OBJECTIVE: Develop a Common Core Combat System (CCCS) modular architecture framework and software component Application Program Interface (API) capable of serving as (i) a core combat system architectural upgrade within the current AEGIS system and (ii) the basis for a new platform-agnostic combat system core implementation for Future Surface Combatant (FSC) platforms.

DESCRIPTION: The Navy needs to expand its sea-based advantage through increased capability. This need can be addressed by providing technology that has the potential to improve ship combat effectiveness and efficiency by providing an updated 21st century combat system implementation capable of handling the increased complexity of today’s battlespace environment. Such a combat system will provide increased task automation utilizing Artificial Intelligence (AI) technology and Autonomous software concepts, thus reducing the management complexity of the overall battlespace as presented to the combat systems operator. This may allow for potential reduced shipboard manning requirements, increased duty time productivity by reducing the potential stress and fatigue levels experienced by the combat systems operator while performing his duties, and improved affordability.

The current implementation of the AEGIS Combat System has fundamental architectural limitations deriving from its initial hardware and software design constraints. These architectural limitations have forced the use of inefficient “bolt-on” style modernization modifications and enhancements to meet evolving 21st century threats. Currently available commercial systems and software, which might be considered for adaptation to partially address the Navy’s ever-growing needs for advanced situational awareness (e.g., the FAA Air Traffic Control System hardware and software), are dated in their designs. They lack the flexibility and track capacity required to adequately address tactical requirements. Specifically, the currently available commercial technology mentioned above is limited in that it lacks the capability to track, identify, and manage complex air, surface and subsurface entities and threats present in a combat environment. Additionally, such commercial systems have no intrinsic ability to provide the other critical weapon and sensor coordination services provided by an effective combat system implementation. Since no viable commercial alternatives exist or can be adapted, it becomes necessary for the Navy to pursue a different avenue of exploration.

A fundamental redesign of the core architecture of the current AEGIS combat systems is needed to enable the efficient, rapid, and cost-effective addition of new capabilities, such as multi-platform sensor and weapons coordination, off-board and organic on-the-fly sensor and weapons integration, built-in cyber resilience, and real-time fault recovery. The Navy needs a software execution framework and API capable of supporting the real-time addition, removal, and control of the major software components constituting a CCCS implementation. The framework and API must also be capable of the real-time dynamic installation, removal, and control of any software modules supporting multiple on-board and off-board sensor and weapons capabilities, as well as any requisite multi-platform integration packages, all without disrupting the real-time performance of the currently executing combat system. Due to the severe tactical real-time constraints placed on the performance of combat-system related software, such as the need for real-time fire-control quality engagement tracking data, target de-confliction data, and identify, friend, or foe (IFF) verification data all synchronized across multiple platforms, the development of a suitable software framework and API capable of meeting the requirements outlined...
above will demand the application of innovative software architectural design techniques and concepts. To address this critical need, a
new CCCS architecture is currently in the planning stage with the intent of providing a modular set of platform-agnostic common
combat system services. This CCCS implementation, when supplemented by platform-specific sets of weapon, sensor, and
communications capabilities modules, will constitute a new, modular, and dynamically adaptable CCCS design that can evolve to
meet future emerging threats in a rapid and cost-effective manner. The innovative technology sought will improve the reliability and
efficiency of the re-designed AEGIS Combat System, improving multi-platform tactical coordination, as well as significantly improving
battlespace situational awareness, thus reducing the management complexity of the overall battlespace. This allows for a reduction in
the number of platforms needed in a specific tactical arena by improving the overall tactical efficiency of the battle group as a whole.

Development of a CCCS combat system modular architectural framework and software component API is critical to the future needs of
the Navy. The architectural framework and software component API will be combined with: (i) an appropriate CCCS Ecosystem
software execution model and software application/component API; (ii) an appropriate CCCS multi-platform coordination architecture
and inter-platform data exchange algorithm set; and (iii) an appropriate multi-platform coordinated/synchronized Distributed
Common Operational Picture (DCOP) subsystem to provide a comprehensive architectural model for a complete, versatile platform-
agnostic “core” combat system implementation. The term “core” means this system implementation will provide combat system
services and capabilities that are considered necessary to satisfy common tactical warfighting requirements, which span various and
diverse surface combatants. This core combat system implementation, when configured with the appropriate surface warfighting
platform-specific sensor and weapons capability software modules, will be capable of fulfilling the functional warfighting capabilities and
requirements needed to support current U.S. Navy surface platforms well into the future.

