SEI Insights

SEI Blog

The Latest Research in Software Engineering and Cybersecurity

Obstacles in Engineering Safety- and Security-Related Requirements, Second in a Three-Part Series

POSTED ON JULY 18, 2011 BY **DONALD FIRESMITH** [/AUTHOR/DONALD-FIRESMITH] IN **ACQUISITION** [HTTPS://INSIGHTS.SEI.CMU.EDU/SEI_BLOG/ACQUISITION/]

Background: In our research and acquisition work on commercial and Department of Defense (DoD) programs, ranging from relatively simple two-tier data-processing applications to large-scale multi-tier weapons systems, one of the primary problems that we see repeatedly is that **acquisition** [http://www.sei.cmu.edu/acquisition/] and development organizations encounter the following three obstacles concerning safety- and security-related requirements:

1. Safety, security, and requirements engineers typically know little about each others' disciplines.
2. Safety, security and requirements engineers often reside in separate teams that rarely collaborate when engineering safety- and security-related requirements.
3. Safety and security are viewed as specialty engineering disciplines that are not well integrated into the overall systems engineering process until after the architecture is largely finalized.

This is the second in a series exploring the engineering of safety- and security-related requirements. The **first post** [http://insights.sei.cmu.edu/post.cfm/the-importance-of-safety-and-security-related-requirements] in the series explored problems with quality requirements. This post takes a deeper dive into key obstacles that

acquisition and development organizations encounter concerning safety- and security-related requirements. In the third part of this series, we will introduce a collaborative method for engineering these requirements that overcomes the obstacles identified in this blog.

The first obstacle is a lack of understanding of each other's disciplines. The safety, security, and requirements communities each have their own terminology, methods, techniques, models, and documents. They read their own journals and books, and they attend their own conferences. In short, they form separate stovepipes that rarely interact.

Safety engineers know how to perform safety (hazard) analysis, security engineers know how to perform security (threat) analysis, and requirements engineers know how to perform requirements analysis. Unfortunately, they are rarely trained in each other's disciplines. These three communities have independently developed effective techniques and methods for performing their own analyses, but they are largely unaware of each others' disciplines. In practice, however, safety techniques and methods are often quite appropriate--with little or no modifications--for performing security analyses and vice versa. This lack of awareness limits the options available to members of these communities and frequently leads to duplication (often inconsistent or incomplete duplication) of each others' work.

Requirements engineers have an additional problem. Although they know how to engineer functional requirements, we have seen many projects where they utilize functional decomposition or use-case modeling to the exclusion of all other requirements analysis techniques. While these approaches may work well for functional requirements, they are not effective for engineering quality requirements, such as safety and security, as discussed in the **first blog posting** [http://insights.sei.cmu.edu/post.cfm/the-importance-of-safety-and-security-related-requirements].

The second obstacle is a lack of close collaboration among these three types of engineers, which is especially detrimental to safety and security; two sides of the same coin. Safety and security engineering are concerned with preventing negative events and conditions. The primary difference between safety and security is that safety deals with unintentional (accidental) negatives, whereas security deals with intentional (malicious) ones.
Although safety and security engineers have different, albeit complementary responsibilities, they are not completely independent. Accidents can cause vulnerabilities that can be exploited during an attack. Likewise, attacks can cause vulnerabilities that lead to accidents. For example, an electrical power outage (accident) could cause the failure of a physical access control system (vulnerability), leaving security doors unlocked so that unauthorized people can enter into secured areas (attack). Similarly, safety critical software could be infected by malware (attack), causing it to fail

(vulnerability), which is a hazard that can lead to associated accidents.

As mentioned above, however, safety and security engineers rarely interact, so each tends not to appreciate what the other does. As safety and security engineers have recognized the need to prevent accidents and attacks, security engineers are starting to claim parts of safety, while safety engineers are beginning to claim parts of security. The lack of collaboration between the two teams can lead to a duplication of tasks. Stepping on the other's turf often leads to resentment, rather than recognition of the need for, and value of, collaboration.

This brings us to our third and final obstacle: the view that safety and security are specialty engineering areas that need not be incorporated into systems engineering until after the system requirements that drive the architecture are defined and the primary architecture decisions have been made. This omission means that safety and security engineers are rarely empowered to make architectural decisions that ensure that the actual safety or security requirements are met. It is therefore more than just a problem of not working closely together; it's that safety and security engineers rarely become involved in the system engineering until after key requirements and architecture decisions have been completed, which is too late.

This viewpoint causes safety and security engineers to concentrate on fixing architectural weaknesses, rather than specifying the requirements that would prevent these weaknesses from being incorporated into the architecture. Safety and security engineers also tend to apply industry standard controls, such as shielding and interlocks (safety) and encryption and passwords (security) rather than asking themselves how safe and secure the system needs to be. In other words, what are the safety and security-related requirements that should be driving the architecture? It is critical to build safety and security into the system from the beginning because it is very difficult and expensive to add it to an existing architecture afterwards.

Given these obstacles, how can we overcome them to properly engineer safety and security requirements? The answer is by providing safety, security, and requirements engineers with an effective method for collaborating closely when engineering these requirements. In the third part of this series, we will introduce a collaborative method for engineering these requirements that overcomes the obstacles identified in this blog.

Additional Resources:

For more information, please visit
www.sei.cmu.edu/library/abstracts/presentations/icse-2010-tutorial-firesmith.cfm

[http://www.sei.cmu.edu/library/abstracts/presentations/icse-2010-tutorial-firesmith.cfm]

About the Author

Donald Firesmith

Contact Donald Firesmith [https://www.sei.cmu.edu/contact.cfm]
Visit the SEI Digital Library for **other publications by Donald**
[http://resources.sei.cmu.edu/library/author.cfm?authorID=4637]
View **other blog posts by Donald Firesmith**
[/author/donald-firesmith]