SANDIA REPORT

SAND2015-9022
Unlimited Release
Printed September 2015

Model of the Product Development Lifecycle

Sunny L. He
Natalie H. Roe
Evan C. L. Wood
Noel Nachtigal, Principal Investigator
Jovana Helms, Team Lead

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd. Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2015-9022
Unlimited Release
Printed September 2015

Model of the Product Development Lifecycle

Sunny L. He, Systems Research and Analysis III
Natalie H. Roe, Data Science & Cyber Analytics
Evan C. L. Wood, Systems Research and Analysis III
Noel Nachtigal, Cyber Systems Assessments
Jovana Helms, Systems Research and Analysis III

Sandia National Laboratories
P.O. Box 969
Livermore, California 94550

Abstract

While the increased use of Commercial Off-The-Shelf information technology equipment has presented opportunities for improved cost effectiveness and flexibility, the corresponding loss of control over the product's development creates unique vulnerabilities and security concerns. Of particular interest is the possibility of a supply chain attack. A comprehensive model for the lifecycle of hardware and software products is proposed based on a survey of existing literature from academic, government, and industry sources. Seven major lifecycle stages are identified and defined: (1) Requirements, (2) Design, (3) Manufacturing for hardware and Development for software, (4) Testing, (5) Distribution, (6) Use and Maintenance, and (7) Disposal. The model is then applied to examine the risk of attacks at various stages of the lifecycle.

ACKNOWLEDGMENTS

The authors owe thanks to numerous individuals at Sandia National Laboratories, including the staff members of the Systems Research and Analysis III group, 8116. Particular thanks are owed to Jessica Westbrook, Brad Steinfeldt, and Margaret "Magey" Todd for their suggestions and comments, which were quite helpful in clarifying our arguments. We would also like to acknowledge our fellow interns, for their positive attitude and support during this project.

This work has been funded by the Department of Homeland Security, Office of Cybersecurity and Communications, Stakeholder Engagement and Cyber Infrastructure Resilience division.

TABLE OF CONTENTS

FIGURES

TABLES

NOMENCLATURE

CAD	Computer Aided Design
COTS	Commercial Off-The-Shelf
DHS	Department of Homeland Security
DOD	Department of Defense
DOE	Department of Energy
FFIEC	Federal Financial Institutions Examination Council
GSA	General Services Administration
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Standards Organization
IT	Information Technology
NIST	National Institute of Standards and Technology
SDK	Software Development Kit
SNL	Sandia National Laboratories

1. INTRODUCTION

The supply chain security of Commercial Off-The-Shelf (COTS) products has been a well characterized goal towards greater security in information technology (IT) acquisitions. While the utilization of commercial hardware and software presents opportunities for significant improvements in efficiency and cost-effectiveness, COTS products also bring greater risk and less oversight than products created in-house. Of particular interest is the risk of a supply chain attack, where a malicious agent intervenes at a point along the product lifecycle, compromising equipment security before and throughout its operational deployment. The issue of supply chain attacks has garnered attention from a great number of organizations, producing studies of potential attack vectors and standards for corresponding countermeasures. However, a detailed characterization of the typical supply chains for hardware and software products is still noticeably absent from the literature.

In this report, a model for the product development lifecycles of software and hardware COTS products is proposed based on an analysis of existing literature. Concepts drawn from a range of sources, including both academic literature and government regulations and standards, were used to paint a comprehensive picture of commonly used development lifecycles and methodologies. In total, seven important stages were identified and defined: (1) Requirements, (2) Design, (3) Manufacturing for hardware and Development for software, (4) Testing, (5) Distribution, (6) Use and Maintenance, and (7) Disposal.

Information gathered from case studies of products and the literature survey then informed a discussion of the risks inherent in each stage of our proposed model, considering the unique vulnerabilities facing hardware and software products. While the highest-risk portion of the lifecycle for hardware products is the Manufacturing stage, software products encounter the most risk in the Distribution and Development stages, especially when products utilize Web-based content distribution systems and draw heavily third-party source code and APIs.

2. LITERATURE SURVEY

In order to identify existing best practices, an extensive literature survey was conducted, which covered a variety of government, industry, and academic approaches to the product lifecycle. A semi-quantitative method was applied to identify the terminology commonly used in these examples. The most frequent terms were then employed as the framework for defining a unified lifecycle model for both software and hardware.

2.1 Methodology

A wide range of product development lifecycle models have been proposed by various groups and organizations. This research began by locating sources, searching for terms such as "lifecycle" or "acquisition process" in common research databases. From the initial inquiry, both academic and industry/government documentation were located. Then, in order to identify patterns, a reference table was constructed, placing the lifecycle stages identified by each paper or article in its own row.

After a sizeable amount of material had been collected, a custom, word frequency counting program identified which terms had been used the most often. Irrelevant words such as "and" and "of" were disregarded. To more easily examine patterns in the ordering of lifecycle stages, these relevant common terms were then highlighted in color in the original table.

2.2 Data and Analysis

In total, 120 different product lifecycle models were extracted from the 105 sources examined. The majority of references came from academic papers. Industry best practices such as Microsoft's Security Development Lifecycle and ISO standards as well as government processes such as DOD and DHS acquisition procedures made up the remainder of sources reviewed. A full listing of the material used in our literature survey is presented in Appendix A: Literature Survey Data.

The fifteen most frequent relevant terms in the lifecycle models identified by our sources are listed below in descending order.

Table 1. Partial list of most frequent relevant terms in descending order

Rank	Term	Occurrences	Percentage of Sources Using this Term
1	design	86	72.3%
2	manufacturing	44	37.0%
3	production	43	36.1%
4	development	36	30.3%
5	maintenance	33	27.7%
6	disposal	29	24.4%
7	use	25	21.0%
8	requirements	24	20.2%
9	planning	23	19.3%

10	sales	23	19.3%
11	distribution	21	17.6%
12	concept	20	16.8%
13	recycling	19	16.0%
14	support	19	16.0%
15	testing	19	16.0%

The process of selecting the elements for our product lifecycle was largely empirically driven, by considering the frequency of a term's appearance in the literature survey data. After trivial words like "and" and "product" were removed from the table, the remaining terms were compared to each other in order to determine whether any were synonymous.

Further analysis showed that some terms were being used in similar contexts and with similar meanings. For instance, "manufacturing" and "production" both describe the transformation of raw materials into a physical product. While both terms are used in a large percentage of sources, only one is necessary to convey the fabrication and assembly process. In addition, synonyms appear to be mutually exclusive—no source used both terms as separate steps in a product's lifecycle (i.e. some sources use the term "manufacturing", while others use "production", but never both). By looking for these patterns, a number of equivalent pairings were discovered. If two terms were synonyms, identifying them as separate stages of the lifecycle would be misleading, so the term that appeared more frequently was chosen to represent both words.

It became apparent that the software and hardware lifecycles differ in many respects. Some terms have distinct definitions when used in the context of hardware development as compared to software development. For instance, the term "development" is more closely associated with "design" under a hardware context, but in the context of software, "development" takes on the role of "manufacturing." As a result, the term "manufacturing" is replaced with "development" in the software lifecycle. There are a number of other minor distinctions between the hardware and software lifecycles, largely related to the intangibility of software products. Ultimately, separate software and hardware lifecycle models were created to better capture these differences.

The lifecycle steps were selected to represent distinct periods in time. Notice that the terms "use" and "maintenance" are merged into one phase. This is a conscious choice to reflect the fact that both actions occur within the same time frame, since maintenance is performed at various points throughout the use of a product.

3. PRODUCT LIFECYCLE MODEL

Drawing from literature survey, the seven stages of the product lifecycle model emerged, with slightly different definitions for software and hardware products. Recall, these stages are: (1) Requirements, (2) Design, (3) Manufacturing for hardware and Development for software, (4) Testing, (5) Distribution, (6) Use and Maintenance, and (7) Disposal.

Most of our definitions for lifecycle stages draw upon the language of ISO/IEC standards 15288 and 12207, which define life cycle processes for hardware (which ISO refers to as "systems") and software, respectively. These standards describe typical processes and actions that may be taken over the course of a product's lifecycle, but do not prescribe a definite ordering or methodology to be followed (International Organization for Standardization 2008). ISO standards present well-vetted definitions that are representative of a wide range of industries. For these reasons, they were used as a reference point for developing our descriptions.

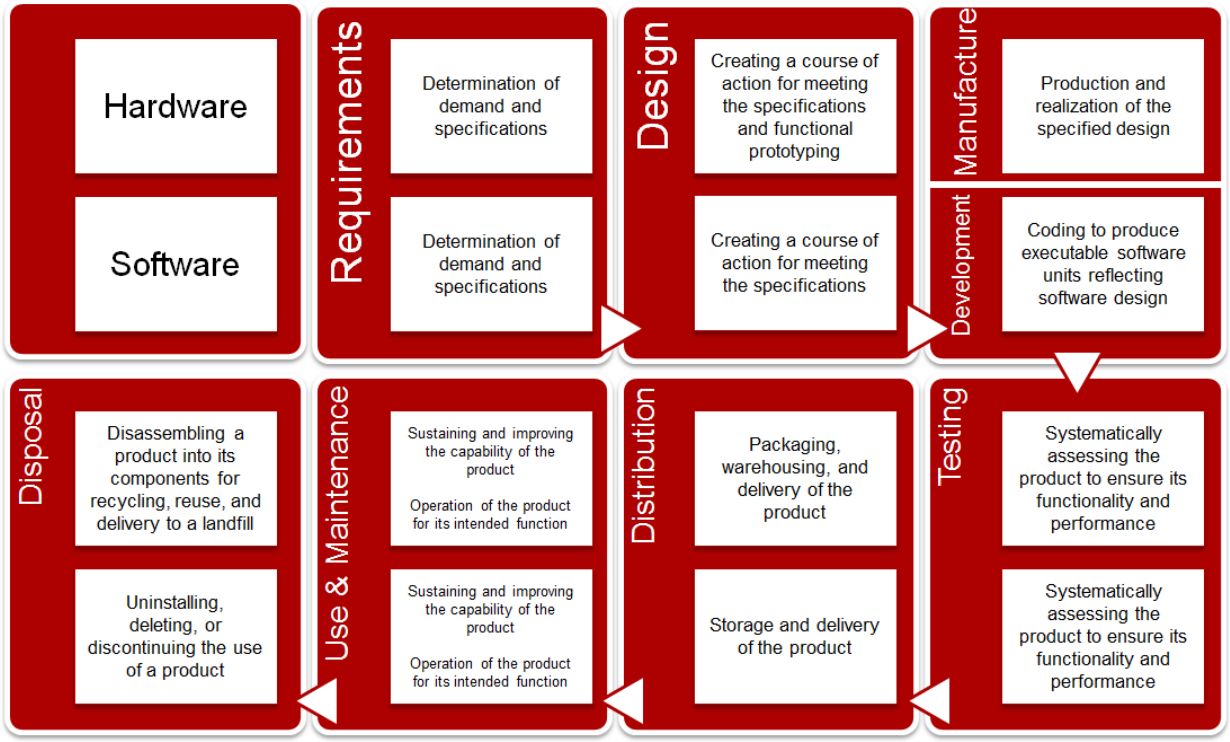

Figure 1. Product Lifecycle Model

3.1 Requirements

3.1.1 Definition

The Requirements stage is defined as the determination of demand and specifications for the product under development. This definition is common to both hardware and software lifecycles.

