RT 193: Framework for Analyzing Versioning and Technical Debt

December 21, 2018

Principal Investigator: Dr. Ye Yang, Stevens Institute of Technology

Co-Principal Investigator: Dr. Jon Wade, Stevens Institute of Technology

Research Team:
Turki Alelyani, Stevens Institute of Technology
Patrick Stanton, Stevens Institute of Technology

Sponsor: RDECOM CERDEC
Copyright © 2018 Stevens Institute of Technology, Systems Engineering Research Center

The Systems Engineering Research Center (SERC) is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Office of the Assistant Secretary of Defense for Research and Engineering (ASD(R&E)) under Contract HQ0034-13-D-0004, Task Order 0014.

Any views, opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense nor ASD(R&E).

No Warranty.
This Stevens Institute of Technology and Systems Engineering Research Center Material is furnished on an “as-is” basis. Stevens Institute of Technology makes no warranties of any kind, either expressed or implied, as to any matter including, but not limited to, warranty of fitness for purpose or merchantability, exclusivity, or results obtained from use of the material. Stevens Institute of Technology does not make any warranty of any kind with respect to freedom from patent, trademark, or copyright infringement.

This material has been approved for public release and unlimited distribution.
# TABLE OF CONTENTS

Table of Contents .................................................................................................................. iii
List of Figures .......................................................................................................................... iv
List of (Tables, Sequences)...................................................................................................... iv
Executive Summary ................................................................................................................ 1

1. Introduction .......................................................................................................................... 2

2. Background and Definitions ............................................................................................... 4
   2.1 Background ....................................................................................................................... 4
   2.2 Definition of COTS ........................................................................................................... 5
   2.3 Definition of Obsolescence Management ......................................................................... 6
   2.4 Definition of Technical Debt ........................................................................................ 6
   2.5 Motivation ....................................................................................................................... 7

3. Literature Review on COTS Obsolescence Management ..................................................... 8
   3.1 Study Design .................................................................................................................. 8
   3.2 Review Questions .......................................................................................................... 9
   3.3 Results .......................................................................................................................... 10
       3.3.1 RQ1: What are the current MPTs relevant to obsolescence management? ............. 10
       3.3.2 RQ2: What data is available/has been used in analyzing COTS obsolescence cost/risk? . 13
       3.3.3 RQ3: What are different COTS categories? ............................................................... 13
       3.3.4 RQ4: What are the metrics for COTS TD? ................................................................. 15
       3.3.5 RQ5: What are existing strategies for OM&TD? ...................................................... 16

4. DECISION FRAMEWORK FOR MAPPING EXISTING OBSOLESCENCE MANAGEMENT
   APPROACHES .................................................................................................................... 19
   4.1 Design Items Decision Making ..................................................................................... 19
   4.2 Tools and Database Decision Making .......................................................................... 20
   4.3 Design Refresh Candidate Decision Making ............................................................... 20
   4.4 Design Modification Decision Making ......................................................................... 21
   4.5 Gap: COTS-Intensive Technical Debt Identification .................................................. 21

5. Need for the Notion of COTS Technical Debts ................................................................... 22
   5.1 Analogy of COTS Technical Debt in CPS Systems Engineering .................................. 22
   5.2 Existing Taxonomy of Software Technical Debt .......................................................... 23
       Rubin’s Taxonomy for Agile Development ........................................................................ 23
       Clark’s Taxonomy for Game Development .................................................................... 23
       Kruchten et al.’s Technical Debt Landscape .................................................................... 24

6. Taxonomy on COTS Technical Debts in CPS Systems ....................................................... 26

7. Future Work - Towards Effective COTS Technical Debt Management (TDM) for CPS
   Systems .................................................................................................................................. 31

8. Conclusions ........................................................................................................................ 33

List of Publications Resulted .................................................................................................. 34
Appendix A: Summary of Included Existing Studies in The Literature Review .............. 35
Appendix B: Cited and Related References ........................................................................... 39

LIST OF FIGURES

Figure 1 Example Figure ................................................................. Error! Bookmark not defined.

LIST OF (TABLES, SEQUENCES)

Table 1 Example Table ................................................................. Error! Bookmark not defined.

Formulation 1 Formulation Example (w/o sequence numbers) Error! Bookmark not defined.

Formulation 1 SoS Architect Optimization Problem (SoS-A) Error! Bookmark not defined.

Algorithm 1 Example If Step List is a References Algorithm Error! Bookmark not defined.
EXECUTIVE SUMMARY

Many Cyber-Physical Systems (CPSs) require arrays of commercial-off-the-shelf (COTS) components, both hardware and software. COTS components have their independent, rapid upgrading cycles. The frequent upgrading of COTS components is one of the root causes for many obsolescence headaches in long-lived CPS from domains such as space, defense, and avionics systems. Most existing studies addressing COTS obsolescence issues have strong emphasis on the sustainment phases. We have identified a gap on methods, processes, and tools for effective COTS risk analysis in the early systems acquisition phases. To fill the gap, this study proposes a taxonomy of COTS-related technical debts in order to support early identification, communication, and assessment of obsolescence risks in CPS system engineering life cycles. It starts with a literature review and synthesize on existing OM work, and contributes to the identification of seven key types of COTS technical debt according to systematic signs discoverable during early COTS activities, which may contribute to obsolescence in later phases. These seven types of COTS technical debt include COTS functionality mismatch, performance mismatch, interoperability difficulty, versioning frequency, documentation and support readiness, and limitation on system evolution. It is expected that such notions will help to increase the efficiency of COTS-based CPS development, readiness, and sustainment, through more informed COTS decision-making and readily-informed and focused-visibility of associated cost-centric obsolescence issues in expedited systems acquisition phases to avoid expensive and unaffordable obsolescence issues in the envisioned systems sustainment phases.
1. INTRODUCTION

Large scale, complex systems typically rely on tightly conjoining sets of computational processes and physical processes, which are commonly referred to as Cyber-physical systems (CPS) [1, 2]. These systems such as aircrafts and ships take many years to design and are typically maintained for decades. Starting from the 1990s, increasing trends of COTS-based systems in commercial domains have also been reported [3, 10], providing cost-effective alternatives to traditional custom development approaches. Consequently, there has been many recommendations [3, 4, 5] and COTS-related policies, e.g. Perry Memo [7, 11] and Federal Acquisition Streamlining Act [7], to encourage and facilitate the expanding usage of COTS. It was a common belief at the time that COTS-based solutions can positively contribute to accelerating military transformation, “using proven commercial technology, allowing for substantial cost savings, with fewer schedule delays, and improved performance characteristics, across numerous programs, DoD-wide” [6].

