 1) Neither the contractor nor the PO have conducted an assessment of the quality of the software reuse product; therefore, there is no insight into the quality attributes for the software. 2) The contractor and/or the PO has conducted an assessment of the quality of the software reuse product and there are major concerns about the quality of the software.	There is insufficient information to form a judgment of risk.	

Risk Templates (Concluded)

Major Risk Areas		The following conditions must be met for a software reuse product to be considered low risk	The following conditions result in a software reuse product being considered medium risk	The following conditions result in a software reuse product being considered high risk	Insufficient Information	Basis of Assessment
12. Software Defects	**Reused As-is/Modified**	The high priority defects for the software reuse product have been fixed and there are no known problems that should affect the reuse of the product.	The high priority defects for the software reuse product have been fixed; however there are remaining defects that must be fixed prior to its fielding. There is a credible plan to repair these defects.	*Presence of either condition results in a high risk:* 1) The software reuse product has open, high priority defects or has a significant number of other open defects that must be fixed prior to its fielding. 2) The software has not been fully tested so that the number of defects in the code is not yet known.	There is insufficient information to form a judgment of risk.	
	COTS/GOTS	N/A	N/A	N/A	N/A	
13. Relationships with Sources of Software Reuse Products		*Both conditions must be met for low risk:* 1) The contractor has, in the past, successfully established working relationships with the source of the software reuse product. 2) The contractor has developed a comprehensive approach to manage cross program relationships, and how to stay informed about the evolving software functionality.	The contractor has no past working relationship with the source of the software reuse product, but has developed a comprehensive approach to manage cross program relationships, and how to stay informed about the evolving software functionality.	The contractor has not developed an adequate approach for managing cross program relationships, and staying informed about the evolving software functionality.	There is insufficient information to form a judgment of risk.	
14. Software Reuse Schedule		*Both conditions must be met for low risk:* 1) The contractor has identified the schedule for the software reuse product, including up-front tasks and alternative strategy decision points. 2) The schedule duration is reasonable.	*Presence of either condition results in a medium risk:* 1) The contractor has identified the schedule for the software reuse product, but the schedule does not include sufficient up-front tasks and/or alternative strategy decision points. 2) The schedule duration is highly optimistic.	*Presence of either condition results in a high risk:* 1) The schedule for the software reuse product is not clearly identifiable within the overall program schedule. 2) The schedule duration is not reasonable.	There is insufficient information to form a judgment of risk.	
15. Software Reuse Sizing		The effective sizing estimate for the software reuse product is reasonable, and reflects sufficient estimation uncertainty and growth.	The effective sizing estimate for the software reuse product is highly optimistic.	The effective sizing estimate for the software reuse product is not reasonable.	There is insufficient information to form a judgment of risk.	

Tutorial for Major Risk Areas

The purported cost and schedule benefits of reusing pre-existing software (as-is), modifying pre-existing software, or using off-the-shelf (OTS) software products may be outweighed by the risks of these acquisition approaches. The following discussion is intended to highlight major risk areas and reasons why a PO should be cautious when incorporating software reuse products into their system's software baseline. The PO needs to understand the risks of a contractor's reuse approach and then determine whether they are willing to accept those risks. If so, the PO must closely monitor the status and progress of the software reuse activities after contract award (ACA).

The following descriptions provide a brief tutorial for the major risk areas. The following tutorial is not intended to provide an all-inclusive list nor a highly detailed discussion of each risk area, but rather helpful information that supplements an assessment of software reuse risks.

1. Applicability

The software that is intended to be reused as-is, including COTS/GOTS, or modified may or may not closely match the requirements of a particular program. It is therefore necessary to understand the specific mismatches. The Worksheet Questions, which will be completed and submitted by an offeror as part of their proposal, ask the offeror to list the functions that each software product will provide and cross-reference these functions to the Technical Requirements Document (TRD). ACA, the contractor will be required to show functional and performance requirements cross-referenced to the TRD. The updated Worksheet Questions will be delivered to the Government as an appendix to the Reuse Management Report.

Both during a source selection and ACA, mismatches in functions/ requirements between the TRD and the reuse product must be clearly identified so that the Government can understand where there are disconnects and assess whether these disconnects can be resolved. In some cases, the software may need minor modification. However, if the mismatches require significant modification of the pre-existing software, the PO should be wary of the planned reuse approach.

The importance of a comprehensive applicability analysis cannot be overstated. Too often, the contractor has not performed a thorough analysis of the software reuse product and thus, the software neither meets critical requirements nor has an architecture or interfaces that are compatible with the new system. An explanation of the evaluation methodology and any criteria used to distinguish between alternative software products should be provided in the proposal. The contractor must have adhered to a formal process for the evaluation. Be skeptical of the depth and breadth of an applicability analysis. Ask to see the analysis.

The analysis results are not necessarily valid without the contractor having performed some sort of internal, hands-on demonstration and testing of the software. Even for OTS software,

both the contractor and the Government should explore vendor claims of the product's capabilities. Additionally, an applicability analysis for a rapidly changing OTS product may be valid for only a short period of time. Therefore, the Government needs to be proactive in understanding changes to these products and the potential impacts to their programs.

2. Hardware/Software Platform

It is important to understand if the software reuse product has been used on a hardware and/or software platform similar to the one proposed for this system. Software development and integration engineers seem to consistently misjudge the effort and time to rehost software onto a new platform. It is easy to convince oneself that high level similarity between computing environments (e.g., between two POSIX compliant operating systems) will necessarily eliminate porting risk. Often problems arise from the details of a particular implementation. If feasible, it would be useful to consult someone with experience porting software between the computing environments identified in the proposal. Even if the contractor, who originally developed the software, or the vendor, who supplies the OTS product, state they have hosted the software on a compatible platform, plan to contact a Government program manager to verify and learn from that program's experiences.

3. Architecture

Any reused software component will be designed and implemented in accordance with certain assumptions concerning the environment in which it will operate. To be a bit more specific, these assumptions might involve such things as compliance to standards, functional partitioning among components, data and control interfaces, and patterns of interaction with other system components. Collectively, these assumptions can be referred to as characteristics of the architecture. The reused software components will, either explicitly or implicitly, be designed with the particular assumptions concerning the architecture in which they will operate.

The software system that will be created by the contractor will also be designed and implemented according to particular architectural assumptions. When assumptions inherent in the reusable software components are not compatible with the overall system architecture, one can expect problems to appear with performance, integration, test, and (ultimately) maintenance and sustainment. Depending on the compatibility of the particular architecture choices, these problems may be minimal, they may be substantial, or they may be nearly insurmountable. Consequently, it is important to evaluate the architecture early in the program.

Fortunately, there are some well-known *ad hoc* standards for packaging software as reusable components. Examples of such approaches include Microsoft's evolving component architecture (OLE, COM, DCOM, .Net), J2EE's JavaBeans, and OMG's CORBA Component Model (CCM). Each of these imposes a set of architectural restrictions and packaging requirements for the reusable software. In return, the software components can be relatively easily integrated into a compatible framework. Furthermore, while these various

approaches are not 100 percent compatible, there is a body of experience in integrating these somewhat related approaches. Once an offeror has proposed particular architecture choices, the PO should seek an experienced consultant to identify potential pitfalls and risks.

Some architecture approaches can facilitate software reuse even though they have not been developed to the extent that the "well-known" *ad hoc* component architectures have. Such approaches might include the use of a software application framework, a plug-in architecture, or using "design patterns." Such approaches, when used well, can decrease the risk of integrating reusable software into the system. However, such approaches are much less rigorously defined and it is easy for a developer to deviate from the optimal approach—sometimes in subtle ways.

The application of design patterns provides a cautionary example. A "design pattern" is a template for a software design that is well documented, is intended to address a particular set of design issues or goals, and is well understood in terms of how effectively it addresses those issues and goals. (Ease of software component reuse is an example of a goal. Scalability, portability, fault tolerance, and security are other examples.) There has been substantial research and practical application of design patterns. Consequently, there is a substantial body of knowledge with respect to various well-known design patterns—including their strengths, weaknesses, and applicability. Unfortunately, it is difficult to enforce the rigorous application of design patterns. Sometimes a contractor will assert that they are using a well-known and well-regarded design pattern when, in fact, they have deviated from the design pattern in ways that negate potential benefits. (This can happen below the radar screen if the technical management for the contractor's development team is not on top of the design and implementation). Other times, a contractor may identify a design pattern, but it is not a well-known or well-understood design pattern. Unfortunately, while design patterns provide an excellent mechanism for reusing "best of breed" design techniques, it is equally easy to employ mediocre or inferior design patterns.

Even seemingly low-level, detailed assumptions concerning the execution environment can have architectural implications. For example, a contractor was developing software that needed to complete extensive mathematical computations (matrix operations, etc.) within tight real-time constraints. To accomplish this requirement, the contractor planned to reuse a well-known, open source package which provided "high performance mathematical library routines." The contractor's system relied on a highly parallel software design. Unfortunately, the open source package that they planned to reuse was not thread-safe—in other words, it could not be reused as-is in the contractor's design. Although it was not particularly difficult to modify the package to make it thread safe, it was not possible to make the package both thread-safe and, at the same time, retain the required performance characteristics.

The program manager should insist that reusable software components be analyzed for architectural incompatibilities as early as practical in the program. Too often a contractor will select particular software reuse products without considering potential architecture

conflicts. Once these product choices have been made, the contractor may not revisit these decisions and may not begin investigating potential "integration issues" until very late in the program. This delayed investigation can result in substantial rework and a significant schedule impact.

4. Interfaces

Interfaces are a feature of the chosen architecture. However, the developer should pay particular attention to the interfaces between the developed software and the reused software components.

First, the contractor should understand the data interface requirements. What data does the reuse software component provide? Does this data satisfy the requirements of the system being built? In order to operate, what data does the reuse software component require from other parts of the system? Is this data available to pass to the reuse software component? What other parts of the system will be responsible for providing this data? Are the data interactions clearly understood?

Secondly, the interfaces need to be understood from a control point of view. How are the reuse software components invoked? Are they called as a procedure or subroutine? Do they need to poll for data?

Thirdly, the chosen component architecture may impose additional requirements on the reuse software component. For example, is the reuse component responsible for signaling some other part of the system when its output data is available? Does it need to signal based on some other event? Are there "events" (mouse clicks, dialog pop-ups, interrupts) that the software reuse product must react to? Are there events that the reuse product must ignore? Is the software reuse product responsible for passing event notification on to some other part of the system?

Finally, if the architecture compatibility between the system and the reuse software components is not clean, then there may be a need for "wrapper code" to isolate the software reuse component and manage the architecture incompatibilities. Has this "wrapper code" been properly addressed in the software size and effort estimates? Has the impact of this "wrapper code" been adequately addressed from a system functionality or performance perspective?

The PO should insist that an evaluation of the interfaces between the developed software and the software reuse products be included as part of the analysis of the software architecture.

5. Modification

To provide a given functionality, when should a contractor attempt to modify pre-existing software versus decide from a program's start to develop the software from scratch? Unfortunately, we often know the answer to this question only in hindsight. A contractor will propose software reuse, but later discover at some point during the contract that the mismatches in requirements, architecture and/or interfaces are more complex than originally thought. The contractor then needs to modify more code than had been intended to make the required changes or needs to write new code simply because they cannot modify as much pre-existing software as planned.

Modifying software can be challenging. It takes time and effort for the software engineer to understand the code intended for reuse and the developer of the pre-existing software is often not available to answer any questions. The complexity of the modification and/or the extent of modification may be significant. Therefore, it may be easier to start from scratch. There are no hard and fast rules as to when a contractor should pursue a software modification or development approach, but the PO should be wary of moderate to high percentages for the amount of code that will need to be changed.

> "Research studies at … the NASA-Goddard Software Engineering Laboratory have shown that if you have to change more than 15 to 20 percent of a component to make it work in your program, it is more economical to build the component from scratch. And few components meet that 15-to-20 percent threshold." [Glass, Robert L., What's Wrong with Software Reuse? <http://www.stickyminds.com/sitewide.asp?ObjectId=2731&Function=edetail &ObjectType=COL>, 13 August 2001.]

Some rules of thumb are not as pessimistic, but generally they recommend that recoding not exceed 25 to 35 percent. Independent of the exact threshold, it is apparent that in order for software reuse to be cost effective, the percent of the code to be changed should be relatively low.

Software engineers typically do not recommend modifying a COTS product. The problems introduced outweigh the benefits of a product tailored to a specific program, even if the vendor is willing to make the modifications. A PO should avoid having a one-of-a-kind product; it becomes exceedingly difficult for it to keep pace with the changes that are released in the commercially available version, thus negating some of the advantages of pursuing a COTS solution in the first place. In addition, maintenance of a unique product by the vendor may be quite costly.

6. Maturity

Software maturity often turns out to be a significant problem for a PO. Time after time, a PO misjudges or is misinformed about the maturity of the software reuse product. Typically, the problem is that the software has not been fully tested. In one undesirable scenario, the

software needs additional testing and is therefore not available when promised to the program. The program's schedule slips since the contractor either is forced to wait longer for the code or decides to develop the functionality themselves. Alternatively, the PO may need to perform more testing on the immature software than had originally been planned. In another undesirable scenario, perhaps equally bad, the supplier (or offeror) will overstate the comprehensiveness and rigor of the testing that has been completed. In this case, a greater number of undiscovered defects may exist in the software which will then complicate later integration and test activities.

The PO should be careful to understand what levels of testing have been completed for the software. Contractors propose to reuse software that is at various stages of completion; therefore the PO must understand the applicable technology readiness levels (TRLs) and the resulting performance, cost, and schedule risks to the program. Since a Department of Defense (DoD) goal is to stop launching acquisition programs before the technologies are mature, the DoD now requires that the technology in Major Defense Acquisition Programs be demonstrated in a relevant environment (TRL 6 or greater) before Milestone B approval. TRL 6 for a software-intensive system means that the modules and/or subsystems have been validated in a relevant end-to-end environment. Certainly if the software has been qualified through test or demonstration in an operational environment (TRL 8) or the software has been fielded (TRL 9), the PO should be more confident that the software is mature, the software defects have been identified and corrected, and the software will be available as scheduled.

Although the PO might assume that a COTS product would be fully tested, this is not always the case. Commercial vendors are pressured to rush their products to market and therefore may forego comprehensive testing. Fast-changing COTS products with frequent releases may be immature. Therefore it is necessary that the contractor plan for sufficient testing of the COTS products to not only demonstrate the capabilities, but also validate the vendor's claims that the software is working correctly.

7. Availability Timeline

The availability timeline for a software reuse product is a major factor that could contribute to a program schedule slip. Availability should be considered a high risk if the software product is not available, in a usable state, at contract award, even if the contractor insists it will be available in time to meet their modification/integration schedules. The contractor should have the software reuse products to support up-front applicability analyses, internal hands-on demonstrations, and demonstrations to the Government.

Most often, the software reuse product's availability timeline is dependent on another source. The source, whether another contractor, a division within the same contractor's organization, a commercial vendor, or a Government entity, is responsible for the product's delivery schedule. Both the contractor and PO should be guarded about the source's claims related to the availability timeline as well as the product itself when it is delivered. Will the software

be available at contract award? When delivered, will the software be at the maturity level the contractor is anticipating? Has the contractor been able to gain an in-depth understanding of the software and the requirements that the software was originally developed, or is being developed, to meet? Are the requirements for this software changing? It is imperative that the contractor stay informed about the status of the software reuse product; the contractor must be knowledgeable about changes related to applicability, availability timeline, maturity, etc. Therefore, the contractor must develop a plan as to how cross program (and contractor to contractor) relationships will be established and managed.

Unfortunately, when a contractor is dependent on another source for a software reuse product, it becomes more difficult for the contractor to manage and mitigate the risks because the risks are not within their control. Even so, it is important that the contractor take a more proactive approach to software reuse. Rather than let the program's schedule slip for each month that the software reuse product's delivery is delayed, the contractor must develop alternative approaches, including the cost and schedule impacts of pursuing these approaches. The contractor needs to identify in their program schedule when these approaches are to be assessed and decisions made.

