**DATA ITEM DESCRIPTION**

**Title: DESIGN REVIEW INFORMATION PACKAGE (DRIP)**

**Number: DI-SESS-81757A Approval Date: 20100406**

**AMSC Number: N9129 Limitation: N/A**

**DTIC Applicable: N/A GIPDEP Applicable: N/A Office of Primary Responsibility: SH/PEO IWS 1.0**

**Applicable Forms: N/A**

**Use/Relationship:** The Design Review Information Package (DRIP) is used by the government to determine and ensure that the emerging system has demonstrated sufficient maturity to enter the next phase of development (i.e., ).

This Data Item Description (DID) contains the format and content preparation instructions for the data product resulting from the discrete task requirements as delineated in the contract (i.e., ).

**Requirements:**

1.0 Format. The DRIP shall be in contractor’s format (i.e., ).

* 1. Content. The DRIP shall contain the following sections:
	2. Section 1 – Alternative System Review (ASR). The ASR section of the DRIP shall contain the following:

N/A

* 1. Section 2 – System Requirements Review (SRR). The SRR section of the DRIP shall contain the following:

N/A

* 1. Section 3 – System Functional Review (SFR). The SFR section of the DRIP shall contain the following:

N/A

* 1. Section 4 – Software Specification Review (SSR). The SSR section of the DRIP shall contain the following:

N/A

* 1. Section 5 – Hardware Preliminary Design Review (HPDR). The HPDR section of the DRIP shall contain the following:

N/A

* 1. Section 6 – Software Preliminary Design Review (SPDR). – renamed “Maturity and End User Engagement Review” to align with agile development. The SPDR section of the DRIP shall contain the following. Bullets below have been modified to better align with agile software development.
		1. Description of the system architecture (i.e., a view of the prospective prototype solution at this point of its maturity as captured in a lean architecture diagram delivered as a model in a digital engineering tool, and export from it, or another computer-aided software/systems engineering tool).
		2. An overview of the system requirements (i.e., requirements are initially reflected in the Epic Hypothesis. These are decomposed into features that get end user vetting through Acceptance Criteria (AC). The Product Owner(s) then create User Stories based on the features with their own AC. The requirements for a component are AC associated User Stories as documented in the issue tracking tool rather than a separate document.).
		3. Allocation of system requirements to SCIs (i.e., Software Configuration Items (SCI) usually manifest themselves as microservices and supporting infrastructure. Infrastructure as Code (IaC) may consist of another configuration control aspect. Allocation of features to SCI is captured in the issue management system such as Jira where code artifacts are associated with issue tracking entries containing User Stories. This allocation includes which versions they will be or are implemented.).
		4. User interfaces, including displays (i.e., these artifacts may consist of workflow diagrams, wireframes, screen mock ups, etc. as appropriate).
		5. Growths of requirements for each system build (i.e., these will be captured in the Features and User Stories and their associated AC).
		6. Interfaces between the CIs, broken out into builds (i.e., in software-intensive systems, these are often Application Programming Interfaces (API) which may be documented in modern approached such as OpenAPI).
		7. Layers of the architecture (if layers have been defined) (i.e., Often there will be API Gateways and other middleware, Policy Enforcement Points (PEP), Policy Decision Points (PDP), and other similar components within the architecture. These should be reflected in any lean architecture diagrams.).
		8. Locations of new and reused software, including COTS, NDI, legacy, and GFI (i.e., Location is often depicted in a Deployment Diagram or codified in the IaC).
		9. Plans for how reused CIs will meet IA requirements (i.e., This deals with Risk Management Framework (RMF) artifacts necessary to obtain an Interim Authority to Test (IATT) or Authority to Operate (ATO).).
		10. An overview of the SCI requirements (i.e., Configuration control is critical to agile software development. The issue tracking tool, software configuration management (SCM), code repository, and related tools ensure tight configuration management. Delivery consists of these repositories).
		11. Where the CI fits into the system architecture and system build plan (i.e., Service composition or orchestration and supporting APIs reveals interrelationships among components. The deployment diagram and other documentation should satisfy this line.).
		12. How the system requirements are allocated to the various CIs (i.e., The deployment diagram and other documentation should satisfy this line).
		13. A breakdown of how the requirements are allocated across builds (i.e., The issue tracking tool contents should reveal the allocation of features and User Stories across versions, and which versions are releases.).
		14. Documentation defining which SRS requirements is allocated to which builds. N/A
		15. Requirements changes made since SSR (i.e., All Feature and User Story changes (at whatever time) are tracking in the issue tracking system with version control and timestamps).
		16. Security critical requirements (i.e., Enabling features (including Non-Functional Requirements (NFR) such as security) are also captured in the issue tracking tool).
		17. Quality requirements (including reliability, availability and maintainability) (i.e., Enabling features (including Non-Functional Requirements (NFR) such as security) are also captured in the issue tracking tool).
		18. Requirements supporting test and analysis (i.e., the DevSecOps process includes unit, integration, and system verification testing. Issue tracking system entries and the code repository capture unit test and other artifacts.).
		19. Safety critical requirements (i.e., Enabling features (including Non-Functional Requirements (NFR) such as security) are also captured in the issue tracking tool).
		20. How the SCI build plan is consistent with the overall system build plan. N/A
		21. Dates when technical agreements are needed on the content of the various requirements and design documentation. N/A
		22. Description of the software development process/methods and tools that are to be used, as documented in the SDP (i.e., the DevSecOps process should be documented in company SOP or modified in a lean SDP).
		23. Explanation of any process and tool changes that have been made since the SSR, why the changes were made, and an assessment of the impact of these changes on the SDP. N/A
		24. Description of the measures to be used in assessing human performance and operator workload (cognitive/temporal/physical) (i.e., these are captured in the issue tracking tool as AC associated with Features and User Stories).
		25. Description of the intra-SCI landscape (i.e., Service composition or orchestration and supporting APIs reveals interrelationships among components. The deployment diagram and architecture diagram(s) capture these.).
		26. An overview of the system architecture (i.e., a view of the prospective prototype solution at this point of its maturity as captured in a lean architecture diagram delivered as a model in a digital engineering tool, and export from it, or another computer-aided software/systems engineering tool).