In many cases over the past few decades, it is a simple fact that “the proven use of COTS products speeds and lowers the cost of military system development” [10] and witnessed by many reported successful case studies [6]. The main driving factor is the advent of the information revolution, the maturity of commercial marketplace, and the directions and rate of innovative technologies in COTS solutions. As a recent example, the F-35 Lightning II Joint Strike Fighter aircraft are using COTS avionics wherever and whenever possible throughout the advanced fighter’s cockpit [9].

Meanwhile, COTS is never the silver bullet that can be plug-n-played in any CPS systems. The adoption of COTS remains complex and faces many challenges due to many obstacles such as mismatch between defense acquisition system and unique COTS integration activities, system complexity and uncertainty, COTS technology complexity, COTS evolution uncertainty, etc. Gansler and Lucyshyn pointed out several challenges including loss of control over COTS vendor and/or COTS evolution, lack of quality in COTS hardware to meet military environment requirements, potential integration and security concerns due to lack of visibility into internal code details in COTS software, additional COTS cost incurred e.g. developmental test and evaluation, loss of flexibility due to user locked into a proprietary technology, etc. [6]. Clark and Clark [12] interviewed 25 project managers on reasons why COTS-intensive systems are so difficult to maintain, and concluded 11 factors having impacts on the true cost of COTS maintenance, including licensing, evaluation of new releases, defect hunting, vendor support, and so on. Such costing sources, if unaware and unplanned, turned out to be triggers for challenging development or maintenance challenges.

COTS challenges if not planned and managed appropriately in acquisition phase, could lead to COTS-intensive solutions that are obsolete, out of vendor support, or even system discontinuation. Given the DoD’s increased push for cost efficiency, reliability, and portability, the increasing role of COTS components in the future of military operations is inevitable. In order to achieve the expected economic benefits of COTS procurement, it is critical to enable program managers to better understand, analyze, and cope with obsolescence cost so that they can make
informed COTS commitment decisions. A wealth of obsolescence management (OM) studies has been reported to develop new processes, models, and tools for better planning for and coping with COTS obsolescence in military CPS systems over the past two decades [3-5, 13-17]. However, the majority of the studies are focusing on predicting, planning, mitigating obsolescence cost and risk during the later maintenance or sustainment phases, and very few shed light on investigating technical, process, people, culture factors which, if overlooked, may lead to pre-mature COTS commitment in the acquisition time and consequently inoperable, unmaintainable, unaffordable, or unsustainable CPS systems later on.

The metaphor of technical debt stemmed from the field of software engineering, first introduced two decades ago by Ward Cunningham to explain to nontechnical business stakeholders the need for what we call now “refactoring” [18]. More broadly, it refers to delayed tasks and immature artifacts that constitute a “debt” because they incur extra costs in the future in the form of increased cost of change during evolution and maintenance [19]. The basic idea is straightforward: quicker built, simpler solutions will be faster to market but often require more maintenance over their useful life. In COTS-intensive CPS context, engineering decisions on acquiring quick COTS solutions would incur more “debt” to the user/customer organizations, which will need to be paid off in later maintenance or sustainment phases in the form of continuous upgrades. As Software Engineering has grown so has the application of this analogy to a wide range of software engineering activities and artifacts, as summarized in the technical debt landscape in [18]. It is our belief that as Systems Engineering grows, esp. for COTS-based CPS systems, the need for a COTS-related technical debt analog grows to facilitate the reevaluation of defense acquisition systems which can better identify and mitigate COTS related risks esp. obsolescence issues in designing CPS systems.

The research project reported in this study aims at adopting the notion of technical debt in studying obsolescence problems in CPS system engineering life cycles, and proposes a taxonomy of COTS-related technical debts in order to support early identification, communication, and assessment of obsolescence risks. It starts with a literature review and synthesize on existing OM work, and concludes 6 types of COTS technical debts according to systematic signs and effects discoverable during early COTS activities (e.g. identification, assessment, tailoring, etc.) that may contribute to obsolescence in later life cycles (e.g. maintenance, operation, etc.). The 5 technical debts include COTS functionality mismatch, non-functional mismatch, interoperability difficulty, versioning frequency, documentation & support readiness, and limitation on system evolution. It is expected that such notion of COTS technical debts will help to increase the efficiency of COTS-based CPS development and sustainment, through more informed COTS decision making in the acquisition phases to avoid expensive and unaffordable obsolescence issues in the sustainment phases.

The rest of the paper is organized as follows. Section 2 discusses the background and some working definitions. Section 3 reports the literature review on OM studies for COTS-intensive systems. Section 4 presents a process decision framework for mapping existing OM work, with the identification of the gap of COTS-related technical debt. Section 5 proposes an initial
taxonomy of COTS technical debt for CPS systems. Section 6 discusses future directions. Section 7 is the conclusion.

2. BACKGROUND AND DEFINITIONS

2.1 BACKGROUND

Many DoD missions require arrays of COTS components, both hardware and software, that were not designed to be in an array. The frequent upgrading of COTS components (e.g. 18 months) is one of the root causes for many obsolescence headaches for systems whose life times are typically more than 30 years. For sustainment-dominated CPS systems, their lifespans (e.g. over 20 or more years) are long enough that a significant portion of the COTS components becomes obsolete prior to the system being deployed. Figure 1 illustrates a case where over 70% of the COTS electronics become un-procurable by fielding time [22]. As another example, Future Combat System had 153 relevant systems to deal with. If every one updated once a year, that would be a change every other day! In order to achieve the expected economic benefits of COTS procurement, it is critical to enable program managers to better understand obsolescence cost before making informed COTS commitment decisions.

![Fig. 1. Portion of the obsolete COTS parts in the first 10 years of a surface ship sonar system's life cycle [22].](image)

COTS components are increasingly imposing long-term management issues such as obsolescence, poor reliability, lack of readiness, and inability to be readily maintaining systems in an efficient and effective manner. The main challenge is the lack of a common metric and measurement framework to serve as basis for understanding, communicating, analyzing and predicting the life cycle consequences incurred by COTS obsolescence issues. The existing landscape of obsolescence management research is dominated by a focus on electrical components, namely in avionics and other complex military systems. The focus of technical debt in academia is on software. Not having a view on both obsolescence risks and technical debts will cause system operators to make short-sighted decision that drive up the life cycle cost of a system.
However, understanding both fields of research is not enough. As COTS components begin to dominate COTS-intensive CPS, the management of these risks needs to be addressed in a dynamic, iterative fashion. This is due to the short-life expectancy of COTS components. As manufacturers evolve their system to keep up with customer demands system’s integrators and operators are forced to repeatedly choose: upgrade the component now and deal with the integration issues or stay with the existing component and risk future obsolescence that may compromise functionality. For this reason, many evaluation tools and methods have been created for choosing between COTS components [20, 21]. These focus on the different phases of life for the component, the complexity of the system and the criticality of the component. The main concern is on the impacts that these components will have on the quality, schedule, and life-cycle cost of the system. The purpose of this work is to align these existing strategies into a high-level framework that allows the different tools, methods, and processes to be used in parallel. A properly built framework will maximize the utilization of the system integrator’s time and enable predictive component management.