3.1.2 Explanation

The Requirements stage is the initiation of the product lifecycle, where the need for a new product is first recognized and general specifications are defined. Specifications may include technical requirements, guidelines on manufacturing or testing procedures, and a variety of other conditions and constraints.

Generally, the Requirements phase is dictated by market demand for a product. For government acquisitions, product specifications are derived from the needs of the acquiring department or agency, culminating in a formal Solicitation for Offers, Request for Proposal, or another requirements document. As an example, the corresponding stage in the Defense Acquisition System is the Material Solution Analysis Phase, where acquisition officers "conduct the analysis and other activities needed to choose the concept for the product that will be acquired, to begin translating validated capability gaps into system-specific requirements" (Department of Defense 2015).

For commercial products, the Requirements stage usually involves internal discussions and market research to determine the existence and scope of demand. Rather than looking to satisfy their internal needs from the development of a new product, commercial companies look to satisfy the needs and desires of their customer base. Most have research groups dedicated to proposing new product ideas and identifying potential improvements in existing technology. The Requirements stage also includes the translation of concepts into more concrete specifications for goods. While the level of rigor in drafting requirements varies from company to company, some form of documentation listing specifications and goals is typically an output of this phase of the lifecycle.

3.1.3 References

- ISO/IEC 15288: "6.4.1 Stakeholder Requirements Definition Process… The purpose of the Stakeholder Requirements Definition Process is to define the requirements for a system that can provide the services needed by users and other stakeholders in a defined environment" (International Organization for Standardization 2008).
- United States Coast Guard Major Systems Acquisition Manual: "The Project Identification Phase… is a pre-acquisition phase conducted by the Coast Guard that provides a foundation for the identification of capability gaps. The Project Identification Phase may also begin as the result of a Congressional mandate, need for technology refreshment, or new technology development that provides a new capability or significant improvement in mission performance" (Acquisition Directorate of the U.S. Coast Guard 2015).

3.2 Design

3.2.1 Definition

For both hardware and software lifecycles, design entails creating a concrete course of action for meeting the specifications that result from the Requirements stage. In the case of hardware, design also involves functional prototyping.

3.2.2 Explanation

Starting from the abstract set of requirements created in the previous lifecycle step, designers and engineers flesh out the details of the product's components, functionality, and structure. The Requirements stage specifies *what* the product must do, whereas the Design stage determines *how* the product will to fulfill the specifications of what the product must do. The Design phase might not be performed by the same firm that establishes the requirements or even the firm that will eventually sell the product. Companies specializing in design may be contracted to perform design work (e.g. Sandia). In addition, designs from third party sources may be incorporated into the product. For instance, software may draw on industry standard protocols and algorithms, and hardware might use prebuilt systems or standardized connectors and form factors. To obtain these external designs, patent licenses and other intellectual property agreements may be needed.

3.2.3 References

- ISO 12207 6.4.3.3 "A top-level architecture of the system shall be established. The architecture shall identify items of hardware, software, and manual operations. It shall be ensured that all the system requirements are allocated among the items. Hardware configuration items, software configuration items, and manual operations shall be subsequently identified from these items" (International Organization for Standardization 2008).
- Zhang, et al. "3.2 Conceptual Design… To transform CRs [customer requirements] specifications to a systematic product developing specification." (Zhang and Fan 2006)

3.3 Manufacture/Development

This stage of the product lifecycle differs greatly both in terminology and in definition for hardware and software products. For this reason, completely separate stages for the hardware and software lifecycles were defined.

3.3.1 Manufacture (Hardware Products)

3.3.1.1 Definition

Manufacturing involves the production and realization of the specified design and prototypes.

3.3.1.2 Explanation

In this stage, final product units intended for delivery to the customer or acquiring agency are fabricated and assembled. For hardware products, manufacturing will likely involve a large network of suppliers to provide subsystems or components. In many cases, manufacturing is performed by companies subcontracted by the product's original designer.

3.3.1.3 References

- DOD Instruction 5000.02: "Production and Deployment (P&D) Phase… The purpose of the P&D Phase is to produce and deliver requirements-compliant products to receiving military organizations… In this phase, the product is produced and fielded for use by operational units" (Department of Defense 2015).

- SAND2015-3667C “Fabrication, including assembly, of systems can be a complex operation with many components and subassemblies coming together to create the final product” (McCrory, Kao and Blair 2015).

3.3.2 Development (Software Products)

3.3.2.1 Definition
Development is the process of coding to produce executable software units that realize the intended design.

3.3.2.2 Explanation
The source code for the software product is produced by teams of software developers. Libraries, APIs, and software components from other sources may also be incorporated as needed. If required, the final code is compiled into binaries for delivery to the customer.

3.3.2.3 References
- ISO 12207 7.1.5.1 “The purpose of the Software Construction Process is to produce executable software units that properly reflect the software design” (International Organization for Standardization 2008).
- SAFECode Best Practices: “Programming… This phase is where programmers translate the design and specification into actual code. Effective coding requires implementers to enforce consistent coding practices and standards throughout all aspects of producing the application” (Software Assurance Forum for Excellence in Code 2008).

3.4 Testing

3.4.1 Definition
Testing is the systematic assessment of the product to ensure its functionality and performance. Testing may also occur in conjunction with other stages of the lifecycle.

3.4.2 Explanation
The rigor of testing may vary greatly, ranging from in-depth examination by dedicated personnel using specialized tools for analysis to ad hoc testing for the sake of debugging and basic validation. Assessment mechanisms such as diagnostic equipment or evaluation software may be acquired from third-party suppliers. Due to the continuous nature of testing, this phase often occurs at multiple points in the product lifecycle.

3.4.3 References
- MITRE Systems Engineering Guide “Test & Evaluation (T&E) is the process by which a system or components are compared against requirements and specifications through testing. The results are evaluated to assess progress of design, performance, supportability, etc.” (The MITRE Corporation 2013).
- FFIEC IT Development and Acquisition Handbook: “The Testing phase requires organizations to complete various tests to ensure the accuracy of programmed code, the inclusion of expected functionality, and the interoperability of applications and other

network components. Thorough testing is critical to ensuring systems meet organizational and end-user requirements" (Federal Financial Institutions Examination Council 2004).

3.5 Distribution

3.5.1 Definition

For hardware, distribution is the packaging, warehousing, and delivery of the product. For software, distribution is defined as the storage and delivery of the product through physical and/or nonphysical means. As with testing, distribution may also occur within other stages of the lifecycle whenever the product or its components are in transit. For instance, there could be a distribution process involved in the maintenance or disposal of a product.

3.5.2 Explanation

Various intermediaries may be involved in the distribution of the completed product. For hardware products or software distributed on physical media, this may include shipping companies or courier services. Depending on the distance and geography involved, air, sea, or land transport may be required. Physical goods may be stored in warehouses and distribution centers while awaiting transport. Software can also be transmitted over the Internet, possibly using third-party web hosting services or content distribution systems.

3.5.3 References

- Chou et al. "In an SW[software]-focused supply chain, because of the nature of SW, SW components can be distributed by nonphysical means, via the electronic platform." (Chou and Ruchika 2006)
- Axelrod. "Physical products must be transported via land, sea and/or air depending on size, weight, toxicity and flammability, for example." (Axelrod 2013)

3.6 Use and Maintenance

3.6.1 Definition

This stage details two simultaneous activities, relevant to both software and hardware products. Use involves the operation of the product for its intended function. Maintenance encompasses sustaining and improving the capability of the product.

3.6.2 Explanation

Use of the product depends on the acquiring agency's needs and policies. The product may need to be configured and connected to other devices as part of its normal use.

Maintenance can be further separated into two categories, corrective maintenance and enhancement maintenance. Corrective maintenance aims to remedy problems and vulnerabilities discovered after the product's delivery. Enhancement maintenance adapts the product to changes in the customer's needs or provides additional functionality.

3.6.3 References

- NIST SP 800-64 "2.3.4 Operations and Maintenance… In this phase, systems are in place and operating, enhancements and/or modifications to the system are developed and

tested, and hardware and/or software is added or replaced" (Grance, Hash and Stevens 2003).

- House Systems Development Life-Cycle (SDLC) "Sustainment (2-11)" "Conduct maintenance and service activity to keep the system operational. This includes response to system failures and notification to users through the system alert process. Activity also includes proactive and preventive actions that help keep the system from failing" (U.S. House of Representatives 1999).

3.7 Disposal

3.7.1 Definition

Disposal covers the final decommissioning and removal of a product at the end of its life. For hardware, disposal involves disassembly for reuse, recycling, or destruction. For software, disposal includes uninstalling, deleting, or otherwise discontinuing the use of the product.

3.7.2 Explanation

The process of disposal also covers migration to replacement systems, if appropriate. Data on the old system may need to be removed or transferred.

3.7.3 References

- DOD Instruction 5000.02 "Disposal. At the end of its useful life, a system will be demilitarized and disposed of in accordance with all legal and regulatory requirements and policy relating to safety (including explosives safety), security, and the environment" (Department of Defense 2015)
- NIST SP 800-64 "2.3.5 Disposition… the final phase in the SDLC, provides for disposal and contract closeout of the system or contract in place" (Grance, Hash and Stevens 2003).

