ACQUISITION
AND SUSTAINMENT

THE UNDER SECRETARY OF DEFENSE
3010 DEFENSE PENTAGON
WASHINGTON, DC 20301-3010

JAN 03 2020

MEMORANDUM FOR: SEE DISTRIBUTION

SUBJECT: Software Acquisition Pathway Interim Policy and Procedures

Purpose. This interim policy establishes direction, responsibilities, and procedures for the management of the Software Acquisition Pathway pursuant to the authorities outlined in DoD Directive 5134.01, the July 13, 2018, Deputy Secretary of Defense Memorandum, and Section 800 of the National Defense Authorization Act for FY 2020. Further, this interim policy:

- Simplifies the acquisition model to enable continuous integration and delivery of software capability on timelines relevant to the Warfighter/end user.
- Establishes the Software Acquisition Pathway as the preferred path for acquisition and development of software-intensive systems.
- Establishes business decision artifacts to manage risk and enable successful software acquisition and development.

This interim policy will be replaced by issuance of a DoD Instruction within a year of signature of this memo.

Applicability. This interim policy applies to:

a) The Office of the Secretary of Defense (OSD), the Military Departments, the Office of the Chairman of the Joint Chiefs of Staff and the Joint Staff, the Combatant Commands, the Office of the Inspector General of the Department of Defense (DoD), the Defense Agencies, the DoD Field Activities, and all other organizational entities within the DoD (referred to collectively in this interim policy as the "DoD Components").
b) Acquisition, development, operations, and sustainment of any DoD software-intensive system approved to use this pathway in order to demonstrate the viability and effectiveness of capabilities for operational use not later than one year after the date on which funds are first obligated to acquire or develop new software capability. Software-intensive systems include software-only systems, such as Command & Control (C2) software or applications; weapon system software, such as Intelligence, Surveillance, and Reconnaissance (ISR) software, embedded mission planning software or embedded Situational Awareness software; and any other custom-developed software running on commercial or modified commercial hardware. Software programs that meet the definition of a Defense Business System (DBS) and primarily acquire Commercial-Off-The-Shelf (COTS) components will follow DoDI 5000.75 procedures but may elect to use this pathway for custom developed software.

Policy.

a) The overarching management principles that govern the Defense Acquisition System (DAS) are described in DoD Directive 5000.01 and DoDI 5000.02. The purpose of the DAS is to deliver effective and affordable solutions to the end user while enabling execution at the speed of relevance. To achieve that objective, the DoD will employ an adaptive acquisition framework composed of acquisition pathways, each tailored for the unique characteristics and risk profile of the capability being acquired. A general listing of statutory requirements associated with each of the pathways is located in the Milestone Document Identification (MDID). This interim policy describes the responsibilities of acquisition officials and the purpose and key characteristics of the software acquisition pathway.
b) The Component Acquisition Executive (CAE) may elect to use this pathway for acquisition of software-intensive systems or sub-systems. The decision authority shall document the decision in an Acquisition Decision Memorandum, which includes the rationale for using the software pathway, and submit to Office of the Under Secretary of Defense for Acquisition and Sustainment (OUSD (A&S)) to provide notification of use of this pathway.
c) Programs executing this pathway will not be subject to the Joint Capability Integration and Development System (JCIDS), except pursuant to a modified process specifically provided for the acquisition or development of software by the Vice Chairman of the Joint Chiefs of Staff, in consultation with Under Secretary of Defense for Acquisition and Sustainment and each service acquisition executive (as defined in section 101(a)(10) of title 3 10, United States Code. The Sponsor and Program Manager (PM) shall develop a Capability Needs Statement (CNS) and a User Agreement (UA) prior to acquisition of software capabilities. CNSs identify mission deficiencies, required enhancements to existing operational capabilities, features, interoperability needs, legacy interfaces, and other attributes required for new software-intensive systems or sub-systems or upgrades to existing systems or sub-systems. Each DoD Component will develop a streamlined process that results in iteratively developed requirements documented in the CNS. UAs capture a commitment between the PM office, the sponsor, and end user(s) of the system. The CNS and UA documents are meant to be flexible products, periodically updated to reflect the capabilities baseline, and will be developed and approved via an expedited Component validation process and expedited joint process if the Joint Staff determines that there are joint implications. CAEs will ensure the approved CNS document is available in the Knowledge Management and Decision Support system.
d) The PM shall develop an Acquisition Strategy that outlines the program's approach to performing software acquisition in increments consistent with the user's requirements that results in demonstrating the viability and effectiveness of capabilities for operational use not later than one year after the date on which funds are first obligated to acquire or develop new software capability, and to continuously engineer and deliver capability updates at least annually. The Acquisition Strategy shall include a lifecycle sustainment strategy that promotes continuous engineering and delivery throughout the software lifecycle, as well as an Intellectual Property (IP) strategy (see paragraph (m)). The acquisition strategy must also clearly identify test and evaluation requirements that have been fully coordinated with the test community.