2.2 Definition of COTS

In the context of the U.S. government, the Federal Acquisition Regulation (FAR) has defined that a "COTS item": 1) Means any item of supply (including construction material) that is a commercial item, sold in substantial quantities in the commercial marketplace; and offered to the Government, under a contract or subcontract at any tier, without modification, in the same form in which it is sold in the commercial marketplace; and 2) Does not include bulk cargo, such as agricultural products and petroleum products [22].

Under this definition, a formal term for commercial items, including services, available in the commercial marketplace that can be considered as COTS. For the purpose of this study, we further introduce the working definitions for COTS hardware and COTS software, respectively. They share some commonalities, and with different characteristics in life-cycle impacts. The definition for COTS software in the COCOTS model [17] is adapted here: 1) A COTS component is sold, leased, or licensed for a fee (which includes vendor support in fixing defects if they are found); 2) The source code is unavailable; 3) The component evolves over time as the vendor provides periodic releases of the product (upgrades) containing fixes and new or enhanced functionality; 4) Any given version of a COTS component will reach eventual obsolescence or end of life in which it will no longer be supported by the vendor. All four parts of this definition have major implications for costs and obsolescence risks during maintenance. The definition for COTS hardware is generally referring to other types of COTS such as electronic parts, mechanical/physical components, or materials.

When dealing with large scale systems – and systems of systems – COTS components are always going to be incorporated in the design. This takes design work, and control, out of the hands of the system integrator in exchange for reduced cost. Furthermore, it simply is not necessary for engineers to design every aspect of a system when there are existing solutions identified as best in industry. Although this offers many benefits COTS components have significantly shorter lifespans than custom solutions, leave the design engineers without a complete understanding of the components in the system, and leave them uniquely at risk for
technical debt incurrence. A goal of the research that this report has been done in tandem with is to characterize this COTS component technical debt risk for cyber-physical systems.

Key attributes that can characterize COTS technical debt are multiplicity, complexity, and interdependence at the system level. Multiplicity refers to the number of different COTS in a system; the more COTS the more risk the system is at of requiring early and frequent updates. Similarly, the more complex the system, or the COTS components that are integrated into it, the higher the risk the system is at of containing obsolete parts that require update. The way that these COTS components are integrated into the system and how the data or functionality flows through the system also offers a certain level of technical debt risk to the system. The more complex the interfaces, or interdependencies, between components the more difficult it is to upgrade a system and have it function in accordance with the system requirements. Together these three major metrics make a working definition of COTS technical debt.

2.3 DEFINITION OF OBSOLESCENCE MANAGEMENT

Obsolescence is often used interchangeably with “diminishing manufacturing sources and material shortages” (DMSMS) [16]. For this reason, the management of both obsolescence and DMSMS are addressed interchangeably by industry, the military, and academia. From these studies the definition of obsolescence management can be viewed as planning how to address the loss of materials, manufacturers, human skill, technology support, or maintenance tools that are integral to the functionality of the system. When something goes obsolete, and is no longer able to function as part of the system without incurring risk, an obsolescence risk mitigation management strategy must be executed. It is the finding of this report that it is optimal for these strategies to be developed and executed by leadership in the early stages of the CPS design. This is in line with “frontloaded development” in engineering design for new product development.

2.4 DEFINITION OF TECHNICAL DEBT

The term ‘technical debt’ refers to delayed tasks and immature artifacts that constitute a “debt” because they incur extra costs in the future in the form of increased cost of “change” during evolution, maintenance, and sustainment [19].

Often, regardless of the size or quality of the initial roll out, a system will require upgrade. A more thoroughly designed system may incur less debt, but may also lack speed to market. The more “homework” completed upfront that focuses on obsolescence risk-mitigating designs, the less obsolescence management activity and less unforeseen cost exposure will likely take place over the life of the CPS. This study treats obsolescence as a form of the consequences of technical debt being incurred. Other forms of technical debt include, but are not limited to, the lack of testability, inflexibility of design, insufficient documentation, and unmaintainability of code. These shortfalls each enable a system to be rolled out early, but may be accompanied with compromised reliability, availability, maintainability, with unaffordable costs to future stakeholders.
2.5 Motivation

The existing landscape of obsolescence management research is dominated by a focus on electrical components, namely the piece-parts within platforms and other complex military systems. The present focus of technical debt in academia is on software. There is also widespread awareness that the use of overly restrictive acquisition requirements with practices and perspectives lacking informed evidence-based views, at new program starts and onward throughout the envisioned systems life cycle, will enable unsubstantiated short-sighted decisions that drive negative outcomes over the life cycle of a system. Hence, there is a compelling need for the consistent advantages associated with an efficient and effective decision-making system that provides data points that readily identify associated risks and which facilitate evidence-based decision making and accountability.

However, understanding both fields of research, and the coupling of systems obsolescence risks to technical debt, is not enough. As COTS components begin to dominate COTS-based CPS, the management of the associated risks needs to be addressed in a dynamic, iterative fashion that concurrently considers these multiple perspectives, and synthesizes them into value-added, insightful output for use by CPS systems design, development, procuring, and sustaining organizations in decision-making associated with new technology and systems introductions and throughout their planned life spans. The velocity of this need is also driven by the short-life expectancy of COTS components, and the compelling need to minimize the lead times of the introductions of new technologies and CPS, particularly within the DOD. As manufacturers evolve their system to keep up with customer demands, system’s integrators and operators are often forced to repeatedly choose between two strategic directions: upgrade the component now and deal with the integration issues, or stay with the existing component and risk future obsolescence, at some unknown or unconsidered level, that may compromise functionality, reliability, availability, maintainability, and/or cost.

In summary, this work has the purposes to leverage the metaphor of “Technical debt”, and developing a counter part concept of “Technical Debt” at the systems-level which categorizes typical patterns of COTS hardware and software technical debts together. The work aims to use these concepts specifically for early TD risk assessment, but also to support the life cycles of of COTS-oriented CPS for the management of the associated risks in a dynamic, iterative fashion.
3. LITERATURE REVIEW ON COTS OBsolescence MANAGEMENT

In this study, we aim to get a comprehensive overview on obsolescence management in COTS-based systems. Our extensive review allows all related studies to be analyzed and identify the research gaps in order to direct future research to address specific challenges in the context of COTS in CPS. In the present study, we identify obsolescence process, methods, and tools that have been investigated in the literature. We also aim to report different types of COTS categories and the strategies that have been used to manage its obsolete components.