4. LIFECYCLE RISK

At every stage, there can potentially be multiple suppliers involved, with varying levels of oversight. In this section, potential attack vectors and adversaries associated with each phase of the product lifecycle are identified. Particularly noteworthy is a report recently released by the Office of the Deputy Assistant Secretary of Defense for Research and Engineering cataloging various supply chain attacks and mapping them to the defense acquisition lifecycle (Reed, Miller and Popick 2014). This paper demonstrates that some phases are riskier than others and that hardware and software products have unique risk profiles. Using the product lifecycle model identified previously, as well as building upon Reed, Miller, and Popick's findings and other pieces of literature, this report more closely examines the risks encountered by software and hardware products moving through the supply chain.

After analysis, it was determined that the Manufacturing stage holds the largest risk for hardware, while the largest risk for software is distributed across both the Development and Distribution phases. In the case of hardware, the Manufacturing step contains the most complex portion of the supply chain, relying on the aggregation of many different components and processes from individual actors. For software, online distribution presents increased risk of interception and forgery, more so than physical distribution. In terms of software development, the introduction of third-party applications and source code introduces a much higher amount of additional risk as opposed to code originating entirely in-house.

Sections 4.1 and 4.2 outline the risks that can be encountered along each phase of the lifecycle for both software and hardware.

4.1 Software Lifecycle Risk

4.1.1 Requirements

The Requirements phase generally presents a low risk to the supply chain since any malicious actions taken at this point in the lifecycle need corresponding actions in subsequent stages in order to be realized. For government agencies, requirements are typically specified by the acquiring organization. Thus it can be assumed that the agency has no intent to harm itself and therefore the product encounters little risk at this point. In COTS products, requirements are general guidelines outlining the direction of software design without much specific technical detail. As a result, only a limited number of attacks can be implemented at this stage, and oversight at other points in the lifecycle will likely prevent the attacks from succeeding.

4.1.2 Design

The goal of a secure design is to create a system that ensures the necessary authentication, authorization, confidentiality, data integrity, and availability laid out by the product requirements (IEEE Cybersecurity Initiative 2014). If software is not designed in accordance with basic security principles, or the designer is unaware of potential security hazards associated with the technology and architecture they propose to the development team, vulnerabilities may be inadvertently introduced into the planning process.

At the Design level, an exploit may result from either the intentional or unintentional decision to implement an unsuitable algorithm, data structure, or technique. For example, certain memory leakage vulnerabilities are specific to languages without garbage collection, and method complexity issues in software can allow an attacker to trigger worst-case behaviors that result in denial of service. For various programming languages (including Javascript, Python, and Ruby), interpreters may leak hash table address information to optimize object tracking. "Some hash table implementations directly store the address information in the table, while others permit inference of address information through repeated table scanning (Lee, Jang and Wang 2014). Plaintext communication, insecure state handling, and non-robust data validation are other design choices that affect the end security of the product. These are all flaws that are inherent in the poor design of the product, regardless of its actual performance. Designers must take security risks like these into account when weighing options because their decisions will either generate problems in later stages of the product lifecycle or help to guard against them.

Besides choice of technology, mistakes in the general design of a software application can also lead to major vulnerabilities. For instance, the 2013 Target data breach was a recent, real-world example of a design flaw leading to a hack. Its "environment was 'crunchy on the outside and chewy in the middle.' As a result, it was 'easy to get to…where all the data was stored' once the attackers had compromised the point-of-sale system" (Higgins 2014). The Target design placed the majority of its defenses on the perimeter, meaning that a hacker could access large amounts of sensitive information once inside the outermost security layer. Reducing the opportunities for attackers to exploit a potential weak spot requires thorough analysis of the overall attack surface, including the disabling or restricting of access to system services, application of the principle of least privilege, and employment of layered defenses wherever possible (Microsoft, Inc. 2015).

4.1.3 Development

The Development phase is when the source code for the entire application is written. Designing for security in software is futile unless the developer follows through and incorporates the necessary controls. Inadvertent or malicious insertion of vulnerabilities, or a failure to implement standard secure programming concepts is fairly common. The industry average for bugs per thousand lines of code is 1 - 25 (McConnell 2004), and most software contains hundreds of thousands, if not millions, of lines of code (McCandless 2014).

While back-door insertion is certainly possible with in-house development, more worrisome is the existence of undiscovered, exploitable vulnerabilities in third party libraries, APIs, and source code. The extension of security checks to include external software components as well as the publishing of a whitelist of tools determined to be safe would greatly reduce potential security bugs (Microsoft, Inc. 2015). Unfortunately, third-party software developers often do not have the budget or the resources to implement thorough quality assurance and consistent patch release cycles (Magalhaes and Magalhaes 2014). The Study of Software Related Cybersecurity Risk in Public Companies reported that out of those surveyed, 80% of respondents currently use commercial software. Less than one in five, however, have performed any formal assessment of their third-party applications. Veracode's State of Software Security report indicates that 30 - 70% of applications initially thought of as internally developed were actually comprised of third-

party components (Veracode 2015). The fact that enterprises are often unaware of non-internal source code interacting with their own software makes the risk extremely high that multiple security flaws arise as a result of third-party vulnerabilities.

4.1.4 Testing

Once the Development stage is completed, software is tested for functionality and performance to ensure the application runs as expected. The Testing phase is intended to reveal vulnerabilities in software code that were not recognized during implementation. At a bare minimum, companies usually check for common software vulnerabilities such as buffer overflow or code injection flaws, and test the software's reaction to random or malformed input formats (fuzz testing). Risk increases, however, when the evaluation process provides insufficient coverage of the program's attack surface.

Automated testing tools finish orders of magnitude more quickly than manual testing and screen for a breadth of common, low-hanging fruit technical flaws. Automation, nevertheless, includes a higher percentage of false positives and cannot uncover errors in logic or design (Kesäniemi and Oy 2009). Companies using exclusively automated testing or manual testing will more than likely miss a large portion of potential vulnerabilities. There is also the possibility that testing tools or employees will insert exploitable code into software functions and mark that section as verified. However, when testing is outsourced to third-party assessors, it is unlikely that they will be able to insert malicious code into the product or otherwise compromise the system given their level of influence. The company normally sends them a copy of the software and takes only their verbal or written feedback in return. Despite this, a hacker has the opportunity to overlook vulnerabilities he or she finds with the intent to exploit the function or interface at a later date.

4.1.5 Distribution

Some software firms such as Juniper Networks distribute physical media to customers in the form of CDs. Many companies, however, are purely digital, and provide downloadable versions of their product as well as updates through their website.

For distribution via physical media, the risk lies in interception of the product. Once in the hands of the attacker, it can be modified and en route to the customer with very little time delay. Although shipping providers and distribution outlets often have security solutions in place, including tamper-proof packaging and online monitoring, an attack may still be able to bypass these restrictions.

When the internet is the means of distribution, risk is higher than with physical transfer and consists of server-to-client payload disruption or spoofing. Malicious sites masquerading as authentic software distributers present a facet of this digital distribution risk. Even if the software itself does not contain malware, it can infect user PCs during the download process. In a study by the International Data Corporation, tracking cookies and spyware were encountered 78% of the time when downloading software from non-company websites and those with peer-to-peer sharing. Trojans and other malicious adware were identified 36% of the time. On the physical media front, Trojans and malicious adware were found 20% of the time (Gantz, et al. 2013).

Malware infection from illegitimate software distributors constitutes a significant risk linked to the Distribution phase.

4.1.6 Use and Maintenance

During the use of a software product, system configuration and misalignment issues present a problem that exposes unforeseen security vulnerabilities. Server configurations, for example, play a key role in the security of software and improper server configuration management can lead to a variety of exposures (Tracy, et al. 2007). Default passwords that remain unchanged after installation, unnecessary services enabled, misconfigured encryption settings, etc. supply hackers with a foothold and most likely more capability to carry out an attack.

The security of stored data is another area of concern associated with the use of software. It involves preventing unauthorized people from accessing sensitive information as well as avoiding accidental or intentional destruction, infection or corruption of information. Regardless of the software company's product testing caliber, Personal Identifiable Information (PII) or limited release data may be exposed to attackers if the storage entity contains flaws in its own code or quality assurance evaluations.

Maintenance risks include spoofed software updates as well as third-party maintenance providers, legitimate or otherwise. As with the fraudulent distribution of software, both can allow viruses, Trojan horses, keystroke-logging software, authentication backdoors, and spyware into a system.

4.1.7 Disposal

Software caches data on a large variety of storage media, ranging from high-speed internal hard drives to optical storage on CDs and DVDs (Adroit Data Recovery Centre Pte Ltd. 2010). Risk in the Disposal stage stems from leftover sensitive information after uninstallation or deletion of software. To prevent attackers from accessing critical material, the data must be rendered unreadable after the product is no longer needed. "Normal 'erasing' of files does not remove data from a storage device, it just tells the computer that the old data sectors are now available to be overwritten" (Yale IT Services 2015). In most cases "erasing" simply changes the indexing of a file, and even if file space is subsequently filled with new data, there are sophisticated scanning methods that can be used to recover data previously stored in those locations (Gutmann 1996). Magnetic media should be demagnetized or overwritten while only physical destruction will do for optical media, and purely digital software should undergo a complete, formal data erasure process.

4.2 Hardware Lifecycle Risk

4.2.1 Requirements

As with software, the Requirements stage presents very little risk due to lack of intent. In the case of an acquisition, the acquiring agency has no intent to harm itself. While a malicious actor

may add requirements to force a product to include malicious functionality, this is unlikely to be documented or otherwise made visible until later stages.

4.2.2 Design

Although design frequently happens in-house for hardware products, designs frequently incorporate components from outside sources. For instance, a product may include prebuilt subsystems designed by a supplier. Also, outside sources may be used to assist in the design process. A company may license designs from another source or directly contract other companies to complete portions of the product design. Anytime an external source becomes involved, there is an increased risk in terms of threat.

For hardware products, attacks during the Design stage usually revolve around malicious modification of the product design. Simple attacks, such as making slight adjustments to parameters or component selection, may have consequences such as lower reliability or shorter useful lifespan. More sophisticated actions may involve the insertion of malicious circuits that enable backdoors, kill switches, or other much more severe attacks. These attacks would naturally have a much higher consequence, but also be much more difficult to execute without detection. Because designs usually undergo multiple stages of review, more sophisticated modifications will usually be easier to identify.

The feasibility of adding "Hardware Trojans" to integrated circuits has been demonstrated and widely developed in academic literature (Chakraborty, Narasimhan and Bhunia 2009). Furthermore, news sources have reported on multiple suspected instances of "kill switches" embedded in military hardware (Adee 2008).

4.2.3 Manufacture

Manufacture is usually the most complex part of the hardware lifecycle, involving the highest number of actors and processes. As a result, the risk of a supply chain attack is highest at this crucial point in the supply chain. The Office of the Deputy Assistant Secretary of Defense for Systems Engineering created a report that outlined possible attack vectors throughout a product's lifecycle, and found that the "Engineering and Manufacturing Development (EMD) phase is susceptible to the greatest number of attacks," with over two-thirds of all attacks applying to this stage (Reed, Miller and Popick 2014).

As with design, malicious insertion and modification are the most likely supply chain attacks in the Manufacturing stage. Components can be swapped for less reliable counterfeits or specially crafted look-alikes with hidden malicious payloads. Switching out a component in an automated production line requires much less skill and subtlety than changing a manually curated design or testing process.

The large number of suppliers involved in manufacturing supply chains increases the potential attack surface during the Manufacturing stage. In the iPhone case study, Apple identified 759 different suppliers and subcontractors involved in the manufacturing of its products, many in foreign countries (Apple Inc. 2015). Any one of those suppliers could intentionally or

accidentally compromise a component of the product, rendering the final output corrupt. In the absence of rigorous oversight, manufacturing presents many opportunities for adversaries to tamper with the product.

4.2.4 Testing

The Testing phase mainly exists to reduce the risk introduced in prior stages of the lifecycle. The majority of risk inherent in the testing process lies in vulnerabilities in testing procedures as implemented by producers. In addition, third parties involved in the testing process introduce additional attack vectors. Malicious actors may attempt to interfere with the testing process, but these attacks do not present a major risk.

As documented in academic literature, testing can aid detection of supply chain attacks performed during the Design and Manufacturing phases. Some basic attacks such as blatant counterfeiting can be caught through normal quality assurance testing. However, testing for malicious modifications often requires disassembly, which is both impractical and expensive when performed on a large scale (Chakraborty, Narasimhan and Bhunia 2009). Attacks on individual electronic components are particularly hard to detect, with a Defense Science Board Task Force report concluding that "electrical testing and reverse engineering cannot be relied upon to detect undesired alterations in military integrated circuits" (Defense Science Board 2005).

A particularly determined and advanced adversary would likely be able to craft attacks that circumvent normal testing procedures, and intense comprehensive testing would be impractical to implement. As a result, vulnerabilities in the Testing stage are difficult to eliminate, and controlling threats in earlier stages would be a more realistic way of managing the risk of malicious insertions.

There are a couple of specific attacks that may be perpetrated in the Testing stage. Products (both hardware and software) may be sent to an external organization for verification, providing another entry point for adversaries. Testing often requires the use of specialized test equipment or tools, which may be tampered with by a malicious actor. For instance, an automated test may be reprogramed to allow defective products to pass quality assurance testing, decreasing the overall reliability of the product. These attacks present relatively little risk, since they are difficult to perform. Furthermore, the type of testing can limit the scope of a successful attack. Notably, if only a random sample of products is tested, then only the small number of products that were selected would be vulnerable to malicious activity.

4.2.5 Distribution

While the Distribution stage presents a number of different threats and possible attacks on hardware products, effective countermeasures exist and are widely implemented.

During transit, the product is generally outside the control of both the manufacturer and the customer, providing opportunities for malicious actors to access and tamper with the product. However, numerous programs already exist to provide security during shipping of products. For

instance, US Customs and Border Protection manage the Customs-Trade Partnership Against Terrorism (C-TPAT), which works with companies to improve security during transit of goods. Certified "C-TPAT Partners" must agree to "protect the supply chain, identify security gaps, and implement specific security measures and best practices" (U.S. Customs and Border Protection 2015). While primarily aimed at preventing terrorism, the security measures implemented as part of C-TPAT also limit access by other malicious actors and enhance the overall supply chain security of the product. Similar programs exist around the world, and most shipping companies implement their own security measures.

4.2.6 Use and Maintenance

During the operation and normal use of the product, adversaries will attempt to stage attacks on the product's vulnerabilities. Risks during this stage have been well studied and categorized. Methodologies such as the NIST Risk Management Framework can be used to evaluate security concerns during this stage. From a supply chain standpoint, configuration management presents the most relevant area of concern in the Use stage. An improperly configured product may have weaker security, which presents additional vulnerabilities. Weak operational practices may also increase the product's vulnerability, such as leaving it in an insecure environment, poorly protecting passwords, and not taking proper care of the product.

Most of the risk during the Maintenance stage concerns the involvement of third parties. According to NIST SP800-161, "With many maintenance contracts, information on mission, organization, and system-specific objectives and requirements is shared between the organization and its system integrators, suppliers, or external service providers, allowing for vulnerabilities and opportunities for attack" (Boyens, et al. 2015).

In general, the original supplier will provide corrective maintenance in some form, such as vulnerability notices or updates. The supplier's diligence in providing updates and the acquiring agency's procedures for applying updates will affect the risk of a subsequent attack. In addition, replacement parts may be necessary to maintain hardware products. The sourcing of these parts presents additional attack vectors, especially if parts are not available from the original manufacturer. If the product is sent to a third party maintainer or technicians are brought onsite, maintenance may pose additional risks involving the trustworthiness of these parties.

The location of maintenance affects the vulnerability of the product. For instance, this research indicates that using a third party maintainer is less secure than using maintenance services from the product's original producer; both are less secure than performing maintenance in-house. NIST SP 800-161 explicitly provides additional guidance for managing risk associated with "Nonlocal Maintenance" and "Maintenance Tools" associated with external maintainers (Boyens, et al. 2015).The way maintenance is performed can be just as large a risk as the need for maintenance.

4.2.7 Disposal

The primary risk of disposal involves the protection of sensitive data. If the product was involved in storing or passing data, memory may retain information even after the device is

decommissioned. Adversaries may attempt to use forensic tools to recover information from erased media. Agencies must ensure memory devices are properly erased prior to sending products to recyclers to prevent loss of confidentiality. DOD procedures recommend that "Military electronics shall be shredded or crushed, preferably to the point of pulverization" (Department of Defense 2011). Ensuring such procedures are in place would effectively mitigate risk when disposing of a hardware product.

The location of disposal can also greatly affect the security of the product during its end of life. The less access outside parties have to information about or pertaining to the product, the less vulnerable a product is to theft and data loss. For this reason, companies must have proper methods of disposal, preferably done in-house rather than by the manufacturer or a third party.

4.3 Product Lifecycle Risk Quantification

From the conclusions of sections 4.1 and 4.2, it becomes clear that the risk is not evenly spread throughout the product lifecycle. Not only do hardware and software products have differences in the lifecycle stages involved, but some of these stages contain more risk than do others. Below are tables for software and hardware, quantifying the approximate values of risk according to each stage of the lifecycle.

Software Risk Weighting						
Req. .05	Design .15	Development .20	Testing .10	Distribution .25	Use/Maint. .15	Disposal .10

Figure 2. Software Risk at Each Lifecycle Phase

As discussed in section 4.1, the phase of the software product lifecycle most vulnerable to attack and malicious activity is Distribution, closely followed by the Development phase. Requirements has extremely low (if any) risk, while the remaining stages are somewhere in the middle.

Hardware Risk Weighting						
Req. .05	Design .10	Manufacturing .30	Testing .10	Dist. .10	Use/Maintenance .20	Disposal .15

Figure 3. Hardware Risk at Each Lifecycle Phase

Even though hardware and software have the same general stages, the amount of risk involved at each stage is significantly different. Manufacturing and Use/Maintenance are the largest stages of risk, with all others less important.

Evaluating the risk at each stage in the product lifecycle is necessary to understand what an acquiring agency should consider when performing a risk assessment of a product they are looking to purchase. The weight they place on each lifecycle stage should be determined by the amount of risk inherent in that phase.

5. APPLICATIONS TO GOVERNMENT ACQUISITIONS

After considering the dangers and vulnerabilities of a product over all stages of its life, it is important to recognize the value of analyzing and quantifying lifecycle risk in the government acquisition process. This report highlights the importance of a thorough understanding of opportunities for exploitation, inherent not just during the manufacturing or design phases, but also continuing all the way up until the end of a product's life. An acquiring agency would be remiss if it selected a product that may appear secure at the time of purchase, but actually contains vulnerabilities that appear in later lifecycle stages.

Following safe operational practices, such as those outlined in the NIST Risk Management Framework (SP800-53), is certainly a necessary step to reduce vulnerable processes over the course of the product lifecycle. However, even with these standards in place, there may be large exposures still unaccounted for. This report provides numerous examples of risks, many of which are not inherently covered by common operational guidelines and all of which should ideally be considered before the purchase and acquisition of a product.

6. REFERENCES

Acquisition Directorate of the U.S. Coast Guard. "Major Systems Acquisition Manual (MSAM)." *COMDTINST M5000.10D.* May 29, 2015. http://www.uscg.mil/directives/cim/5000-5999/CIM_5000_10D.pdf.

Active kernel releases. March 11, 2015. https://www.kernel.org/releases.html (accessed July 10, 2015).

Adee, Sally. *The Hunt for the Kill Switch.* May 1, 2008. http://spectrum.ieee.org/semiconductors/design/the-hunt-for-the-kill-switch.

Adroit Data Recovery Centre Pte Ltd. *About Storage Media* . 2010. http://www.adrc.com/ckr/about_storage_media.html.

Apple Inc. *Apple Recycling Program.* 2015. http://www.apple.com/recycling/.

—. *Apple Service Programs* . 2015. https://www.apple.com/support/programs/.

—. *iPhone 5s & iPhone 5c Arrive on Friday, September 20.* September 17, 2013. https://www.apple.com/pr/library/2013/09/16iPhone-5s-iPhone-5c-Arrive-on-Friday-September-20.html.

—. *Our Suppliers.* 2015. https://www.apple.com/supplier-responsibility/our-suppliers/.

—. "Supplier Responsibility Standards." *Apple Supplier Responsibility.* January 1, 2015. https://www.apple.com/supplier-responsibility/pdfs/supplier_responsibility_standards.pdf.

Axelrod, Warren C. "Mitigating Software Supply Chain Risk." *Information Systems Audit and Control Association*, 2013: 1-9.

Bishop, Bryan. *Apple designer Christopher Stringer reveals iPhone design process, says 'we've been ripped off'.* July 31, 2012. http://www.theverge.com/2012/7/31/3207582/apple-designer-christopher-stringer-samsung-trial-testimony.

Bloom, Gedare, Bhagirath Narahari, and Rahul Simha. "Fab Forensics: Increasing Trust in IC Fabrication." *IEEE International Conference on Technologies for Homeland Security (HST).* Waltham, MA: IEEE, 2010. 99-105.

Boyens, Jon, Celia Paulsen, Rama Moorthy, and Nadya Bartol. *NIST Special Publication 800-161.* Gaithersburg, MD: National Institute of Standards and Technology, 2015.

Brown, Neil. *Linux kernel design patterns.* June 8, 2009. http://lwn.net/Articles/336224/.

Chakraborty, Rajat Subhra, Seetharam Narasimhan, and Swarup Bhunia. "Hardware Trojan: Threats and Emerging Solutions." *High Level Design Validation and Test Workshop.* San Francisco: IEEE, 2009. 166 - 171 .

Chen, Weiwei, Rui Kang, Diganta Das, and Michael Pecht. "Study on Electronics Recycling Process." *Industrial Engineering and Engineering Management.* Beijing: IEEE, 2009. 1602-1606.

Chou, Mabel C., and A. Ruchika . "An In-depth Study of the Software Supply Chain." *IEEE International Conference on Industrial Informatics.* Singapore: IEEE, 2006. 753 - 758.

Debian Wiki. *How To Upgrade Kernel.* April 25, 2014. https://wiki.debian.org/HowToUpgradeKernel.

Defense Science Board. *Defense Science Board Task Force on High Performance Microchip Supply.* Washington, D.C.: Office of the Under Secretary of Defense For Acquisition, Technology, and Logistics, 2005.

Department of Defense. "Department of Defense Instruction 5000.02." January 7, 2015. http://www.acq.osd.mil/fo/docs/500002p.pdf.

—. "Department of Defense Manual 4160.28, Volume 3." June 7, 2011. http://www.dtic.mil/whs/directives/corres/pdf/416028m_vol3.pdf.
Duhigg, Charles, and Keith Bradsher. *How the U.S. Lost Out on iPhone Work.* January 21, 2012. http://www.nytimes.com/2012/01/22/business/apple-america-and-a-squeezed-middle-class.html.
Ellingwood, Justin. *How To Migrate Linux Servers.* Feburary 27, 2014. https://www.digitalocean.com/community/tutorials/how-to-migrate-linux-servers-part-1-system-preparation.
Federal Financial Institutions Examination Council. "Development and Acquisition IT Examination Handbook." April 2004. http://ithandbook.ffiec.gov/ITBooklets/FFIEC_ITBooklet_DevelopmentandAcquisition.pdf.
FinancesOnline. *How and Where iPhone Is Made: A Surprising Report on How Much of Apple's Top Product is US-manufactured.* n.d. http://financesonline.com/how-iphone-is-made/.
Gantz, John F, et al. *The Dangerous World of Counterfeit and Pirated Software.* Framingham, MA: International Data Corporation, 2013.
Grance, Tim, Joan Hash, and Marc Stevens. *Security Considerations in the Information System Development Life Cycle.* Special Publication 800-64, Gaithersburg, MD: National Institute of Standards and Technology, 2003.
Gutmann, Peter. "Secure Deletion of Data from Magnetic and Solid-State Memory." *Sixth USENIX Security Symposium.* San Jose, CA: USENIX, 1996. 77-90.
Hansen, Jessica. *Happy Earth Day! Be Green: Recycle Your Old Electronics Today.* April 22, 2014. http://scoop.intel.com/happy-earth-day-green-recycle-electronics-today/.
Heisler, Yoni. *How Apple conducts Market Research and keeps iOS source code locked down.* Network World. August 3, 2012. http://www.networkworld.com/article/2222892/wireless/how-apple-conducts-market-research-and-keeps-ios-source-code-locked-down.html (accessed July 14, 2015).
Higgins, Kelly Jackson. *10 Common Software Security Design Flaws.* August 27, 2014. http://www.darkreading.com/application-security/10-common-software-security-design-flaws/d/d-id/1306776.
IEEE Cybersecurity Initiative . *Avoiding the Top 10 Security Flaws.* 2014. http://cybersecurity.ieee.org/center-for-secure-design/avoiding-the-top-10-security-flaws.html.
iFixit. *iFixit Store.* 2015. https://www.ifixit.com/Store.
Independent Media. *FCAPS.* December 4, 2012. http://www.tech-faq.com/fcaps.html.
Intel Corporation. "Intel Global Manufacturing Facts." 2010. http://www.intel.com/content/www/us/en/silicon-innovations/standards-global-manufacturing-facts.html.
—. "Intel Quality System Handbook, Dec. 2013." December 2013. http://www.intel.com/content/www/us/en/manufacturing/quality-system-handbook.html.
—. *Intel Tick-Tock Model.* 2015. http://www.intel.com/content/www/us/en/silicon-innovations/intel-tick-tock-model-general.html.
—. "Introduction to Intel Architecture." *The Basics of Intel Architecture.* 2014. http://www.intel.com/content/www/us/en/intelligent-systems/embedded-systems-training/ia-introduction-basics-paper.html.

—. "Transistors to Transformations." *How Intel Makes Chips: Transistors to Transformations.* 2012. https://www-ssl.intel.com/content/www/us/en/history/museum-transistors-to-transformations-brochure.html.
International Organization for Standardization. "ISO/IEC 12207:2008." 2008.
International Organization for Standardization. "ISO/IEC 15288-2008." 2008.
Ippolito, Greg. *Linux Tutorial - Software Development on Linux.* 2014. http://www.yolinux.com/TUTORIALS/LinuxTutorialSoftwareDevelopment.html.
Juniper Networks, Inc. *Company Profile.* 2015. https://www.juniper.net/us/en/company/profile/.
—. *Junos Space - Download Software.* 2015. https://www.juniper.net/support/downloads/space.html.
—. *Junos Space.* October 2014. http://www.juniper.net/assets/us/en/local/pdf/datasheets/1000297-en.pdf.
—. *Junos Space Network Management Platform - Download Software.* 2015. https://www.juniper.net/support/downloads/?p=space#sw.
—. *Junos Space SDK Release Notes.* January 2015. http://developer.juniper.net/shared/jdn/docs/releasenotes/Junos%20Space%20SDK%20Release%20Notes.pdf.
—. *Network Operating System Evolution.* October 2010. http://www.juniper.net/us/en/local/pdf/whitepapers/2000264-en.pdf.
—. *Product Reclamation and Recycling Program.* January 30, 2014. http://www.juniper.net/techpubs/en_US/release-independent/cse-series/topics/reference/general/cse-environ-recycle-pgm-weee.html.
—. *Strategic Alliances.* 2015. http://www.juniper.net/us/en/partners/strategic-alliances/.
Kesäniemi, Ari, and Nixu Oy. *Automatic vs. Manual Code Analysis.* November 17, 2009. https://www.owasp.org/images/5/53/Ari_kesaniemi_nixu_manual-vs-automatic-analysis.pdf.
Khan, Shuah. *Linux Kernel Testing and Debugging.* July 10, 2014. http://www.linuxjournal.com/content/linux-kernel-testing-and-debugging.
Kroah-Hartman, Greg, Jonathan Corbet, and Amanda McPherson. "Linux Kernel Development." August 2009. http://www.linuxfoundation.org/sites/main/files/publications/whowriteslinux.pdf.
Lee, Byoungyoung, Yeongjin Jang, and Tielei Wang. *Abusing Performance Optimization Weaknesses to Bypass ASLR.* August 2014. https://www.blackhat.com/us-14/briefings.html#abusing-performance-optimization-weaknesses-to-bypass-aslr.
Linux Foundation. *Training and Certification.* 2015. http://training.linuxfoundation.org/.
Linux Test Project. *Linux Test Project.* 2012. http://linux-test-project.github.io/.
Lowensohn, Josh. *Inside the building where Apple tortures the iPhone 6.* September 25, 2014. http://www.theverge.com/2014/9/25/6845611/inside-apples-iphone-6-torture-building.
Magalhaes, Ricky M., and Monique L. Magalhaes. *Third-Party Software is a Security Threat (Part 1).* September 17, 2014. http://www.windowsecurity.com/articles-tutorials/misc_network_security/third-party-software-security-threat-part1.html.
McCandless, David. *Codebases.* November 26, 2014. http://www.informationisbeautiful.net/visualizations/million-lines-of-code/.
McConnell, Steve. "Code Complete." In *Code Complete*, by Steve McConnell, 521. Redmond, WA: Microsoft Press, 2004.

McCrory, Mitch F., Gio K. Kao, and Dianna S. Blair. *Supply Chain Risk Management: The Challenge in a Digital World.* SAND2015-3667C, Albuquerque, NM: Sandia National Laboratories, 2015.

Meunier, Pascal. *Classes of Vulnerabilities and Attacks.* April 2010. http://homes.cerias.purdue.edu/~pmeunier/aboutme/classes_vulnerabilities.pdf.

Microsoft, Inc. *Security Development Lifecycle.* 2015. http://www.microsoft.com/en-us/sdl/default.aspx.

Mollick, Ethan. "Establishing Moore's Law." *IEEE Annals of the History of Computing* 28, no. 3 (2006): 62 - 75 .

Panzarino, Matthew. *This is how Apple's top secret product development process works.* January 24, 2012. http://thenextweb.com/apple/2012/01/24/this-is-how-apples-top-secret-product-development-process-works/.

Reed, Melinda, John F. Miller, and Paul Popick. "Supply Chain Attack Patterns: Framework and Catalog." August 2014. http://www.acq.osd.mil/se/docs/Supply-Chain-WP.pdf (accessed July 14, 2015).

Satariano, Adam. *The IPhone's Secret Flights From China to Your Local Apple Store.* September 11, 2013. http://www.bloomberg.com/news/articles/2013-09-11/the-iphone-s-secret-flights-from-china-to-your-local-apple-store.

Slivka, Eric. *Apple Returns 5-8 Million Defective iPhones to Foxconn?* April 22, 2013. http://www.macrumors.com/2013/04/22/apple-returns-5-8-million-defective-iphones-to-foxconn/.

Software Assurance Forum for Excellence in Code. "Software Assurance: An Overview of Current Industry Best Practices." Feburary 2008. http://www.safecode.org/publication/SAFECode_BestPractices0208.pdf.

Software in the Public Interest, Inc. *Debian on CDs.* July 4, 2015. https://www.debian.org/CD/.

Sprott, David, and Lawrence Wilkes. *Understanding Service-Oriented Architecture.* January 2004. https://msdn.microsoft.com/en-us/library/Aa480021.aspx.

The MITRE Corporation. *Test and Evaluation.* September 2013. http://www.mitre.org/publications/systems-engineering-guide/se-lifecycle-building-blocks/test-and-evaluation.

Topolsky, Joshua. *Inside Apple's 'black lab' wireless testing facilities.* July 16, 2010. http://www.engadget.com/2010/07/16/inside-apples-black-lab-wireless-testing-facilities/.

Torvalds, Linus. *How the Development Process Works.* October 28, 2014. https://www.kernel.org/doc/Documentation/development-process/2.Process.

—. *How The Development Process Works.* October 28, 2014. https://www.kernel.org/doc/Documentation/development-process/2.Process.

—. *How to Get Your Change Into the Linux Kernel or Care And Operation Of Your Linus Torvalds.* May 29, 2015. https://www.kernel.org/doc/Documentation/SubmittingPatches.

—. *Linux Kernel Source Tree.* 2015. https://github.com/torvalds/linux.

Tracy, Miles, Wayne Jansen, Karen Scarfone, and Theodore Winograd. *NIST Special Publication 800-44 Version 2.* Gaithersburg, MD: National Institute of Standards and Technology, 2007.

Turley, Jim. *Intel vs. ARM: Two titans' tangled fate.* March 3, 2014. http://www.itworld.com/article/2831623/mobile/intel-vs--arm--two-titans--tangled-fate.html.

U.S. Customs and Border Protection. *C-TPAT: Customs-Trade Partnership Against Terrorism.* 2015. http://www.cbp.gov/border-security/ports-entry/cargo-security/c-tpat-customs-trade-partnership-against-terrorism.

U.S. House of Representatives. "Systems Development Life-Cycle Policy." March 24, 1999. http://www.house.gov/content/cao/procurement/ref-docs/SDLCPOL.pdf.

U.S. Senate Committee on Armed Services. *Inquiry into Counterfeit Electronic Parts in the Department of Defense Supply Chain.* Washington, D.C.: U.S. Goverment Printing Office, 2012.

Veracode. *What is Third-Party Software Security.* 2015. http://www.veracode.com/products/vendor-application-security-testing/what-is-third-party-software-security.

Walters, Helen. *Apple's Design Process.* Bloomberg Business. March 8, 2008. http://www.businessweek.com/the_thread/techbeat/archives/2008/03/apples_design_p.html (accessed July 14, 2015).

Wikimedia Foundation, Inc. *FCAPS.* June 8, 2015. https://en.wikipedia.org/wiki/FCAPS.

—. *Juniper Networks.* July 7, 2015. https://en.wikipedia.org/wiki/Juniper_Networks.

Yale IT Services. *Secure removal of data or disposal of computing devices.* 2015. http://its.yale.edu/secure-computing/security-standards-and-guidance/data-and-application-security/protecting-yales-data/secure-removal-data-or-disposal-computing-devices.

Zhang, Wenlei, and Yushun Fan. "A Conceptual Framework for Product Lifecycle Modeling." *Proceedings of the First International Conference on Innovative Computing, Information and Control.* Beijing, 2006.

GLOSSARY

Design For both hardware and software lifecycles, design entails creating a concrete course of action for meeting the specifications that result from the Requirements stage. In the case of hardware, design also involves functional prototyping.

Development The process of coding to produce executable software units that realize the intended design.

Disposal The final decommissioning and removal of a product at the end of its life. For hardware, disposal involves disassembly for reuse, recycling, or destruction. For software, disposal includes uninstalling, deleting, or otherwise discontinuing the use of the product.

Distribution For hardware, distribution is the packaging, warehousing, and delivery of the product. For software, distribution is defined as the storage and delivery of the product through physical and/or nonphysical means. As with testing, distribution may also occur within other stages of the lifecycle whenever the product or its components are in transit. For instance, there could be a distribution process involved in the maintenance or disposal of a product.

Maintenance The process of sustaining and improving the capability of the product.

Manufacturing The production and realization of the specified design and prototypes.

Product Lifecycle The series of phases that characterize a product's development from conception to its end of life.

Requirements The determination of demand and specifications for the product under development. This definition is common to both hardware and software lifecycles.

Supply Chain The network of processes involved in the evolution of a product over its lifecycle.

Testing The systematic assessment of the product to ensure its functionality and performance. Testing may also occur in conjunction with other stages of the lifecycle.

Use The operation of the product for its intended function.

APPENDIX A: LITERATURE SURVEY DATA

The following table lists all of the sources that contain product lifecycle models. These models were compiled and analyzed according to the specific steps of the product lifecycle. After a consensus was formed as to the seven main steps of the product lifecycle, each source was labeled by the number of matching steps it contained. The "*" symbol refers to a commonality between the source's product lifecycle term and one of the seven terms used in this paper. For example, the first source, Abramovici, uses a product lifecycle model that utilizes, among others, the Manufacture/Development stage, the Use/Maintenance stage, and the Disposal stage.

	Requirements	Design	Manufacture/ Development	Testing	Distribution	Use/Maintenance	Disposal
Abramovici, M. "Future trends in product lifecycle management (PLM)." In *The future of product development*, pp. 665-674. Springer Berlin Heidelberg, 2007.			*			*	*

The table begins on the next page.

Table 2. Sources used in Literature Survey

	Requirements	Design	Manufacture/ Development	Testing	Distribution	Use/Maintenance	Disposal
Abramovici, M. "Future trends in product lifecycle management (PLM)." In *The future of product development*, pp. 665-674. Springer Berlin Heidelberg, 2007.			*			*	*
"Acquisition Management Directive," Department of Homeland Security, January 20, 2010, accessed June 8, 2015. https://www.dhs.gov/xlibrary/assets/foia/mgmt_directive_102-01_acquisition_management_directive.pdf	*		*	*		*	
Aitken, James, Paul Childerhouse, and Denis Towill. "The impact of product life cycle on supply chain strategy." *International Journal of Production Economics* 85, no. 2 (2003): 127-140.	*	*	*		*		
Al-Mudimigh, Abdullah S., Mohamed Zairi, and Abdel Moneim M. Ahmed. "Extending the concept of supply chain: The effective management of value chains." International Journal of Production Economics 87, no. 3 (2004): 309-320.			*		*		
Alting, D. Leo, and D. Jørgen Jøgensen. "The life cycle concept as a basis for sustainable industrial production." CIRP Annals-Manufacturing Technology 42, no. 1 (1993): 163-167.		*	*			*	*
Amann, K. "Product lifecycle management: empowering the future of business." *CIM Data, Inc* (2002).		*	*			*	
Axelrod, Warren C. "Mitigating Software Supply Chain Risk," Information Systems Audit and Control Association, Vol. 4, 2013, accessed June 9, 2015. http://www.isaca.org/Journal/archives/2013/Volume-4/Documents/jol13v4-Mitigating-Software.pdf.	*	*	*	*	*	*	*
Beamon, Benita M. "Supply chain design and analysis: Models and methods." International journal of production economics 55, no. 3 (1998): 281-294.			*		*		
Ben Khedher, A.; Henry, S.; Bouras, A., "Integration between MES and Product Lifecycle Management," Emerging Technologies & Factory Automation (ETFA), 2011 IEEE 16th Conference on , vol., no., pp.1,8, 5-9 Sept. 2011		*	*			*	*

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Additional lifecycle from above source		*	*			*	*
Additional lifecycle from above source		*	*				*
Boehm, Barry W. "A spiral model of software development and enhancement." *ACM SigSoft Software Engineering Notes*, 1986: 22-42.	*		*	*			
Borg, S. "Securing the supply chain for electronic equipment: A strategy and framework." Internet Security Alliance Publication (2008).	*		*		*	*	
Brissaud, D., and S. Tichkiewitch. "Product models for life-cycle." CIRP Annals-Manufacturing Technology 50, no. 1 (2001): 105-108.		*	*			*	
Broekel, J., and G. Scharr. "The specialities of fibre-reinforced plastics in terms of product lifecycle management." *Journal of Materials Processing Technology* 162 (2005): 725-729.	*		*			*	*
Budde, O.; Golovatchev, J.; Chin-Gi Hong, "Next generation PLM-process management for the development of telecommunications products in the multi-lifecycle environment," *Management of Innovation and Technology, 2008. ICMIT 2008. 4th IEEE International Conference on* , vol., no., pp.391,396, 21-24 Sept. 2008	*						*
Cameron, Ian T., and G. D. Ingram. "A survey of industrial process modelling across the product and process lifecycle." *Computers & Chemical Engineering* 32, no. 3 (2008): 420-438.	*	*	*			*	*
Chiabert, Paolo, Franco Lombardi, J. Martinez, and J. Sauza. "Visualization model for product lifecycle management." *Annals of Faculty Engineering Hunedoara* 11, no. 1 (2013).		*	*		*	*	*
Chiang, Tzu-An, and Amy JC Trappey. "Development of value chain collaborative model for product lifecycle management and its LCD industry adoption." *International Journal of Production Economics* 109.1 (2007): 90-104.	*	*	*	*		*	*
Additional lifecycle from above source	*	*	*	*	*	*	
Chou, Mabel C., and A. Ruchika. "An in-depth study of the software supply chain." (2007). Cyber Security and Information Systems Information Analysis Center, accessed June 8, 2015.			*		*	*	
"Cost Structure and Life Cycle Cost (LCC) for Military Systems," North Atlantic Treaty Organization, June			*	*		*	*

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Daniel Mahler, Johan Aurik, Emmanuel Hembert, and Kristen Schrieber. “A Product Lifecycle Approach to			*		*	*	*
Dori, Dov, and Moshe Shpitalni. "Mapping knowledge about product lifecycle engineering for ontology construction via object-process methodology." *CIRP Annals-Manufacturing Technology* 54, no. 1 (2005): 117-122.		*	*	*		*	*
Additional lifecycle from above source	*	*				*	
Emami, Mir Shahriar, Norafida Binti Ithnin, and Othman Ibrahim. "Software process engineering: Strengths, weaknesses, opportunities and threats." In Networked Computing (INC), 2010 6th International Conference on, pp. 1-5. IEEE, 2010.	*	*	*	*	*	*	
Fandel, Günter, and M. Stammen. "A general model for extended strategic supply chain management with emphasis on product life cycles including development and recycling." *International journal of production economics* 89, no. 3 (2004): 293-308.		*	*		*		*
Finger, Susan, and John R. Dixon. "A review of research in mechanical engineering design. Part I: Descriptive, prescriptive, and computer-based models of design processes." *Research in engineering design* 1, no. 1 (1989): 51-67.	*	*	*			*	
Fleischmann, Bernhard, Herbert Meyr, and Michael Wagner. "Advanced planning." In *Supply chain management and advanced planning*, pp. 81-106. Springer Berlin Heidelberg, 2005.			*		*		
Gecevska, Valentina, Paolo Chiabert, Zoran Anisic, Franco Lombardi, and Franc Cus. "Product lifecycle management through innovative and competitive business environment." *Journal of Industrial Engineering and Management* 3, no. 2 (2010): 323-336.	*	*				*	
Geoffrion, Arthur M., and Richard F. Powers. "Twenty years of strategic distribution system design: An evolutionary perspective." *Interfaces* 25, no. 5 (1995): 105-127.			*		*		
Georgiadis, Patroklos, Dimitrios Vlachos, and George Tagaras. "The impact of product lifecycle on capacity planning of closed loop supply chains with remanufacturing." Production and Operations management 15, no. 4 (2006): 514-527.			*		*		*

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
H.-J.or. Bullinger, K.-P. Fahnrich, T. Meiren. "Service engineering—methodical development of new service products," International Journal of Production Economics, 85 (September (3)) (2003), pp. 275–287.	*	*	*		*		
Harrison, Mark, Duncan McFarlane, Ajith Kumar Parlikad, and Chien Yaw Wong. "Information management in the product lifecycle-the role of networked RFID." In *Industrial Informatics, 2004. INDIN'04. 2004 2nd IEEE International Conference on*, pp. 507-512. IEEE, 2004.		*	*		*	*	*
He, W., X. G. Ming, Q. F. Ni, W. F. Lu, and B. H. Lee. "A unified product structure management for enterprise business process integration throughout the product lifecycle." *International journal of production research* 44, no. 09 (2006): 1757-1776.	*	*	*				
Hepperle, C., et al. "An integrated lifecycle model of product-service-systems." *CIRP IPS2 Conference*. Vol. 2010. 2010.	*	*		*	*		
Additional lifecycle from above source	*	*	*		*	*	*
Hines, Peter, Mark Francis, and Pauline Found. "Towards lean product lifecycle management: a framework for new product development." *Journal of Manufacturing Technology Management* 17, no. 7 (2006): 866-887.		*	*				
Horvath, L.; Rudas, I.J., "Human-computer communication based enhanced decision assistance in lifecycle management of product information," Automation Congress, 2008. WAC 2008. World , vol., no., pp.1,6, Sept. 28 2008-Oct. 2 2008	*		*			*	*
"How does Sustainable Procurement affect You," University of Greenwich, accessed June 9, 2015. http://www.gre.ac.uk/offices/procurement/suppliers/sustainability-matters/how-does-sustainable-procurement-affect-you!		*	*		*	*	*
Hu, Guiping, and Bopaya Bidanda. "Modeling sustainable product lifecycle decision support systems." *International Journal of Production Economics* 122, no. 1 (2009): 366-375.		*	*			*	*
Additional lifecycle from above source			*		*	*	*
International Electrotechnical Commission. *Medical device software: software life cycle processes*. IEC, 2006.	*	*	*	*			

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Iyer, Natraj, Subramaniam Jayanti, Kuiyang Lou, Yagnanarayanan Kalyanaraman, and Karthik Ramani. "Shape-based searching for product lifecycle applications." *Computer-Aided Design* 37, no. 13 (2005):1435-1446.	*	*	*	*		*	
J. Aurich, C. Fuchs, C. Wagenknecht. "Modular design of technical product–service systems," D. Brissaud (Ed.), et al., Innovation in Life Cycle Engineering and Sustainable Development, Springer, The Netherlands (2006), pp. 303–320.							
J. Broekel, G. Scharr, The specialities of fibre-reinforced plastics in terms of product lifecycle management, Journal of Materials Processing Technology, 162/163 (2005), pp. 725–729	*		*			*	*
Jian Cui; Guoning Qi, "Research on Integration Technology for Product Lifecycle Management System," *Intelligent Systems Design and Applications, 2006. ISDA '06. Sixth International Conference on* , vol.2, no., pp.1109,1113, 16-18 Oct. 2006	*	*	*			*	
Jiang, X. Y., X. M. Zhang, S. J. Wang, and W. W. Liu. "Product lifecycle oriented quality management system for supply chain." In *Industrial Engineering and Engineering Management (IE&EM), 2011 IEEE 18Th International Conference on*, pp. 1040-1043. IEEE, 2011.		*	*			*	*
K. Amann, Product lifecycle management: empowering the future of business, CIM Data, Inc. (2002).		*	*		*	*	
Kang, Lifeng, Bong Geun Chung, Robert Langer, and Ali Khademhosseini. "Microfluidics for drug discovery and development: From target selection to product lifecycle management." *Drug discovery today* 13, no. 1 (2008): 1-13.			*				
Kumar, Anand, and Tejas Vithani. "A comprehensive mobile application development and testing lifecycle." In *IT Professional Conference (IT Pro), 2014*, pp. 1-27. IEEE, 2014.		*	*	*		*	
Le Duigou, Julien, Alain Bernard, and Nicolas Perry. "Framework for product lifecycle management integration in small and medium enterprises networks." *Computer-Aided Design and Applications* 8.4 (2011): 531-544.			*			*	*
Lee, S. G., Y-S. Ma, G. L. Thimm, and J. Verstraeten. "Product lifecycle management in aviation maintenance, repair and overhaul."Computers in industry (2008): 296-303.	*	*	*	*	*	*	*

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Lihong Jiang; Boyi Xu; Hongming Cai, "A multi-views modeling approach for product lifecycle management in supply chain," *Computer Supported Cooperative Work in Design (CSCWD), 2012 IEEE 16th International Conference on* , vol., no., pp.533,539, 23-25 May 2012		*	*			*	
Lipner, Steve. "The trustworthy computing security development lifecycle." In Computer Security Applications Conference, 2004. 20th Annual, pp. 2-13. IEEE, 2004.	*	*		*			
M. Abramovici, O.C. Sieg, Status and development trends of product lifecycle management systems, Proceedings of the IPPD 2002 Wroclaw, Wroclaw, Poland (2002)		*	*				*
M. Lindhal, E. Sundin, T. Sakao, Y. Shimomura. "Integrated product and service engineering versus design for environment –a comparison and evaluation of advantages and disadvantages," 14th CIRP Conference on Life and Cycle Engineering (2004), pp. 137-142.	*	*	*	*		*	
Ma Mingxu; Fan Yushun; Yin Chaowan, "Research on product structural tree for product lifecycle," *Industrial Technology, 2005. ICIT 2005. IEEE International Conference on* , vol., no., pp.1147,1152, 14-17 Dec. 2005	*	*	*			*	
Ma, Y.-S.; Song, B.; Yang, Q.Z., "Product Lifecycle Analysis and Optimization in an Eco-value Based, Sustainable and Unified Approach," *Industrial Informatics, 2006 IEEE International Conference on* , vol., no., pp.537,541, 16-18 Aug. 2006	*	*	*	*	*	*	*
Ma, Yongsheng, and Jerry YH Fuh. "Product lifecycle modelling, analysis and management." *Computers in Industry* 59, no. 2 (2008): 107-109.		*	*			*	*
Marchetta, Martín G., Frédérique Mayer, and Raymundo Q. Forradellas. "A reference framework following a proactive approach for Product Lifecycle Management." *Computers in Industry* 62.7 (2011): 672-683.	*	*	*				
Maropoulos, Paul G., and Darek Ceglarek. "Design verification and validation in product lifecycle." *CIRP Annals-Manufacturing Technology* 59, no. 2 (2010): 740-759.		*	*	*			
Additional lifecycle from above source		*	*	*			
Martin, James. *Rapid application development*. Macmillan publishing company, 1991.	*	*	*	*		*	