8. Reuse History

Reuse history is another factor to consider when evaluating the risks of software reuse. Studies in the commercial sector show that the amount of code that can be successfully reused increases with each sequential reuse attempt. Have other Government programs reused as-is or modified the software product? Was the reuse successful? A PO can be more optimistic about successful reuse on their program if the product had been reused in similar applications and environments. However, the PO should contact the program managers for these Government programs or the program managers currently responsible for the software reuse products for lessons learned. The PO may be able to gain insights from past experience that would streamline the reuse activity and further minimize the risks of integrating the product into the new software system.

9. Developer's Experience with Software

It is desirable that the contractor has experience with the code that is intended to be reused as-is or modified. Preferably the software engineers, who will be modifying and/or integrating the code, were part of the original development and integration teams. If not, it is advantageous for the software developers or integrators to have familiarity with the code and thus have reused, modified, or integrated it for another program. The effort and time to learn code, with which one is unfamiliar, is often underestimated. This learning activity is complicated when no software engineers, who developed the code, are available to answer questions. In fact, even if the new contractor and the original contractor are the same entity, there still may be no one available to field the questions and explain the code. Software is less reusable, or at least more time-consuming to understand, when created by different

teams. The intricacies of the design and the subtleties of the documentation may require that the two groups communicate, even informally.

Likewise, it is helpful for a contractor to have previous experience integrating the specific OTS software product. It is always a challenge to integrate different software entities and make them play together, so prior experience with the software typically makes integration easier.

10. Documentation

The importance of clear, complete, and up-to-date documentation of the software to be reused and/or modified cannot be overstated. The contractor should not be proposing software reuse without good documentation to support the modification, maintenance, and enhancement activities that occur throughout the life cycle. The time and effort to learn, modify, integrate, test, maintain, and enhance code that was poorly documented could easily be far greater than developing the code from scratch.

Documentation refers to both the internal (source code resident) and external (printed) documentation. The internal documentation must be able to provide sufficient and concise insight into the functioning of the code. MITRE's Software Quality Assessment Exercise (SQAE) states the following for the quality factor of self-descriptiveness:

> "Modules should have standard formatted prologue sections. These sections should contain module name, version number, author, date, purpose, inputs, outputs, function, assumptions, limitations and restrictions, accuracy requirements, error recovery procedures, commercial software dependencies, references, and side effects. White space and naming conventions should be used to help the legibility and comprehensibility of the code. The judicious use of comments to highlight special features and to clarify the codes functionality is also desired." [Martin, R. A., S. A. Morrison, 1994, *Managing Software Quality Throughout the LifeCycle,* p. 7.]

For the quality factor of external documentation, the SQAE states that:

> "Both high level functional descriptions and characterizations of the system as well as low level design details are needed to support the maintainer's need to understand the system's functionality as well as identify where to make changes and corrections." [Martin, R. A., 1994]

11. Software Quality

Software quality is a term used to describe how well a software product has been designed and implemented. A "high quality" software product will be easier (and less expensive) to own and maintain. It will be more flexible in accommodating new requirements and easier to move to new computing environments in the future. If needed, the adaptation of high quality software to new situations would be more straightforward. In the context of software

reuse, it would be easier (and less expensive) to integrate “high quality” software reuse components into a system than it would be integrate “low quality” components.

At a summary level, people feel that they understand the concept of software quality but may have difficulty articulating what it means to them. (“I will recognize quality when I see it.”) In a more rigorous sense, software quality is often described in terms of software quality attributes, such as: reliability, usability, efficiency, portability, evolvability, maintainability, descriptiveness, understandability, consistency, testability, and security. While the concept of “software quality” may be vague, a more tractable exercise would be to determine objective assessment criteria against each of the quality attributes.

MITRE has developed the SQAE, a structured and repeatable methodology, for assessing software quality based on analysis of the software’s physical artifacts (source code, documentation, etc.). This methodology has been applied to approximately a hundred Electronic Systems Center (ESC) programs. Other organizations may offer similar services or quality evaluation methods.

The contractor or offeror should evaluate any candidate software reuse products in terms of the software quality. It is in their best interest to do so, since the software quality of the reuse products has a direct bearing on the level of effort entailed in integration and test. However, software quality has an even greater impact on “cost of ownership” issues, such as maintenance, reliability, flexibility, adapting to new situations, etc. Consequently, it is the customer who has an even larger stake in ensuring that an adequate software quality evaluation is performed. The PO should insist that a comprehensive and thorough evaluation be accomplished.

12. Software Defects

More often than not, when an offeror proposes software to be reused or modified, little information is provided to the Government regarding the number of defects or severity of the defects in this software. Obviously the PO does not want to integrate defect-laden software into their system, so it is necessary to understand the nature and severity of the defects, when these defects are planned to be fixed, and by whom. Without significant knowledge about these defects, the offeror cannot develop a realistic schedule or effort estimate for their software development and test activities. Additionally, without this knowledge, the PO cannot evaluate, with any confidence, the offeror’s schedule or effort estimates.

Depending upon the maturity of the software reuse product, the defects may not have even been discovered yet. Thus the defect information provided to the PO may not provide adequate insight. Since software defects are detected as the software progresses through integration and test, it is necessary to evaluate the defect information in the context of software maturity.

One might expect COTS products to be defect-free, but this is seldom the case. COTS vendors rush products to market, so testing is often not as thorough as needed.

13. Relationships with Sources of Software Reuse Products

As previously mentioned, when a contractor is dependent on another source for a software reuse product, there is increased uncertainty about that product's availability timeline, functionality, quality, maturity, etc. To manage and mitigate the risks of that dependency, it is imperative that the contractor establish and maintain a good working relationship with the reuse product's source. The contractor should provide a plan to the PO as to how cross program (and contractor to contractor) relationships will be managed. The contractor should collaborate with these sources (or ideally have personnel embedded within these organizations) so that the contractor stays knowledgeable about product changes and potential slips in availability.

It is equally important to establish good working relationships with COTS vendors. The contractor and PO should stay informed about planned changes to the COTS product, long-term strategic visions for product changes, schedules for product releases, and changes in licensing and/or maintenance strategies.

Even if the contractor has established good relationships with the sources of the software reuse products, the PO should not rely on information solely provided by the contractor about these products. The PO should contact directly the sources to verify updated product information. The PO should be proactive about anticipating problems that will in turn affect their program's performance, cost, and/or schedule.

14. Software Reuse Schedule

The PO must often make a subjective judgment of the reasonableness of the contractor's planned schedule. Whether early in the acquisition process, during source selection, or ACA, the PO must assess the realism of the software reuse schedule since it is often a schedule driver. Actual data for analogous software developments are most often not readily available, and the schedule impact of reusing as-is or modifying software complicates any comparisons. So the PO must assess the schedule based on their understanding of the requirements and the scope of the work to be completed, the sizing associated with the reused as-is and/or modified software, and the productivity rate. Given the information provided, do the durations for the development, integration, and test schedules appear to be realistic? Do these schedules reflect up-front tasking so that problems can be detected early and resolved? Have decision points for the alternative strategies been clearly identified?

15. Software Reuse Sizing

The PO must also make a subjective judgment of the reasonableness of the contractor's sizing estimates. Similar to schedule, the PO must assess sizing based on their understanding of the requirements and the scope of the work to be completed. Does the effective sizing estimate seem realistic given the work to be performed? Does the sizing estimate reflect appropriate assessments for the percent redesign, reimplementation (i.e., recode) and retest?

2 Software Reuse Management

Notes:

1. *The words below provide new software reuse-specific sections in the PO RFP documents.*
2. *The words for Section H or K must be added to the licensing clauses in the contract.*
3. *Worksheet Questions are to be filled out for every software reuse product, and are part of the plan for software reuse management. The PO may choose to exempt the completion of the worksheets for small, low risk COTS products.*
4. *Worksheet Questions should be first provided by the offerors as part of their proposal; worksheets should be exempt from the proposal page count limits. Worksheets will be updated ACA as part of the Reuse Management Report (ReMR).*
5. *The Worksheet Questions for Reused As-is/Modified Software is provided in Appendix A; the Worksheet Questions for COTS/GOTS Software is provided in Appendix B.*
6. *The data item description (DID) for the ReMR is provided in Appendix C.*
7. *The SOW requires the contractor to inform the Government immediately of all substantial changes to the information contained in the worksheets. The PO should define "substantial" in the context of their program and include this clarification in the SOW.*

Section H or K

In addition to the licensing clauses under DFARS 227.72, the following terms and conditions shall apply for any and all third party commercial off-the-shelf (COTS) software obtained by the Contractor and intended to be transferred to the U.S. Government during performance of the present Government contract (Contract No. _____). The Government will not accept or execute a DD Form 250 for the software deliverables under the present contract until the Contractor obtains agreement to the terms described below from any and all third party COTS software suppliers and/or vendors for which the Contractor has licensed software for incorporation into deliverables to the Government:

1. Any license shall be perpetual in nature and may not be unilaterally terminated by the Licensor. The Licensor may, however, seek other remedies at law.
2. The Licensee shall not be restricted from copying or embedding elements of accessible code into other applications (e.g., nesting code, derivative works).
3. The Licensor shall not include any indemnification clauses.
4. The Licensor shall not use the fact that the Licensee is using the Licensor's products in any notification to the public (e.g., no publicity rights permitted).
5. The Licensee is a Federal entity governed by Federal Statutes, Case Law, and Federal Regulations. Therefore, the Licensor shall remove any references to

binding the Licensee through any laws of any municipality, state, or foreign country.

6. The Licensor shall not include any clauses indicating a right to enter the premise of the Licensee for the purpose of auditing the use of any license, as the Licensee cannot allow an auditor physical access to the Licensee's facility due to security concerns. The Licensor may submit to the Licensee written notice indicating a substantiated belief that the Licensee is not using the software within the terms described in the license and the Licensee may consider conducting its own internal audit and providing a certified statement of its findings to the Licensor.
7. The Licensor shall not use any integration clauses.
8. The Licensor shall not use any injunctive relief clauses as the Licensor cannot prevent the Licensee from performing mission operations. The Licensor may seek other remedies at law (e.g., monetary damages).
9. The Licensor shall include the following clause (and no other) for disputes: "Since the Licensee is a Government entity, any dispute arising from or in connection with this agreement shall be subject to resolution by the Disputes Clause included in the basic contract and/or the Government may also consider resolving any disputes using an appropriate Alternate Dispute Resolution (ADR) remedy."
10. If the Licensor will not agree to the terms and conditions cited herein and/or as contained in DFARS 227.72, the Licensor shall retain the current license on behalf of the U.S. Government.
11. The Licensor shall add the clause described below to all third party COTS software licenses intended to be transferred to the Government:

    The Government agrees to the provisions of the present Software License, as set forth above, to the extent that the provisions of the Software License are consistent with Federal procurement law(s) and satisfies the Government's needs, as prescribed at least by the Defense Federal Acquisition Regulation Supplement (DFARS) Section 227.7202-1. In the event that any of the provisions of the present Software License are determined to be inconsistent with Federal procurement law(s) and/or do not otherwise satisfy the Government's needs, the parties to the present Software License hereby agree that such provisions shall be null and void."

Section L

Mission Capability and Proposal Risk Volume

The Offeror shall:

- Provide the technical approach for each software reuse product, including its proposed use within the system, approach to integration, challenges to integration, user interface integration, and usability.

Integrated Program Management and Systems Engineering Processes Volume

The Offeror shall provide a draft Reuse Management Report to include:

- An executive summary that identifies and briefly describes all software products that will be reused as-is, including COTS/GOTS, or modified (existing software requiring change).
- Populated Worksheet Questions for each reused as-is/modified and COTS/GOTS software product, in accordance with the embedded instructions.

In accordance with DFARS 227.70, DFARS 227.71, DFARS 227.72, the Offeror shall:

- Provide the license agreement information for all commercial software licenses (as known at the time of the proposal) to be obtained on behalf of or transferred to the U.S. Government under this contract.

Section M

The Government will evaluate the technical soundness, applicability, and achievability of the proposed software reuse.

The Government will evaluate the achievability in terms of effort and schedule of the proposed approach for software reuse. This evaluation will include the approach for assessing, acquiring, documenting, and maintaining reused as-is/modified and COTS/GOTS software products.

The Government will evaluate the acceptability of the licensing agreement information for all commercial software licenses (as known at the time of the proposal) to be obtained on behalf of or transferred to the U.S. Government under this contract.

SOW

Program Management

Software Reuse Management

The Contractor shall:

- Execute and manage software reuse activities, in accordance with the Integrated Master Schedule (IMS) and the software reuse plan provided in the Software Development Plan (SDP), and report on the status of these activities.
- Inform the Government immediately if a decision is made to implement an alternative approach, if any reuse cannot be accomplished as planned.
- Inform the Government immediately of all substantial changes to the information contained in the worksheets (especially in the areas of applicability and requirements traceability to the TRD; extent of modification and effective size; product maturity; status of the availability timeline; dependencies on other programs and the current status of these programs; availability of software

documentation; vendor viability; and updated status of critical defects) for all the software reuse products.

- Prepare, update and provide a Reuse Management Report (DI-MGMT-81650, DI-IPSC-81427A/T, DI-SESS-81771).

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Integrated Master Schedule

Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-81650

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Software Development Plan

Block 4. AUTHORITY (Data Acquisition Document No.): DI-IPSC-81427A/T

Block 16. REMARKS

Block 4 tailored as follows:

Contractor format acceptable

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Reuse Management Report

Block 4. AUTHORITY (Data Acquisition Document No.): DI-SESS-81771

3 Risk *[and Opportunity]* Management

Notes:

1. *It is the decision of the PO whether to require a Risk Management Plan (RMP) or Risk and Opportunity Management Plan (ROMP).*
2. *The words below provide software reuse-specific additions to the risk [and opportunity] management sections in the PO RFP documents.*
3. *The PO should require the offeror to specifically address software reuse risks [in the RMP, ROMP, or in the body of the proposal].*

Section L

The Offeror shall:

- Identify the technical, cost, and schedule risks associated with the specific reused as-is/modified and COTS/GOTS software products.
- *[Identify any software reuse opportunities that might offer future benefit to the program from a technical, cost or schedule standpoint.]*
- Discuss risk mitigation strategies for any software reuse risks identified.
- Describe alternative approaches for any reused as-is/modified and COTS/GOTS software products that are considered high or moderate risk.
- Describe alternative approaches for any reused as-is/modified and COTS/GOTS software products that are not available (fully documented and tested) at the time of the proposal.
- Estimate the technical and programmatic (i.e., effort and schedule) impacts of implementing alternative approaches.
- Identify in the IMS when each alternative approach would have to be implemented in the event that the planned software reuse products are not available in time to preserve the program schedule.

Section M

The Government will evaluate the risks of the planned software reuse activities, the risk mitigation strategies, and the potential technical and programmatic impacts that would result from not being able to reuse the software as planned. The Government will evaluate alternative approaches for any reused as-is/modified and COTS/GOTS software products that are considered high or moderate risk. The Government will evaluate alternative approaches identified for any reused as-is/modified and COTS/GOTS software product that is not available (fully documented and tested) at contract award. The efficacy of the proposed decision points for implementing alternative approaches will also be evaluated.

SOW

Program Management

Risk *[and Opportunity]* Management

The Contractor shall:

- Identify, track, manage, mitigate and report the technical, cost and schedule risks associated with software reuse.
- *[Identify any software reuse opportunities that might offer future benefit to the program from a technical, cost, or schedule standpoint.]*
- Identify and monitor risk mitigation strategies associated with the specific reused as-is/modified and COTS/GOTS software products.
- Monitor the status and viability of alternative approaches identified for software reuse and modify approaches to minimize the impact to the program.
- Update the decision points in the IMS, as needed. (DI-SESS-81771, DI-MGMT-81650).

