Tactical Service-Oriented Architecture  
Combat Operations Center   
Command and Control  
Software Package XXXX

Software Design Description



Prepared By:

SPAWAR Systems Center, Atlantic

P.O. Box 190022

North Charleston, South Carolina 29419-9022

Distribution Statement C: Distribution authorized to U.S. Government Agencies and their contractors for administrative or operational use as of 11 May 2017. Other requests for this document shall be referred to Marine Corps Systems Command, PM, MC3, PdM, MC2S.

Destruction Notice: For unclassified, limited documents, destroy by any method that will prevent disclosure of contents or reconstruction of the documents

Tactical Service-Oriented Architecture  
Combat Operations Center   
Command and Control  
Software Package XXXX

Software Design Description

Submitted By:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
<Name, Organization>  
<Position (if available)>

**Reviewed By**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
<Name, Organization>  
<Position (if available)>

**Approved By**:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
<Name, Organization>

<Position (if available)>

Record of Changes

\***A** - ADDED **M** - MODIFIED **D** - DELETED

| Revision Identifier | Date | Figure, Table or Paragraph Number | A  M  D | Title or Brief Description | Change Request Number |
| --- | --- | --- | --- | --- | --- |
| .01 |  |  | A  M  D | Initial Document |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

[1 Introduction 1-1](#_Toc335983264)

[1.1 Scope 1-1](#_Toc335983265)

[1.2 Identification 1-1](#_Toc335983266)

[1.3 System Overview 1-2](#_Toc335983267)

[1.4 Document Overview 1-2](#_Toc335983268)

[2 Reference Documents 2-1](#_Toc335983269)

[3 Design Considerations 3-1](#_Toc335983270)

[3.1 Assumption and Dependencies 3-1](#_Toc335983271)

[3.2 General Constraints 3-1](#_Toc335983272)

[3.3 Goals and Guidelines 3-2](#_Toc335983273)

[3.4 Development Methods 3-2](#_Toc335983274)

[4 Components 4-1](#_Toc335983275)

[5 System Architecture 5-1](#_Toc335983276)

[5.1 Subsystem Architecture 5-1](#_Toc335983277)

[6 Detailed System Design 6-1](#_Toc335983278)

[6.1 Classification 6-1](#_Toc335983279)

[6.2 Definition 6-1](#_Toc335983280)

[6.3 Responsibilities 6-1](#_Toc335983281)

[6.4 Constraints 6-1](#_Toc335983282)

[6.5 Composition 6-1](#_Toc335983283)

[6.6 Uses/Interactions 6-1](#_Toc335983284)

[6.7 Resources 6-2](#_Toc335983285)

[6.8 Processing 6-2](#_Toc335983286)

[6.9 Interface/Exports 6-2](#_Toc335983287)

[7 Traceability 7-1](#_Toc335983288)

[8 Notes 8-1](#_Toc335983289)

[Appendix A Acronyms and Abbreviations A-1](#_Toc335983290)

List of Tables

[Table 1‑1 Configuration Management (CM) Identification 1-1](#_Toc335982993)

[Table 2‑1 Reference Documents 2-1](#_Toc335982994)

[Table 7‑1: Traceability 7-1](#_Toc335982995)

**List of Figures**

No table of figures entries found.

# Introduction

## Scope

<Provide the Scope of this document.>

## Identification

This is the Software Design Description (SDD) for <Software Name/Version>.

Table 1‑1 Configuration Management (CM) Identification

|  |  |
| --- | --- |
| CM Official Name | <To be completed by CM representative> |
| CM Acronym | <To be completed by CM representative> |
| Version Number | <To be completed by CM representative> |
| Increment Number | <To be completed by CM representative> |
| Material Provider  Name/Address | SPAWAR Systems Center Atlantic  Code 53160, Bldg. 198 PO Box 190022  North Charleston SC 29419-9022 |

## System Overview

The TSOA is a collection of software components that comply with the Marine Air Ground Task Force (MAGTF) Command and Control (C2) Software Architecture and, when composed for a system, provides C2 and Situational Awareness capabilities. When the TSOA is fully fielded, the end-state is an end-to-end software C2 system that supports the seamless flow of data on the battlefield, focusing on C2 platforms as the center of data integration. This provides Marines a software architecture that can quickly adapt to the mission in order to ensure an efficient and effective traversing of the Observe, Orient, Decide and Act (OODA) Loop. An example of this heightened efficiency is the ability to tailor the flow and presentation of data for a commander and his staff, supporting both their Standard Operating Procedures (SOPs) and decision making heuristics. The underlying software architecture is scalable and tailorable due to its basis in software engineering Best Practices such as Service-Oriented Architecture (SOA). TSOA entails a truly interoperable Marine Corps C2 capability that is relevant across the full range of military operations, from Major Theater War to Policing Actions to Humanitarian Assistance and Disaster Relief operations.