The main contribution of this literature review consists of two parts: First, we do an extensive review of the stat of art related to obsolescence in COTS-based systems. We report the current approaches and how they have been applied to manage obsolescence. In addition, we show the current gabs and the future research direction. Second, we propose a novel framework that synthesizes the existing literature to contribute in management obsolescence in CPS. Our framework can be introduced for management level to better formulate a decision when addressing obsolescence.

3.1 STUDY DESIGN

We apply systematic approach to review the relevant obsolescence studies in the context of COTS-based CPS systems. These include hardware, software, and electronics. Our literature review process based on well-established method in software engineering by Kitchenham [23], to systematically identify, evaluate, and interpret all relevant articles to our topic. The predefined search strategy will also allow our research to be evaluated. Our search strategy process includes the following steps:

- Defining a searching term or query.
- Defining the target for the searching term.
- Selecting different data sources to identify candidate articles.

Our search term or query contains some specific keywords we deemed the most relative to our target. Keywords were combined using the Boolean “AND” operator, which entails that reviewed articles should include both of the terms. The character * will be used as a wildcard to match one or more inflected form of the searching term. The final search keywords are:

(“Technical debt” OR “Obsolescence”) AND

(“COTS” OR “NDI” OR “GOTS” OR “Component*)”) AND

(“cyber physical system” OR “military systems” OR (“embedded systems”)

We retrieved studies from several databases including ACM Digital Library, IEEExplore, ScienceDirect, SpringerLink, Scopus, Web of Science, and google search engine. Beside the first five academic databases, the google search engine is included because we expect there are some
industry-specific articles or presentations which might not be included in any of the academic repository. This helps to identify DMSMS and other online articles.

The inclusion and exclusion criteria will be based on the following:

- Only include publications that define or discuss COTS and obsolescence issues in Cyber Physical Systems (CPS) context.
- Only publications written in English are included.
- Publications where the full text can’t be located are excluded.
- Publications earlier than 1980 are excluded.

This results a list of 56 literatures in our final selection.

![Summary of Literature Review Process](image)

### 3.2 Review Questions

In order to better understand both the state of art and state of practice in COTS obsolescence management in CPS systems, we formulate five review questions which guide the data extraction from the selected studies from the above step.

**RQ1:** What are the current methods, processes, and tools (MPTs) relevant to obsolescence management?

**RQ2:** What data is available/has been used in analyzing COTS obsolescence cost/risk?
RQ3: What are different COTS categories?

RQ4: What are the metrics for COTS TD?

RQ5: What are existing strategies for OM&TD?

3.3 RESULTS

This section presents the findings from the analysis of the 56 studies we reviewed regarding COTS obsolescence in CPS systems.

3.3.1 RQ1: WHAT ARE THE CURRENT MPTS RELEVANT TO OBSOLESCENCE MANAGEMENT?

To answer RQ1, we further define four attributes to characterize the selected studies, in terms of type of new ideas proposed in the study, type of the main author’s affiliation, phases that obsolescence occurred/addressed, and granularity of the obsolescence issues discussed. Table 1 summarizes the attributes and their detailed description to guide review data collection.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPTs</td>
<td>Four input options for recording the main contribution of the study when it is proposing either:</td>
</tr>
<tr>
<td></td>
<td>- A method: if the paper introduces a new method</td>
</tr>
<tr>
<td></td>
<td>- A processes: if the paper introduces a new process</td>
</tr>
<tr>
<td></td>
<td>- A tool: if the paper introduces a new tool, or</td>
</tr>
<tr>
<td></td>
<td>- Others: position papers, reviews, case studies, etc.</td>
</tr>
<tr>
<td>Sector</td>
<td>Four input options to classify the authors’ affiliation types:</td>
</tr>
<tr>
<td></td>
<td>- Academia</td>
</tr>
<tr>
<td></td>
<td>- Government</td>
</tr>
<tr>
<td></td>
<td>- Industry</td>
</tr>
<tr>
<td></td>
<td>- Others</td>
</tr>
<tr>
<td>LC Phase</td>
<td>Five DoD acquisition systems compatible phases to map the focus or applicability of the study results:</td>
</tr>
<tr>
<td></td>
<td>- Materiel solution analysis (i.e. Milestone A)</td>
</tr>
<tr>
<td></td>
<td>- Technology maturation and risk reduction (i.e. Milestone B)</td>
</tr>
<tr>
<td></td>
<td>- Technology maturation and risk reduction (i.e. Milestone C)</td>
</tr>
<tr>
<td></td>
<td>- Production and deployment (i.e. Initial Operational Capability, IOC)</td>
</tr>
<tr>
<td></td>
<td>- Operations and support</td>
</tr>
<tr>
<td>Granularity</td>
<td>Two input options for capturing to what level the study is addressing the obsolescence issue?</td>
</tr>
<tr>
<td></td>
<td>- System level</td>
</tr>
<tr>
<td></td>
<td>- Component</td>
</tr>
</tbody>
</table>
According to the attributes’ definition in Table 1, we extract the corresponding attribute data from each of the 56 studies. Appendix A summarizes the extracted attribute data.

Figure 3 shows the number of studies in the identified dataset of 56 related studies since 1995. The peak period of 2005-2009, corresponding to 20 studies published during this time frame. In order to understand the context of studies in terms of their levels of applicability, we examined the reviewed literature in two aspects: System level and Component level. In the system level, authors tend to look at the problem from a system level and results implication can be generalized to the system as a whole, where in component level, authors were only interested to study specific components within the systems. Though there is a seemingly decreasing trend in total number of studies over the years, however, there is a significant shift of study focus from component-level issues to the system level issues. The implication is that more comprehensive strategies for both hardware and software are needed to overcome some of the challenges obsolescence brings to CPS systems [7,14].

![Fig. 3. Distributions of the reviewed studies](image)

Figure 4 shows the paper type distribution across methods, processes, tools, and others. The results show that 46% of studies were about proposing new methods, 22% were about proposing new processes, where tools receive 12% suggesting that the existing studies lack tools that can address obsolescence problems. Table 2 lists some of the reviewed MPTs [3,6, 7, 8, 10, 22, 26, 33].

