ARCHITECTURE EVALUATION FOR UNIVERSAL COMMAND AND CONTROL

John Klein, Patrick Donohoe, Phil Bianco, Harry Levinson, Reed Little, and Jason Popowski

June 2020

Introduction

The allocation of command-and-control (C2) functions to systems, subsystems, and computing nodes impacts architectural qualities such as latency and composability. The SEI developed an analysis method to assess allocations in existing C2 systems and to reason about design choices and tradeoffs during the design of new C2 systems.[1] A program manager or portfolio manager (for example, a Program Executive Office or a service) can use the analysis method early in the life cycle to

- compare architecture drivers across systems and understand the types of requirements changes that could allow increased architecture commonality across systems;
- identify *edge cases* – systems with requirements or constraints that preclude architecture commonality;
- focus architecture design changes to improve commonality; and
- evaluate new architecture approaches.

The method prioritizes the architecture qualities for the system or systems under analysis, and then rates candidate solutions (allocations) based on how well each candidate satisfies each architecture quality. The prioritization and ratings are defined and synthesized using the Analytic Hierarchy Process (AHP), a mature multi-criteria decision-making method. The synthesis produces a ranking of the candidate allocations for each system. Sensitivity analysis allows exploration of tradeoffs in requirements (by altering the prioritization of the architecture qualities) and design (by modifying a candidate solution) that would improve, for example, the reusability or replaceability of system elements.

We tested the method with a set of scenarios that use notional requirements and system architectures, constructed to stress the analysis method and to illustrate capabilities of the method. We found that the method should facilitate analysis of the requirements-architecture tradespace, providing early insights into which requirements and design decisions affect the solution ranking and providing a framework for stakeholders to identify and understand concerns and impacts. However, the solution ranking produced by AHP only identifies a one solution as best *among the candidates*; it cannot determine if that best solution meets all requirements. This AHP-based method must be part of a broader suite of analyses that screen candidates prior to AHP ranking synthesis or perform further analysis on the candidate ranked best by AHP synthesis.

---

[1] This work was sponsored by OSD (R&E) as part of the Fully Networked Command, Control & Communications (FNC3) Modernization Priority.

This white paper begins by discussing the C2 function allocation that defines candidate solutions and the architecture qualities that are affected by those allocation design decisions. We next introduce the Analytic Hierarchy Process and present an example of using the method to explore a tradespace. We conclude with a summary of the strengths and limitations of the analysis method and discuss possible future directions for this work.

C2 Function Allocation

Functions perform information processing, decision making, data access, and user interaction tasks necessary for a C2 system to accomplish its mission. This study focused on systems where all functions are executed automatically without human intervention.

C2 functions may be derived from or organized by a process model such as the Sense-Process-Compare-Decide-Act model [Lawson 1981], the Observe-Orient-Decide-Act model [Boyd 2018], or the Find-Fix-Track-Target-Engage-Assess model [Tirpak 2000]. We found no authoritative taxonomy of C2 functions; however, from our experience and from reviewing the literature, particularly Lawson [1981], Dekker [2005], and Eisenberg [2018], we developed a list of twenty C2 functions for defensive systems, including object detection, target identification, sensor resource management, engagement decision, and effector activation. Further work is needed to align this list with taxonomies such as the National Information Exchange Model [NIEM 2019] or the Joint C3 Information Exchange Data Model [MIP 2012].

Having identified C2 functions, we can allocate the functions to subsystems of a system and to computing resources. We focused on military C2 systems [Lawson 1981], as distinguished from other types of command and control (for example, C2 as a more general management activity [Alberts 2006]). Figure 1 decomposes a military C2 system, showing three types of subsystems or components: Applications, Sensors, and Effectors. C2 functions are mapped to a subsystem of one of these types to define the *subsystem allocation*.