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Reuse Management Report

Block 4. AUTHORITY (Data Acquisition Document No.): DI-SESS-81771

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Integrated Master Schedule

Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-81650

4 Software Reuse Demonstration(s)

Notes:

1. *The words below provide new software reuse-specific sections in the PO RFP documents.*
2. *The PO can choose to conduct a Software Reuse Demonstration either during source selection, or ACA, or both. The purpose of the source selection demonstration is to provide the source selection team with an assessment of the maturity of and availability timeline for the proposed software reuse products. The purpose of the ACA demonstration is to provide the PO early insight into the maturity, functionality, and performance of the software reuse products. The PO can decide to have more than one ACA demonstration.*
3. *The PO's decision to conduct a demonstration during source selection should depend on the expected maturity of the offerors' proposed software to be reused as-is/modified.*
4. *Demonstration(s) should be required for any <u>significant</u> reused as-is/modified or COTS/GOTS software product. The PO may choose to exempt small, low risk COTS products from the demonstrations.*
5. *Offerors should be required to provide demonstrations during source selection if they are bidding software reuse; if an offeror has no software reuse, no demo is required and the absence of a demo cannot be held against them. To ensure full and fair assessment during source selection, the PO must fully evaluate the overall software approach, including all new software development, bid by the offerors.*
6. *In addition to witnessing the demonstrations, the Government should obtain information about each software reuse product by contacting the program manager, who is currently responsible for the reused as-is/modified or COTS/GOTS software product, or the responsible entity or source of the software, if the Government is not responsible for it.*

Section L

The Offeror shall demonstrate to the Government the proposed software reuse products and include in the demonstration:

- Both (1) the products that are proposed to be reused as-is and (2) the products that are proposed to be modified (the demonstration will show the capabilities of these products prior to their modification).
- Execution in a computing environment that is as similar as practical to the intended target environment.
- Illustrations of key functions that the software reuse products are planned to provide for the delivered system.
- Highlights of specific features or characteristics that were factors in their choice of the product as a reuse candidate.

In addition, the Offeror shall:

- Identify the demonstrated products by name, supplier, and version. Include definition of the specific configuration of the computing environment that supports the demonstration.
- Identify any proposed software reuse products that are not part of the Software Reuse Demonstration, with specific rationale explaining why such products have not been included.

Section M

The Government will observe and evaluate the demonstration in terms of the availability of the proposed products and appropriateness of the proposed products for their intended use. Lack of demonstration of a proposed software reuse product will be considered as part of the Government assessment of risk of that reuse.

SOW

Software Engineering

Software Reuse Demonstration(s)

The Contractor shall demonstrate the software reuse products to the Government to include:

- Both (1) the products that will be reused as-is and (2) the products that will be modified (the demonstration(s) will show the capabilities of these products prior to their modification).
- Demonstration of the maturity, functionality, and performance of the software reuse products.
- Illustration of all functions that these products will provide for the delivered system.
- Execution in a computing environment that is as similar as practical (e.g., operationally representative) to the intended target environment.

In addition, the Contractor shall:

- Identify in the IMS the schedule for the Software Reuse Demonstration(s), which should be conducted prior to the Preliminary Design Review (PDR).
- Have the Software Quality Assurance (SQA) organization document and certify as correct the software and hardware configuration of the Software Reuse Demonstration(s) and witness the Software Reuse Demonstration(s).
- Provide an agenda for the Software Reuse Demonstration(s) identifying objectives for the demonstration(s).
- Provide a report documenting the minutes of the Software Reuse Demonstration(s) that includes an identification of the products and functions that were demonstrated; the software and hardware configurations; a description of the

demonstration(s) as conducted, including a description of inputs and resulting outputs; all non-conformances, deviations, anomalies and notable observations that occurred in the demonstration(s); and all presentation material. (DI-MGMT-81650, DI-ADMN-81249A/T, DI-ADMN-81505/T)

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Integrated Master Schedule

Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-81650

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Conference Agenda

Block 3. SUBTITLE: Software Reuse Demonstration(s) Agenda

Block 4. AUTHORITY (Data Acquisition Document No.): DI-ADMN-81249A/T

Block 16. REMARKS

Block 4 tailored as follows:

Only paragraphs 10.2 a, b, c, i and l apply

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Report, Record of Meetings/Minutes

Block 3. SUBTITLE: Software Reuse Demonstration(s) Report

Block 4. AUTHORITY (Data Acquisition Document No.): DI-ADMN-81505/T

Block 16. REMARKS

Block 4 tailored as follows:

Delete paragraph 10.2.1.3 b

5 Software Quality Assessment(s)

Notes:

1. *The words below provide new software reuse-specific sections in the PO RFP documents.*
2. *It is recommended that the PO conduct a Software Quality Assessment Exercise (SQAE) ACA to gain insight into the potential risks associated with the software reuse products. Additional information about the SQAE may be found at http://www.mitre.org/work/tech_transfer/technologies/sqae.html.*
3. *It is especially recommended that an SQAE be conducted for any software products not developed in accordance with known development standards (e.g., code developed for experimentation, internal research and development (IR&D)).*
4. *If an SQAE is to be conducted ACA, words for Section H or K should be added to the contract.*

Section H or K

At its sole discretion, the Government will conduct one SQAE per reused as-is/modified software product. The Government will provide the Contractor with 30 days notice of an SQAE**.**

Section L

The Offeror shall:

- Describe their assessment of software quality for the software products that are proposed as a basis for reuse (either reused as-is/modified or COTS/GOTS software products).
- Describe the processes and tools used in the assessment and the results of the assessment.

Section M

The Government will evaluate the Offeror's choice of software products that are proposed for reuse based on the following software quality attributes: reliability, usability, efficiency, portability, evolvability, maintainability, descriptiveness, understandability, consistency, testability, and software security/vulnerability.

SOW

Software Engineering

Software Quality Assessment Exercise

The Government will conduct SQAE(s) for all reused as-is/modified software products. The purpose of an SQAE is to assess the status of the software for consistency,

independence, modularity, documentation, self descriptiveness, anomaly control, and design simplicity. The Contractor shall support this Government SQAE as follows:

- Submit all relevant source code and current documentation for the reused as-is/modified software products, for which an SQAE will be conducted.
- Provide current information on the status of each system under assessment, as follows:
  - Application domain (e.g., C2, Financial Management, Simulation)
  - Product type (e.g., database management system, graphical user interface (GUI), client/server)
  - Estimated software size
  - Software languages and tools used in development
  - Target hardware and operating system
  - Applicable standards
  - Host development system
  - Classification, proprietary, or source selection sensitive (unclassified desirable)
- Provide access to the Government to any software applications necessary to access (read) the documentation for the duration of the SQAE.
- Support a meeting per each reused as-is/modified product to discuss the conduct of the SQAE and the information needed by the Government to perform the assessment.
- Respond to any Government questions about the software and information provided. (DI-IPSC-81488/T)

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Computer Software Product

Block 3. SUBTITLE: Software Quality Assessment Exercise

Block 4. AUTHORITY (Data Acquisition Document No.): DI-IPSC-81488/T

Block 16. REMARKS

Block 4 tailored as follows:

The completeness of the information provided has a direct bearing on how well the SQAE process may be conducted. Information to support the SQAE shall be submitted as follows:

- The complete source code will be delivered exclusively in electronic form and consist solely of ASCII (ANSI X3.4-1986) or Unicode (ISO/IEC 10646:2003) source files. All unclassified source code, package specifications/header files, makefiles, shell scripts, templates, readme files, etc., should be included in the delivery. Additionally, where system libraries, third party bindings, commercial off-the-shelf (COTS) application program intefaces (APIs), etc.,

have been utilized, the ASCII/Unicode header files and specification (but not the actual binary archives) should be included. All items in the delivery must be referenced via relative path names.

- Software source code in this context should include any and all human-generated and/or modified machine-generated source code (including scripts, files, templates, macro source, package specifications/header files, makefiles, etc.). Machine-generated source code should be delivered only if such code is intended (wholly or in part) to be modified or otherwise maintained through human inspection and modification (requiring a human to be able to read and reasonably comprehend) it. Inputs to the machine-generation of GUI code should also be delivered in appropriate form, if that input is to be used as the point of maintenance. Machine-generated source need not be delivered if that code that will only be processed by tools in the development environment, and will not be subject to any modification or processing by humans.
- As complete a set of the current documentation, as possible, will be delivered. Several of the Exercise questions are concerned with how well the system is documented for future maintainability, so it is important that the documentation be available, including all change pages in effect, as well as a thorough bibliography of every relevant document and reference, whether available or not. Any missing documentation should have a brief description of its essence and relevancy. The types of documentation to be provided are:
  - All the design, implementation, and release documents, such as plans, function allocations, specifications (above the implementation level, unless specifying coding standards)
  - Design documents
  - Interface (internal, external, user, hardware, and software) definitions (above the detailed level)
  - Style guides (programming, as well as user interface)
  - User manuals (sometimes contain useful design information)
  - Data and control flow diagrams (above the procedure level)
  - COTS manuals (of products that the code is heavily dependent on)
  - Code generation (e.g., GUI Builders) documentation, and
  - Code libraries descriptions (if utilized)
- If available, the following system documentation, which supplies insight into the following areas, should be delivered:
  - Coding standards, input/output (I/O) and exception handling policies, naming conventions, and documentation standards
  - High level description of the project’s architecture, functional summaries of the system’s major tasking threads, the system's control flow and data flows

- Documentation that includes plain language descriptions and justifications of any complex algorithms, self-modifying code, recursive code, or other non-standard programming practices

- In the case of other systems which may be running on platforms with the capability to support parallel processing, any additional documentation and software which would emphasize and otherwise explain any differences from uni-processing operations and help the assessment processes will be delivered.
- The physical media must be accompanied by a hard copy listing detailing the contents of the archive, the particular archive format used, restoration/ installation instructions and any assumptions made about the host configuration.

Note that implementation-based, detailed documentation such as PDL, procedure (or lower) level flow charts, and set-use diagrams are not required or desired. Object (executable) code is neither necessary nor wanted for the assessment.

Delete paragraphs 10.4.3, 10.4.4 and 10.5.

Software and related information provided via CDs, DVDs, zip files or web access, or other format as mutually agreed upon.

6 Software Size

Notes:

1. *In order to better understand the sizing associated with the reused as-is/modified and COTS/GOTS software products, it is necessary to assess the size of these software products in the context of the size of all software (new, modified and reused) being proposed. Format M-1 (Revised) not only provides insight into the sizing of the new software, but it importantly includes the needed detail into the factors associated with reuse as-is and/or modification.*
2. *Format M-1 (Revised) and the instructions for filling out the format are provided in Appendix D.*

Section L

The Offeror shall:

- Present estimates of software size in either source lines of code (SLOC) or unadjusted function points, for each Computer Software Configuration Item (CSCI) or, if known, for each Computer Software Component (CSC). If the Offeror does not use SLOC or function points as measures of software size, an equivalent table using the applicable measure to describe size should be provided.
- Describe their method for estimating software size (e.g., developer/expert opinion, previous development experience, analysis of required functions, interfaces, code blocks, other method, or a combination of methods). The description should identify the information that is collected as a basis for the estimate, and how that information is analyzed in the development of the estimate.
- Complete Format M-1 (Revised) based on the instructions included with the sample format. The instructions provide standard definitions for software size, nature of the code (new, existing, deleted, modified, reused as-is/lifted, etc.), productivity, schedule phase, etc. Any non-standard definitions used in the estimating process should be included.
- Provide the tables electronically in Excel such that all formulas are able to be audited by the Government.

Section M

The technical soundness of the Offeror's estimating methodologies and sizing estimates, including new, reused as-is, and modified code, will be evaluated.

7 Software Development Plan

Notes:

1. *The words below provide software reuse-specific additions to the Software Development Plan (SDP) sections in the PO RFP documents.*
2. *The current versions of any required standards should be listed in the RFP.*
3. *It is recommended that the PO require the contractor to adhere to IEEE Std 1517 (IEEE Standard for Information Technology-Software Life Cycle Processes-Reuse Processes) if the software is being developed in accordance with IEEE/EIA Std 12207.0 (International Standard ISO/IEC 12207:2008 Systems and software engineering—Software life-cycle processes).*

Section L

In a draft SDP, the Offeror shall:

- Provide a well thought-out software reuse approach identifying specific engineering and management tasks, and a corresponding schedule of these tasks for each software reuse product. Completion of these tasks should be linked to program milestones, such as program and design reviews.
- Describe when and how the reused as-is/modified and COTS/GOTS software products will be introduced into the build plan.
- Reference the software development standards to which the Contractor will adhere, including standards for reuse processes.
- Describe the software development processes that have been tailored for reuse as well as any new reuse processes that have been implemented. In particular, the Offeror should discuss how the software products will be evaluated for applicability for this program and how the Offeror plans to collaborate with the sources of the software products to stay informed about the products' applicability, maturity, availability timeline, etc.
- Identify metrics that will status and track the reused as-is/modified and COTS/GOTS software products during the performance of the contract. The proposed metrics, which should be reflective of the Offeror's overall approach to software reuse, should report planned versus actual performance, including trigger points for when alternative approaches should be implemented. Quality metrics should also be included. Metrics should also include the types of measures required for the Software Metrics Report. Brief descriptions and notional examples of each metric should be provided as part of the proposal.

Section M

The Government will evaluate the proposed software reuse approach, and the associated software development standards, software development processes, and software metrics as they pertain to reused as-is/modified and COTS/GOTS software products.

SOW

Software Engineering

Software Development Plan

The Contractor shall:

- Update, as needed, the draft SDP submitted with the proposal.
- Implement and manage an SDP.
- Implement development processes for all software (new, modified, and reused as-is) in accordance with *[the standards specified by the PO and/or contractor-identified standards, and]* standard organizational practices.
- Ensure all reuse software (e.g., other programs, IR&D, demo) with the exception of COTS/GOTS (reused as-is), is in accordance with the standards *[specified by the PO and/or identified by the contractor]*.

In the SDP, the Contractor shall:

- Provide the software reuse management approach, including a plan of detailed engineering and management tasks for each software reuse product and a corresponding schedule of these tasks.
- Identify and describe the software metrics that will be reported during the performance of the contract in the Software Metrics Report, with particular attention paid to metrics pertaining to software reuse.
- Describe the software development processes that have been tailored for reuse as well as any new reuse processes that have been implemented, with particular attention paid to how the software products are evaluated for applicability for this program and how the Contractor plans to collaborate with the sources of the software products to stay informed about the products' applicability, maturity, availability, timeline, etc.
- Reference any software development standards to which the Contractor will adhere. (DI-IPSC-81427A/T)

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Software Development Plan

Block 4. AUTHORITY (Data Acquisition Document No.): DI-IPSC-81427A/T

Block 16. REMARKS

Block 4 tailored as follows:

Contractor format acceptable

8 Software Metrics

Notes:

1. *The words below provide software reuse-specific additions to the software metrics sections in the PO RFP documents.*

SOW

Systems Engineering

Systems Engineering Processes

The Contractor shall:

- Track and report metrics pertaining to software reuse.
- Provide metrics for the reused as-is/modified and COTS/GOTS software products, including, but not be limited to, sizing, effective sizing, effort by phase, progress by phase, staffing, requirements assigned to reuse, requirements stability, product stability, components integrated, defect discovery and repair rates, and interface definition and implementation as well as any other metric identified by the Contractor in the Software Development Plan. (DI-MGMT-80227/T)

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Contractor's Progress, Status and Management Report

Block 3. SUBTITLE: Software Metrics Report

Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-80227/T

Block 16. REMARKS

Block 4 tailored as follows:

Only paragraphs 10.3 a, b, c, o, and p apply

9 Past Reuse Performance

Notes:

1. *The words below provide software reuse-specific additions to the past performance sections in the PO RFP documents.*
2. *Relevancy criteria need to be tailored by the Performance Confidence Assessment Group (PCAG) to reflect the expected type of software reuse [e.g., software reuse as-is, modification of pre-existing software, use of COTS/GOTS software].*
3. *The PCAG will determine what constitutes significant past contracts and how relevant these past efforts are.*

Section L

Past Performance Questionnaire

(#) Did the Contractor reuse as-is/modify software developed for other programs or from other sources?