e) PMs shall leverage Enterprise Level Services as a first choice, if available, before creating unique services. These may come from portfolio, Component, or DoD-wide services and/or commercial providers. Enterprise Services include technical services (e.g., Platform as a Service (PaaS), common containers/virtual machines, or infrastructure) and business services (e.g., contract vehicles or requirements management).

f) The PM shall ensure that software teams use iterative and incremental software development methodologies (such as Agile or Lean), and use modern technologies (e.g. DevSecOps pipelines) to achieve automated testing, continuous integration and continuous delivery of user capabilities, frequent user feedback/engagement (at every iteration if possible), security and authorization processes, and continuous runtime monitoring of operational software. Software development teams shall consider the program's lifecycle objective and use sound software practices to achieve improved software quality (e.g., manage technical debt, actively refactor design and code, and create effective modular open systems approaches to support future capabilities). Embedded systems should tailor software development activities to this pathway to the greatest extent possible.

g) The PM shall ensure that software teams address persistent cyber security requirements starting at program inception and include a risk-based lifecycle management approach to secure development, secure capabilities, and secure lifecycle to address software vulnerabilities. The PM shall also ensure that automated test processes, tools and/or environments are certified by the test community and the automated test process includes, to the greatest extent practicable, frequent and recurring tests that address cyber and software assurance considerations throughout the software lifecycle. The automated build scripts and test results shall be available to government testers, so they can reuse/recreate any test artifact.

h) The PM shall ensure that software teams develop and/or leverage a Software Architecture that aligns, supports, and reflects the continuous engineering of capability over the software lifecycle. Interoperability and interfaces with legacy systems are to be addressed early in the software lifecycle. The PM shall ensure that software teams demonstrate that the architecture complies with good design principles including but not limited to modular open systems architecture, and supports frequent iterative capability releases. Periodic evaluation of the architecture should include refactoring and adopting newly available technologies when appropriate to continuously "pay down" technical debt.

i) The PM and the sponsor shall define and develop a Minimum Viable Product (MVP) and if appropriate, a develop a Minimum Viable Capability Release (MVCR) to deliver software products that will meet an initial set of Warfighter/end-user needs, and a Product Roadmap to guide the delivery of remaining capabilities. The MVP is an early version of the software that has just enough working features to meet basic minimum functional capabilities. The goal of an MVP is to quickly deliver basic capabilities into users' hands for evaluation, feedback, and improvements. The MVCR is a set of features suitable to be delivered to an operational environment that provides value and capability to the Warfighter/end user on a reduced delivery timeline. The goal of the MVCR is to deliver initial warfighting capabilities that can be fielded to enhance some mission outcome(s). Subsequent capability deliveries must occur at least annually.

j) The PM, in collaboration with developmental and operational test organizations, shall seek to streamline, automate and integrate contractor test, Developmental Test and Evaluation (DT&E), and Operational Test (OT). The test and user communities shall participate in early program development and test planning activities. The software build and automated test process and associated data should be leveraged to enable timely satisfaction of DT and OT test criteria and user acceptance.

k) The PM shall identify, collect, and use management, cost, schedule, and performance metrics to enable effective program execution by the PM and other stakeholders. Metrics collection should leverage automated tools to the maximum extent practicable. The minimum set of metrics used to manage the program should include process efficiency, software quality, software development progress, cost, and capability delivery (i.e. value). Programs using this pathway shall report a minimal set of data to OUSD (A&S) on a quarterly basis as defined in the Software Acquisition Pathway Guidance located at www.dau.edu/aaf/.

l) The Sponsor and PM shall conduct value assessments at least annually that will provide a determination of whether the operational value of the software from the end user perspective, as informed by testing, is commensurate with the resources invested. Value assessments inform future resource investment decisions.

m) The PM shall develop and maintain an Intellectual Property (IP) strategy in accordance with DoDI 5010.44 that is tailored to the unique circumstances of the program to achieve its supportability and cyber security requirements. The PM should be aware of and understand the rights and obligations of both the Government and industry, as well as the system architecture and life-cycle requirements, in order to negotiate for computer software deliverables and license rights. In the case of open source software and software developed in whole or in part at Government expense, the IP strategy should include, to the maximum extent practicable, negotiation for and periodic delivery of executable and source code and all associated scripts, tools, databases, libraries, other software executables, and anything else necessary to compile, test, debug, deploy, and successfully operate the software. The PM will ensure any restrictive markings on software and software documentation deliverables are consistent with the Government's license rights prior to acceptance by the Government. The PM shall approve the use of any commercial or proprietary software prior to its insertion into the software developed for the government, and the PM will protect all IP associated with commercial or proprietary software. The PM, to the maximum extent practicable, shall require that any commercial or proprietary software used in or interoperable with software developed for the Government has documented open interfaces to allow for technology insertion, and to support the use of modular open systems approaches, as applicable.