In order to meet evolving operational C2 needs, MAGTF C2 Systems Applications (MC2SA) will employ a SOA approach in acquiring and designing services that will not only provide the Ground Combat Element (GCE) the ability to link services together, but also the ability to easily add new services. TSOA goals are to:

* Provide an improved, standards-based approach to achieve information sharing
* Increase agility through cost and resource-effective re-use of services and capabilities
* Eventually replace the information “stovepipes” of the currently deployed Tactical Data Systems (TDSs) with open architecture-based integration

## Document Overview

<Summarize the contents of this document, stating its purpose, intended audience, etc. as deemed appropriate.>

# Reference Documents

This section lists the number, title, revision, and date of all documents referenced in this plan. This section also identifies the source of all documents not available through normal government stocking activities.

Table 2‑1 Reference Documents

| **Document Number** | **Document Title** | **Revision/Date** |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

.

# Design Considerations

<This section should describe many of the issues that need to be addressed or resolved before attempting to devise a complete design solution.>

## Assumption and Dependencies

<Describe any assumptions or dependencies regarding the software and its use. These may concern such issues as:

* Related software or hardware
* Operating systems
* End user characteristics
* Possible and/or probable changes in functionality
* External interface dependencies>

## General Constraints

<Describe any global limitations or constraints that have a significant impact on the design of the system's software (and describe the associated impact). Such constraints may be imposed by any of the following (the list is not exhaustive):

* Hardware or software environment
* End-user environment
* Availability or volatility of resources
* Standards compliance
* Interoperability requirements
* Interface/protocol requirements
* Data repository and distribution requirements
* Security requirements (or other such regulations)
* Memory and other capacity limitations
* Performance requirements
* Network communications
* Verification and validation requirements (testing)
* Other means of addressing quality goals
* Other requirements described in the requirements specification>

## Goals and Guidelines

<Describe any goals, guidelines, principles, or priorities that dominate or embody the design of the system's software. Such goals might be:

* The KISS principle ("Keep It Simple, Stupid!")
* Emphasis on speed versus memory use
* Working, looking, or "feeling" like an existing product>

<For each such goal or guideline, unless it is implicitly obvious, describe the reason for its desirability. Feel free to state and describe each goal in its own subsection if you wish.>

## Development Methods

<Briefly describe the method or approach used for this software design. If one or more formal/published methods were adopted or adapted, then include a reference to a more detailed description of these methods. If several methods were seriously considered, then each such method should be mentioned, along with a brief explanation of why all or part of it was used or not used.>

# Components

<List all the software components that the application depends on and that are not found in the standard hosting environment. The components function in the application should also be described.>

# System Architecture

<This section should provide a high-level overview of how the functionality and responsibilities of the system were partitioned and then assigned to subsystems or components. Do not go into too much detail about the individual components themselves (there is a subsequent section for detailed component descriptions). The main purpose here is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together to provide the desired functionality.

At the top-most level, describe the major responsibilities that the software must undertake and the various roles the system (or portions of the system) must play. Describe how the system was broken down into its components/subsystems (identifying each top-level component/subsystem as well as the roles/responsibilities assigned to it). Describe how the higher-level components collaborate with each other in order to achieve the required results. Do not forget to provide some sort of rationale for choosing this particular decomposition of the system (perhaps discussing other proposed decompositions and why they were rejected). Feel free to make use of design patterns, either in describing parts of the architecture (in pattern format), or for referring to the elements of the architecture that employ them.

If there are any diagrams, models, flowcharts, documented scenarios or use-cases of the system behavior and/or structure, they may be included here (unless you feel they are complex enough to merit being placed in the Detailed System Design section). Diagrams that describe a particular component or subsystem should be included within the particular subsection that describes that component or subsystem.>

## Subsystem Architecture

<If a particular component is one which merits a more detailed discussion than what was presented in the System Architecture section, provide that more detailed discussion in a subsection of the System Architecturesection (or, it may even be more appropriate to describe the component in its own design document). If necessary, describe how the component was further divided into subcomponents, and the relationships and interactions between the subcomponents (similar to what was done for top-level components in the System Architecturesection).