- If yes, how well did the Contractor assess the applicability of the performance and functions of the reused as-is/modified software products?
- Did the Contractor's final software baseline use the software products that were originally proposed?
- Did any problems encountered with software reuse result in code growth? How much?
- How well did the Contractor manage the risks associated software reuse?
- Were there any software reuse risks that required Government involvement?

(#) Add the following relevancy criteria:

a. Past contracts with significant *[e.g., software reuse as-is, modification of pre-existing software, use of COTS/GOTS software]*

Section M

Past Performance Factor

Add the following relevancy criteria:

a. Past contracts with significant *[e.g., software reuse as-is, modification of pre-existing software, use of COTS/GOTS software]*

10 Integrated Master Schedule

Notes:

1. *The words below provide software reuse-specific additions to the Integrated Master Schedule (IMS) sections in the PO RFP documents.*

Section L

In the IMS, the Offeror shall:

- Schedule key engineering and management tasks that are associated with the reused as-is/modified and COTS/GOTS software products.
- Identify all interactions with the source organization for each software reuse product, including intermediate and final deliveries for the product and its supporting artifacts (e.g., design specifications).
- Identify tasks that provide the Government with sufficient insight into a credible plan for accomplishing the software reuse activities.
- Describe how the software reuse tasks fit into the program schedule and link the completion of these tasks to program milestones, such as design and program reviews.
- Provide all critical dates pertaining to software reuse that could affect program success, including, but not limited to, the delivery schedules for the reused as-is/modified and COTS/GOTS software products, the Contractor demonstration(s) for the reused as-is/modified and COTS/GOTS software products, and decision points for when alternative approaches need to be implemented.

Section M

The Government will evaluate the comprehensiveness and reasonableness of the scheduling of the tasks associated with all reused as-is/modified and COTS/GOTS software products.

SOW

Program Management

Integrated Master Schedule

In the IMS, the Contractor shall:

- Provide key engineering and management tasks associated with the reused as-is/modified and COTS/GOTS software products, and link the completion of these tasks to program milestones, such as design and program reviews.
- Identify all critical dates pertaining to software reuse, including, but not limited to, the delivery schedules for the reused as-is/modified and COTS/GOTS software products, all interactions with the source organization for each software reuse product, the Contractor’s demonstration for the reused as-is/modified and

COTS/GOTS software products, and all decision points for when alternative strategies need to be implemented. (DI-MGMT-81650)

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Integrated Master Schedule

a. Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-81650

11 Integrated Master Plan

Notes:

1. *The words below provide software reuse-specific additions to the Integrated Master Plan (IMP) sections in the PO RFP documents.*
2. *If Integrated Product Teams (IPTs) will be used, it is recommended that the PO extend the communications approach to include a description of formal and informal methods of communications within and across the IPTs.*

Section L

In the IMP, the Offeror shall:

- Develop a communications approach to facilitate the timely exchange of management, technical, and risk information related to the reused as-is/modified and COTS/GOTS software products.
- Describe all formal and informal methods of communications with all internal and external stakeholders *[as well as within and across the IPTs]*.

Section M

The Government will evaluate the comprehensiveness of the communications approach and its ability to facilitate the timely exchange of management, technical, and risk information related to the reused as-is/modified and COTS/GOTS software products.

12 Contract Work Breakdown Structure

Notes:

1. *The words below provide software reuse-specific additions to the Contract Work Breakdown Structure (CWBS) sections in the PO RFP documents.*
2. *The DID for DI-MGMT-81334C/T states that routine reporting shall be at CWBS level 3 for prime contractors. Extensions of the CWBS can be tailored to the specific program, but will be consistent with MIL-HDBK-881, current edition. More detailed reporting of the CWBS will be required only for those elements that address high-risk, high-value, or high-technical-interest areas of a program.*

Section L

The Offeror shall extend the CWBS to include detailed engineering and management activities associated with all reused as-is/modified and COTS/GOTS software products.

Section M

The Government will evaluate the comprehensiveness of the engineering and management activities identified in the CWBS for all reused as-is/modified and COTS/GOTS software products.

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Contract Work Breakdown Structure

Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-81334C/T

Block 16. REMARKS

Block 4 tailored as follows:

Report the CWBS at level 3, except for report the CWBS to the lowest level of tasks that will provide the Government with visibility into the engineering and management activities associated with all reused as-is/modified and COTS/GOTS software products.

13 Contract Performance Report

Notes:

1. *The words below provide software reuse-specific additions to the Contract Performance Report (CPR) sections in the PO RFP documents.*
2. *The level of detail to be reported in Format 1 normally will be at level 3 of the CWBS, but lower levels may be specified for high risk or high cost items. The words below ask for detailed reporting against all software reuse products; however, it is recommended that the PO identify specific high risk or high cost products for which detailed reporting is required.*

CDRL (Form 1423)

Block 2. TITLE OF DATA ITEM: Contract Performance Report

Block 4. AUTHORITY (Data Acquisition Document No.): DI-MGMT-81466A/T

Block 16. REMARKS

Block 4 tailored as follows:

Report data on Format 1 to level 3 of the CWBS, except for report to the lowest level of tasks that will provide the Government with visibility into the engineering and management activities associated with all reused as-is/modified and COTS/GOTS software products.

14 Statement of Objectives

Notes:

1. *If a PO is not developing a Statement of Work (SOW) as part of the RFP, the PO should incorporate the following objective for software reuse in the Statement of Objectives (SOO).*
2. *The PO should also attempt to incorporate as many of the SOW requirements from sections 2 through 13 as possible, into other program products/deliverables (e.g., IMP, SDP, System Engineering Management Plan (SEMP), IMS, CPR).*

For any software reuse, it is a Government objective to have a comprehensive reuse strategy that identifies, tracks, manages, mitigates, and reports the technical, cost, and schedule risks associated with such reuse, providing early visibility into the availability timeline, maturity, quality, functionality, and performance of the reuse products. The strategy must include alternative approaches to be implemented in the event that software reuse cannot be implemented as planned. The Contractor will be expected to assume responsibility for the technical performance of all reused as-is/modified software products, except for COTS/GOTS software products that will be reused as-is and supported by the vendor/Government.

15 Request for Information

Notes:

1. *The words below provide software reuse-specific additions to a Request for Information (RFI).*
2. *High level information about potential software reuse products should be gathered when conducting market research.*

[The USAF] invites industry to respond with information about any software reuse product being considered *[for the program]*, including the source, applicability, extent of modification (if any), availability timeline, maturity, and reuse history.

16 Award Fee Plan

Notes:

1. *An AFPEO/C2&CS Memorandum states that:*
   "Award Fee type contracts should only be used when it is neither feasible nor effective to devise predetermined objective incentive targets applicable to cost, technical, performance or schedule. … Our priority must be to utilize objective incentive criteria, whenever possible, across all programs in the AFPEO/C2&CS portfolio." [AFPEO/C2&CS, "AFPEO/C2&CS Award Fee Contracting Policy," 12 October 2007.]
2. *Objective incentive criteria, in general, are not easily defined for software reuse products. However, POs may be able to define objective incentive criteria for their specific program.*
3. *If subjective Award Fee Plan (AFP) criteria are used, the PO should select a few of the following software reuse criteria most appropriate to the program circumstances.*
4. *These criteria may also be considered for use by an on-going program that uses AFPs.*
5. *The AFP criteria vary with each phase of the program and serve to incentivize desirable contractor behaviors in the areas of highest risk. The criteria cover a wide range of possible behaviors to be incentivized over the various periods of performance.*
6. *The criteria for the satisfactory category must demonstrate that the contractor has met the basic (minimum essential) requirements of the SOO, SOW, and other contract documents.*
7. *Software reuse focus for the AFP is provided for the Schedule, Technical, and Program Management areas. No software reuse-specific Cost area evaluation criterion has been generated; it is recommended that cost evaluation criteria address the total program.*

Schedule				
Outstanding	Excellent	Good	Satisfactory	Unsatisfactory
The contractor has incorporated into the IMS all tasks associated with all software reuse products. These tasks are scheduled to start as soon as software reuse products and their artifacts (e.g., design specifications) are available to support them. The contractor has clearly identified when each software reuse product and its supporting artifacts (e.g., design specifications) will be obtained from its source and how they feed into the program activities. Intermediate and final deliveries have been identified. The schedule identifies all interactions with the source organization for each software reuse product. The schedule includes activities to accelerate the availability timeline of the software products and their artifacts.	The contractor has incorporated into the IMS all major tasks associated with all software reuse products. The contractor has clearly identified when each software reuse product and its supporting artifacts (e.g., design specifications) will be obtained from its source and how they feed into the program activities. Intermediate and final deliveries have been identified. The schedule identifies all interactions with the source organization for each software reuse product.	The contractor has incorporated into the IMS most major tasks associated with all significant software reuse products. The contractor has identified when each software reuse product and its supporting artifacts (e.g., design specifications) will be obtained from its source and how they feed into the program activities. Intermediate and final deliveries have been identified. The schedule identifies all critical interactions with the source organization for each software reuse product.	The contractor has incorporated into the IMS some major tasks associated with all significant software reuse products. The contractor has identified when most software reuse products and their supporting artifacts (e.g., design specifications) will be obtained from their source and how they feed into the program activities. Intermediate and final deliveries have been identified. The schedule identifies most critical interactions with the source organization for each software reuse product.	The contractor has incorporated into the IMS few tasks associated with the significant software reuse products. The contractor has identified when some of the software reuse products and their supporting artifacts (e.g., design specifications) will be obtained from their source and how they feed into the program activities. Intermediate and final deliveries have been identified. The schedule identifies few critical interactions with the source organization for each software reuse product.

Demonstrations of software reuse products are scheduled to provide the Government with visibility into the maturity, functionality, and performance of the products significantly earlier than required by the contract (i.e., prior to PDR). Demonstrations are scheduled so as to provide results/data in support of the TRD requirements trace to the software reuse products.	Demonstrations of software reuse products are scheduled to provide the Government with visibility into the maturity, functionality, and performance of the products significantly earlier than required by the contract (i.e., prior to PDR).	Demonstrations of software reuse products are scheduled to provide the Government with visibility into the maturity, functionality, and performance of the products earlier than required by the contract (i.e., prior to PDR).	Demonstrations of software reuse products are scheduled to provide the Government with visibility into the maturity, functionality, and performance of the products IAW the schedule constraint required by the contract (i.e., prior to PDR).	Demonstrations of only some software reuse products are scheduled as required by the contract.
The IMS shows all decision points for implementing alternative approaches (to be used if the software cannot be reused as originally planned). These decision points provide sufficient time to pursue the alternative approaches without negative impact to the program schedule. The contractor has scheduled activities to reduce the risks of the alternative approaches prior to their decision points.	The IMS shows all decision points for implementing alternative approaches (to be used if the software cannot be reused as originally planned). These decision points provide sufficient time to pursue the alternative approaches without negative impact to the program schedule.	The IMS shows key decision points for implementing alternative approaches (to be used if the software cannot be reused as originally planned). These decision points provide sufficient time to pursue the alternative approaches without negative impact to the program schedule.	The IMS shows key decision points for implementing alternative approaches (to be used if the software cannot be reused as originally planned). These decision points provide sufficient time to pursue the alternative approaches without major negative impact to the program schedule.	The IMS shows few key decision points for implementing alternative approaches (to be used if the software cannot be reused as originally planned). These decision points provide insufficient time to pursue the alternative approaches without major negative impact to the program schedule.

The contractor has executed as scheduled all major tasks associated with all software reuse products. Some tasks were executed ahead of schedule. The contractor has conducted additional proactive activities (i.e., beyond those contained in their plan) to obtain early insight into the content, performance, and applicability of the software reuse products.	The contractor has executed as scheduled all major tasks associated with all software reuse products. Some tasks were executed ahead of schedule.	The contractor has executed as scheduled all major tasks associated with all significant software reuse products.	The contractor has executed as scheduled most major tasks associated with all significant software reuse products.	The contractor has executed as scheduled only a few tasks associated with the significant software reuse products.
The contractor has obtained as scheduled each software reuse product and its supporting artifacts (e.g., design specifications) from their source. The contractor has executed as scheduled all interactions with the source organization for each software reuse product. Proactive activities conducted by the contractor have accelerated the availability timeline for the software products and their artifacts.	The contractor has obtained as scheduled each software reuse product and its supporting artifacts (e.g., design specifications) from their source. The contractor has executed as scheduled all interactions with the source organization for each software reuse product.	The contractor has obtained as scheduled each software reuse product and its supporting artifacts (e.g., design specifications) from their source. The contractor has executed as scheduled all critical interactions with the source organization for each software reuse product.	The contractor has obtained as scheduled all significant software reuse products and their supporting artifacts (e.g., design specifications) from their source. The contractor has executed as scheduled most critical interactions with the source organization for each software reuse product.	The contractor has obtained as scheduled only some of the software reuse products and their supporting artifacts (e.g., design specifications) from their source. The contractor has executed as scheduled only some critical interactions with the source organization for each software reuse product.

The contractor's Software Reuse Demonstration(s) was successfully concluded significantly earlier than required by the contract (i.e., prior to PDR). Results/data from the demonstrations were available to support the TRD requirements trace to the software reuse products.	The contractor's Software Reuse Demonstration(s) was successfully concluded significantly earlier than required by the contract (i.e., prior to PDR).	The contractor's Software Reuse Demonstration(s) was successfully concluded earlier than required by the contract (i.e., prior to PDR).	The contractor's Software Reuse Demonstration(s) was successfully concluded IAW the schedule constraint required by the contract (i.e., prior to PDR).	The contractor's Software Reuse Demonstration(s) was concluded later than required by the contract (i.e., prior to PDR).

Technical				
Outstanding	Excellent	Good	Satisfactory	Unsatisfactory
TRD requirements have been clearly derived/ allocated to all software reuse products and the products meet or exceed these requirements. Requirements allocation is supported by data from analyses, demonstrations, and test/certification activities by other agencies resulting in a high level of confidence in the applicability of the products. The contractor's Software Reuse Demonstration(s) provided the Government with excellent insight into the maturity, functionality and performance of all software reuse products. The demonstration(s) included all key TRD requirements that were allocated to each product, and showed conclusively that the contractor's choice of software reuse products is sound.	TRD requirements have been clearly derived/ allocated to all software reuse products and the products meet or exceed these requirements. The contractor's Software Reuse Demonstration(s) provided the Government with excellent insight into the maturity, functionality and performance of all software reuse products. The demonstration(s) included all functions that were allocated to each product, and showed conclusively that the contractor's choice of software reuse products is sound.	TRD requirements have been clearly derived/ allocated to all software reuse products and the products meet these requirements. The contractor's Software Reuse Demonstration(s) provided the Government with good insight into the maturity, functionality and performance of all significant software reuse products. The demonstration(s) included all significant functions that were allocated to each product, and showed that the contractor's choice of the products is sound.	TRD requirements have been derived/allocated to all software reuse products and the products meet most requirements. There is a mitigation plan for those requirements not being met by the software reuse products. The contractor's Software Reuse Demonstration(s) provided the Government with adequate insight into the maturity, functionality and performance of most significant software reuse products. The demonstration(s) included most significant functions that were allocated to each product, and showed that the contractor's choice of software reuse products is basically sound, although technical issues may exist	TRD requirements have not been clearly and completely derived/ allocated to software reuse products. Some products do not meet these requirements. The contractor's Software Reuse Demonstration(s) provided the Government with poor insight into the maturity, functionality and performance of the software reuse products. The demonstration(s) did not support the contractor's choice of software reuse products.