n) The PM shall ensure technical manuals are developed/verified incrementally and synchronized with software deliveries throughout the software development lifecycle; and shall also ensure that receiving users and military units are trained to the appropriate level of proficiency and readiness to successfully execute the individual and collective tasks necessary to accomplish the mission supported by the software. The PM shall ensure delivery of technical operator and maintainer manuals required to operate and maintain the system.

o) The decision authority should formally tailor in applicable processes and documentation consistent with modern software development principles to fit the size, scope, and

complexity of the development activity to achieve cost, schedule, performance, and delivery objectives.

p) It is the implementing components' responsibility to ensure that the program is conducted in accordance with all applicable laws and regulations. For example, procedures for compliance with statutory requirements applicable to DoD programs that acquire information technology (including acquisition of software), such as the Clinger Cohen Act, are established in DoDI 5000.02 Change 5, and in the Acquisition of Information Technology DoDI upon cancellation of DoDI 5000.02 Change 5. A general listing of statutory requirements associated with each of the pathways is located in the MDID at www.dau.edu .

q) All safety critical software standards and guidance apply when using this pathway policy.

Responsibilities. The organizational responsibilities associated with this policy include

DEFENSE ACQUISITION EXECUTIVE (DAE) WILL:

a. Establish policy and provide guidance for the Software Acquisition Pathway.
b. Serve as the Approval Authority for programs that request to use the Software Acquisition Pathway or delegate authority to a Component Acquisition Executive.
c. Act as decision authority for programs in the Software Acquisition Pathway when appropriate, or delegate decision authority to a Component Acquisition Executive.
d. Determine when a program is not appropriate for the Software Acquisition Pathway and direct that the program be executed using another acquisition pathway.

DOD COMPONENT HEADS WILL:

a. Establish a streamlined and coordinated requirements, budget, and acquisition process to support rapid fielding of software applications and of software upgrades to embedded systems for operational use.

COMPONENT ACQUISITION EXECUTIVES (CAEs):

a. Serve as the Approval Authority for programs that request to use the Software Acquisition Pathway or delegate authority to a designated official.
b. Act as decision authority for programs in the Software Acquisition Pathway when appropriate, or delegate decision authority to a designated official.
c. Ensure that the procedures in this interim policy are implemented.

DECISION AUTHORITY:

a. Serve as the Decision Authority for programs that adopt the Software Acquisition Pathway and shall implement the procedures in this interim policy.
b. Designate a PM for each Software Acquisition Pathway program.

c. Submit sufficient program metrics data to support adequate DoD leadership insight into the operation of the pathway and support development and/or evolution of tools for modeling software acquisition cost and performance.

PROGRAM MANAGER WILL:

a. Develop and submit an acquisition strategy, timing and scope of decision points, metrics, and required documentation to the decision authority.
b. In conjunction with the user community representative, develop a UA that identifies the appropriate level of user involvement and expectations for a collaborative method of evolving capability delivery timelines.
c. In conjunction with the user community, the test and evaluation community, and cost estimating community, develop an acquisition strategy and program cost estimate.

SPONSOR, AS THE END USER REPRESENTATIVE FOR THE SOFTWARE WARFIGHTING CAPABILITY, WILL:

a. In conjunction with the PM, develop a UA that identifies the user involvement strategy, any operational constraints, and expectations for a collaborative method of evolving capability delivery and/or deployment timelines.
b. Ensure that the UA is executed to provide end-user input on capability needs, prioritization of work, and approve working software.
c. In conjunction with the PM, develop a CNS and provide advocacy for program resources.

Procedures. See Enclosure.

Guidance. The Software Acquisition Pathway interim policy guidance can be found at www.dau.edu/aaf .

Releasability. Cleared for public release.

Ellen M. Lord

Enclosure:
Interim Procedures for Operation of the Software Acquisition Pathway

DISTRIBUTION:

CHIEF MANAGEMENT OFFICER OF THE DEPARTMENT OF DEFENSE
SECRETARIES OF THE MILITARY DEPARTMENTS
CHAIRMAN OF THE JOINT CHIEFS OF STAFF
UNDER SECRETARIES OF DEFENSE
CHIEF OF THE NATIONAL GUARD BUREAU
GENERAL COUNSEL OF THE DEPARTMENT OF DEFENSE
DIRECTOR OF COST ASSESSMENT AND PROGRAM EVALUATION
INSPECTOR GENERAL OF THE DEPARTMENT OF DEFENSE
DIRECTOR OF OPERATIONAL TEST AND EVALUATION
CHIEF INFORMATION OFFICER OF THE DEPARTMENT OF DEFENSE
ASSISTANT SECRETARY OF DEFENSE FOR LEGISLATIVE AFFAIRS
ASSISTANT TO THE SECRETARY OF DEFENSE FOR PUBLIC AFFAIRS
DIRECTOR OF NET ASSESSMENT
DIRECTORS OF THE DEFENSE AGENCIES
DIRECTORS OF THE DOD FIELD ACTIVITIES

ENCLOSURE

PROCEDURES

To allocate resources to the most relevant capability needs, DoD or Component leadership must make software acquisition and development investment decisions within a framework that addresses tradeoffs between capabilities, affordability, risk tolerance, and other considerations. The Software Acquisition Pathway (Figure 1) outlines key business decision artifacts that support the software acquisition and development activities.