aa. Identification of all Computer Software Components (CSCs). N/A – redundant given the other rows in this description

bb. Documentation defining the SCI’s architecture. N/A – redundant given the other rows in this description

cc. Where, within the architecture, development/new and reuse components are located, including NDI, COTS, legacy equipment and GFI. N/A – redundant given the other rows in this description

dd. How reused components will meet IA requirements. N/A – redundant given the other rows in this description

ee. Where industry standards are applied (i.e., applicable standards should be captured as NFR in the issue tracking tool as AC in Features and User Stories).

ff. Maturity of the functional/subsystem design. N/A – redundant given the other rows in this description

gg. Description of the SCI’s performance requirements. N/A – redundant given the other rows in this description

hh. Summary of risk items identified for the SCI that impact successful completion of the SCI (i.e., risks are identified in the issue tracking tool).

ii. Test approaches planned for the SCI testing. N/A – redundant given the other rows in this description

jj. Lower-level tests to be performed on the components, from unit-level to top-level. N/A – redundant given the other rows in this description

* 1. Section 7 – Software Design Review (SDR). The SDR section of the DRIP shall contain the following:

N/A

Section 8 – Hardware Critical Design Review (HCDR). The HCDR section of the DRIP shall contain the following:

N/A

Section 9 – Software Critical Design Review (SCDR) – renamed “Maturity and End User Engagement Demo” to align with agile development. The SCDR section of the DRIP shall contain the following:

See Section 6 – *Maturity and End User Engagement Review*.

* 1. Section 10 – Test Readiness Review (TRR). The TRR section of the DRIP shall contain the following:

N/A

* 1. Section 11 – Functional Configuration Audit (FCA). The FCA section of the DRIP shall contain the following:

N/A

* 1. Section 12 – Physical Configuration Audit (PCA). The PCA section of the DRIP shall contain the following:

N/A

3.0 End of DI-SESS-81757A