Program Management				
Outstanding	Excellent	Good	Satisfactory	Unsatisfactory
The contractor updates the Reuse Management Report for all software reuse products. The contractor immediately informed the Government of all major changes to the Worksheet Questions, and solicited the Government point of view on key decisions regarding software reuse. The contractor has reported the status of all software reuse activities and changes to the reuse plans, including the potential impact of these changes on technical performance, cost and schedule, at the design and management reviews. The contractor has shared their performance/cost/schedule trade analyses in support of any changes/decisions on software reuse. The contractor has personnel embedded within the organizations of the entities or sources	The contractor updates the Reuse Management Report for all software reuse products. The contractor immediately informed the Government of all major changes to the Worksheet Questions. The contractor has reported the status of all software reuse activities and changes to the reuse plans, including the potential impact of these changes on technical performance, cost and schedule, at the design and management reviews. The contractor regularly collaborates with the entities or sources	The contractor updates the Reuse Management Report for all software reuse products. The contractor immediately informed the Government of all major changes to the Worksheet Questions. The contractor has reported the status of the software reuse activities and changes to the reuse plans, including the potential impact of these changes on technical performance, cost and schedule, at the design and management reviews. The contractor regularly collaborates with the entities or sources	The contractor updates the Reuse Management Report for the significant software reuse products. The contractor informed the Government of most major changes to the Worksheet Questions. The contractor has reported the status of key software reuse activities and key changes to the reuse plans, including the potential impact of these changes on technical performance, cost and schedule, at the design and management reviews. The contractor occasionally collaborates with the entities or sources responsible for the	The contractor does not update, as required, the Reuse Management Report. The contractor does not keep the Government informed of major changes to the Worksheet Questions. The contractor inconsistently reports the status of the software reuse activities and changes to the reuse plans at the design and management reviews. The contractor seldom collaborates with the entities or sources

responsible for all software reuse products and is very knowledgeable about the software's applicability, availability timeline, maturity, and other attributes. The contractor has implemented a comprehensive set of metrics that are used to pro-actively manage the software reuse activities as well as provide the Government with excellent insight into the status of these activities. The contractor has monitored the status and viability of all alternative approaches for software reuse, modified all approaches to minimize the impact to the program, and updated decision points for implementation. The contractor has continued to explore new alternative approaches to minimize program risks. Prior to reaching all	responsible for all software reuse products and stays informed about the software's applicability, availability timeline, maturity, and other attributes. The contractor has implemented a comprehensive set of metrics that are used to manage the software reuse activities as well as provide the Government with excellent insight into the status of these activities. The contractor has monitored the status and viability of all alternative approaches for software reuse, modified all approaches to minimize the impact to the program, and updated decision points for implementation.	responsible for the key software reuse products and stays informed about the software's applicability, availability timeline, maturity, and other attributes. The contractor has implemented a comprehensive set of metrics that are used to manage the software reuse activities as well as provide the Government with insight into the status of these activities. The contractor has monitored the status and viability of key alternative approaches for software reuse, modified key approaches to minimize the impact to the program, and updated decision points for implementation.	key software reuse products and maintains awareness of the software's applicability, availability timeline, and maturity. The contractor has implemented a set of metrics and often uses these metrics to manage the software reuse activities. The metrics provide the Government with some insight into the status of these activities. The contractor has monitored the status and viability of key alternative approaches for software reuse and updated decision points for implementation.	responsible for the key reuse software products. The contractor has implemented limited metrics and seldom uses these metrics to manage the software reuse activities. The metrics provide the Government with only limited insight into the status of these activities. The contractor has minimally monitored the status and viability of alternative approaches for software reuse.

decision point(s) for implementing the alternative approaches, the contractor has thoroughly analyzed the options (i.e., alternative approaches versus current plan), presented them in detail to the Government, and selected an approach that minimizes the impact to the cost, schedule and performance of the program. When decision points were reached, the contractor promptly selected the course of action and immediately proceeded to execute.	Prior to reaching all decision point(s) for implementing the alternative approaches, the contractor has thoroughly analyzed the options (i.e., alternative approaches versus current plan), presented them in detail to the Government, and selected an approach that minimizes the impact to the cost, schedule and performance of the program.	Prior to reaching the key decision point(s) for implementing the alternative approaches, the contractor has analyzed the options (i.e., alternative approaches versus current plan), presented them to the Government, and selected an approach that minimizes the impact to the cost, schedule and performance of the program.	Prior to reaching the key decision point(s) for implementing the alternative approaches, the contractor has reviewed the options (i.e., alternative approaches versus current plan), presented them to the Government, and selected an approach that minimizes the impact to the cost, schedule and performance of the program.	The contractor has minimally reviewed the options (i.e., alternative approaches versus current plan) at key decision points. It is unclear if the selected approach will minimize the impact to the cost, schedule or performance of the program.

Appendix A Worksheet Questions for Reused As-is/Modified Software

The questions below should be answered in the corresponding worksheet format for <u>each</u> software product for which the Offeror plans to assume responsibility for the performance of the product. Software products may be reused as-is or modified. Information about the commercial off-the-shelf (COTS) or Government off-the-shelf (GOTS) software products that will be reused as-is should be provided in the worksheet format titled "Worksheet Questions for COTS/GOTS Software."

<u>Product and Contact Information</u>

1. What is the name of the software product to be reused as-is or modified?
2. What is the version number and date of release for the software product that is being reused as-is/modified?
3. What are the programming language(s) of this software?
4. For which system/program was the software originally developed?
5. Provide contact information, including the contact's name, the office symbol (if applicable), phone number and address for the: - Program manager currently responsible for the reused as-is/modified software - Responsible entity or source of the software, if the Government is not responsible for the software

<u>Applicability</u>

6. To which Computer Software Configuration Item (CSCI) (and Computer Software Component (CSC), if known) is this reused as-is/modified software product assigned?
7. What functions/requirements will the software provide? (Attach a separate sheet that shows functions cross referenced to the Technical Requirements Document (TRD). Identify any mismatches in functionality between the TRD and the reused as-is/modified software product.)
8. Has the Offeror conducted an internal demonstration(s) to evaluate the applicability of this software product for this system/program? - If yes, provide additional information
9. Have the software product's interfaces that provide access to the functionality been evaluated? - If yes, provide additional information
10. Has the software product's architecture been evaluated for compatibility with the system architecture? - If yes, provide additional information

11. Has the software product been used on a hardware/software platform similar to the one proposed for this system/program?

- If yes, provide additional information

Extent of Modification

12. Briefly describe the tasks (e.g., modification, integration, test) required to make the reused as-is/modified software functional within this system.

13. What organization will perform the modifications to this software product?

14. What organization will integrate the reused as-is/modified software with the system's software?

15. What is the effective size of the reused as-is/modified software product and extent of the modification, if applicable? Complete this table, in an Excel workbook, according to the definitions and instructions attached.

A	B	C	D	E	F	G	H	I	J
Total Delivery or Delivered Build	ID Number of CSCI Contained In	Name of CSCI Contained In	CSC Name	Module or Class Level	Development Contractor/ Subcontractor	Sizing Method	New Software	Total Pre-existing Software	Deleted Software

K	L	M	N	O	P	Q	R
Modified Software	Redesign Required (%)	Reimple-mentation Required (%)	Retest Required (%)	Weight for Design Phase (%)	Weight for Implementation Phase (%)	Weight for Test Phase (%)	Effective Size for Modified Software

S	T	U	V	W	X
Reused As-is/Lifted Software	Reuse As-is/Lift Factor Required (%)	Effective Size for Reused As-is/Lifted Software	Total Effective Size	Effective Size Representing Software Growth	Total Size

Maturity

16. What is the extent of testing of the software that is to be reused as-is/modified (e.g., completed unit tests, completed CSC tests, completed CSCI tests)?

17. Has formal qualification has been conducted?

- If yes, provide additional information

18. Has the software been certified and accredited?

- If yes, provide additional information (e.g., specific certifications and accreditations)

19. Has the software been fielded in an operational environment?

- If yes, provide additional information

20. Has the software been fielded in an operational environment?

- If yes, provide additional information (e.g., which systems/programs, whether these systems/programs have fielded the software)

21. Is the software in long-term maintenance?

- If yes, provide additional information (e.g., organization maintaining the software)

Availability

22. How does the Offeror have access to the software to be reused as-is/modified (e.g., developed the software in-house, has or will acquire the software from another contractor/vendor, or requesting the software be provided by the Government)?

23. Is the software currently available?

- If not, describe the software delivery schedule, including all critical dates that could affect program success

24. Is the Offeror's solution dependent on another Government program for this software?

- If yes, briefly discuss if cross program (and contractor to contractor) relationships have been established, how they will be managed, and how the Offeror plans to stay informed about the evolving software functionality

Other Attributes

Designed for Reuse

25. Identify any attributes (e.g., standards, design patterns, architecture paradigms) of the reused as-is/modified software that support reuse.

Offeror's Experience with Software

26. Will the Offeror have any access to the software developers, who were part of the original software development team?

- If yes, provide additional information

27. Has the organization (that will be performing the modifications to this software product for this program) reused as-is or modified (e.g., altered the design, made changes to the code) the software previously?

- If yes, provide additional information (e.g., for what systems/programs, how many of the software developers have modified this software product before)

28. Has the organization (that will be integrating this software product for this program) integrated the software previously?

- If yes, provide additional information (e.g., for what systems/programs, how many of the software integration engineers have integrated this software product before)

Documentation

29. What supporting engineering and management documentation for the reused as-is/modified software is available for the software developers?

30. What supporting documentation for the reused as-is/modified software is available for the end users?

31. What documentation (both development and end user) will be delivered to the Government?
32. Describe the test procedures that will support the conduct of the comprehensive regression testing for the reused as-is/modified software? - Do these procedures exist or do they need to be created?

Standards

33. What development standards (e.g., IEEE/EIA Std 12207.0-2008) were followed during the development of the software intended to be reused as-is/modified?

Data and Software Rights

34. What rights will the Government have to the data and software? Identify what data and software rights are being provided to the Government using the relevant Defense Federal Acquisition Regulation Supplement (DFARS) clause definitions (DFARS 227.7103-3 and 227.7203-3.1). What is the name of the COTS/GOTS software product to be reused as-is?
35. Does the reused as-is/modified software require the Government to purchase any COTS software licenses? - If yes, provide the commercial software licenses for review
36. Do you intend to transfer any COTS software licenses to the Government? - If yes, provide the commercial software licenses for review

Defect Reports

37. How many Defect Reports (DRs) are currently open for the software?
38. Provide a listing of all (open and closed) DRs by category/priority, date when opened, description of problem and planned/actual date of closure.

Maintenance and Support Strategy

39. What organization is expected to maintain the modified software?

Releases/Updates

40. Will the Offeror incorporate future releases of the reused as-is/modified product into the system's software baseline? - If yes, how will these releases be incorporated?

Dead and Unused Code

41. Identify any dead code (i.e., unreachable, unnecessary, and/or inoperative code that is not required for any purpose) and/or unused code (i.e., code used in applications other than this program) from the reused as-is and/or modified software products. Discuss how dead and/or unused code will be handled, how it will be tested, and whether it presents any risks to the program.

Worksheet Questions for Reused As-is/Modified Software
Instructions for Completing Question 15

Total Delivery or Delivered Build (Col. A): If there are multiple delivered builds (blocks, increments, etc.), enter the build identifier for the sizing information provided. Enter "Total" if the sizing information represents the total delivery. A separate table should be completed for each delivered build as well as the total delivery.

ID Number of CSCI Contained In (Col. B): Enter the identification numbers for the Computer Software Configuration Item (CSCI), in which the reused as-is/modified software product is contained.

Name of CSCI Contained In (Col. C): Enter the name of the CSCI, in which the reused as-is/modified software product is contained.

CSC (Col. D): Enter the names of the CSCs, if known. A separate row should be completed for each CSC.

Module or Class Level (Col. E): Enter the software module or class level, if known. A separate row should be completed for each software module or class level.

Development Contractor/Subcontractor (Col. F): Enter the name of the contractor or subcontractor responsible for the development of each CSCI.

Sizing Method (Col. G): Enter either source lines of code (SLOC) or function points (FP). Standard definitions for SLOC and FPs are provided below. Any non-standard definition should be fully explained on a separate sheet. If an alternative sizing measure is used, the counting method should be described in detail. This table can be adapted to accommodate an alternative measure, but the type of information requested in these instructions must be included.

> Lines of Code: Non-Comment lines of source code for the computer program. Source lines to include are: All executable source lines such as (1) Control, (2) Mathematical, (3) Conditional, (4) Deliverable Job Control, (5) Data Declaration Statements, and (6) Data Typing and Equivalence; and input/output/format. Source lines to exclude are: debug statements, continuation of single statement to multiple lines, machine/library generated statement, and non-deliverable test statements.
>
> Function Points: Unadjusted function points, IFPUG compatible. Use this only if your size methods are function based rather than line based.

New Software (Col. H): Enter the new non-comment lines of source code or the new number of unadjusted function points, IFPUG compatible for the computer program. New code is software developed from scratch and is not modified or reused as-is in any way from any pre-existing design or code.

Total Pre-existing Software (Col. I): Enter the number of lines of code or functions in a pre-existing software package (before reuse as-is/modification/deletion), including lines of code or functions that may not be pertinent to this program/system.

Deleted Software (Col. J): Enter the number of lines of code or functions which will be deleted from the pre-existing software package (Col. I). The deletion will be accomplished by physical omission or commenting out.

Modified Software (Col. K): Enter the total number of lines of code or functions that will be modified from a pre-existing software package through re-design and/or re-implementation, and then integrated and tested in the new software product baseline. If there are multiple delivered builds, this number should represent the code developed in previous builds that may need to be modified and/or re-tested with the code being developed for the current build.

Redesign Required (Col. L): Enter the percentage of the pre-existing software to be modified (Col. K) that requires redesign to make this software functional within the new environment.

Reimplementation Required (Col. M): Enter the percentage of the pre-existing software to be modified (Col. K) that requires reimplementation (i.e., code and unit test) to make this software functional within the new environment.

Retest Required (Col. N): Enter the percentage of the pre-existing software to be modified (Col. K) that requires retesting (i.e., CSC integration/test and CSCI integration/test, but excluding CSCI-to-CSCI integration/test) to ensure this software functions within performance, reliability, and other criteria after the modifications.

Weight for Design Phase (Col. O): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the design phase. The weights (Col. O-Q) represent the phase distribution of

effort, i.e., the distribution typically observed for each phase. Note that the sum of the weights for the design, implementation and test phases must equal 100 percent.

Weight for Implementation Phase (Col. P): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the implementation phase.

Weight for Test Phase (Col. Q): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the test phase.

Effective Size for Modified Software (Col. R): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be modified (Col. K) and are adjusted based on the applicable percentages (Col. L-N) and weights (Col. O-Q). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the Modified Software is:

Col. R = Col. K * ((Col. L * Col. O) + (Col. M * Col. P) + (Col. N * Col. Q))

The Offeror shall explain the method used for calculating effective size if it differs from the formula in this table.

Reused As-is/Lifted Software (Col. S): Enter the number of lines of code or functions that will be reused as-is or lifted, with no modification of design or code, from a pre-existing software package.

Reuse As-is/Lift Factor Required (Col. T): Enter the percentage that is applied to the pre-existing software to be reused as-is/lifted (Col. S) to estimate effective size. This percentage is similar in concept to the percentages for redesign required, reimplementation required, and retest required, but is a composite factor applied to the software that will be reused as-is/lifted. Reused as-is/lifted code, by definition, will not require modification.

Reused as-is software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the software product itself will remain unchanged. The new lines of source code or functions associated with the software (e.g., glue code, integration code) should be included as New Software (Col H). The effort required to understand the product and its interfaces, integrate the product as part of a CSC and/or CSCI, and perform testing should be reflected in the percentage in order to estimate effective size.