Several methods have been reviewed which help in enabling the sustainability of any potential component or system that may go obsolete. For instance, design refresh [66] schedule predicts the design refresh content for each of the scheduled design refreshes. Open Source Software Products as a process to mitigate obsolescence effect means that the COTS vendor authorizes access to the source code, so it enables more customized interfaces to the product. Finally, we reviewed several tools and found that most of the existing tools only deal with hardware and electronics components rather than software obsolescence issues.
Several studies have introduced management tools/techniques to cope with obsolescence in cyber physical systems [39, 40]. For instance, COCOTS is a model/tool for estimating cost associated with COTS evaluation, tailoring, and integration [17, 34]. It contains three sub-models to estimate the COTS selection, tailoring, and integration effort for COTS-intensive application development. In estimating obsolescence cost, these sub-models can be adapted to predict necessary re-evaluation, re-tailoring, and re-qualification costs with respect to every changes between the version in usage and the new released version. It also has a risk analyzer component to obtain a COTS glue code integration risk analysis with no user inputs other than the set of glue code cost drivers that should be submitted to get a glue code integration effort estimate.

**Table 2. List of example methods, processes, tools identified in the review**

<table>
<thead>
<tr>
<th>Category</th>
<th>Existing Work</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Methods</strong></td>
<td>Design Refresh; Life Time Buy; Last Time Buy; Substitution; Forecasting Model; VHDL-Based Model; Design Longevity Agreements via Life-of-Need (LON) buys [30]</td>
</tr>
<tr>
<td><strong>Processes</strong></td>
<td>Open source software products; Software Application programming Interfaces (API) and wrappers; After-market Supplier; Emulation/Cloning; Software Obsolescence Trigger Map</td>
</tr>
<tr>
<td><strong>Tools</strong></td>
<td>COCOTS tool for estimating cost associated with COTS evaluation, tailoring, and integration [17, 34]; MOCA (mitigation of obsolescence cost analysis) tool [27]; Total Obsolescence Management Capability Assessment Tool (TOMCAT) [38]; Component Information Management System [39];</td>
</tr>
</tbody>
</table>

When considering the distribution of studies based on the type of venue, we found that 26 studies published in conference proceedings, 26 published in journals, and another 3 theses. Furthermore, we also classified the reviewed studies according to authors affiliation's type and found that 40% Academia, 38% industry, while government gets 22%. This distribution suggests that there are strong interests for all three sectors in studying obsolescence.
3.3.2 RQ2: **What data is available/has been used in analyzing COTS obsolescence cost/risk?**

For RQ2, we use the following five input options to record the data in analyzing COTS obsolescence cost/risk in the corresponding studies:

- Technology forecasting
- Business trending (demand forecasting)
- Obsolescence data
- Logistics data
- Others

We examined existing studies in which different types of data has been used in analyzing obsolescence in COTS-based systems, and the results is shown in Figure 5. These data include Technology Forecasting, Business Trending, Obsolescence, and logistics data.

![Fig. 5. Distribution of analysis data](image)

Traditional methods of life cycle forecasting for managing obsolescence utilized available data that can be used to develop models to assist in quantifying the risks and dates. For instance, some forecasting that was used in commercially available databases are based on the use of ordinal scales that determine the life cycle stage of the part or component [13]. Existing tools such as TACTRAC, Total Parts Plus, and Q-Star use historical parts records data and sales information in the forecasting process. For example, Bill of Materials (BOM) used in the strategic planning and forecasting where a product’s BOMs are analyzed and each part scored as a way to identify any potential risk [24].

3.3.3 RQ3: **What are different COTS categories?**

For RQ3, we employ 6 categories to capture different COTS components in CPS engineering processes:
• Electronic components/Mechanical components
• s/w and media support tooling
• test equipment
• documentation
• skills/personnel/training
• Others

As Figure 6 illustrates, studies on COTS Software and media support constitute the largest portion with 22 studies. COTS electronics corresponds to 15 studies. The remaining categories receive between 1 to 6 studies. The results suggest that there are intensive obsolescence issues associated with software and electronics COTS.

![Fig. 6. COTS category](image)

Software obsolescence research has been undertaken to develop different strategies to mitigate obsolescence issue. Software becomes obsolete when there is a new technological advancement; the software functionality is not required or other market factors [2]. In addition, prior work identified three causes for software to go obsolete: (1) Functional obsolescence; (2) Technological Obsolescence; (3) Logistical Obsolescence [14]. Functional obsolescence occurs when there are changes to the hardware or software in the same system. Technological obsolescence occurs where vendor is no longer supporting the product. Finally, logistical obsolescence occurs when the media or hard drive, for example, doesn’t support the new software version anymore.

For COTS hardware such as electronics/mechanical parts, Baker has applied emulation as an electronic obsolescence technique in order to replace the original obsolete circuit with another design which can result in a form fit and function replacement [44]. Another study explored the loss of original manufacturers or the end of production for hardware/electronic component [31]. Simply replacing obsolete components with newer component is not always optimal.
Humans can be one of the major factor in producing technical debt. This can be explained in legacy systems as sometimes the loss of critical human skills is a problem for support organizations as they try to understand and mitigate the effects of an aging workforce with highly specialized low-demand skill sets. To cope with this problem, Sandborn and Prabhakar [25] developed a model for forecasting the loss of critical human skills and the impact of that loss on the future cost of system support.

3.3.4 RQ4: WHAT ARE THE METRICS FOR COTS TD?

In order to effectively manage quality and risks in CPS, COTS metrics can aid decision makers in managing obsolescence. One way to be a proactive in managing Obsolescence is being able to forecast it and predict it. This would improve the response time which can mitigate its effect when occurs. Sandborn et al. identified two types of forecasting: long-term and short term [23]. Long-term forecasts that are used when obsolescence is one year or further into the future to enable pro-active management of obsolescence events and strategic life cycle planning for the sustainment of systems. Short-term forecasting observes the supply chain for precursors to a part’s obsolescence.

Prior conducting our literature review, we have identified 7 groups of metrics that can likely contribute to or measure degree of COTS obsolescence:

- Multiplicity (e.g. # of COTSs, # of components, etc.)
- Complexity (e.g. system complexity, application complexity, Requalification complexity, etc.)
- Interdependency (e.g. Coupling level and package density, etc.)
- Platform diversity
- PBS: product breakdown structure
- OM strategy
- Financial Metrics (e.g. RO, NPV, etc.).

Figure 7 shows each metric and its distribution across years. The most intensively studied metrics for COTS-related decision analysis and making are complexity (i.e. 23), interdependency (i.e. 20), and obsolescence management strategy (i.e. 17). While metrics related to product breakdown structure (PBS), Platform diversity (PD) and multiplicity are less studied. Multiplicity metric quantifies the number of different COTS components at system level, which are frequently overlooked in component level studies. One of the challenge in CPS is purchasing sufficient parts to meet current and future demands [1]. This can be demonstrated by complexity of multiple concurrent buys as in lifetime buy cost there are different financial costs that should be
considered including procurement cost, inventory cost, disposition cost and penalty cost. Complexity measures assess different level of integration or parts criticality. For example, when forecasting obsolescence one factor should be considered which is complexity of the component (e.g. low complexity such as resistors or high complexity such as microprocessors or LCD displays [7]. In COTS assessment and quality evaluation, complexity metric quantifies component interface and middleware or integration code complexity. Middleware represents a combo of hardware and software which acts as interface between two disparate systems/platforms. The level of complexity increases when the level of interaction increases to maintain barriers between different platforms. Financial metrics works as an approach to reduce the impact of obsolescence by proposing a central database which includes information about suppliers, customers and data warehousing.