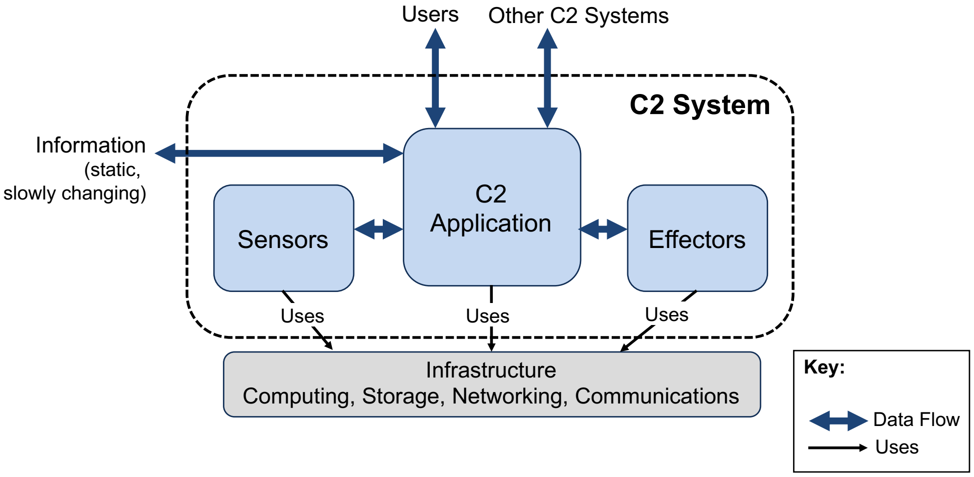

Figure 1: C2 Software Architecture Context Diagram

The software in a C2 system requires computing, storage, and networking resources. We call an instance of these resources a *node*, and a node's type characterizes the quantity of compute and storage available, the relative physical location of the node, and the types of network connections that the node instance uses to communicate with other nodes. Node types are summarized in Table 1. C2 functions are mapped to a node of one of these types to define the *node allocation*.

Table 1: Node Types

Type	**Characteristics**
Tactical	An embedded computing environment with limited size, weight, and power (SWaP) constraints A demand for hyperresponsive exchange of local C2 function message traffic Employment within disconnected, intermittent, limited-bandwidth environments
Expeditionary	A movable but stationary environment (e.g., a land-based combat operations center or maritime-based guided-missile destroyer) Relaxed SWaP constraints and some elasticity[2] with respect to compute and storage resources Employment within constrained access to an electromagnetic spectrum, geographic location and native infrastructure, and minimum bandwidth requirements
Enterprise	A stationary data center with high elasticity and access to availability zones (if applicable) Unconstrained internal / central data computing and message exchanges Communication variability within a system of systems (e.g., incoming / outgoing exchanges) defined by interfaces

In order to provide the most generality when specifying allocations, we do not define or constrain a direct relationship between subsystems and nodes—they are related only by the C2 function instances mapped to each. This allows the mappings to be constructed for any existing system. There may be cases where the same set of functions is mapped to both a single subsystem and a single node, or cases where a set of functions is allocated to a subsystem and also subsets of that set are allocated to several nodes (or vice versa, first to a node and then to several subsystems).

Architecture Quality Criteria

There are many quality criteria that can be used to judge an architecture [Bass 2013]. In order to allow efficient use of the analysis method, we wanted to keep the number of criteria small. Starting with a survey of more than 70 architecture evaluations conducted by the SEI and a literature review (especially Lawson [1981], Dekker [2005], and Eisenberg [2018], and various Naval Postgraduate School theses on C2), we filtered the criteria based on those that are *observable* and provide *differentiation*. An observable criterion can be measured, either qualitatively or quantitatively, using the subsystem and node allocations defined above. A criterion provides differentiation if an allocation can be judged as better or

---

[2] Elastic computing uses a pool of shared resources, usually with more capacity than needed for steady-state operation. Capacity can be dynamically and automatically assigned to running services as needed and returned to the pool when no longer needed.

worse with respect to the criterion. For example, scalability of compute and storage resources may be harder to achieve at the tactical edge because of size, weight, power, and networking constraints, so a node allocation that uses only tactical nodes would be less scalable than one that uses enterprise nodes. Such filtering produced the initial set of quality criteria shown in Table 2.