Effective Size for Reused As-is/Lifted Software (Col. U): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be reused as-is/lifted (Col. 18) and are adjusted based on the applicable percentage (Col. 19). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the Reused As-is/Lifted Software is:

Col. U = Col. S * Col. T

The Offeror shall explain the method used for calculating effective size if it differs from the formula in this table.

Total Effective Size (Col. V): Enter the number of lines of code or functions that represent all new lines or functions (Col. H) as well as the pre-existing lines or functions that are modified (Col. K) or reused as-is/lifted (Col. S) and are adjusted based on the applicable percentages (Col. N-N, T) and weights (Col. O-Q). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Total Effective Size is:

Col. V = Col. H + Col. R + Col. U

Effective Size Representing Software Growth (Col. W): Enter the number of effective lines or functions that are included in the effective size estimate (Col. V) to capture software growth. The Offeror shall provide the definition of software growth used for the sizing estimate and the method used to estimate software growth.

Total Size (Col. X): Enter the number of new (Col. H) and pre-existing (Col. K and S) lines of code or functions. Total size represents the total amount of new software that would need to be developed for the new software baseline, if no code were to be reused as-is and/or modified. Note that Total Size does not include the pre-existing software that will be deleted.

Appendix B Worksheet Questions for COTS/GOTS Software

The questions below should be answered in the corresponding worksheet format for <u>each</u> commercial off-the-shelf (COTS) or Government off-the-shelf (GOTS) software product that will be reused as-is. COTS/GOTS software include the products, for which the software provider, either a commercial vendor or the Government, assumes responsibility for the performance of the software product. The source code is not necessarily provided to the Offeror. The COTS/GOTS software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the COTS/GOTS product itself will remain unchanged.

<u>Product and Contact Information</u>

1. What is the name of the COTS/GOTS software product to be reused as-is?
2. What is the version number and date of release for the COTS/GOTS software product that is being reused as-is?
3. Provide contact information, including the contact's name, the office symbol (if applicable), phone number and address for the: - COTS/GOTS software provider - Government program manager currently responsible for a program, if any, that uses the COTS/GOTS software product

<u>Applicability</u>

4. To which Computer Software Configuration Item (CSCI) (and Computer Software Component (CSC), if known) is this software product assigned?
5. What functions/requirements will the software provide? (Attach a separate sheet that shows functions cross referenced to the Technical Requirements Document (TRD). Identify any mismatches in functionality between the TRD and the COTS/GOTS product.)
6. Has the Offeror conducted an internal demonstration(s) to evaluate the applicability and usability of this product for this system/program? - If yes, provide additional information
7. Have the COTS/GOTS product's interfaces that provide access to the functionality been evaluated? - If yes, provide additional information
8. Has the software product's architecture been evaluated for compatibility with the system architecture? - If yes, provide additional information
9. Are the COTS/GOTS product's development and target hardware/software platforms similar to the ones proposed for this system/program? - If yes, provide additional information

Approach to Integration/Test

10. Briefly describe the tasks (e.g., development of glue code, integration, test) required to make the COTS/GOTS software functional within this system.
11. What organization will be responsible for developing any new code (e.g., glue code, integration code) needed?
12. What organization will integrate this COTS/GOTS product with the system's software?
13. Does the Offeror need access to the source code? - If yes, does the Offeror have access to the source code?
14. What is the effective software size associated with the COTS/GOTS software product? Complete this table, in an Excel workbook, according to the definitions and instructions attached.

A	B	C	D	E	F	G
Total Delivery or Delivered Build	**ID Number of CSCI Contained In**	**Name of CSCI Contained In**	**CSC Name**	**Development Contractor/ Subcontractor**	**Sizing Method**	**New Software**

H	I	J	K	L	M
Reused As-is/ Lifted Software	**Reuse As-is/Lift Factor Required (%)**	**Effective Size for Reused As-is/Lifted Software**	**Total Effective Size**	**Effective Size Representing Software Growth**	**Total Size**

Maturity

15. When was the COTS/GOTS product first released?
16. How many versions (both major and minor updates) have subsequently been released?
17. Has this COTS/GOTS product been successfully used on any Government program? - Has it been system-level tested? - Has it been certified and accredited? - Has it been fielded? - If yes, provide additional information (e.g., for what Government program, when, Government agency witnessing test, specific accreditations and certifications).

Availability

18. Is the COTS/GOTS software currently available? - If not, describe the software delivery schedule, including all critical dates that could affect program success

19. Is the Offeror's solution dependent on another Government program for this software? - If yes, discuss if cross program (and contractor to contractor) relationships have been established, how they will be managed, and how the Offeror plans to stay informed about the evolving functionality of the COTS/GOTS software

Other Attributes

Offeror's Experience with COTS/GOTS Product

20. Does the Offeror have experience using the proposed COTS/GOTS software? - If yes, for what systems/programs?
21. Does the Offeror have experience integrating the proposed COTS/GOTS software? - If yes, for what systems/programs?
22. Has the Offeror worked with the vendor of the COTS software product before? - If yes, on which program or in what capacity?

Documentation

23. What documentation will be provided to the Government?

Data and Software Rights

24. What rights will the Government have to the data and COTS/GOTS software? Identify what data and software rights are being provided to the Government using the relevant Defense Federal Acquisition Regulation Supplement (DFARS) clause definitions (DFARS 227.7103-3 and 227.7203-3).
25. Does the reused COTS/GOTS software require the Government to purchase any COTS software licenses? - If yes, provide the commercial software licenses for review
26. Do you intend to transfer any COTS software licenses to the Government? - If yes, provide the commercial software licenses for review

Licensing

27. How will the COTS/GOTS software be licensed (e.g., per seat, per site, per host) for both development and run-time for this program?
28. Will the Government be expected to keep track of run-time licenses?
29. Will there be any automated enforcement mechanisms (e.g., license managers, activation)?

Maintenance and Support Strategy

30. What organization is expected to maintain the COTS/GOTS software?
31. For what time frame, will the COTS/GOTS software be maintained?

Releases/Updates

32. Will the Offeror incorporate future releases of the COTS/GOTS product into the system's software baseline? - If yes, how will these releases be incorporated?

COTS Vendor Viability

33. How long has the COTS vendor been in business?
34. How many of the vendor's employees are dedicated to the implementation of the product and to the product's support?
35. What was the funding source for the development of this product?
36. What is the vendor's customer base (e.g., commercial, Government defense, or Government nondefense)?

Worksheet Questions for COTS/GOTS Software

Instructions for Completing Question 14

Total Delivery or Delivered Build (Col. A): If there are multiple delivered builds (blocks, increments, etc.), enter the build identifier for the sizing information provided. Enter "Total" if the sizing information represents the total delivery. A separate table should be completed for each delivered build as well as the total delivery.

ID Number of CSCI Contained In (Col. B): Enter the identification numbers for the Computer Software Configuration Item (CSCI), in which the COTS/GOTS software product is contained.

Name of CSCI Contained In (Col. C): Enter the name of the CSCI, in which the COTS/GOTS software product is contained.

CSC (Col. D): Enter the names of the CSCs, if known. A separate row should be completed for each CSC.

Development Contractor/Subcontractor (Col. E): Enter the name of the contractor or subcontractor responsible for the development of each CSCI.

Sizing Method (Col. F): Enter either source lines of code (SLOC) or function points (FP). Standard definitions for SLOC and FPs are provided below. Any non-standard definition should be fully explained on a separate sheet. If an alternative sizing measure is used, the counting method should be described in detail. This table can be adapted to accommodate an alternative measure, but the type of information requested in these instructions must be included.

> Lines of Code: Non-Comment lines of source code for the computer program. Source lines to include are: All executable source lines such as (1) Control, (2) Mathematical, (3) Conditional, (4) Deliverable Job Control, (5) Data Declaration Statements, and (6) Data Typing and Equivalence; and input/output/format. Source lines to exclude are: debug statements, continuation of single statement to multiple lines, machine/library generated statement, and non-deliverable test statements.
>
> Function Points: Unadjusted function points, IFPUG compatible. Use this only if your size methods are function based rather than line based.

New Software (Col. G): Enter the new non-comment lines of source code or the new number of unadjusted function points, IFPUG compatible for the computer program. New code is software developed from scratch and is not modified or reused as-is in any way from any pre-existing design or code. The new lines of source code or functions associated with reusing as-is any COTS/GOTS software (e.g., glue code, integration code) should be included in this column.

Reused As-is/Lifted Software (Col. H): Enter the number of lines of code or functions that will be reused as-is or lifted, with no modification of design or code, from a pre-existing COTS/GOTS software package. If COTS/GOTS software size is not encompassed in the Offeror's overall sizing methodology, the Offeror shall attach a separate sheet to explain the estimating methodology and fully discuss all efforts associated with the COTS/GOTS products in the Basis of Estimates (BOEs).

Reuse As-is/Lift Factor Required (Col. I): Enter the percentage that is applied to the pre-existing COTS/GOTS software to be reused as-is/lifted (Col. H) to estimate effective size.

Reused as-is software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the software product itself will remain unchanged. The new lines of source code or functions associated with the software (e.g., glue code, integration code) should be included as New Software (Col G). The effort required to understand the product and its interfaces, integrate the product as part of a CSC and/or CSCI, and perform testing should be reflected in the percentage in order to estimate effective size. If these activities associated with the COTS/GOTS software are not part of the Offeror's software size methodology, the Offeror shall fully discuss these efforts in the BOEs.

Effective Size for Reused As-is/Lifted Software (Col. J): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be reused as-is/lifted (Col. H) and are adjusted based on the applicable percentage (Col. I). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the COTS/GOTS software is:

> Col. J = Col. H * Col. I

The Offeror shall explain the method used for calculating effective size if it differs from the formula in this table.

Total Effective Size (Col. K): Enter the number of lines of code or functions that represent all new lines or functions (Col. G) as well as the pre-existing lines or functions that are reused as-is/lifted (Col. J). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Total Effective Size is:

Col. K = Col. G + Col. J

Effective Size Representing Software Growth (Col. L): Enter the number of effective lines or functions that are included in the effective size estimate (Col. K) to capture software growth. The Offeror shall provide the definition of software growth used for the sizing estimate and the method used to estimate software growth.

Total Size (Col. M): Enter the number of new (Col. G) and pre-existing (Col. H) lines of code or functions. Total size represents the total amount of new software that would need to be developed for the new software baseline, if no code were to be reused as-is and/or modified.

Appendix C Data Item Description for the Reuse Management Report

The Data Item Description for DI-SESS-81771 is available on the ASSIST database.

DATA ITEM DESCRIPTION

Title: REUSE MANAGEMENT REPORT (ReMR)

Number: DI-SESS-81771
Approval Date: 20090520
AMSC Number: F9071
Limitation: N/A
DTIC Applicable: N/A
GIDEP Applicable: N/A
Preparing Activity: 13 **(**ESC/AQT)
Applicable Forms:
Worksheet Questions for Reused As-is/Modified Software
Worksheet Questions for COTS/GOTS Software
Use/Relationships: The Reuse Management Report (ReMR) provides information about existing software products intended to be reused as-is or modified as part of the delivered operational software. The report also provides the acquirer insight into the current status of the activities associated with the reuse of these products as compared to the planned activities, and alternative approaches.

This Data Item Description (DID) contains the format, content and intended use information for the data product resulting from the work tasks described in the contract.

Requirements:

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 the contract.

2. Format. Contractor format is acceptable.

3. Content. The report shall contain the following:

3.1 Executive Summary. This section shall identify and briefly describe all software products that will be reused as-is or modified (existing software requiring change) and integrated into the delivered operational software. Both commercial off-the-shelf (COTS) and government off-the-shelf (GOTS) shall be included.

3.2. Current Status. This section shall present the current status of the software reuse activities compared to the contractor's planned activities. Status shall include progress made and accomplishments for the engineering and management activities for each software reuse product.

3.3. Variance. This section shall identify any activities where work is not progressing in accordance with the plans and schedules, including the reasons for this lack of progress.

3.4. Milestones. This section shall describe the progress made against program milestones during the reporting period.

3.5. Alternative Approaches. This section shall describe alternative approaches for any reused as-is/modified and COTS/GOTS software products that are 1) considered high or moderate risk or 2) not available (i.e., fully documented and tested) at contract award. Alternative approaches provide options if these software reuse products cannot be implemented as planned.

3.6. Impacts of Implementing Alternative Approaches. This section shall include an assessment of the technical impacts to the program and estimates of the programmatic (i.e., effort and schedule) impacts of implementing alternative approaches.

3.7. Decision Points. This section shall include the decision points for implementing alternative approaches. These decision points shall identify when the alternative approach would need to be implemented in the event that the planned software reuse products are not available in time to preserve the program schedule.

3.8. Worksheet Questions. This section shall include the completed forms (updated, if needed) for all reused as-is/modified and COTS/GOTS software products in accordance with the instructions embedded in the forms. Forms will be included for any newly identified software reuse products.

WORKSHEET QUESTIONS FOR
REUSED AS-IS/MODIFIED SOFTWARE

The questions below should be answered in the corresponding worksheet format for each software product for which the Contractor plans to assume responsibility for the performance of the product. Software products may be reused as-is or modified. Information about the commercial off-the-shelf (COTS) or Government off-the-shelf (GOTS) software products that will be reused as-is should be provided in the worksheet format titled "Worksheet Questions for COTS/GOTS Software."

Product and Contact Information

1. What is the name of the software product to be reused as-is or modified?
2. What is the version number and date of release for the software product that is being reused as-is/modified?
3. What are the programming language(s) of this software?
4. For which system/program was the software originally developed?
5. Provide contact information, including the contact's name, the office symbol (if applicable), phone number and address for the: - Program manager currently responsible for the reused as-is/modified software - Responsible entity or source of the software, if the Government is not responsible for the software

Applicability

6. To which Computer Software Configuration Item (CSCI) (and Computer Software Component (CSC), if known) is this reused as-is/modified software product assigned?
7. What functions/requirements will the software provide? (Attach a separate sheet that shows performance requirements cross referenced to the Technical Requirements Document (TRD). Identify any mismatches in requirements between the TRD and the reused as-is/modified software product.)
8. Has the Contractor conducted an internal demonstration(s) to evaluate the applicability of this software product for this system/program? - If yes, provide additional information
9. Have the software product's interfaces that provide access to the functionality been evaluated? - If yes, provide additional information
10. Has the software product's architecture been evaluated for compatibility with the system architecture? - If yes, provide additional information
11. Has the software product been used on a hardware/software platform similar to the one proposed for this system/program? - If yes, provide additional information

Extent of Modification

12. Briefly describe the tasks (e.g., modification, integration, test) required to make the reused as-is/modified software functional within this system.
13. What organization will perform the modifications to this software product?
14. What organization will integrate the reused as-is/modified software with the system's software?
15. What is the effective size of the reused as-is/modified software product and extent of the modification, if applicable? Complete this table, in an Excel workbook, according to the definitions and instructions attached.