![Fig. 7. COTS metrics distribution](image)

**3.3.5 RQ5: WHAT ARE EXISTING STRATEGIES FOR OM&TD?**

Previous studies on obsolescence management can be mapped to three different categories: reactive, proactive and strategic [15, 25, 26]. Reactive approaches are used to determine the immediate resolution to the problem of an obsolete part, executes the resolution process, and tracks the actions taken. Pro-active approaches provide the ability to forecast the obsolescence risk for parts. Strategic approaches aim at using obsolescence data, logistics data, technology forecasting, and business trending (demand forecasting) to enable strategic planning, life-cycle optimization, and long-term business case development for system support. Pro-active management of obsolescence requires the identification of critical parts that: a) are at risk of becoming obsolete, b) will have an insufficient quantity available after obsolescence to satisfy expected demand, and c) will represent a problem to manage if/when they become obsolete. Once critical parts are identified they are managed prior to their actual obsolescence event. Strategic techniques may employ three different strategies: Obsolescence Forecasting, Obsolescence Mitigation, and Strategic Planning.

For RQ5, we focus on the following 5 types of OM strategies, adopted from [15, 25-26]:

- **Strategic:**

---

Report No. SERC-2018-TR-120  
December 21, 2018
• Supply-chain: life-time buy and partnering agreement

Pro-active:

• Design: open system architecture, modularity, use of multi-sourced components
• Planning: obsolescence mgmt. plan, technology roadmap, monitoring tools

Reactive

• Form, fit & function(FFF) replacement (e.g. equivalent-component)
• Emulation or redesign (e.g. use of state-of-art technology to replicate or redesign the component)

Figure 8 shows that 23 studies were in the planning phase, design phase comes after with 19 studies. The rest of the phases are less studied.

In the planning phase which received the most number of studies, several approaches have been used to manage obsolescence. For instance, Material Risk Index (MRI) approaches analyze a product’s bill of materials and scores each part within the context of the application and the enterprise using the part [43]. Another strategy is defining a clear plan so it can take into account all major events during the systems life time and provides a comprehensive life-cycle ma includes proactive and reactive strategies. For instance, Robinson et al. explored different techniques in managing hardware lifecycle for COTS-systems [47]. Several of techniques have been discussed including monitoring vendor supply and services, market trends, and technology roadmap. Baker’s study [44] proposed emulation as the replacement of a device with another or combination of other devices to provide a form fit and function replacement. The proposed technique was applied on an electronic device where the design is split into three easily defined areas; Interface, Utilities and Core. The technique shows that it can reduce the complexity of the mitigation design enabling more suitable solutions. COTS can be adapted at different stages in system life cycle which requires more adequate planning to avoid and future issues. Blom et al. illustrate the characteristics of “switch or struggle” situations and proposes an initial set of risk
factors to be considered at the late stage of software development [33]. Their proposed method accounts for late COTS component integration, and especially strategies for weighing risks vs. benefits in such “switch or struggle” scenarios.

In the design phase, one of the most cited approach was open system architecture. Recent study proposed what is called Hardware Open Systems Technologies (HOST) as a cost effective and sustainable embedded open system architecture [22]. HOST provides an open, interoperable, upgradeable, and sustainable set of embedded system standards. It promotes reuse of both hardware and software design for future systems. This type of design can overcome some of the obsolescence issues such as upgrading, modification, integration and replacement. Torchiano et al. studied COTS selection and defined general COTS characterization attributes [4]. The resulting framework provides a structure, which facilitate the learning process. This framework provides more insights into the homogeneity of products and provide a tool for characterizing them. By following systematic approach, the study introduced different COTS attributes in order to be measured before going into the development phase. These attributes include: product maturity, market share, performance, safety/security, reliability, hardware requirements, product support, documentation, usability, learnability, modifiability, change frequency, license type, cost of use, software requirements, conformance, and domain specific. These attributes can assist different stakeholders to select specific component, technology or tool.

In supply chain, two measures can be taken including risk mitigation buy (RMB) and partnering agreements with suppliers. Mitigation buy involves purchasing and storing enough obsolete items in case of hardware and electronic components. Based on the lifetime forecast, the number of components will be acquired to minimize the obsolescence risk. However, this approach may not be suitable for software obsolescence which is one of the current challenge.

Software related study by Bhuta and Boehm proposed an attribute-based framework that can be used to perform high-level and automated interoperability assessment to filter out COTS product combinations whose integration will not be feasible within the project constraints [34]. The framework was built upon standard definitions of both COTS components and connectors and is intended for use by architects and developers during the design phase of a software system. Kotonya and Hutchinson proposed an approach to help developers to understand the impact of change [38]. Their approach relies on the use of a COTS component-oriented development process and an architecture description language (ADL) for documenting component system architectures; both elements That includes adapting COTS at different stages in the system lifecycle. For instance, Blom et al. illustrate the characteristics of “switch or struggle” situations and proposes an initial set of risk factors to be considered at the late stage of development [33]. Their proposed method accounts for late COTS component integration, and especially strategies for weighing risks vs. benefits in such “switch or struggle” scenarios.
3.4 DECISION FRAMEWORK FOR MAPPING EXISTING OBsolescence MANAGEMENT APPROACHES

Fig. 9 shows a high level decision framework to map existing obsolescence management studies. This framework has blended multiple existing methodologies and results in a chart that is meant to be flowed through top to bottom at every major stage of the design. Each decision node will be discussed in terms of four characteristic stages: entry conditions, actors involved, main steps, and exit conditions.

3.4.1 DESIGN ITEMS DECISION MAKING

The first step of this process, the initial entry condition, is conducting the preliminary system design or at least identifying the core design items of the system. This design will manifest in the form of the alternative, preliminary, and critical design reviews. However, this procedure can be implemented before a complete system design or between these reviews. This will serve to refine all assumptions and define all constraints.

In the design element stage, the main actors are the design engineers. The design engineering team is tasked with coming to a final list of components to be incorporated in the system. As aforementioned, in large scale systems it is assumed that many of these components
will be COTS-components and will not be created by the design team’s organization but purchased from a third-party.

The traditional system engineering life cycle steps are the main proceedings of this stage. Engineers are expected to meet with customers, architect multiple solutions, identify options between key components, and understand the limitations of those options in this stage. This will produce the most robust design and facilitate the rest of this process.