Software Acquisition Pathway. The Software Acquisition Pathway has two phases (Planning and Execution) as depicted in Figure 1.

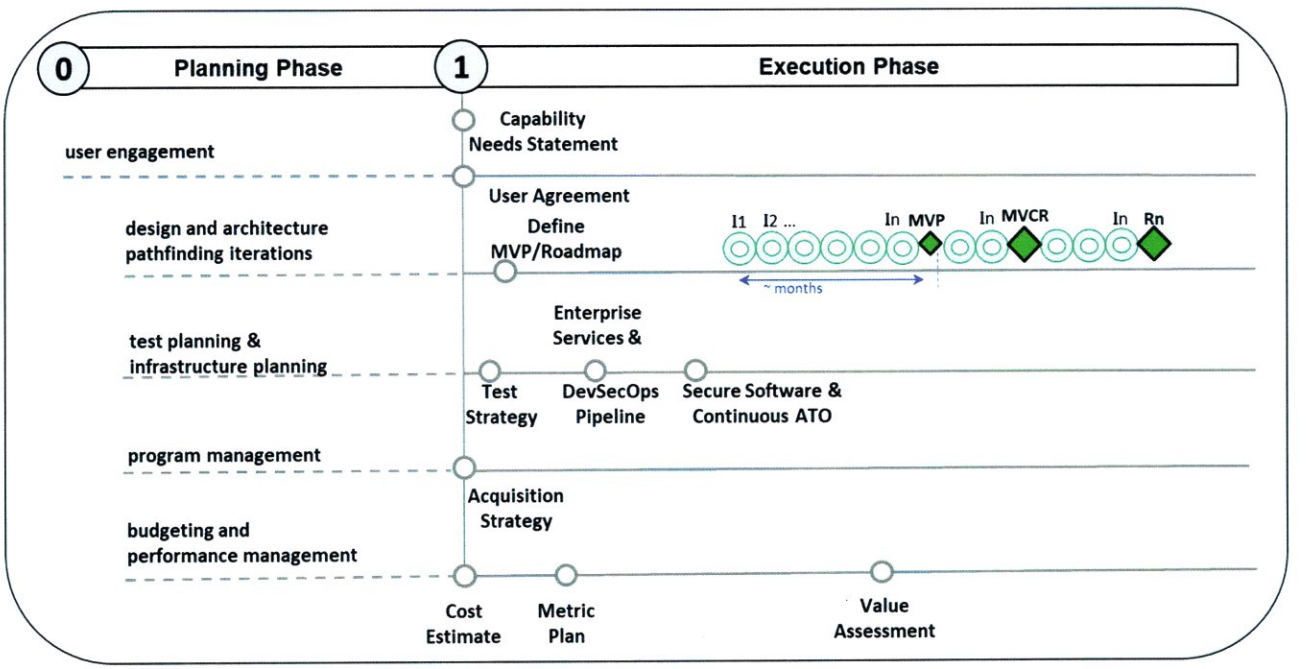

Figure 1 - Software Acquisition Pathway Phases

Planning Phase

The planning phase focuses on understanding the users'/systems' needs and planning the approach to deliver capabilities to meet those needs. This requires active engagement with the users to understand their concepts of operations, environment, external systems with which the required capability must interface, interoperability requirements, threats, existing capabilities, and other specific needs. The software development team shall begin to plan the software design and architecture, leveraging enterprise services to the maximum extent possible. Government and/or vendor systems engineering and software engineering teams may develop capabilities to

explore possible solutions, confirm architecture decisions, and solicit user feedback. The chosen software development methodology shall incorporate security as a persistent requirement and include a risk-based lifecycle management approach to address software vulnerabilities through secure development, secure capabilities, and secure lifecycle.

The program office will develop strategies for acquisition, funding, contracting, Intellectual Property (IP), test and evaluation, systems engineering, software security, and sustainment in a single or minimum set of tailored documents. The team will estimate costs, identify funding, and develop metrics and value assessment plans. These planning efforts should be tightly aligned but can occur independently to support individual business decisions.

The Intellectual Property (IP) strategy should be tailored to meet the needs of the program throughout the lifecycle, with specific emphasis on acquiring the computer software and license rights required to support the acquisition strategy, and with consideration of the IP rights of both the Government and industry. To the maximum extent practicable, the strategy should include the requirement for periodic delivery of all source code developed in whole or in part at Government expense (e.g., after each iteration or each release)), along with any other software or documentation necessary to compile, test, debug, deploy, and successfully operate the software. The IP strategy should also identify technological areas where IP may result from Government investment, and treat those appropriately. Prior to software transition to a different organization (e.g. a DoD Lifecycle Software Center), the PM should require delivery of all the software developed at government expense, including all software capability descriptions (e.g. features, story points, use cases) and all as-built architecture and design products, traceability products, interface definitions including interfaces to proprietary software elements, and any other requisite documentation. This facilitates managing program risk, understanding IP rights, and supports the software transition to another organization for sustainment. The PM shall define the software transition plan in a lifecycle software support plan and shall capture the final software package at the point of transition in the product roadmap.