A	B	C	D	E	F	G	H	I	J
Total Delivery or Delivered Build	**ID Number of CSCI Contained In**	**Name of CSCI Contained In**	**CSC Name**	**Module or Class Level**	**Development Contractor/ Subcontractor**	**Sizing Method**	**New Software**	**Total Pre-existing Software**	**Deleted Software**

K	L	M	N	O	P	Q	R
Modified Software	**Redesign Required (%)**	**Reimple-mentation Required (%)**	**Retest Required (%)**	**Weight for Design Phase (%)**	**Weight for Implementation Phase (%)**	**Weight for Test Phase (%)**	**Effective Size for Modified Software**

S	T	U	V	W	X
Reused As-is/ Lifted Software	**Reuse As-is/ Lift Factor Required (%)**	**Effective Size for Reused As-is/Lifted Software**	**Total Effective Size**	**Effective Size Representing Software Growth**	**Total Size**

Maturity

16. What is the extent of testing of the software that is to be reused as-is/modified (e.g., completed unit tests, completed CSC tests, completed CSCI tests)?
17. Has formal qualification has been conducted? - If yes, provide additional information
18. Has the software been certified and accredited? - If yes, provide additional information (e.g., specific certifications and accreditations)
19. Has the software been fielded in an operational environment? - If yes, provide additional information
20. Has the software been fielded in an operational environment? - If yes, provide additional information (e.g., which systems/programs, whether these systems/programs have fielded the software)
21. Is the software in long-term maintenance? - If yes, provide additional information (e.g., organization maintaining the software)

Availability

22. How does the Contractor have access to the software to be reused as-is/modified (e.g., developed the software in-house, has or will acquire the software from another contractor/vendor, or requesting the software be provided by the Government)?
23. Is the software currently available? - If not, describe the software delivery schedule, including all critical dates that could affect program success.
24. Is the Contractor's solution dependent on another Government program for this software? - If yes, briefly discuss if cross program (and contractor to contractor) relationships have been established, how they will be managed, and how the Contractor plans to stay informed about the evolving software functionality.

Other Attributes
Designed for Reuse

25. Identify any attributes (e.g., standards, design patterns, architecture paradigms) of the reused as-is/modified software that support reuse.

Contractor's Experience with Software

26. Will the Contractor have any access to the software developers, who were part of the original software development team? - If yes, provide additional information
27. Has the organization (that will be performing the modifications to this software product for this program) reused as-is or modified (e.g., altered the design, made changes to the code) the software previously? - If yes, provide additional information (e.g., for what systems/programs, how many of the software developers have modified this software product before)
28. Has the organization (that will be integrating this software product for this program) integrated the software previously? - If yes, provide additional information (e.g., for what systems/programs, how many of the software integration engineers have integrated this software product before)

Documentation

29. What supporting engineering and management documentation for the reused as-is/modified software is available for the software developers?
30. What supporting documentation for the reused as-is/modified software is available for the end users?
31. What documentation (both development and end user) will be delivered to the Government?
32. Describe the test procedures that will support the conduct of the comprehensive regression testing for the reused as-is/modified software?

- Do these procedures exist or do they need to be created?

Standards

33. What development standards (e.g., IEEE/EIA Std 12207.0-2008) were followed during the development of the software intended to be reused as-is/modified?

Data and Software Rights

34. What rights will the Government have to the data and software? Identify what data and software rights are being provided to the Government using the relevant Defense Federal Acquisition Regulation Supplement (DFARS) clause definitions (DFARS 227.7103-3 and 227.7203-3.1). What is the name of the COTS/GOTS software product to be reused as-is?

35. Does the reused as-is/modified software require the Government to purchase any COTS software licenses? If yes, provide the commercial software licenses for review.

36. Do you intend to transfer any COTS software licenses to the Government? If yes, provide the commercial software licenses for review.

Defect Reports

37. How many Defect Reports (DRs) are currently open for the software?

38. Provide a listing of all (open and closed) DRs by category/priority, date when opened, description of problem and planned/actual date of closure.

Maintenance and Support Strategy

39. What organization is expected to maintain the modified software?

Releases/Updates

40. Will the Contractor incorporate future releases of the reused as-is/modified product into the system's software baseline?
- If yes, how will these releases be incorporated

Dead and Unused Code

41. Identify any dead code (i.e., unreachable, unnecessary, or inoperative code that is not required for any purpose) and unused code (i.e., code used in applications other than this program) from the reused as-is and modified software products. Discuss how dead and unused code will be handled, how it will be tested, and whether it presents any risks to the program.

WORKSHEET QUESTIONS FOR REUSED AS-IS/MODIFIED SOFTWARE
Instructions for Completing Question 15

Definitions: Total Delivery or Delivered Build (Col. A): If there are multiple delivered builds (blocks, increments, etc.), enter the build identifier for the sizing information provided. Enter "Total" if the sizing information represents the total delivery. A separate table should be completed for each delivered build as well as the total delivery.

ID Number of CSCI Contained In (Col. B): Enter the identification numbers for the Computer Software Configuration Item (CSCI), in which the reused as-is/modified software product is contained.

Name of CSCI Contained In (Col. C): Enter the name of the CSCI, in which the reused as-is/modified software product is contained.

CSC (Col. D): Enter the names of the Computer Software Components (CSCs), if known. A separate row should be completed for each CSC.

Module or Class Level (Col. E): Enter the software module or class level, if known. A separate row should be completed for each software module or class level.

Development Contractor/Subcontractor (Col. F): Enter the name of the contractor or subcontractor responsible for the development of each CSCI.

Sizing Method (Col. G): Enter either Source Lines of Code (SLOC) or Function Points (FP). Standard definitions for SLOC and FPs are provided below. Fully explain any non-standard definition on a separate sheet. If an alternative sizing measure is used, the counting method should be described in detail. This table can be adapted to accommodate an alternative measure, but the type of information requested in these instructions must be included.

> Lines of Code: Non-Comment lines of source code for the computer program. Source lines to include are: All executable source lines such as (1) Control, (2) Mathematical, (3) Conditional, (4) Deliverable Job Control, (5) Data Declaration Statements, and (6) Data Typing and Equivalence; and input/output/format. Source lines to exclude are: debug statements, continuation of single statement to multiple lines, machine/library generated statement, and non-deliverable test statements.
>
> Function Points: Unadjusted function points, IFPUG compatible. Use this only if your size methods are function based rather than line based.

New Software (Col. H): Enter the new non-comment lines of source code or the new number of unadjusted function points, IFPUG compatible for the computer program. New code is software developed from scratch and is not modified or reused as-is in any way from any pre-existing design or code.

Total Pre-existing Software (Col. I): Enter the number of lines of code or functions in a pre-existing software package (before reuse as-is/modification/deletion), including lines of code or functions that may not be pertinent to this program/system.

Deleted Software (Col. J): Enter the number of lines of code or functions which will be deleted from the pre-existing software package (Col. I). The deletion will be accomplished by physical omission or commenting out.

Modified Software (Col. K): Enter the total number of lines of code or functions that will be modified from a pre-existing software package through re-design or re-implementation, and then integrated and tested in the new software product baseline. If there are multiple delivered builds, this number should represent the code developed in previous builds that may need to be modified or re-tested with the code being developed for the current build.

Redesign Required (Col. L): Enter the percentage of the pre-existing software to be modified (Col. K) that requires redesign to make this software functional within the new environment.

Reimplementation Required (Col. M): Enter the percentage of the pre-existing software to be modified (Col. K) that requires reimplementation (i.e., code and unit test) to make this software functional within the new environment.

Retest Required (Col. N): Enter the percentage of the pre-existing software to be modified (Col. K) that requires retesting (i.e., CSC integration/test and CSCI integration/test, but excluding CSCI-to-CSCI integration/test) to ensure this software functions within performance, reliability, and other criteria after the modifications.

Weight for Design Phase (Col. O): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the design phase. The weights (Col. O-Q) represent the phase distribution of effort, i.e., the distribution typically observed for each phase. Note that the sum of the weights for the design, implementation and test phases must equal 100 percent.

Weight for Implementation Phase (Col. P): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the implementation phase.

Weight for Test Phase (Col. Q): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the test phase.

Effective Size for Modified Software (Col. R): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be modified (Col. K) and are adjusted based on the applicable percentages (Col. L-N) and weights (Col. O-Q). Effective size represents the software size equivalent to developing the code from

scratch. The formula, in this spreadsheet, used for calculating Effective Size for the Modified Software is:

Col. R = Col. K * ((Col. L * Col. O) + (Col. M * Col. P) + (Col. N * Col. Q))

Explain the method used for calculating effective size if it differs from the formula in this table.

Reused As-is/Lifted Software (Col. S): Enter the number of lines of code or functions that will be reused as-is or lifted, with no modification of design or code, from a pre-existing software package.

Reuse As-is/Lift Factor Required (Col. T): Enter the percentage that is applied to the pre-existing software to be reused as-is/lifted (Col. S) to estimate effective size. This percentage is similar in concept to the percentages for redesign required, reimplementation required and retest required, but is a composite factor applied to the software that will be reused as-is/lifted. Reused as-is/lifted code, by definition, will not require modification.

Reused as-is software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the software product itself will remain unchanged. The new lines of source code or functions associated with the software (e.g., glue code, integration code) should be included as New Software (Col H). The effort required to understand the product and its interfaces, integrate the product as part of a CSC and CSCI, and perform testing should be reflected in the percentage in order to estimate effective size.

Effective Size for Reused As-is/Lifted Software (Col. U): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be reused as-is/lifted (Col. 18) and are adjusted based on the applicable percentage (Col. 19). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the Reused As-is/Lifted Software is:

Col. U = Col. S * Col. T

Explain the method used for calculating effective size if it differs from the formula in this table.

Total Effective Size (Col. V): Enter the number of lines of code or functions that represent all new lines or functions (Col. H) as well as the pre-existing lines or functions that are modified (Col. K) or reused as-is/lifted (Col. S) and are adjusted based on the applicable percentages (Col. N-N, T) and weights (Col. O-Q). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Total Effective Size is:

Col. V = Col. H + Col. R + Col. U

Effective Size Representing Software Growth (Col. W): Enter the number of effective lines or functions that are included in the effective size estimate (Col. V) to capture software growth. Provide the definition of software growth used for the sizing estimate and the method used to estimate software growth.

Total Size (Col. X): Enter the number of new (Col. H) and pre-existing (Col. K and S) lines of code or functions. Total size represents the total amount of new software that would need to be developed for the new software baseline, if no code were to be reused as-is or modified. Note that Total Size does not include the pre-existing software that will be deleted.

WORKSHEET QUESTIONS FOR COTS/GOTS SOFTWARE

The questions below should be answered in the corresponding worksheet format for <u>each</u> commercial off-the-shelf (COTS) or Government off-the-shelf (GOTS) software product that will be reused as-is. COTS/GOTS software include the products, for which the software provider, either a commercial vendor or the Government, assumes responsibility for the performance of the software product. The source code is not necessarily provided to the Contractor. The COTS/GOTS software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the COTS/GOTS product itself will remain unchanged.

Product and Contact Information

1. What is the name of the COTS/GOTS software product to be reused as-is?
2. What is the version number and date of release for the COTS/GOTS software product that is being reused as-is?
3. Provide contact information, including the contact's name, the office symbol (if applicable), phone number and address for the: - COTS/GOTS software provider - Government program manager currently responsible for a program, if any, that uses the COTS/GOTS software product

Applicability

4. To which Computer Software Configuration Item (CSCI) (and Computer Software Component (CSC), if known) is this software product assigned?
5. What functions/requirements will the software provide? (For submissions with the proposal, attach a separate sheet that shows functions cross referenced to the Technical Requirements Document (TRD). Identify any mismatches in functionality between the TRD and the COTS/GOTS product. After contract award, attach a separate sheet that shows performance requirements cross referenced to the TRD. Identify any mismatches in requirements between the TRD and the COTS/GOTS product.)
6. Has the Contractor conducted an internal demonstration(s) to evaluate the applicability and usability of this product for this system/program? - If yes, provide additional information
7. Have the COTS/GOTS product's interfaces that provide access to the functionality been evaluated? - If yes, provide additional information
8. Has the software product's architecture been evaluated for compatibility with the system architecture? - If yes, provide additional information
9. Are the COTS/GOTS product's development and target hardware/software platforms similar to the ones proposed for this system/program? - If yes, provide additional information

Approach to Integration/Test

10. Briefly describe the tasks (e.g., development of glue code, integration, test) required to make the COTS/GOTS software functional within this system.
11. What organization will be responsible for developing any new code (e.g., glue code, integration code) needed?
12. What organization will integrate this COTS/GOTS product with the system's software?
13. Does the Contractor need access to the source code? - If yes, does the Contractor have access to the source code?
14. What is the effective software size associated with the COTS/GOTS software product? Complete this table, in an Excel workbook, according to the definitions and instructions attached.

A	B	C	D	E	F	G
Total Delivery or Delivered Build	**ID Number of CSCI Contained In**	**Name of CSCI Contained In**	**CSC Name**	**Development Contractor/ Subcontractor**	**Sizing Method**	**New Software**

H	I	J	K	L	M
Reused As-is/ Lifted Software	**Reuse As-is /Lift Factor Required (%)**	**Effective Size for Reused As-is/Lifted Software**	**Total Effective Size**	**Effective Size Representing Software Growth**	**Total Size**

Maturity

15. When was the COTS/GOTS product first released?
16. How many versions (both major and minor updates) have subsequently been released?
17. Has this COTS/GOTS product been successfully used on any Government program? - Has it been system-level tested? - Has it been certified and accredited? - Has it been fielded? - If yes, provide additional information (e.g., for what Government program, when, Government agency witnessing test, specific accreditations and certifications)

Availability

18. Is the COTS/GOTS software currently available? - If not, describe the software delivery schedule, including all critical dates that could affect program success
19. Is the Contractor's solution dependent on another Government program for this software?

- If yes, discuss if cross program (and contractor to contractor) relationships have been established, how they will be managed, and how the Contractor plans to stay informed about the evolving functionality of the COTS/GOTS software

Other Attributes
Contractor's Experience with COTS/GOTS Product

20. Does the Contractor have experience using the proposed COTS/GOTS software? - If yes, for what systems/programs?
21. Does the Contractor have experience integrating the proposed COTS/GOTS software? - If yes, for what systems/programs?
22. Has the Contractor worked with the vendor of the COTS software product before? - If yes, on which program or in what capacity?

Documentation

23. What documentation will be provided to the Government?

Data and Software Rights

24. What rights will the Government have to the data and COTS/GOTS software? Identify what data and software rights are being provided to the Government using the relevant Defense Federal Acquisition Regulation Supplement (DFARS) clause definitions (DFARS 227.7103-3 and 227.7203-3.)
25. Does the reused COTS/GOTS software require the Government to purchase any COTS software licenses? If yes, provide the commercial software licenses for review.
26. Do you intend to transfer any COTS software licenses to the Government? If yes, provide the commercial software licenses for review.

Licensing

27. How will the COTS/GOTS software be licensed (e.g., per seat, per site, per host) for both development and run-time for this program?
28. Will the Government be expected to keep track of run-time licenses?
29. Will there be any automated enforcement mechanisms (e.g., license managers, activation)?

Maintenance and Support Strategy

30. What organization is expected to maintain the COTS/GOTS software?
31. For what time frame, will the COTS/GOTS software be maintained?

Releases/Updates

32. Will the Contractor incorporate future releases of the COTS/GOTS product into the system's software baseline? - If yes, how will these releases be incorporated?

COTS Vendor Viability

33. How long has the COTS vendor been in business?
34. How many of the vendor's employees are dedicated to the implementation of the product and to the product's support?
35. What was the funding source for the development of this product?
36. What is the vendor's customer base (e.g., commercial, Government defense, or Government nondefense)?

WORKSHEET QUESTIONS FOR COTS/GOTS SOFTWARE
Instructions for Completing Question 14

Total Delivery or Delivered Build (Col. A): If there are multiple delivered builds (blocks, increments, etc.), enter the build identifier for the sizing information provided. Enter "Total" if the sizing information represents the total delivery. A separate table should be completed for each delivered build as well as the total delivery.

ID Number of CSCI Contained in (Col. B): Enter the identification numbers for the Computer Software Configuration Item (CSCI), in which the COTS/GOTS software product is contained.

Name of CSCI Contained in (Col. C): Enter the name of the CSCI, in which the COTS/GOTS software product is contained.

CSC (Col. D): Enter the names of the Computer Software Components (CSCs), if known. A separate row should be completed for each CSC.

Development Contractor/Subcontractor (Col. E): Enter the name of the contractor or subcontractor responsible for the development of each CSCI.

Sizing Method (Col. F): Enter either Source Lines of Code (SLOC) or Function Points (FP). Standard definitions for SLOC and FPs are provided below. Fully explain any non-standard definition on a separate sheet. If an alternative sizing measure is used, the counting method should be described in detail. This table can be adapted to accommodate an alternative measure, but the type of information requested in these instructions must be included.

> Definitions:
> Lines of Code: Non-Comment lines of source code for the computer program. Source lines to include are: All executable source lines such as (1) Control, (2) Mathematical, (3) Conditional, (4) Deliverable Job Control, (5) Data Declaration Statements, and (6) Data Typing and Equivalence; and input/output/format. Source lines to exclude are: debug statements, continuation of single statement to multiple lines, machine/library generated statement, and non-deliverable test statements.
>
> Function Points: Unadjusted function points, IFPUG compatible. Use this only if your size methods are function based rather than line based.

New Software (Col. G): Enter the new non-comment lines of source code or the new number of unadjusted function points, IFPUG compatible for the computer program. New code is software developed from scratch and is not modified or reused as-is in any way from any pre-existing design or code. The new lines of source code or functions associated with reusing as-is any COTS/GOTS software (e.g., glue code, integration code) should be included in this column.

Reused As-is/Lifted Software (Col. H): Enter the number of lines of code or functions that will be reused as-is or lifted, with no modification of design or code, from a pre-existing COTS/GOTS software package. If COTS/GOTS software size is not encompassed in the Contractor's overall sizing methodology, attach a separate sheet to explain the estimating methodology.

Reuse As-is/Lift Factor Required (Col. I): Enter the percentage that is applied to the pre-existing COTS/GOTS software to be reused as-is/lifted (Col. H) to estimate effective size.

Reused as-is software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the software product itself will remain unchanged. The new lines of source code or functions associated with the software (e.g., glue code, integration code) should be included as New Software (Col G). The effort required to understand the product and its interfaces, integrate the product as part of a CSC and CSCI, and perform testing should be reflected in the percentage in order to estimate effective size.

Effective Size for Reused As-is/Lifted Software (Col. J): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be reused as-is/lifted (Col. H) and are adjusted based on the applicable percentage (Col. I). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the COTS/GOTS software is:

Col. J = Col. H * Col. I

Explain the method used for calculating effective size if it differs from the formula in this table.

Total Effective Size (Col. K): Enter the number of lines of code or functions that represent all new lines or functions (Col. G) as well as the pre-existing lines or functions that are reused as-is/lifted (Col. J). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Total Effective Size is:

Col. K = Col. G + Col. J

Effective Size Representing Software Growth (Col. L): Enter the number of effective lines or functions that are included in the effective size estimate (Col. K) to capture software growth. Provide the definition of software growth used for the sizing estimate and the method used to estimate software growth.

Total Size (Col. M): Enter the number of new (Col. G) and pre-existing (Col. H) lines of code or functions. Total size represents the total amount of new software that would need to be developed for the new software baseline, if no code were to be reused as-is.

4. End of DI-SESS-81771

Appendix D Format M-1 (Revised)

Sizing, Schedule and Historical Information

1	2	3	4	5	6	7	8	9
					SIZE			
Total Delivery or Delivered Build	CSCI ID Number	CSCI Name	CSC Name	Development Contractor/ Subcontractor	Sizing Method	New Software	Total Pre-existing Software	Deleted Software

10	11	12	13	14	15	16	17
SIZE							
Modified Software	Redesign Required (%)	Reimple-mentation Required (%)	Retest Required (%)	Weight for Design Phase (%)	Weight for Implementation Phase (%)	Weight for Test Phase (%)	Effective Size for Modified Software

18	19	20	21	22	23	24	25
SIZE							
Reused As-is/ Lifted Software	Reuse As-is/Lift Factor Required (%)	Effective Size for Reused As-is/ Lifted Software	Total Effective Size	Effective Size Representing Software Growth	Total Size	Productivity	Software Language

26	27	28	29	30	31	32	33	34
HISTORICAL DATA		SCHEDULE (MONTHS ESTIMATED)						
Software Program Analogy	Productivity Range	Requirements Analysis	Preliminary Design	Detailed Design	Implementation	CSC-to-CSC Integration and Test	CSCI-to-CSCI Integration and Test	Total Schedule

INSTRUCTIONS FOR FORMAT M-1 (REVISED)
Sizing, Schedule, and Historical Information

1. **Total Delivery or Delivered Build** (Col. 1): If there are multiple delivered builds (blocks, increments, etc.), enter the build identifier for the sizing information provided. Enter "Total" if the sizing information represents the total delivery. A separate Format M-1 should be completed for each delivered build as well as the total delivery.

2. **CSCI ID Number** (Col. 2): Enter the identification numbers for the Computer Software Configuration Items (CSCIs). A separate row should be completed for each CSCI.

3. **CSCI** (Col. 3): Enter the names of the CSCIs.

4. **CSC** (Col. 4): Enter the names of the CSCs, if known. The size for the CSCs must sum to the size for the respective CSCI. A separate row should be completed for each CSC.

5. **Development Contractor/Subcontractor** (Col. 5): Enter the name of the contractor or subcontractor responsible for the development of each CSCI.

6. **SIZE:**

6.1 Sizing Method (Col. 6): Enter either source lines of code (SLOC) or function points (FP). Standard definitions for SLOC and FPs are provided below. Any non-standard definition should be fully explained on a separate sheet. If an alternative sizing measure is used, the counting method should be described in detail. This table can be adapted to accommodate an alternative measure, but the type of information requested in these instructions must be included.

6.1.1 Lines of Code: Non-comment lines of source code for the computer program. Source lines to include are: All executable source lines such as (1) Control, (2) Mathematical, (3) Conditional, (4) Deliverable Job Control, (5) Data Declaration Statements, and (6) Data Typing and Equivalence; and input/output/format. Source lines to exclude are: debug statements, continuation of single statement to multiple lines, machine/library generated statement, and non-deliverable test statements.

6.1.2 Function Points: Unadjusted function points, IFPUG compatible. Use this only if your size methods are function based rather than line based.

6.2 New Software (Col. 7): Enter the new non-comment lines of source code or the new number of unadjusted function points, IFPUG compatible for the computer program. New code is software developed from scratch and is not modified or reused as-is in any way from any pre-existing design or code. The new lines of source code or functions associated with reusing as-is any COTS/GOTS software (e.g., glue code, integration code) should be included in this column.

6.3 Total Pre-existing Software (Col. 8): Enter the number of lines of code or functions in a pre-existing software package (before reuse as-is/modification/deletion), including lines of code or functions that may not be pertinent to this program/system.

6.4 Deleted Software (Col. 9): Enter the number of lines of code or functions which will be deleted from the pre-existing software package (Col. 8). The deletion will be accomplished by physical omission or commenting out.

6.5 Modified Software (Col. 10): Enter the total number of lines of code or functions that will be modified from a pre-existing software package through re-design or re-implementation, and then integrated and tested in the new software product baseline. If there are multiple delivered builds, this number should represent the code developed in previous builds that may need to be modified or re-tested with the code being developed for the current build.

6.6 Redesign Required (Col. 11): Enter the percentage of the pre-existing software to be modified (Col. 10) that requires redesign to make this software functional within the new environment.

6.7 Reimplementation Required (Col. 12): Enter the percentage of the pre-existing software to be modified (Col. 10) that requires reimplementation (i.e., code and unit test) to make this software functional within the new environment.

6.8 Retest Required (Col. 13): Enter the percentage of the pre-existing software to be modified (Col. 10) that requires retesting (i.e., CSC integration/test and CSCI integration/test, but excluding CSCI-to-CSCI integration/test) to ensure this software functions within performance, reliability, and other criteria after the modifications.

6.9 Weight for Design Phase (Col. 14): Enter the percentage of the software development effort (i.e., design, implementation and test) attributed to the design phase. The weights (Col. 14-16) represent the phase distribution of effort, i.e., the distribution typically observed for each phase. Note that the sum of the weights for the design, implementation, and test phases must equal 100 percent.

6.10 Weight for Implementation Phase (Col. 15): Enter the percentage of the software development effort (i.e., design, implementation, and test) attributed to the implementation phase.

6.11 Weight for Test Phase (Col. 16): Enter the percentage of the software development effort (i.e., design, implementation, and test) attributed to the test phase.

6.12 Effective Size for Modified Software (Col. 17): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be modified (Col. 10) and are adjusted based on the applicable percentages (Col. 11-13) and weights (Col. 14-

16). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the Modified Software is:

Col. 17 = Col. 10 * ((Col. 11 * Col. 14) + (Col. 12 * Col. 15) + (Col. 13 * Col. 16))

The Offeror shall explain the method used for calculating effective size if it differs from the formula in this table.

6.13 Reused As-is/Lifted Software (Col. 18): Enter the number of lines of code or functions that will be reused as-is or lifted, with no modification of design or code, from a pre-existing software package. This column should also include any COTS/GOTS software that will be reused as-is. If COTS/GOTS software size is not encompassed in the Offeror's overall sizing methodology, the Offeror shall attach a separate sheet to explain the estimating methodology and fully discuss all efforts associated with the COTS/GOTS products in the Basis of Estimates (BOEs).

6.14 Reuse As-is/Lift Factor Required (Col. 19): Enter the percentage that is applied to the pre-existing software to be reused as-is/lifted (Col. 18) to estimate effective size. This percentage is similar in concept to the percentages for redesign required, reimplementation required, and retest required, but is a composite factor applied to the software that will be reused as-is/lifted. Reused as-is/lifted code, by definition, will not require modification. The percentage, if applicable to the Offeror's sizing methodology, applied to the COTS/GOTS software that will be reused as-is should be included in this column.

Reused as-is software products may require new code, such as glue code, wrappers, or plug-ins, or parameterization, but the software product itself will remain unchanged. The new lines of source code or functions associated with the software (e.g., glue code, integration code) should be included as New Software (Col. 7). The effort required to understand the product and its interfaces, integrate the product as part of a CSC and CSCI, and perform testing should be reflected in the percentage in order to estimate effective size. If these activities associated with the COTS/GOTS software are not part of the Offeror's software size methodology, the Offeror shall fully discuss these efforts in the BOEs.

6.15 Effective Size for Reused As-is/Lifted Software (Col. 20): Enter the number of lines of code or functions that represent the pre-existing lines or functions that will be reused as-is/lifted (Col. 18) and are adjusted based on the applicable percentage (Col. 19). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Effective Size for the Reused As-is/Lifted Software, including COTS/GOTS software, is:

Col. 20 = Col. 18 * Col. 19

The Offeror shall explain the method used for calculating effective size if it differs from the formula in this table.

6.16 Total Effective Size (Col. 21): Enter the number of lines of code or functions that represent all new lines or functions (Col. 7) as well as the pre-existing lines or functions that are modified (Col. 10) or reused as-is/lifted (Col. 18) and are adjusted based on the applicable percentages (Col. 11-13, 19) and weights (Col. 14-16). Effective size represents the software size equivalent to developing the code from scratch. The formula, in this spreadsheet, used for calculating Total Effective Size is:

Col. 21 = Col. 7 + Col. 17 + Col. 20

6.17 Effective Size Representing Software Growth (Col. 22): Enter the number of effective lines or functions that are included in the effective size estimate (Col. 21) to capture software growth. The Offeror shall provide the definition of software growth used for the sizing estimate and the method used to estimate software growth.

6.18 Total Size (Col. 23): Enter the number of new (Col. 7) and pre-existing (Col. 10 and 18) lines of code or functions. Total size represents the total amount of new software that would need to be developed for the new software baseline, if no code were to be reused as-is or modified. Note that Total Size does not include the pre-existing software that will be deleted.

7. **Productivity** (Col. 24): Enter the lines of code or functions per staff month estimated for this development effort. Attach a separate sheet that identifies the number of hours per staff month as well as which software development phases and labor categories from the lists below are included in this estimate.

Software development phases:

- Software requirements analysis (derived requirements)
- Preliminary design
- Detailed design
- Code and unit test
- Software component, CSC and CSCI integration and test
- CSCI-to-CSCI integration and test

Software development labor categories:

- Direct software management/supervision
- Software requirements analysts
- Software design, code and unit testers
- Software component, CSC and CSCI integration and testing personnel
- CSCI-to-CSCI integration and testing personnel
- Software engineering data, configuration management and quality assurance personnel

8. **Software Language** (Col. 25)**:** Enter the software language for the new code for each CSCI and CSC, if known.

9. **HISTORICAL DATA:**

9.1 Software Program Analogy (Col. 26): Enter the name(s) of any software development effort(s) similar to this effort.

9.2 Productivity Range (Col. 27): Enter the lines of code or functions per staff month observed for this development effort. Attach a separate sheet that identifies which software development phases and labor categories from the lists below are included in this metric.

Software development phases:

- Software requirements analysis (derived requirements)
- Preliminary design
- Detailed design
- Code and unit test
- Software component, CSC and CSCI integration and test
- CSCI-to-CSCI integration and test

Software development labor categories:

- Direct software management/supervision
- Software requirements analysts
- Software design, code, and unit testers
- Software component, CSC and CSCI integration and testing personnel
- CSCI-to-CSCI integration and testing personnel
- Software engineering data, configuration management and quality assurance personnel

10. **SCHEDULE (MONTHS ESTIMATED):** Enter the schedule estimates, in months, for each CSCI. Schedule estimates do not need to be included on a CSC basis. The schedule inputs below are based on a waterfall approach to software development. If the Offeror is proposing an alternative schedule structure, provide the equivalent type of information and a brief explanation of the schedule.

10.1 Requirements Analysis (Col. 28): Enter the number of schedule months estimated for the software requirements phase.

10.2 Preliminary Design (Col. 29): Enter the number of schedule months estimated for the preliminary and detailed design phase.

10.3 Detailed Design (Col. 30): Enter the number of schedule months estimated for the code and unit test phase.

10.4 Implementation (Col. 31): Enter the number of schedule months estimated for the implementation (i.e., code and unit test) phase.

10.5 CSC-to-CSC Integration and Test (Col. 32): Enter the number of schedule months estimated for the integration and test phase for the software components, CSCs and individual CSCIs.

10.6 CSCI-to-CSCI Integration and Test (Col. 33): Enter the number of schedule months estimated for the CSCI-to-CSCI integration and test phase.

10.7 Total Schedule (Col. 34): Enter the total number of elapsed schedule months estimated from the start of requirements analysis through CSCI-to-CSCI integration and test. The total number of months elapsed may be different from the sum of columns 28 through 33.

Glossary

ACA	after contract award
ADR	Alternate Dispute Resolution
AF	Air Force
AFP	award fee plan
API	application program interfaces
BOE	basis of estimate
C&A	certification and accreditation
CCM	CORBA Component Model
CDRL	Contract Data Requirements List
COTS	commercial off-the-shelf
CPR	Contract Performance Report
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CWBS	Contract Work Breakdown Structure
DFARS	Defense Federal Acquisition Regulation Supplement
DID	Data Item Description
DoD	Department of Defense
DR	Defect Report
ELSG	Electronic Systems Group
FAR	Federal Acquisition Regulation
FP	function points
GOTS	Government off-the-shelf
GUI	graphical user interface
IMP	Integrated Master Plan
IMS	Integrated Master Schedule
I/O	input/output
IPT	Integrated Product Team
IR&D	internal research and development
ITO	Information to Offerors
OTS	off-the-shelf

PCAG	Performance Confidence Assessment Group
PDR	Preliminary Design Review
PO	program office
ReMR	Reuse Management Report
RFI	Request for Information
RFP	Request for Proposal
RMP	Risk Management Plan
ROMP	Risk and Opportunity Management Plan
SDP	Software Development Plan
SEMP	System Engineering Management Plan
SLOC	source lines of code
SOO	Statement of Objectives
SOW	Statement of Work
SSET	Source Selection Evaluation Team
SQA	Software Quality Assurance
SQAE	Software Quality Assessment Exercise
TRD	Technical Requirements Document
TRL	technology readiness level