Table 2: Initial Set of Architecturally Significant Quality Criteria

Quality Criterion Label	Definition	What Makes an Allocation Better?
C2 Loop Latency	The ability to reliably and predictably achieve sub-second response time (detect/react/deploy)	No essential functions are executed remotely.
Composability / Integrability	The ease of replacing the sensor and/or effector and/or C2 application with a different COTS/GOTS product (e.g., MOSA)	Functions allocated to the sensor and effector do not require system-specific information (data, configuration, algorithms, etc.) to execute (good information-hiding decomposition). Coupling is from C2 application to sensor and effector, with no direct coupling between sensor and effector.
Modifiability / Extensibility	The ease with which a software system can accommodate change to existing capabilities and new capabilities	Related functions are allocated together to localize changes. Tactical < Expeditionary < Enterprise
Scalability	The ease of adding and using compute and storage resources to accomplish a bigger and/or more complicated mission	Access to elastic compute/storage resources is provided. Function execution can be distributed across multiple nodes.
Decision Parameter Updatability	The ease of making updates to parameters used for automated decision making and decision support Focused on runtime logistics of the process; could be push or pull (i.e., auto-update mechanism)	Functions are executed in fewer places (fewer nodes to update). Nodes have high quality-of-service network connection to allow updates to be easily and dependably distributed (could be centralized nodes or edge nodes).

Analysis Method Workflow

The Analytic Hierarchy Process (AHP) is a technology to deal with the complexity of multicriteria decision making [Saaty 1987]. The AHP was chosen for this method because it is transparent, repeatable, mature, and well represented in the literature. It also supports the use of qualitative and quantitative characterizations of both the problem space and the solution space, allowing system engineers and architects to apply the analysis method early in the life cycle and refine the results as more data become available. Saaty presents a very accessible tutorial example of applying the AHP, starting near the middle of page 163 [Saaty 1987].

The analytic hierarchy consists of a goal, a set of quality criteria, and a set of solution alternatives to achieve the goal. Our goal is to identify the best allocations of C2 functions for a system or set of systems. The AHP prioritizes the quality criteria by performing a pairwise comparison of criteria to establish their relative importance. The process continues with a pairwise comparison of the solution allocations with respect to each criterion. The judgments can be performed by an individual or by a group using a consensus process or by combining individual judgments.

The final step in making a decision using the AHP is to rank the allocation alternatives to identify the best overall solution when all criteria are considered. The crucial (manual) part of the AHP is making judgments about the relative priorities of the quality criteria and the allocations; the rest is matrix algebra and can be automated.

The workflow steps are shown in Table 3, and each step is summarized briefly below.

Table 3: Analysis Workflow Steps

Workflow Step	Description
Judge each solution alternative with respect to each decision criterion.	Use expert judgment to perform pairwise comparisons based on allocation models. (Or use results of other analyses to directly rate allocations with respect to a quality criterion.)
Prioritize quality criteria.	Create a **profile** that prioritizes the architecturally significant quality criteria for a single C2 system or set of similar C2 systems. Use expert judgment to perform pairwise comparisons between quality criteria.
Compute ranking of alternatives based on prioritized criteria.	Use an AHP calculator to compute rankings.
Check stability of ranking.	Inspect sensitivity to criteria prioritization and allocation ratings.

Judge Solution Alternatives

A judgment is created by making (pairwise) allocation-to-allocation comparisons of a set of solution candidates, with respect to a single quality criterion, which will establish an ordering of the candidates with respect to that criterion. The process is repeated for each of the five quality criteria shown in Table 2. These orderings show the relative satisfaction of an individual criterion across the different allocation alternatives. Each allocation will support each quality criterion differently. These per-criterion judgments reveal quality criteria tradeoffs across solution candidates.

Note that these allocation judgments are made independent of any particular C2 system's requirements. The judgments are only about relative satisfaction of a single quality criterion by the solution candidate, so the allocation judgments can be reused to make decisions about the solution candidate's suitability for any C2 system's requirements.

Judgments can be subjective or objective and based on qualitative or quantitative data. Furthermore, these can be combined. For example, there may be measurements or simulation results from two alternatives for a quality criterion such as C2 Loop Latency. This allows an objective, quantitative comparison between those alternatives. A third alternative may use an architecture approach that is known to be slightly better than one of the first two alternatives, allowing an objective, qualitative comparison. Finally, a fourth alternative is introduced with no data and only subjective judgments. Judgments can be refined over time, replacing subjective judgments with objective judgments and qualitative with quantitative. Sensitivity analysis, described below, can indicate where refinement can improve the quality of the decision. Table 4 shows an example of how such judgments are made using the 9-point scale of the AHP.

Table 4: Judging Allocation Alternatives with Respect to a Criterion

Pairwise allocation comparisons with respect to criterion C2 Loop Latency																		
Better than									Equal	Worse than								
A1	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	**A2**
A1	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	**A3**
A1	9	8	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9	**A4**

This hypothetical example starts with the quality criterion C2 Loop Latency, and there are allocations for four solution candidates (labeled A1 to A4) to be compared. Each allocation is compared in turn with all other allocations, and a judgment is made about the relative satisfaction of the criterion between that pair. Allocation A1 is judged to be 4 times better than allocation A2 with respect to C2 Loop Latency, so that entry in the first row of the table is selected, as indicated by the shading. It's also 4 times better when A1 is compared with A3 (second row), and 6 times better when A1 is compared with A4 (third row).

Although not shown in this example, the full comparison table would include rows for the comparisons of A2 with A3 and A4, and the comparison of A3 with A4, for a total of six comparisons. The full set of pairwise comparison results is used to populate a matrix, such as the one shown in Table 5. The accompanying vector on the right shows the computed relative ratings of the four allocations with respect to C2 Loop Latency.

Table 5: Example Allocation Comparison and Rating with Respect to C2 Loop Latency

Solution Candidate	**A1**	**A2**	**A3**	**A4**		**Rating**
A1	1	4	4	6		0.57
A2	1/4	1	1	5		0.19
A3	1/4	1	1	5		0.19
A4	1/6	1/5	1/5	1		0.05

The rating is computed as the scaled principal eigenvector of the comparison matrix. Off-the-shelf AHP calculators such as SuperDecisions[3] can be used to perform this computation.

---

[3] https://www.superdecisions.com

This pairwise comparison among solution candidates is repeated for the other four quality criteria, producing similar matrices and rating vectors.

Prioritize Decision Criteria

The previous allocation-judgment step used pairwise comparisons to judge the satisfaction of each quality criterion by each allocation. This step judges the priorities of all the quality criteria in the context of one or more C2 systems to construct a profile—a prioritization of the quality criteria. Once the relative importance of the set of decision criteria has been determined for the profile, the resulting prioritization enables the allocations to be ranked as to which one will best meet the profile's objectives.

Note that these criteria prioritization judgments are made independent of any solution candidate. The judgments are only about relative priority of quality criteria for the system or systems covered by the profile. This step can be performed before or in parallel with the previous step that judges allocations.

Like the allocation judgments described in the previous section, these criteria prioritization judgments can be performed by an individual or a group. For a new system, the criteria prioritization judgments might be made during requirements development, while for existing systems, the judgments could be based on established requirements. As with the allocation judgments, these criteria prioritization judgments can be subjective or objective and based on qualitative or quantitative data. However, in this case, we expect that there may be less quantitative data available. As before, sensitivity analysis, described below, can indicate where refinement can improve the quality of the decision.

The same pairwise comparison process described above is used to make these judgments, and the same linear algebra-based transformations are used to go from pairwise relative judgments to a prioritization vector.

Rank Solution Alternatives

A ranking is a comparison of a set of solution candidates against the prioritized quality criteria for a C2 system or set of systems. The ranking for each solution candidate is calculated as the sum of the candidate's rating for each quality criterion weighted by the priority of that criterion for the system or set of systems. This calculation is performed by an off-the-shelf AHP calculator tool.

Although the calculation produces a numerical score for each alternative, the ranking is relative and not absolute. The AHP calculation is constrained so that the rankings for all solution candidates sum to 1, so adding another candidate for consideration reduces the ranking scores for the original candidates. Similarly, removing a candidate from consideration raises the ranking scores for the remaining candidates. Furthermore, an analyst cannot set a threshold and reject solution candidates that rank below that threshold. Finally, an AHP ranking does not mean that the best-ranked candidate satisfies all system requirements, so this method must be paired with other analyses to assess requirements satisfaction.

Check Stability of Rankings

The ranking of solution candidates produced by the AHP depends on the relative importance assigned to the quality criteria for the C2 system or set of systems. Changing that prioritization may produce a change in the ordering of the candidates. Sensitivity analysis will show how changes in criteria prioritization affect the ranking.

If the sensitivity analysis shows that a small change in the priority of a particular criterion changes the order of the solution ranking, that should trigger review of the rationale for the prioritization of that criterion. Can the rationale be refined, for example, by investing to produce quantitative data to support that decision? Furthermore, consider the rating of a solution alternative with respect to this criterion, and decide whether investment is warranted to improve the quality of the data to support that rating.

On the other hand, the sensitivity analysis might show that no matter how much the priority of a particular quality criterion is increased or decreased, it does not change the ranking order, so there is no need to invest to improve the rationale for that criterion.

Sensitivity analysis will typically be performed by the systems engineer, architect, or analyst who is coordinating the analysis process, using an off-the-shelf AHP calculator tool.

Tradespace Analysis Example

The brief example shows how to use the method described above to understand the impact of changing quality criteria priorities during the evolution of a set of multisector defense systems. The prioritization profile labeled P-1, shown in Figure 2, puts a high priority on the quality criterion of C2 Loop Latency, with other criteria prioritized significantly lower. As we evolve these systems, we decide that we want to increase the priority of the criteria Composability and Decision Parameter Updatability. This is reflected in the criteria prioritization for profile P-2 in Figure 2. This profile still puts C2 Loop Latency as the most important quality criterion but increases the importance of Composability and Decision Parameter Updatability.

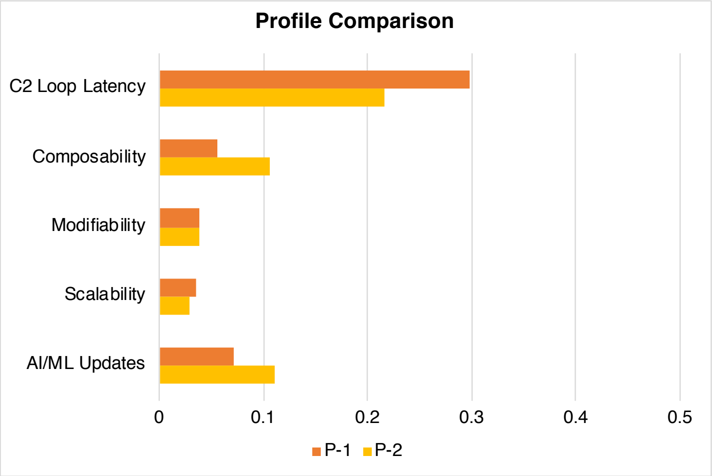

Figure 2: Criteria Prioritization for Profiles P-1 and P-2

Four solution candidates are under consideration, with allocations labeled A-1, A-2, A-3, and A-4.

Applying the method as described above produces an allocation ranking for both profiles that is shown in Table 6. We will look first at the baseline prioritization profile P-1 shown in Table 6. Allocation A-1 is clearly ranked highest.

Table 6: AHP Ranking of Candidate Solutions for Profile P-1

	Profile P-1
Allocation A-1	0.37
Allocation A-2	0.25
Allocation A-3	0.27
Allocation A-4	0.11

We should check the stability of the solution by analyzing the sensitivity of the ranking to changes in the profile's quality criteria prioritization. Figure 3 shows one sensitivity chart for this profile, for the C2 Loop Latency criterion.

Here we see that a small decrease in the priority of C2 Loop Latency changes the solution ranking, with allocation A-3 becoming the highest ranked solution, so a change to the quality criteria prioritization could cause us to prefer a solution other than allocation A-1. (Note that in some sensitivity charts, no change in the criterion priority will change the highest ranked solution.)

Although beyond the scope of this paper, we have also examined a scenario in which a ranking is extremely sensitive to changes to a profile. Options for dealing with such unstable solutions include reexamining the judgments and prioritizations, reworking the allocations (if feasible) to reduce the sensitivity, and accepting the instability at this point and dealing with it in the downstream analysis of alternatives.

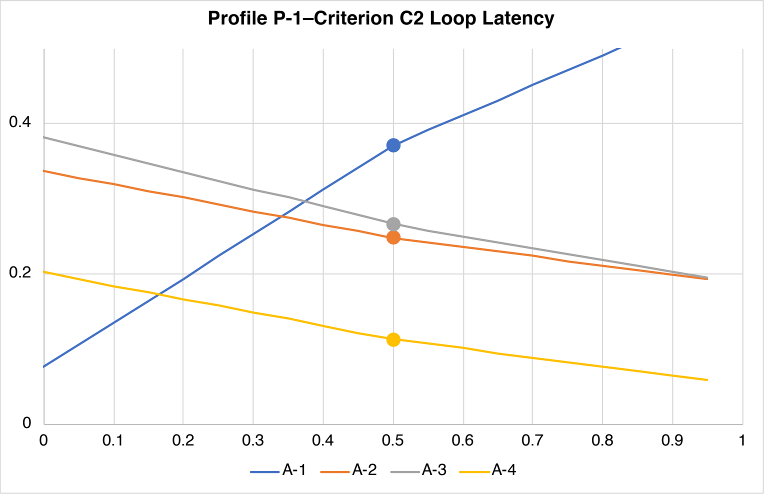

Figure 3: Sensitivity of Profile P-1 to Changes in C2 Loop Latency Priority

Conclusions

The method described here enables lightweight data-driven exploration of the tradespace early in the system life cycle.

The method has several main strengths:

- Relative comparisons allow the use of any available data: subjective, objective, qualitative, and quantitative. Types of data can be combined or mixed, and data can be replaced with more accurate or precise data as it becomes available.
- Sensitivity analysis indicates where better quality data is informative, and where it may not be necessary to invest to improve data quality.
- Sensitivity analysis allows stakeholders to focus discussions and decisions on the parameters that influence the solution selection decision.

The main limitation of the method is that the solution ranking produced by AHP only identifies one solution as best *among the candidates*; it cannot determine whether the highest-ranked solution candidate (or any solution candidate) meets all functional and quality requirements for the system or set of systems under analysis. This method must be applied as part of a broader suite of analyses that screen or filter candidate solutions and further examine the highest-ranked solution.

The method may be best applied as shown in Figure 4, as a way to quickly consider and filter a broad range of alternatives, prior to a deeper analysis of alternatives (AoA) for a smaller number of candidates. The sensitivity analysis from this method can inform the AoA, indicating the quality criteria where precise analysis is needed and those that have less influence on the selection. Finally, there may be iteration back from AoA, as more precise data is produced and the AHP is repeated with that data.