Business decision artifacts that decision authority must approve at the end of the Planning Phase so that the program can move into execution include, but are not limited to the artifacts described in the following paragraphs.

Capability Needs Statement: A high-level capture of need that provides enough information to define the software solution space, considering the threat environment. The CNS must accompany the UA. The CNS must be periodically updated to reflect the actual baseline as necessary.

User Agreement: An agreement between the operational and acquisition communities to gain commitment to continuous user involvement and assign decision-making authority in the development and delivery of software capability releases, as well as operational tradeoffs among software features, cadence, and management of the requirements backlog. The UA will ensure proper resourcing of operational user involvement, which should occur as frequently as possible to support the development process.

Acquisition Strategy: A description of the overall approach to the acquisition, including key decisions such as:

- the contract type(s) and contracting strategy to be used
- the approach to cost-effectively obtaining appropriate intellectual property rights and maintaining an open system architecture
- development and test platforms, test resources and infrastructure
- identification of dependencies with other programs, either in development or in operation
- cadence for operational delivery of the software being acquired
- a roadmap that describes short-term plans with fidelity while outlining longer term objectives, among others
- identification of sustainment factors such as for upgrades, security and performance

The acquisition strategy documents the program approach to obtain the required capabilities with acceptable risk, aiming to field a software capability within one year of initiation of the execution phase, and to collect data on fielded software to facilitate continuous engineering. Follow-on software development efforts will be established in increments of one year or less and evaluated based on software performance in the field. The acquisition strategy should also align to modern software development principles, emphasizing iterative deliveries of quality software early and often, and incorporating frequent feedback from users and other stakeholders.

Cost Estimates: An initial cost estimate for the program lifecycle. The estimate should consider the technical content of the program described in the CNS, user agreement, and acquisition strategy; factors associated with continuous capability delivery; and factors associated with an emphasis on software security, software quality, and functionality. Where applicable, cost and software data reporting, to include software resources data reports, must be submitted in accordance with the policies and procedures in DoDI 5000.73. The initial cost estimate shall be completed prior to entry into the execution phase to support contract initiation, and shall be updated at least annually thereafter.

Execution Phase

Design decisions made during the Planning Phase are critical (e.g., systems and software architecture, software integration strategy, software development factory and pipelines) and will have a major impact on future program cost and schedule during the Execution Phase. The decision authority should consider developing planning completion criteria based on the business decision artifacts to assess program readiness to execute this software acquisition pathway. The decision authority can assess program achievement of planning completion criteria at the end of the Planning phase and approve program transition to Execution Phase 1.

The execution phase focuses on scoping, developing, and deploying an MVP and MVCR to the Warfighter/end user as quickly as possible. MVPs provide users with working software to demonstrate initial capabilities, test external interfaces as needed, accelerate learning, and shape needs/requirements, designs, and future iterations. While time to MVP is of high importance, utility of the functionality is diminished if quality control, testing, and user engagement are not integrated into the software development process. For example, security and testing should be integrated into the process to deliver the most value. Sponsors, users, acquirers and developers

shall maintain active continual engagement throughout the software development process. Annually, PMs and Sponsors will use holistic program health data from metrics, value assessments, test results, etc. to determine if the effort should continue as planned.

The software development team further refines capabilities and features with the users to decompose capabilities into a prioritized backlog of functional and performance requirements, features, mission threads, and/or user stories, use cases, etc. The development efforts leverage enterprise software development services to the maximum extent possible to accelerate deliveries, reduce costs, and improve security and interoperability. The team works with key stakeholders to achieve a continuous authority to operate or an aggressively streamlined approval process for each software delivery. The PM tracks metrics to manage the software teams' progress and convey insights to key stakeholders.

Following delivery of the MVP, the software development team will iteratively design, develop, test, and deliver capabilities that meet the users' and/or systems' highest priorities. The team will establish a cadence for iterating on the broader design, architecture, and infrastructure; allow adequate time for software refactoring; and plan for the MVCR. Users should be involved in the planning of each iteration and evaluation of the software at the end of each iteration if possible. The MVCR focuses on delivery of a minimum set of features that represent a deployable release to mission operations. Subsequent releases are intended to add further sets of capabilities and align with system development schedules. The team uses product roadmaps to convey the integration and synchronization of activities to align with system development activities and planned operational releases. Program and development teams should assess themselves at every iteration to ensure the strategic goals have not changed and are being met, and to remain responsive to emergent and changing requirements by delivering more frequent and timely releases to end users.