If any subcomponents are also deemed to merit further discussion, then describe them in a separate subsection of this section (and in a similar fashion). Proceed to go into as many levels/subsections of discussion as needed in order for the reader to gain a high-level understanding of the entire system or subsystem (but remember to leave the gory details for the Detailed System Design section).

If this component is very large and/or complex, you may want to consider documenting its design in a separate document and simply including a reference to it in this section. If this is the option you choose, the design document for this component should have an organizational format that is very similar (if not identical to) this document.>

# Detailed System Design

<Most components described in the System Architecturesection will require a more detailed discussion. Other lower-level components and subcomponents may need to be described as well. Each subsection of this section will refer to or contain a detailed description of a system software component. The discussion provided should cover each of software component listed below.>

## **Classification**

<Insert the kind of component, such as a subsystem, module, class, package, function, file, etc.>

## **Definition**

<Insert the specific purpose and semantic meaning of the component. This may need to refer back to the requirements specification.>

## **Responsibilities**

<List the primary responsibilities and/or behavior of this component. What does this component accomplish? What roles does it play? What kinds of services does it provide to its clients? For some components, this may need to refer back to the requirements specification.>

## **Constraints**

<List any relevant assumptions, limitations, or constraints for this component. This should include constraints on timing, storage, or component state, and might include rules for interacting with this component (encompassing preconditions, postconditions, invariants, other constraints on input or output values and local or global values, data formats and data access, synchronization, exceptions, etc.)>

## **Composition**

<Insert a description of the use and meaning of the subcomponents that are a part of this component.>

## **Uses/Interactions**

<Insert a description of this component’s collaborations with other components. What other components is this entity used by? What other components does this entity use (this would include any side-effects this entity might have on other parts of the system)? This concerns the method of interaction as well as the interaction itself. Object-oriented designs should include a description of any known or anticipated subclasses, superclasses, and metaclasses.>

## **Resources**

<Insert a description of any and all resources that are managed, affected, or required by this entity. Resources are entities external to the design, such as memory, processors, printers, databases, or a software library. This should include a discussion of any possible race conditions and/or deadlock situations, and how they might be resolved.>

## Processing

<Insert a description of precisely how this component goes about performing the duties necessary to fulfill its responsibilities. This should encompass a description of any algorithms used, changes of state, relevant time or space complexity, concurrency, methods of creation, initialization, and cleanup, and handling of exceptional conditions.>

## Interface/Exports

<List the set of services (resources, data, types, constants, subroutines, and exceptions) that provided by this component. The precise definition or declaration of each such element should be present, along with comments or annotations describing the meanings of values, parameters, etc. For each service element described, include a description (or provide a reference) of its important software component attributes (Classification, Definition, Responsibilities, Constraints, Composition, Uses, Resources, Processing, and Interface).>

<Much of the information that appears in this section is not necessarily expected to be kept separate from the source code. In fact, much of the information can be gleaned from the source itself (especially if it is adequately commented). This section should not copy or reproduce information that can be easily obtained from reading the source code (this would be an unwanted and unnecessary duplication of effort and would be very difficult to keep up-to-date). It is recommended that most of this information be contained in the source (with appropriate comments for each component, subsystem, module, and subroutine). Hence, it is expected that this section will largely consist of references to or excerpts of annotated diagrams and source code. Any referenced diagrams or source code excerpts should be provided at any design reviews.>

# **Traceability**

<This section should map design elements in the SDD to the requirement or User Story it satisfies.>

Table 7‑1: Traceability

| **User Story ID (Forge ID)** | **Title** | **Description** | **Vendor**  **Verification Method** | **CSC** | **SDD Paragraph** |
| --- | --- | --- | --- | --- | --- |
| artf<xxxxx> | <user story title> | <user story description> | <verification method> | <CSC> | <paragraph> |
| artf<xxxxx> | <user story title> | <user story description> | <verification method> | <CSC> | <paragraph> |
| artf<xxxxx> | <user story title> | <user story description> | <verification method> | <CSC> | <paragraph> |

# Notes

<This section is for additional design/architecture considerations poorly aligned with the previous sections.>

Acronyms and Abbreviations

| Term | Definition |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |