

# Understanding Cybersecurity Risk Freshness in an AVCDL Context

## Revision

Version 5  
4/14/25 2:24 PM

## Author

Charles Wilson

## Abstract

This document describes how the freshness of cybersecurity risks is achieved within the **AVCDL**.

## Audience

The audience of this document are the cybersecurity development lifecycle practice leads who will be guiding **AVCDL** adoption within their organization.

**Note:** This document is not subject to certification body review.

## License

This work was created by **Motional** and is licensed under the **Creative Commons Attribution-Share Alike (CC BY-SA-4.0)** License.

<https://creativecommons.org/licenses/by/4.0/legalcode>

# Overview

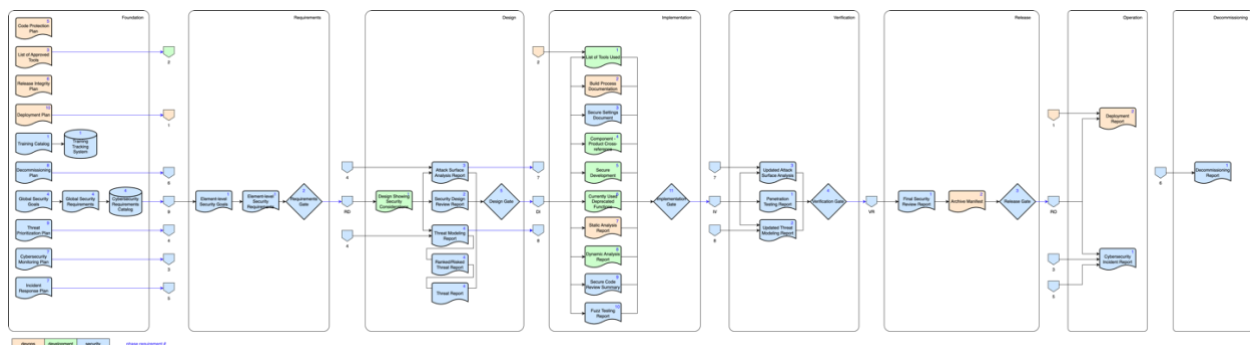
Although the **AVCDL** <sup>[1]</sup> does not specifically discuss the topic of cybersecurity risk freshness it was designed with the intent that the processes and recommendations for the embodiment of machine readable information be used in such a way as to ensure that information is available and current to all processes requiring it. This document will show how the processes of the AVCDL enable effective and efficient mechanisms for ensuring cybersecurity risk freshness.

## Freshness Motivation

Cybersecurity risk freshness is motivated by the need to ensure that the overall system cybersecurity risk is both properly characterized and also minimized. Cybersecurity risk freshness itself is a measure of the delta between the last assessed state of the product (or element thereof) and the current state with respect to cybersecurity risk. The fundamental basis for assessing the cybersecurity risk is the application of the threat prioritization process document in the **Threat Prioritization Plan** <sup>[2]</sup>. Ideally there would be near zero difference between the last assessed and current states.

## Feedforward

The **AVCDL** provides a visualization of the feedforward nature of the phase product dependencies (shown below).



**Note:** This information is shown by phase in the **AVCDL** primary document section 18 – **AVCDL Product Dependencies**.

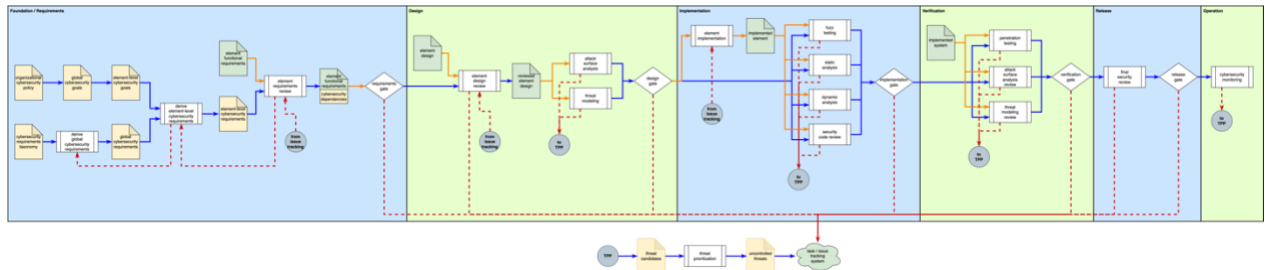
**Note:** Symbology for this diagram is described in the AVCDL elaboration document, **Understanding Workflow Graphs** <sup>[3]</sup>.

This diagram is useful in establishing traceability of documentation but is not sufficient to demonstrate how freshness is maintained, even for the forward case.

# Global View

The following shows the global feedforward / feedback enabled by the **AVCDL**.

**Note:** Diagram [symbolology](#) is described in the following section. This symbolology differs from that used in the preceding diagram.



**Note:** Only items relevant to feedforward / feedback are shown.


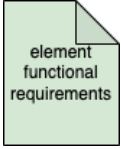


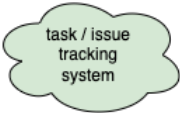




The remainder of the document will examine the enabled feedback as the product progresses through the **AVCDL** processes.

**Note:** For the purpose of this document, a new product is presumed. In the case of product revisions, it is presumed that learnings from earlier revisions of the product are integrated into the base information.

**Note:** In order to focus on feedback aspects, the details of the **AVCDL** processes have been greatly simplified. References to the relevant **AVCDL** secondary documents are included in the discussion.

# Symbology

The following table covers the symbology used in this document's diagrams.

Symbol	Description
	Cybersecurity artifact
	Non-cybersecurity artifact
	Phase activity
	Phase gate
	System (here a non-cybersecurity system is shown)
	Transfer to or from another area of the diagram (used to reduce diagram clutter)
	Cybersecurity process flow
	Non-cybersecurity process flow
	Feedback flow

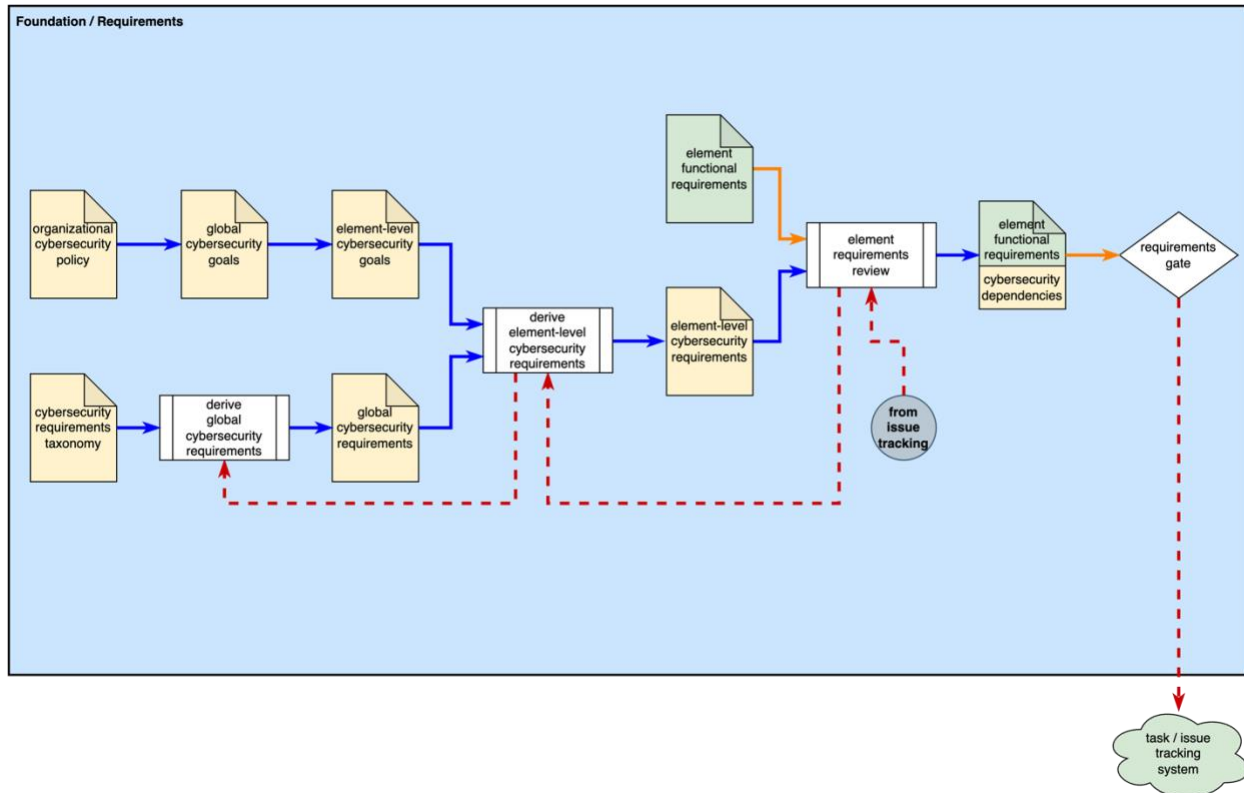
**Note:** There is no significance to the background color of each phase block. They differ in order to improve visual uptake.

# Feedback by Phase

## Foundation / Requirements

**Note:** Because the foundation phase is considered to be under constant review, it is combined with the requirements phase for the sake of convenience.

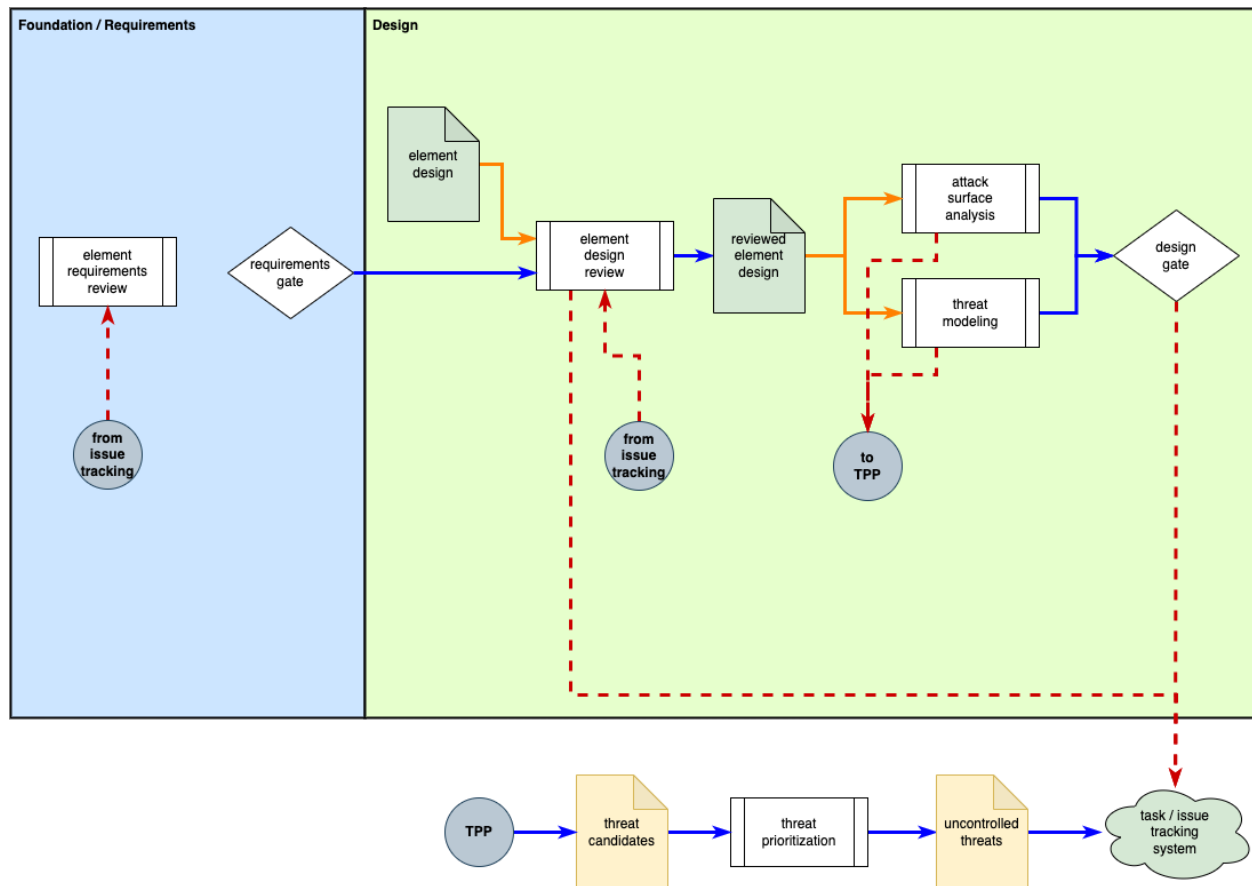
The foundation and requirements phases have the following feedback behavior.



Three main activities dominate the feedback within the foundation and requirements phases. The **global cybersecurity requirements** derivation activity receives feedback during the creation of the **element-level cybersecurity requirements**. This feedback covers gaps in the global requirements themselves. The **element-level cybersecurity requirements** derivation activity receives feedback during the **element requirements review** activity. Again, this is requirements coverage gap feedback. Finally, additional **element requirements review** may be necessitated by incomplete or inadequate work being evidenced during the **requirements gate**. In this case, we see that an issue is created in the **task / issue tracking system**.

# Design

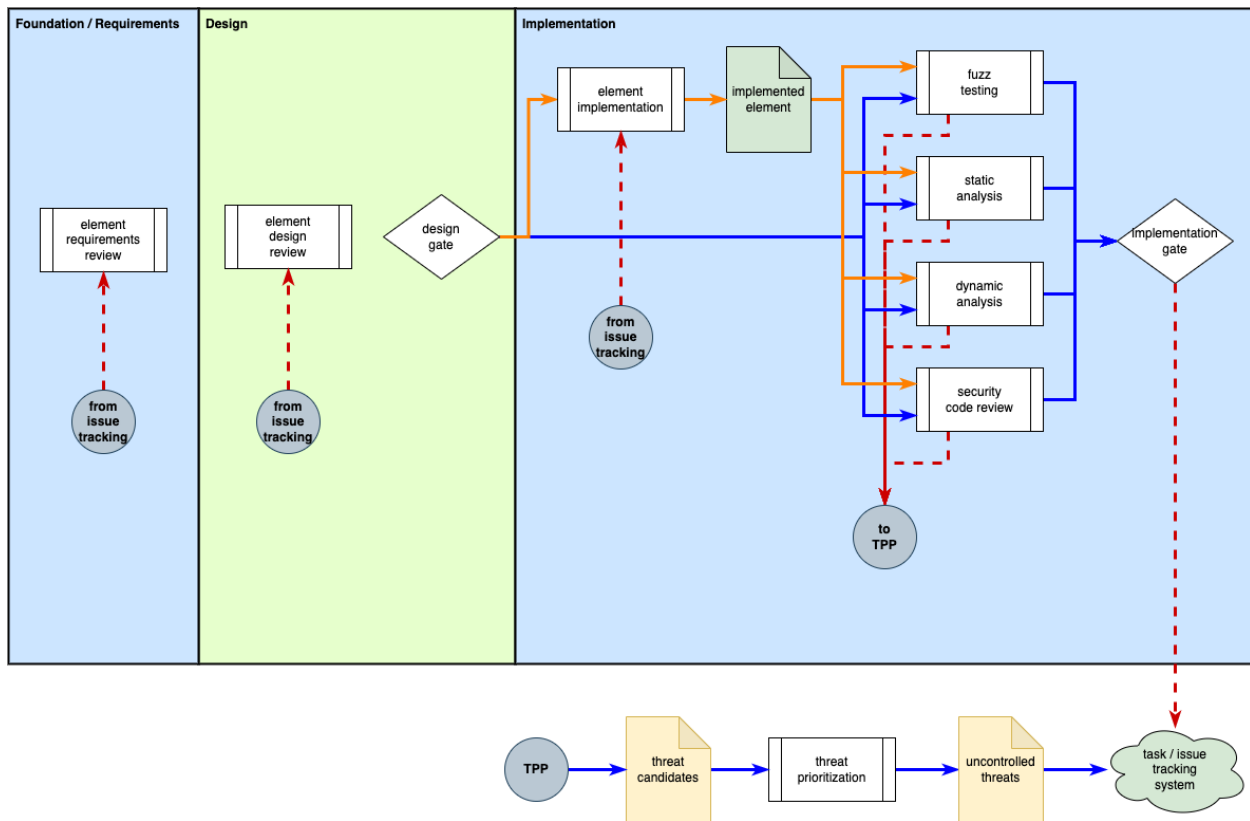
The design phase has the following feedback behavior.



The design phase has a single point of feedback. The **element design review** may be revisited as the result of deficiencies being identified during the **attack surface analysis** or **threat modeling** activities. Both of these activities feedback through application of the **threat prioritization process** [2]. This process eventually feeds into the **task / issue tracking system**. In addition, the **design gate** may result in issues requiring an updated **element design review**. Finally, issues may require a new **element requirements review** as seen by the inclusion of the **foundation / requirements phases** (abbreviated form).

# Implementation

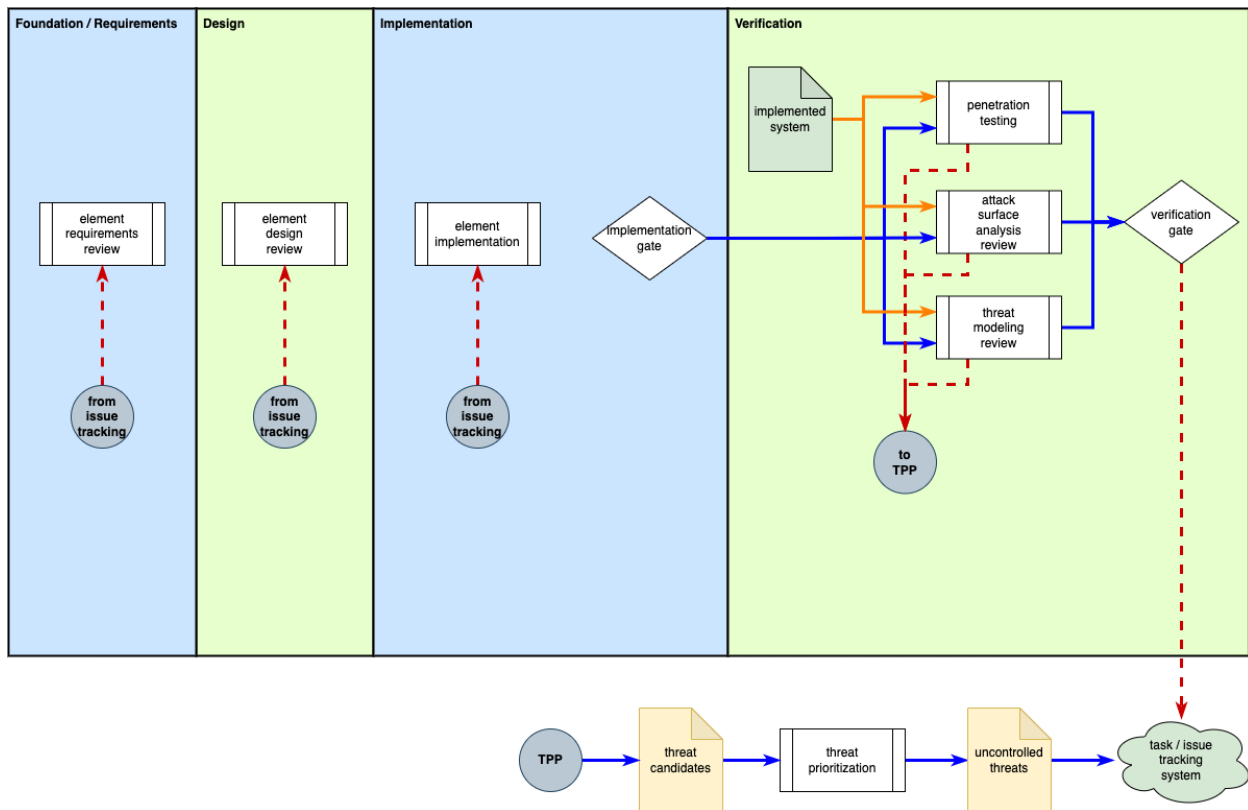
The implementation phase has the following feedback behavior.



The implementation phase has a single point of feedback. The **element implementation** may be revisited as the result of deficiencies being identified during the **fuzz testing**, **static analysis**, **dynamic analysis**, or **security code review** activities. These activities feedback through application of the **threat prioritization process** [2]. In addition, the **implementation gate** may result in issues requiring an updated **element implementation**. Finally, issues may require revisiting activities in previous phases.

# Verification

The verification phase has the following feedback behavior.

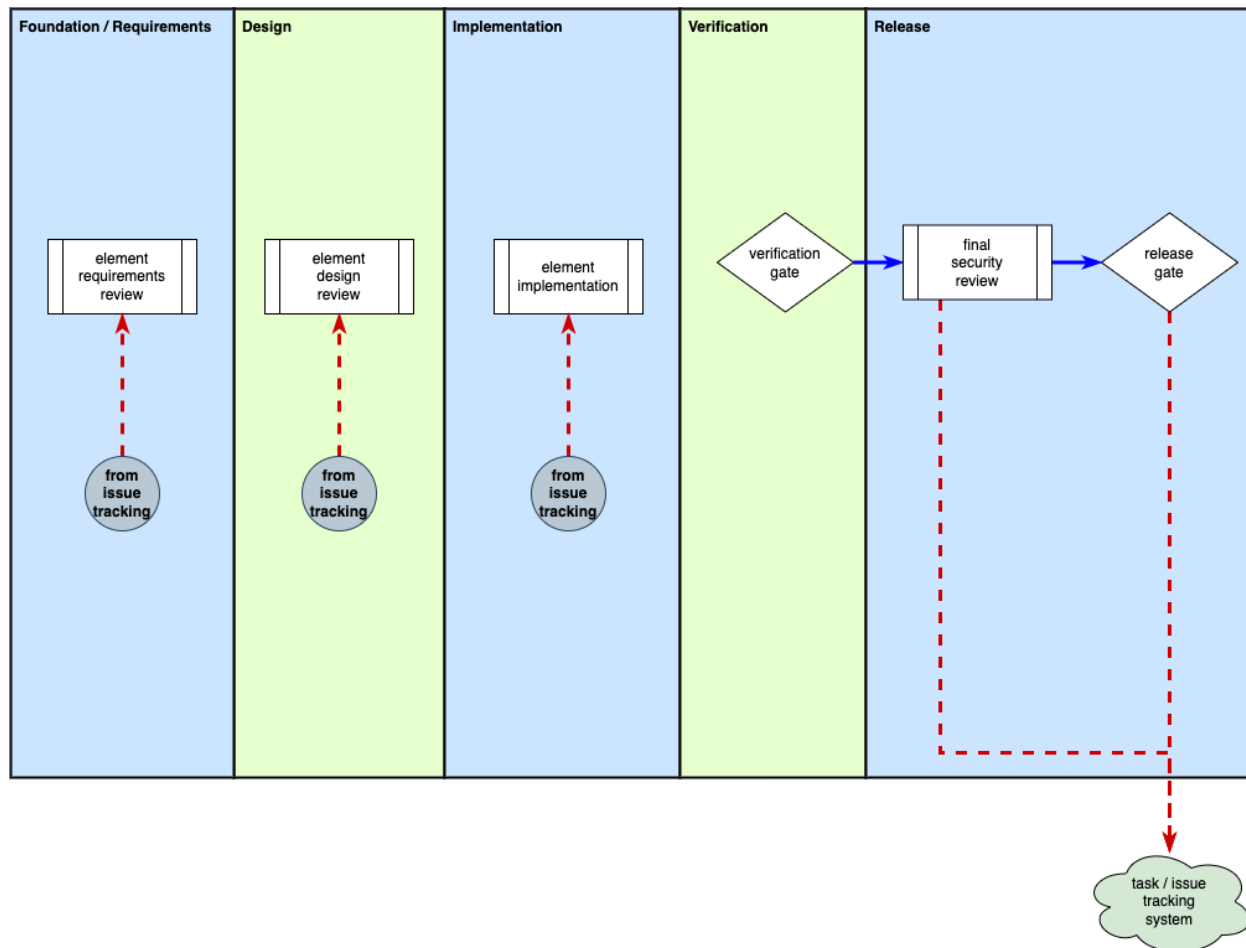


The verification phase no feedback points. Activities in previous phase (shown above) may be revisited as the result of deficiencies being identified during the **penetration testing**, **attack surface analysis review**, or **threat modeling review** activities. These activities feedback through application of the **threat prioritization process** [2]. In addition, the **verification gate** may result in issues also requiring revisiting activities in previous phases.



# Release

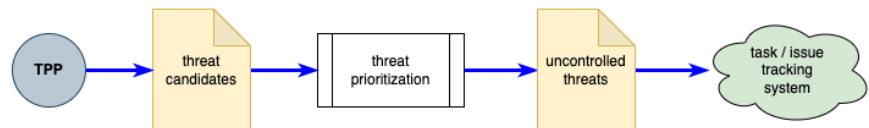
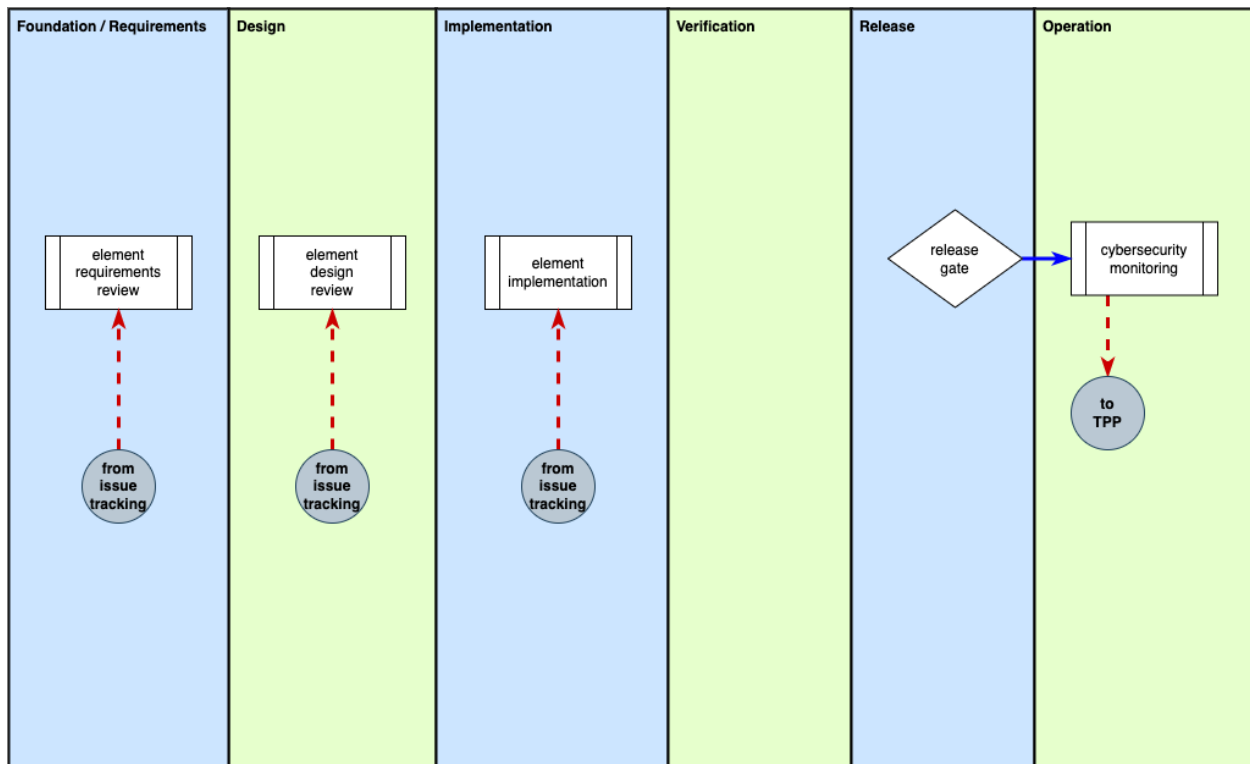
The release phase has the following feedback behavior.



The release phase has no feedback points. Activities in previous phase (shown above) may be revisited as the result of deficiencies being identified during the **final security review** activity. In addition, the **release gate** may result in issues also requiring revisiting activities in previous phases.

# Operation

The operation phase has the following feedback behavior.



The operation phase has no feedback points. Activities in previous phase (shown above) may be revisited as the result of deficiencies being identified during the **cybersecurity monitoring** activity. Feedback is accomplished through application of the **threat prioritization process** [\[2\]](#).

# Additional Freshness Mechanisms

Although the process feedback is important to how the **AVCDL** achieves cybersecurity risk freshness, it is not the only mechanism at play. Throughout the **AVCDL** several technologies and mechanisms are called out.

## Databases

In order to ensure the feedforward is fresh, the **AVCDL** specifies the use of databases. This eliminates the need for traditional text data scanning and aggregation prior to use. It also enables a much richer analytic capability within the individual **AVCDL** activities. The **AVCDL** elaboration document **Software Bill of Materials Lifecycle** <sup>[8]</sup> illustrates how these database allow for the clean integration of multiple **AVCDL** processes.

## Requirements Management System

The use of a requirements management system is critical to enable the traceability aspects of the **AVCDL**. Not only is it used to hold the global cybersecurity requirements, but it also allows the derivation of product and element specific tailoring of this “master catalog.” Additionally, since cybersecurity requirements are non-functional, a requirements management system is necessary to handle the binding of these non-functional requirements to the product and element level functional requirements.

## Task / Issue Tracking System

In conjunction with the requirements management system, the task / issue tracking system provides the product teams with a centralized means of implementing the activities called out in the **AVCDL**. This system ties the requirements to the implementation and the tests needed for verification, as well as to the issues that arise from those tests. It also provides a mechanism for validating that all intended controls have been emplaced.

## Standardized Data Interchange Formats

Finally, the **AVCDL** prefers use of standardized data interchange formats such as **SARIF** <sup>[4]</sup>, **SCAP** <sup>[5]</sup>, **CSAF** <sup>[6]</sup>, and **SPDX** <sup>[7]</sup>. The use of standardized formats greatly reduces the likelihood that information is lost during communication between internal and external organizations such as suppliers and regulatory organizations.

# References

1. **AVCDL** (AVCDL primary document)
2. **Threat Prioritization Plan** (AVCDL secondary document)
3. **Understanding Workflow Graphs** (AVCDL elaboration document)
4. **Static Analysis Results Interchange Format (SARIF) Version 2.1.0**  
<https://docs.oasis-open.org/sarif/sarif/v2.1.0/os/sarif-v2.1.0-os.pdf>
5. **NIST SP800-126r3 The Technical Specification for the Security Content Automation Protocol (SCAP) v1.3**  
<https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-126r3.pdf>
6. **OASIS Common Security Advisory Framework (CSAF)**  
<https://oasis-open.github.io/csaf-documentation/>
7. **Software Package Data Exchange® (SPDX®)**  
<https://spdx.dev/wp-content/uploads/sites/41/2017/12/spdxversion2.1.pdf>
8. **Software Bill of Materials Lifecycle** (AVCDL elaboration document)