Depending on the software practices used by the team, subsequent deliveries could be continuously integrated and delivered when ready or iteratively queued in a staging/testing environment and transitioned to a production/operational environment via an established, consistent timeframe agreed to by the users and other stakeholders. Once a delivery cadence is agreed upon, adherence to it will become a key indicator of program health and effectiveness. Programs must maintain rigorous focus on maintaining delivery cadence. The amount of capability delivered during each rigid, time-bound delivery can and should be allowed to change, but quality should be fixed. Test and evaluation should be automated to the maximum extent possible by defining tests up front, with tight alignment of government and vendor testers. Active user engagement is critical throughout development, delivery, and capability release to shape priorities and provide insights into operations, feedback on early capability iterations, design mock-ups, and previous developments to ensure rapid delivery of capabilities that will have an impact on the mission.

Government and vendor personnel should continuously improve their processes, practices, and roles via small empowered teams and should scale efforts across teams, tracking improvements via established metrics. Issues, errors, and defects identified during operations should be captured in the program's backlog to be addressed in future development iterations. The software architecture should be continuously monitored and updated to reflect as-built design using

automated tool scans as the preferred means. Continuous monitoring may feed value assessments to balance investments between short-term capabilities delivery and the need for longer term enduring solutions.

Business decision artifacts that decision authority should consider during the execution phase include, but are not limited to the artifacts described in the following paragraphs.

Enterprise Services and DevSecOps Pipeline: The program should develop a plan to leverage enterprise software development services, if available, and use modern tools and technology to develop and deliver software. This includes the architectural tradeoff decision to use an enterprise-level platform focused on continuous integration and continuous delivery that can provide integrated software tools, services, and standards that enable partners and programs to develop, deploy, and operate applications in a secure, flexible, and interoperable fashion. Enterprise Services will span multiple programs and will be scaled and resourced to support demand. The program should establish enterprise agreements as appropriate, and make enterprise services discoverable to assist in the evaluation and use of the software by other programs if the capability to discover services exists. If a PM determines that the program requires an enterprise service that is not available in the enterprise, the PM should add the solution to the enterprise repository. Use of enterprise services will enable rapid start-up, real-time delivery, and scalability.

MVP, MVCR, and Product Roadmap: An MVP is working software, delivered to the Warfighter/end user that provides a meaningful first version of the software capability as agreed to by the users. The MVP must be defined early in the execution phase with active user engagement, and may evolve as users interact with the software and user needs are better understood. The delivery of an MVP demonstrates to the users and other acquisition stakeholders that the program is capable of cost-effectively delivering reasonable increments of needed functionality.

An MVCR is the first version of the software that contains sufficient capability to be fielded for operational use. The MVCR contains the minimum set of functions that provide value to the Warfighter/end user and requires the minimum amount of software development effort. MVCRs provide a baseline set of capabilities to validate assumptions and determine if the proposed system will deliver the expected or acceptable business/mission value. The Warfighter/end-user shall determine when the MVCR and subsequent software releases are to be delivered operationally and will ensure that each operational release can achieve interoperability certification, if applicable.

The product roadmap is a plan to facilitate the delivery of multiple releases of software over time to reach full capability. It identifies which releases will be evaluated or fielded for operational use, and the point at which the software transitions to sustainment. The product roadmap can evolve over time to address changing threats and technology. The roadmap informs the planned evolution of the solution capabilities and architecture, is updated periodically, communicates the capabilities/feature sets targeted for delivery at discrete times in an iterative fashion, identifies the point at which the software transitions to sustainment, and aligns with Warfighter/end-user needs.

Test Strategy: The test strategy defines the process by which capabilities, user features, user stories, use cases, etc., will be tested and evaluated to satisfy DT criteria and demonstrate operational effectiveness for OT to the maximum extent possible. The strategy shall:

- Include system-level performance requirements, non-functional requirements, and the metrics that will be used to verify that the system will meet user needs.
- Identify key independent test organizations and their roles and responsibilities, and establish agreements on how they will be integrated early into the development activities and throughout the software lifecycle.
- Identify the tools and resources necessary to assist in data collection and transparency to support DT and OT.
- For embedded software, include a risk assessment of any safety critical implications as well as a companion mitigation strategy, and a strategy, including resources, to describe testing and evaluation as the software transitions from the development environment to the test environment to the operational environment.
- Assess the findings/recommendations of the PMs Software System Safety assessment accomplished in accordance with Military Standard 882E, "Standard Practice for System Safety".

Automated testing should be used at the unit level, for Application Programming Interface (API) and integration tests, and to the maximum extent possible for user acceptance and to evaluate mission effectiveness. Automated testing tools and automated security tools should be accredited by an Operational Test Authority as "fit for purpose". Continuous runtime monitoring of operational software will provide additional data collection opportunities to support test and continuous OT.