The exit conditions for this decision making node can be either a list of core COTS-components or a full list of design elements compatible with the existing life-cycle forecasting tools.

### 3.4.2 Tools and Database Decision Making

After the system design stage there are two streams of work created. With the components identified the engineering management team is required to plan for acquisition and execution, this will leverage traditional tools like Microsoft Project and is not the focus of this report.

With the parts selected the next decision node is entered: tools and database construction or selection. Knowing what components will incorporate in the design are the only entry conditions for this decision node. Following this identification, the design team is required to select from existing tools and databases – such as MOCA [27], COCOTS [17], and TOMCAT [38] among others – that will be used to conduct life-cycle cost estimation.

The main actors at the stage are the design engineers that are required to evaluate how these different tools operate and what will be best for their system: one tool, or a combination. Moreover, the engineers are required to select a tool with a proper database, construct a database with historical data and BOMs from the manufacturer or reach out to similar enterprises that have conducted such work in the past and build off their progress.

The exit conditions for this decision node are a clear choice between what tools will be used, what data they have (or do not), and how those tools will be leveraged for life-cycle estimation.

### 3.4.3 Design Refresh Candidate Decision Making

Management is tasked with creating the schedule for the operation and construction of the system. This will manifest in an execution, acquisition, and production plan. Within the phases of the design, integration, test, and roll-out the management team must identify dates that, in accordance with the execution/acquisition/production plans, allow for the system to be taken offline and obsolete components be upgraded. These dates are known as refresh dates.

Design is tasked with identifying, via proactive and reactive monitoring, what components are going to need to be refreshed. Proactive monitoring is conducted utilizing the life-cycle estimation tools and can accurate predict when a component needs to be removed from the system before it compromises system functionality. Often times a component will be removed after it has gone obsolete; however, it is possible that a component must be refreshed before its obsolescence. This occurs when the time between refresh dates is too great and leaving the component past the upcoming refresh date will compromise an unacceptable amount of functionality before the next refresh date.
Combined these refresh candidates and dates are the entry conditions for this decision node. Armed with this information the design and management teams are tasked with working together to leverage the life-cycle estimation tools, like MOCA [27], to select what components must be refreshed at each refresh date. If this process is conducted early in the design stages and frequently it will allow both teams to realize issues in their planning and adjust accordingly.

The exit condition from this decision node is the completion of a design refresh plan that accounts for all obsolescence and technical debt issues in the life of the system.

**3.4.4 Design Modification Decision Making**

When the teams have decided on a design refresh plan they must understand how the modification of each of those components will affect the rest of the system. This task will be conducted by the engineering design team and is the main entry condition for the design modification decision node. For this reason, the flow between the candidate selection and refresh plan is bilateral, as is the flow between the refresh plan and design modification. If a refresh will incur more obsolescence risk or technical debt in the system a better refresh plan, or design, is required.

This is the final and most important node of this procedure. After passing through each stage the engineering management and design teams should have a clearer perspective on how each component will interact, what technical debt issues are incurred by COTS choices, and how to properly leverage the life-cycle estimation tools. With this perspective they can enhance the design to combat the technical debt within their system.

**3.4.5 Gap: COTS-Intensive Technical Debt Identification**

Utilization plan of these existing OM MPTs must be coupled with capabilities that can capture COTS-intensive technical debts and project their effects on the system in the acquisition stages by capturing the different interdependencies of those COTS in the system, as shown in the upper-right yellow box in Fig. 9.

Ideally, after selection the proper tools and capturing the COTS-intensive technical debts components will be categorized by the design team in two ways: proactive and reactive monitoring. If a distinction cannot be made for a component the design team must conduct a risk based criticality review to move forward. The aforementioned tools allow for proactive planning of refreshing components. Existing practices widely cover reactive monitoring in the form of life-time buys, last time buy, redesign, or substitution among other things. However, this is a largely overlooked topic and there is lack of study in this regard. The ultimate goal of the study reported here is to bridge this gap by introducing the notion of COTS technical debt, which will be discussed in the next section.
4. NEED FOR THE NOTION OF COTS TECHNICAL DEBTS

4.1 ANALOGY OF COTS TECHNICAL DEBT IN CPS SYSTEMS ENGINEERING

The idea of technical debt is straightforward: quickly built and delivered solutions will be faster to market but often require more maintenance and sustainment actions over their useful life. In other words, organizations that adopt the utilization of COTS-based solutions may unknowingly incur more systems-centric costs that accrue over the system’s life span. For example, such costing sources are frequently associated with the requirement to maintain operational reliability, availability, and maintainability, as these rapidly-delivered systems solutions are sustained over their officially-planned life spans.

These known, or perhaps unknown costs in lower-performing organizations, can be considered “organizational” debts which will need to be repaid later. A few exemplar forms for such debt repayment include planned systems upgrade, systems replacement costs, or in the worst case, defaulted systems. These unaccounted-for programmatic costs may have very negative consequences which necessitate funding shifts amongst competing systems developments, to the unforeseen detriment of other planned systems modernization programs.

Conversely, a solution more collectively beneficial to the organization may be properly planned for, efficiently and effectively vetted, technology or system introduction with an outcome requiring substantially less resources to maintain operational reliability, availability, and maintainability over the system's life span – thus incurring less “technical debt”.

As the software engineering practice has grown, so has the application of the TD analogy. There is a large body of research that has looked into ways to overcome TD in software development. For instance, Sharma et al. developed its Designite Tool that can detect design “smells” and classify them based on the principle violated, along with a corresponding smell density, which is measured by total smells detected per thousand lines of code [23].

As systems engineering competencies and practices grow, the compelling and critical need for a systems engineering technical debt metaphor grows as well. This need is further amplified by the increasing transition to, and population of, cyber-physical systems (CPS) in applications across all technical efforts within the commercial and military industrial sectors. Introducing COTS products may contribute to an increase in Technical Debt (TD) due to their often shorter commodity life spans, and may also be due to the consumer environments within which they may operate during that life span. Offsetting “obsolescence” to reduce performance risk and maintain systems reliability, availability, maintainability, and cost-control is an underpinning competency of systems engineering, with importance across all system life cycle phases. Tools to manage obsolescence at the component-level exist, and there are tools to assist with diminishing manufacturing sources and material shortages (DMSMS), but there are none that consider the systems-level and composition of cyber-physical systems. The critical need to competently assess technical debt during different life cycle states to provide decision-making leadership, for their synthesis, with evidence-based technically-substantiated assistance. It is our intention to leverage existing engineering methods associated with software technical debt, with utilization of hardware obsolescence management tools and methods to develop a hybrid approach that
aligns those two schools of thoughts with a specific focus on its application to COTS-intensive cyber-physical systems.