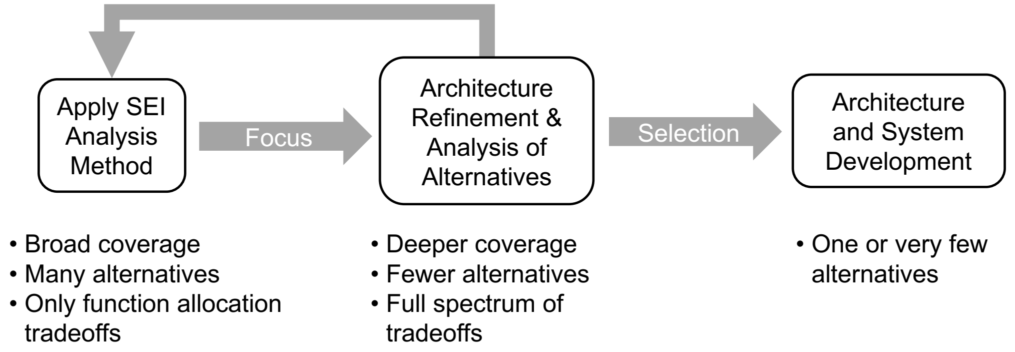

Figure 4: Navigating the Architecture Tradespace

Future work on this method should include richer definitions of the quality criteria and heuristics for rating solution candidates with respect to each criterion, which will make application of the method more systematic. Further validation of the usability and utility of the method is also needed, including pilot application with one or more acquisition programs.

References

URLs are valid as of the publication date of this document.

[Alberts 2006]
Alberts, David S. & Hayes, Richard E. *Understanding Command and Control.* CCRP Publications. 2006.

[Bass 2013]
Bass, Len; Clements, Paul; & Kazman, Rick. *Software Architecture in Practice, 3rd Edition.* Addison-Wesley. 2013.

[Boyd 2018]
Boyd, John R. *A Discourse on Winning and Losing.* Maxwell AFB, AL: Air University Press. 2018.

[Dekker 2005]
Dekker, Anthony. “A Taxonomy of Network Centric Warfare Architectures.” *Proc. 2005 Systems Eng. Test and Evaluation Conf.* (SETE). Brisbane, Queensland, Australia. November 2005.

[Eisenberg 2018]
Eisenberg, D. A.; Alderson, D. L.; Kitsak, M.; Ganin, A.; & Linkov, I. “Network Foundation for Command and Control (C2) Systems: Literature Review.” *IEEE Access 6* (14 November 2018): 68782–68794. doi: 10.1109/ACCESS.2018.2873328

[Lawson 1981]
Lawson, Joel S. “Command Control as a Process.” *IEEE Control Systems Magazine 1,* 1 (March 1981): 5–11. doi: 10.1109/MCS.1981.1100748

[MIP 2012]
Multilateral Interoperability Programme. “The Joint C3 Information Exchange Data Model.” NATO, Standard JC3IEDM Main. 2012. https://public.mip-interop.org/Pages/Default.aspx

[NIEM: 2019]
NIEM. *National Information Exchange Model.* 2019. http://niem.github.io/niem-releases/

[Saaty 1987]
Saaty, R.W. “The Analytic Hierarchy Process—What It Is and How It Is Used.” *Mathematical Modelling, 9,* 3–5 (1987): 161–176. doi:10.1016/0270-0255(87)90473-8

[Tirpak 2000]
Tirpak, John A. “Find, Fix, Track, Target, Engage, Assess.” *Air Force Magazine* (July 2000): 24–29. http://www.airforcemag.com/MagazineArchive/Documents/2000/July%202000/0700find.pdf

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu

Copyright 2020 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

Internal use:* Permission to reproduce this material and to prepare derivative works from this material for internal use is granted, provided the copyright and “No Warranty” statements are included with all reproductions and derivative works.

External use:* This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other external and/or commercial use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

* These restrictions do not apply to U.S. government entities.

DM20-0557