***Secure Software & Cyber Security Plan*:** PMs shall establish and/or leverage a secure software development pipeline and a security lifecycle plan. Software tests shall be run automatically where possible, and at a predetermined cadence sufficient to ensure that cybersecurity controls and other considerations are addressed early and throughout the acquisition process. PMs shall establish the conditions to enable a continuous Authority to Operate (ATO) where appropriate. Ensuring software security includes secure development (coding, test, identity/access management, supply chain risk management), secure capabilities (software patching, encryption, runtime monitoring, and logging) and secure lifecycle management (vulnerability management and configuration control). Automated build scripts and test results shall be available to the test community so that critical verification functions (e.g., performance, reliability, etc.), validation functions (e.g., effectiveness, suitability and survivability) can be assessed iteratively and incrementally. The automated cyber testing shall be designed to support a continuous ATO if possible or an aggressive accreditation process otherwise; and shall be augmented with additional testing where appropriate in accordance with the DoD Cybersecurity Guidelines.

Metrics Plan: The metrics plan identifies key metrics that allow the PM and other stakeholders to manage cost, schedule, and performance. It also organizes metrics by common types (or classes) and provides guidance on how to read and interpret each metric. Each program shall tailor the set of metrics for the unique considerations of the program. All software acquisition

programs must have a set of core metrics in addition to the existing requirements outlined in the Software Resources Data Report (SRDR).[1]

Value Assessment: The sponsor and user community shall perform a value assessment at least annually and provide justification for resources expended to deliver capabilities. End users will determine if the mission improvements and/or efficiencies realized from the delivered software capabilities are worth the investment. The decision authority will use the value assessments to measure progress on the program and inform resourcing decisions. Additional interim value assessments can be performed at any periodicity agreed to by the user and the PM.

[1] http://acqnotes.com/acqnote/careerfields/software-resources-data-report

GLOSSARY -- ACRONYMS AND DEFINITIONS

G.1. ACRONYMS

ATO	authority to operate
C2	command & control
CAE	Component Acquisition Executive
CNS	capability needs statement
DAE	Defense Acquisition Executive
DAS	Defense Acquisition System
DoDI	Department of Defense Instruction
DT	developmental test
IP	Intellectual property
MVCR	minimum viable capability release
MVP	minimum viable product
OT	operational test
PM	Program Manager
UA	user agreement
USD(A&S)	Under Secretary of Defense (Acquisition and Sustainment)

G.2. DEFINITIONS. Unless otherwise noted, these terms and their definitions are for the purpose of this interim policy.

Adaptive Acquisition Framework. A defined series of pathways for material capability acquisition to adopt that recognizes the different characteristics of capabilities to be acquired and describes the general approach and expectations that results in accelerated delivery of improved secure, prioritized capability to the end user with acceptable risk.

Backlog. A product backlog is a list of the new capabilities / features, changes to existing Capabilities / Features, bug fixes, infrastructure changes or other activities that a team may deliver in order to achieve a specific outcome. The development team will work with the user community to decompose and prioritize the roadmap Capabilities/Features.

An iteration backlog is a list of the new stories, changes to existing stories, bug fixes, infrastructure changes or other activities that a team may deliver in order to achieve a specific outcome, within an iteration cadence.

Capability. Higher level solutions typically spanning multiple releases. Capabilities are made up of multiple Features to facilitate implementation.

Capability Needs Statement. Capability Needs Statements identify mission deficiencies, or enhancements to existing operational capabilities, features, interoperability needs, legacy

interfaces and other attributes required for new software intensive systems or upgrades to existing systems.

Continuous ATO. The core concept of continuous ATO is to build software security into the software development methodology so that the ATO process (as with the testing process) is done alongside development. If done correctly, an ATO is nearly guaranteed once the software is release ready.

Continuous Engineering. A practice that merges requirements, design, development, quality assurance, security, test, integration, delivery, and deployment to operations into a single, continuous set of processes to continually, or iteratively, provide working functional systems to internal and external customers and to deliver high quality software more frequently.

Defect. A defect is a condition in a (software, system, hardware) product which does not meet its requirements or end-user expectation, causes it to malfunction or to produce incorrect/unexpected results, or causes it to behave in unintended ways.

- Escaped Defects are defects detected, or resolved, after release of the product and version containing the defect.

- Contained Defects, also known as Saves, are defects detected and resolved before release of the product and version containing the defect.

Note: a project may decide to add detected defects to the backlog and resolve them in a later iteration or release.

DevSecOps. An organizational software engineering culture and practice that aims at unifying software development (Dev), security (Sec) and operations (Ops). The main characteristic of DevSecOps is to automate, monitor, and apply security at all phases of the software lifecycle: plan, develop, build, test, release, deliver, deploy, operate, and monitor. In DevSecOps, testing and security are shifted left through automated unit, functional, integration, and security testing – this is a key DevSecOps differentiator since security and functional capabilities are tested and built simultaneously.

DevSecOps Pipeline. A collection of DevSecOps tools, upon which the DevSecOps process workflows can be created and executed. DevSecOps tools are comprised of a tailored series of software products configured to integrate end-to-end software definition, design, development, test, delivery, and potentially deployment in a highly automated and secure way.