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
McGroarty, Christopher, Christopher J. Metevier, Scott Gallant, Lana McGlynn, and Joseph S. McDonnell. "How Does an Analyst Select M&S to Support the Entire DOD Acquisition Lifecycle Process?" 20.		*	*			*	
Naka, Y., et al. "Technological information infrastructure for product lifecycle engineering." *Computers & Chemical Engineering* 24.2 (2000): 665-670.		*	*				*
Additional lifecycle from above source		*	*			*	
Additional lifecycle from above source	*		*				*
Additional lifecycle from above source			*		*		*
Ncube, L.B.; Crispo, A.W., "Toward meaningful learning in a global age: How Gestalt principles can facilitate Organization Of student learning of product lifecycle management concepts," *Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports, 2007. FIE '07. 37th Annual* , vol., no., pp.S4D-9,S4D-14, 10-13 Oct. 2007		*	*			*	
Additional lifecycle from above source	*	*	*	*	*	*	*
Nori, K.; Swaminathan, N., "A framework for software product engineering," *Software Engineering Conference, 2006. APSEC 2006. 13th Asia Pacific* , vol., no., pp.285,292, 6-8 Dec. 2006.	*		*		*		
"Operation of the Defense Acquisition System," Department of Defense, January 7, 2015, accessed June 8, 2015. http://www.acq.osd.mil/fo/docs/500002p.pdf.			*			*	*
Park, J.-H., Seo, K.-K., Wallace, D., Lee, K.-I., 2002, Approximate Product Life Cycle Costing Method for the Conceptual Product Design, Annals of the CIRP, 51/1:421-424.		*	*			*	*
Rachuri, Sudarsan, Eswaran Subrahmanian, Abdelaziz Bouras, Steven J. Fenves, Sebti Foufou, and Ram D. Sriram. "Information sharing and exchange in the context of product lifecycle management: Role of standards." *Computer-Aided Design* 40, no. 7 (2008): 789-800.		*	*			*	*
Radack, Shirley. "The System Development Life Cycle (SDLC)." Communications (2002).			*			*	*
Rangan, Ravi M., Steve M. Rohde, Russell Peak, Bipin Chadha, and Plamen Bliznakov. "Streamlining product lifecycle processes."	*	*	*	*		*	*

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Ren Genquan; Zhang Li; Wang Jianmin; Liu Yinbo, "One Method for Provenance Tracking of Product Lifecycle Data in Collaborative Service Environment," *Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), 2011 International Conference on* , vol., no., pp.347,356, 10-12 Oct. 2011						*	*
Roebuck, William. "The Supply Chain, Numbering Systems and Identifiers," May 25, 2011, accessed June 9, 2015. http://eradar.eu/supply-chain-identities/.	*	*	*	*		*	
Royce, Winston W. "Managing the development of large software systems." *Proceedings, IEEE Wescon.* August 1970. 1-9.	*	*	*	*	*		
Ruparelia, Nayan B. "Software development lifecycle models." *ACM SIGSOFT Software Engineering Notes* 35, no. 3 (2010): 8-13.		*	*	*	*		
Sabaragamu Koralalage, K. H. S., S. Mohammed Reza, Junichi Miura, Yuichi Goto, and Jingde Cheng. "POP method: an approach to enhance the security and privacy of RFID systems used in product lifecycle with an anonymous ownership transferring mechanism." In *Proceedings of the 2007 ACM symposium on Applied computing*, pp. 270-275. ACM, 2007.			*		*	*	*
"SE Life-Cycle Building Blocks." MITRE Corportation Systems Engineering Guide, September 2013, accessed June 8, 2015. http://www.mitre.org/publications/systems-engineering-guide/se-life-cycle-building-blocks.	*	*	*	*		*	
Shaojun Qin; Hongming Cai; Lihong Jiang, "A Product Lifecycle Data Management Framework Based on Resource Meta-model," *Services Computing Conference (APSCC), 2012 IEEE Asia-Pacific* , vol., no., pp.305,308, 6-8 Dec. 2012		*	*			*	*
Silcher, S.; Konigsberger, J.; Reimann, P.; Mitschang, B., "Cooperative service registries for the service-based Product Lifecycle Management architecture," *Computer Supported Cooperative Work in Design (CSCWD), 2013 IEEE 17th International Conference on* , vol., no., pp.439,446, 27-29 June 2013		*	*			*	*

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Silcher, S.; Minguez, J.; Scheibler, T.; Mitschang, B., "A service-based approach for next-generation Product Lifecycle Management," *Information Reuse and Integration (IRI), 2010 IEEE International Conference on* , vol., no., pp.219,224, 4-6 Aug. 2010		*	*			*	*
Siller, Héctor R., et al. "Modeling workflow activities for collaborative process planning with product lifecycle management tools." *Journal of Intelligent Manufacturing* 19.6 (2008): 689-700.		*	*			*	
Simpson, Stacy, Ed. "The Software Supply Chain Integrity Framework." SAFECode, July 21, 2009, accessed June 8, 2015. http://www.safecode.org/publication/SAFECode_Supply_Chain0709.pdf			*	*	*		
Singh, Raghu. "International Standard ISO/IEC 12207 software life cycle processes." *Software Process Improvement and Practice* 2, no. 1 (1996): 35-50.			*			*	*
"Software Supply Chain Risk Management & Due-Diligence." Department of Homeland Security, Volume 2, Version 1.2, June 16, 2009, accessed June 8, 2015. https://buildsecurityin.us-cert.gov/sites/default/files/DueDiligenceMWV12_01AM090909.pdf		*	*	*	*	*	
Srinivasan, Vijay. "An integration framework for product lifecycle management." *Computer-aided design* 43, no. 5 (2011): 464-478.			*			*	*
"S-SDLC: Secure Software Development Life Cycle," Project Management Planet, October 14, 2013, accessed June 9, 2015. http://www.projectmanagementplanet.com/ssdlc-the-secure-software-development-life-cycle/.	*	*	*	*	*		
Subrahmanian, Eswaran, Sudarsan Rachuri, Steven J. Fenves, Sebti Foufou, and Ram D. Sriram. "Product lifecycle management support: a challenge in supporting product design and manufacturing in a networked economy." *International Journal of Product Lifecycle Management* 1 (2005): 4-25.		*	*		*	*	*
Sudarsan, Rachuri, Steven J. Fenves, Ram D. Sriram, and Fujun Wang. "A product information modeling framework for product lifecycle management." *Computer-aided design* 37, no. 13 (2005): 1399-1411.		*	*	*			

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Tan, Keah Choon. "A framework of supply chain management literature." *European Journal of Purchasing & Supply Management* 7, no. 1 (2001): 39-48.	*	*	*		*	*	
Tang, Xiaoqing, and Hu Yun. "Data model for quality in product lifecycle." *Computers in industry* 59, no. 2 (2008): 167-179.	*	*	*		*	*	
Terzi, Sergio, Jacopo Cassina, and Hervé Panetto. "Development of a metamodel to foster interoperability along the product lifecycle traceability." *Interoperability of Enterprise Software and Applications*. Springer London, 2006. 1-11.		*	*		*	*	*
Thimm, G., S. G. Lee, and Y-S. Ma. "Towards unified modelling of product life-cycles." Computers in Industry 57, no. 4 (2006): 331-341.	*	*	*			*	*
Tzong-Ming Cheng; Chuang, S., "Developing interactive 3D documentations for product lifecycle," Computers and Industrial Engineering (CIE), 2010 40th International Conference on , vol., no., pp.1,6, 25-28 July 2010		*	*			*	
Additional lifecycle from above source	*	*	*			*	*
U.S. House of Representatives. "Systems Development Life-Cycle Policy." *United States House of Representatives.* March 24, 1999. http://www.house.gov/content/cao/procurement/ref-docs/SDLCPOL.pdf (accessed June 9, 2015).	*	*	*			*	
Vezzetti, Enrico. "Product lifecycle data sharing and visualisation: Web-based approaches." *The International Journal of Advanced Manufacturing Technology* 41.5-6 (2009): 613-630.		*	*	*		*	
Wei He; Lee, I.B.H.; Eng Wan Lee, "An Integrated Design Task Management Approach for Product Development Lifecycle," *Industrial Informatics, 2006 IEEE International Conference on* , vol., no., pp.554,559, 16-18 Aug. 2006	*	*	*		*		
Additional lifecycle from above source		*	*		*		
Wenlei Zhang; Yushun Fan, "A Conceptual Framework for Product Lifecycle Modeling," *Innovative Computing, Information and Control, 2006. ICICIC '06. First International Conference on* , vol.2, no., pp.486,489, Aug. 30 2006-Sept. 1 2006	*	*	*			*	

	Requirements	Design	Man. / Dev.	Testing	Distribution	Use/Maint.	Disposal
Westkämper, Engelbert. "Life cycle management and assessment: approaches and visions towards sustainable manufacturing (keynote paper)." CIRP Annals-Manufacturing Technology 49, no. 2 (2000): 501-526.	*	*	*			*	*
Additional lifecycle from above source		*	*			*	*
Additional lifecycle from above source			*			*	*
Westkämper, E., 2002, Platform for the Integration of Assembly, Disassembly and Life Cycle Management, Annals of the CIRP, 51/1:33-36.			*			*	*
Wiesner, Stefan, Mike Freitag, Ingo Westphal, and Klaus-Dieter Thoben. "Interactions between Service and Product Lifecycle Management." *Procedia CIRP* 30 (2015): 36-41.	*	*	*			*	*
Xu, X., J. L. Q. Chen, and S. Q. Xie. "Framework of a product lifecycle costing system." *Journal of Computing and Information Science in Engineering* 6, no. 1 (2006): 69.		*	*		*	*	*
Yang, Q.; Song, B., "Eco-Design for Product Lifecycle Sustainability," *Industrial Informatics, 2006 IEEE International Conference on* , vol., no., pp.548,553, 16-18 Aug. 2006	*	*	*		*	*	*

DISTRIBUTION

2 Evan Wood
3869 Frist Campus Center
Princeton University
Princeton, NJ 08544

2 Natalie Roe
Brown University
69 Brown Street, Box 5867
Providence, RI 02912

2 Sunny He
3047 Frist Campus Center
Princeton University
Princeton, NJ 08544

[List external recipient names and addresses]

4 Lawrence Livermore National Laboratory
Attn: N. Dunipace (1)
P.O. Box 808, MS L-795
Livermore, CA 94551-0808

[List in order of lower to higher Mail Stop numbers.]

1	MS9159	Noel Nachtigal	Org. 8966
1	MS9406	Jovana Helms	Org. 8116

[The housekeeping entries are required for all SAND reports.]

1	MS0899	Technical Library	9536 (electronic copy)

For LDRD reports, add:

1	MS0359	D. Chavez, LDRD Office	1911

For CRADA reports add:

1	MS0115	OFA/NFE Agreements	10012

For Patent Caution reports, add:

1	MS0161	Legal Technology Transfer Center	11500


Sandia National Laboratories