The CCCS core architectural model and software framework must first and foremost be capable of supporting underlying CCCS
functions (i.e., those combat systems capabilities and functions that are common across the vast majority of major surface combatant
platforms, such as track management and weapons engagement planning). The required capabilities and functionality are currently
outlined in the PEO IWS1 Combat Systems Top Level Requirements (TLR) documentation suite (currently in draft form). This
documentation will be made available on request from NAVSEA PEO IWS 1SP for any Phase II efforts. The intent of this SBIR topic is on
the design and prototype of an overarching software architectural framework for a CCCS capable of supporting “on-the-fly” addition,
deletion, or upgrade of the modular software capabilities within any arbitrary CCCS implementation (hereafter referred to as an
“instance”). The solution will allow for the retention of the real-time response capabilities needed to support modern weapons and
sensors control and data linkages in a modular manner and within a modular software architecture with native run-time “on-the-fly” no
impact combat systems new capability installation.

Commonly used combat systems services will be implemented within the CCCS via the use of software capability modules. The intent
is to support all combat systems capabilities, which would benefit from frequent or periodic updates or upgrades (in order to address
our rapidly changing threat environment) through the use of such software capability modules. Those modules, which provide
common capabilities across multiple platforms (such as Anti-Aircraft Warfare (AAW), Anti-submarine Warfare (ASW), Anti-Surface
Warfare (ASuW), and Ballistic Missile Defense (BMD) track management, weapons assignment, scheduling and control.), as well as
modules providing platform-specific weapons and sensor capabilities, will be dynamically upgradable at run-time. This allows for rapid
capability upgrade via a capability software module run-time installation or removal management facilities supported by the CCCS.
Design of an overarching CCCS architecture and software framework capable of supporting “on-the-fly” addition, deletion, upgrade,
and control of such capabilities within any arbitrary CCCS instance, with each capability implemented through the dynamic (i.e., run-
time) installation of a loadable software “capability” module is the focus. Installation, removal, activation, and deactivation of such
software modules within an executing CCCS implementation should have no adverse effect on the real-time performance of the CCCS
system or the services it provides to the host platform and operator at the time those changes are implemented. An exemplar of this
type of no-impact behavior during runtime installation and removal of capabilities within an executing system can be observed in the
Linux operating system kernel module control facilities (kernel 4.4 and above), such as the insmod, rmmod, depmod, Ismod, modinfo,
and modprobe commands [Ref. 2&3]. The process of installing, removing, or otherwise controlling CCCS services and capabilities
within an executing CCCS installation should be easily executed by combat systems watch personnel without the need for specially
trained software maintenance personnel.

The CCCS system architecture shall incorporate a native suite of inter-module and inter-combat-system (CS)-application data
exchange and communications services (IM/ICS service suite). The IM/ICS service suite is intended to support real-time data exchange
between capability modules internal to a specific CCCS instance (such as a ship CCCS instance), as well as across multiple CCCS
instances (like CCCS instances hosted on multiple warfighting platforms). Access to this suite of data exchange services shall be using a
well-defined and documented IM/ICS API, which provides a level of software abstraction between the client capability module software and
the CCCS service provision layer and associated software capability modules. This allows the potential future upgrade of core
CCCS services without affecting existing compiled CCCS capability modules. When supporting data exchange and communications
services between ship and non-organic (such as off-board) CCCS instances, the IM/ICS API shall internally utilize a set of services
One example of a potential software capability module for use within the CCCS is a Multi-platform DCOP support module, which would provide real-time access to a battlespace-wide distributed (multi-platform) common operational picture. Access to such a battlespace common operational picture (COP) will supply critically needed situational awareness (SA) to the combat systems watch stander. The CCCS would support the dynamic run-time installation of such a DCOP capability via the use of a CCCS dynamically loadable DCOP subsystem module. This DCOP capability module will make use of the IM/ICS API and service suite provided by CCCS to achieve COP coherency across multiple warfighting platforms.