Embedded Software. Software with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints, or software applications embedded in a platform (e.g. air vehicle, ground vehicle, or ship). In the context of this policy, embedded software does not apply to firmware or software dedicated to controlling devices.

End user. Those who will ultimately use the software solution. Users convey operational concepts and requirements/needs, participate in continuous testing activities, and provide feedback on developed capabilities.

Enterprise Services. A term for services that have the proper scope to play a productive role in automating business processes in enterprise computing, networking, and data services. Responsibility for these functions is generally above the program manager.

Features. A Service or distinguishing characteristic of a software item (e.g., performance, portability, or functionality) that fulfills a stakeholder need and includes benefit and acceptance criteria within one release. Features are used to complete capabilities and are comprised of multiple Stories (or tasks, use cases, etc.).**Government Developed Source Code.** The product resulting from the implementation of needed capabilities into executing software that is created by the government workforce.
Iteration. A small internal time block in which the development team develops and demonstrates a set of Stories or use cases. An iteration is a full development cycle that can result in a Release. In some methodologies, an iteration is called a Sprint.

Lifecycle sustainment strategy. A strategic vision that shapes the overall development, deployment, and improvement strategy and processes so that the most important capabilities are preserved in the software while new and emerging capabilities are delivered over the entire software life cycle.

Man-Machine Interface. (also Human-Computer Interaction, Human-Machine Interaction). The actions, reactions, and interactions between humans and other system components. This also applies to a multi-station, multi-person configuration or system. Term also defines the properties of the hardware, software or equipment which constitute conditions for interactions.

Minimum Viable Capability Release. A set of features and\or capabilities suitable to be delivered to an operational environment. It provides value and capability on a reduced delivery timeline. The MVCR is analogous to a Minimum Marketable Product (MMP) in commercial industry.

Minimum Viable Product. An early version of the software that has just enough working features to meet basic minimum functional capabilities and fill a user's need. The goal of an MVP is to quickly deliver basic capabilities into users' hands for evaluation, feedback, and improvements.

Mission Operations. The real-world environment within which the needed military capability is required and must perform.

Mission Thread. A sequence of end-to-end activities and events, given as a series of steps that accomplish the execution of one or more capabilities that the system of systems supports.

Modern software development practices. Modern software development practices make it easier to focus teams for rapid results. Continuous engagement with end users encourages the development team to concentrate on usability and most important functionality. Cloud architecture facilitates more rapid project starts with both speed and scale as needed.

Operational Release. A release that has been approved for operational use.

Product Roadmap. A high-level visual summary that maps out the vision and direction of product offerings over time. It describes the goals and features of each software iteration and increment.

Program Manager. Designated individual with responsibility for and authority to accomplish program objectives for development, production, and sustainment to meet the user's operational needs.

Release. A grouping of Capabilities and\or Features that can be used for demonstration or evaluation. A release may be internal for integration, testing, or demonstration; or external to system test or as user delivery. A release may be based on a time block or on product maturity.

Safety-critical. Used in the context of software system safety, is a term applied to a condition, event, operation, process, or item whose mishap severity consequence is either Catastrophic or Critical (e.g., safety-critical function, safety-critical path, and safety-critical component).

Software-intensive. A system in which software represents the largest segment in one or more of the following criteria: system development cost, system development risk, system functionality, or development time.

Sponsor. The organization that holds the authority and advocates for needed end user capabilities and associated resource commitments.

Task. Steps to be completed to satisfy a Story or use case.

Technical Debt. A concept in software development that reflects the extra development work that arises when code that is easy to implement in the short run is used instead of applying the best overall solution. It represents the cost that is incurred by implementing a software solution that is expedient rather than choosing a better approach that would take longer. Technical debt often accrues over the life of a program as code is expanded and patched. Technical debt can often be "paid down" by investing in refactoring or re-architecting the code.

Use Case. In software and systems engineering, a use case is a list of actions or event steps, typically defining the interactions between a user and a system (or between software elements), to achieve a goal. Use cases can be used in addition to or in lieu of user stories.

User Agreement. A commitment between the Program Manager's systems engineering and software program teams, the sponsor, and end-user(s) of the system.

User Story. A small desired behavior of the system based on a user scenario that can be implemented and demonstrated in one iteration. A story is comprised of one or more tasks. In software development and product management, a user story is an informal, natural language description of one or more features of a software system. User stories are written from the perspective of an end user or user of a system.

Value Assessment. The sponsor and user community shall perform a value assessment at least annually and provide justification for resources expended to deliver capabilities. End users will determine if the mission improvements and/or efficiencies realized from the delivered software

capabilities are worth the investment. The decision authority will use the value assessments to measure progress on the program and inform resourcing decisions. Additional interim value assessments can be performed at any periodicity agreed to by the user and the PM.

Weapon System. A combination of one or more weapons with all related equipment, materials, services, personnel, means of delivery and deployment (if applicable) required for self-sufficiency.[i]

[i] Definition from DoD Dictionary of Military and Associated Terms