Both the CCCS architectural model and its associated APIs shall be well documented and conform to open systems architectural principles and standards [Ref. 4]. Implementation attributes should include scalability and the ability to run within the computing resources available within the AEGIS Combat Systems BL9 or hardware later environment.

Any delivered software prototypes will be compatible with the C++ programming language and capable of running in a Linux (Redhat RHEL 7.5/Fedora 29/Ubuntu 18.4.1 or later) software execution environment as a standalone application (that is, no critical dependencies on network-based remotely hosted resources, save for sensor data emulators). The prototype CCCS implementation will demonstrate the following abilities: (i) the ability to install, remove, and control (i.e., start and stop) various CCCS capability software modules in an executing system with no impact on system performance; (ii) the ability to demonstrate real-time performance when executing various data exchange operations between organic capability software modules; and (iii) the ability to demonstrate real-time performance when executing various data exchange operations between organic and non-organic CCCS-hosted capability software modules.

The Government will furnish AEGIS BL9 or later combat systems design or architecture documentation, draft Common Core Combat Systems TLR documentation, and any other appropriate material needed to assist in the development of the design effort.

Work produced in Phase II may become classified. Note: The prospective contractor(s) must be U.S. Owned and Operated with no Foreign Influence as defined by DOD 5220.22-M, National Industrial Security Program Operating Manual, unless acceptable mitigating procedures can and have been be implemented and approved by the Defense Security Service (DSS). The selected contractor and/or subcontractor must be able to acquire and maintain a secret level facility and Personnel Security Clearances, in order to perform on advanced phases of this contract as set forth by DSS and NAVSEA in order to gain access to classified information pertaining to the national defense of the United States and its allies; this will be an inherent requirement. The selected company will be required to safeguard classified material IAW DoD 5220.22-M during the advance phases of this contract.

PHASE I: Design, develop, and deliver an initial concept design for a CCCS architectural model and software framework capable of meeting the subsystem and API requirements and capabilities outlined in the Description. Establish feasibility to accomplish the requirements through modeling and analysis. Develop a Phase II plan. The Phase I Option, if exercised, will include the initial design specifications and capabilities description to build a prototype solution in Phase II.

PHASE II: Produce a software prototype that is compatible with the C++ programming language and capable of running in a Linux (Redhat RHEL 7.5/Fedora 29/Ubuntu 18.4.1 or later) software execution environment as a standalone application (i.e., no critical dependencies on network-based remotely hosted resources, save for sensor data emulators). Demonstrate that the prototype CCCS implementation has the following abilities: (i) the ability to install, remove, and control (i.e., start and stop) various CCCS capability software modules in an executing system with no impact on system performance; (ii) the ability to demonstrate real-time performance when executing various data exchange operations between organic capability software modules; and (iii) the ability to demonstrate real-time performance when executing various data exchange operations between organic and non-organic CCCS-hosted capability software modules. Ensure that the prototype meets the capabilities outlined in the Description during a functional test to be held at an AEGIS or Future Surface Combatant (FSC) prime integrator-supported Land Based Test Site (LBTS) provided by the Government, representing an AEGIS BL9 or newer combat system hardware environment. It is probable that the work under this effort will be classified under Phase II (see Description section for details).

PHASE III DUAL-USE APPLICATIONS: Support the Navy in transitioning the CCCS system software for Navy use. Integrate the CCCS system software along with other CCCS compliant capability software modules into a prototype combat system implementation on a virtualized hardware environment within an AEGIS-compliant land-based testbed.

This capability has potential for dual-use capability within the commercial Air Traffic Control system in future development of an air traffic control system capable of rapid upgrade to handle increasingly complex traffic control patterns.
REFERENCES:


KEYWORDS: Software Execution Environment; Platform-agnostic Combat System Core Implementation; Application Data Exchange and Communications Services; Tactical Warfighting Requirements which Span Various and Diverse Surface Combatants; Platform-specific Sensor and Weapons Capability Software Modules; Software Module Run-time Installation or Removal Management Facilities