### 4.2 Existing Taxonomy of Software Technical Debt

Several taxonomies of technical debt for software system development have been proposed. Most of them evolve from practices centred on agile software development. We summarize three popular ones below, and discuss possible linkage with the COTS-oriented CPS context.

**Rubin’s Taxonomy for Agile Development**

Ken Rubin [28] proposed a high level taxonomy to describe technical debt and how it might be accrued, which includes naïve technical debt, unavoidable technical debt, and strategic technical debt.

1. Naïve TD refers to irresponsible behaviors or immature practices on the people involved, e.g. engineers, business people, and so on. Example causes of naïve technical debt are sloppy design, poor engineering practices, and insufficient testing, as well as over-constrained project date, scope, and budget objectives.

2. Unavoidable TD is usually unpredictable and unpreventable due to system complexity and uncertainty. Examples are the unpredictable need to evolve a design over time as we gain more information, and the unpreventable third-party component (e.g. COTS, open source, etc.) interface changes.

3. Strategic TD is used as a tool to help organizations better quantify and leverage the economics of important, often time-sensitive, decisions. Examples are strategic decisions to take shortcuts during product development to achieve an important short-term goal, e.g. meeting a time to market deadline.

Using Rubin’s taxonomy, COTS changes can be categorized as the unavoidable TD in Rubin’s taxonomy. Other COTS related risk such as insufficient COTS evaluation and planning may correspond to the Naïve TD, and if COTS-over-build strategy is followed at the organizational level, it may correspond to strategic TD.

**Clark’s Taxonomy for Game Development**

In a recent online article, Bill Clark [29] introduced a framework for measuring four types of technical debt used at Riot Games, the company which developed the PC game entitled League of Legends. The measurement framework consists of three major axes to evaluate technical debt including impact, fixed cost, and contagion. The impact axis takes the form of player-facing issues (e.g. bugs, missing features, etc.), and developer-facing issues (e.g. workflow issues, random elements). The fixed cost axis has to do with the cost to fix the tech debt, which should consider both required time to fix and risk for deploying the fix. The contagion axis measures the extent to which the technical debt will spread if it is allowed to continue to exist. Bill Clark’s proposed taxonomy of technical debt includes the following four categories:
Local debt: Local debt resembles the classic “black box” model of programming. Examples of local debt at Riot are facts like “as far as the rest of the game is concerned, the local system (spell, network layer, and script engine) works pretty reliably”. No one may be keeping the debt in mind as they develop around the system, however, if anyone notionally “opens the lid and looks inside”, they may be horrified, disgusted, or completely confused by what they observe.

MacGyver debt: MacGyver debt is named after the TV show from the mid-1980s. Angus MacGyver would solve problems using his Swiss army knife, duct tape, and whatever else was on hand. His solutions often involved attaching two unlikely pieces; in the context of tech debt, this means two conflicting systems are “duct-taped” together at their interface points throughout the codebase. One example of MacGyver debt at Riot Games in the LoL codebase is the use of C++’s std::string vs. its custom AString class. While both provide standard methods for String operations, std::string leads to lots of “hidden” memory allocations and performance costs, and makes it easy to write code that has undesired side effects. AString is specifically designed with thoughtful memory management in mind. Riot’s strategy for replacing std::string with AString was to allow both to exist in the codebase and provide conversions between the two.

Foundational debt: Foundational debt is when some assumption lies deep in the heart of your system and has been baked into the way the entire thing works. Foundational debt is sometimes hard to recognize for experienced users of a system because it’s seen as “just the way it is.”

Data debt: Data debt starts with a piece of technical debt from one of the other categories, but then content gets built on top of that existing deficiency. As time elapses, fixing the initial technical debt becomes extremely risky and difficult. An example for understanding data debt is DNA. The genome of an organism is slowly built up over millions of years through mutations, transcription errors, and evolutionary pressure.

All four categories can be adapted to COTS-oriented CPS context. Though local TD is largely inapplicable due to the fact that there generally is no access inside of COTS black-boxes. However, it is not unusual that defense/government COTS contracting practices frequently involve the purchase of COTS source code. This practice provides opportunities for introducing COTS local debt through modifications to COTS. MacGyver TD can be related to the scenarios where trying to integrate two or more COTS components with conflicting application programming interface specifications. Foundational TD and Data TD can be related to maintenance difficulty due to mismatches between COTS imposed requirements.

**Kruchten et al.’s Technical Debt Landscape**

Kruchten et al. [18] proposed possible organization of a technical debt landscape, as shown in Figure 10. It differentiates technical debts from other user-visible elements such as new features, additional functionalities, and defects to fix. Technical debts refer to those aspects (as grouped in the box) that could potentially impact system evolution and/or quality issues. They are considered “mostly invisible” to business people, and visible only to software/system developers.

This landscape characterizes various forms of software TD according to major software engineering activities introducing such TDs. Therefore, it is reasonable to adapt these to COTS CPS context, for example, architectural debt, structural debt, test debt, documentation debt.
Figure 10. Technical Debt Landscape
5. TAXONOMY ON COTS TECHNICAL DEBTS IN CPS SYSTEMS

Using a COTS-intensive CPS context, engineering and program leadership decisions to rapidly acquire COTS solutions would incur more “debt” to the user/customer/sustaining organization, which needs enduring visibility and accountability from its onset, as its liability is paid off in later systems maintenance or sustainment phases. Objectively, the aim is to better identify and mitigate COTS related risks including the many facets of potential obsolescence issues in designing, developing, maintain, and sustaining CPS systems.

5.1 TAXONOMY ON COTS TECHNICAL DEBTS IN CPS SYSTEMS

Based on the literature review of existing OM work in the CPS domain, and analysis of existing technical debt taxonomies, we propose a new taxonomy of COTS technical debts for CPS systems, as shown in Table 3. It consists of seven COTS TD types, characterized according to risk indicators during early COTS activities from the acquisition phase (e.g. identification, assessment, tailoring, etc.), that may contribute to obsolescence in later life cycles (e.g. maintenance, operation, etc.).

Function TD refers to functionality mismatch between the desired systems needs and that which is delivered by proposed COTS alternative, as well as the excess functionality delivered by COTS that is not necessary for the system. The former will necessitate COTS modification, custom development to bridge the functionality gap, and the later will require additional evaluation and/or wrapper effort to disable/mask unnecessary COTS capabilities for security assurance. This category is similar to the local TD and data TD from Clark’s taxonomy.

Performance TD refers to the mismatches between COTS capabilities and system needs, w.r.t. quality/extra-functional properties such as reliability (e.g. mainly of hardware), safety assurance (e.g. of software and hardware), and performance in terms of e.g. bandwidth, processing capability, memory, etc. This is particularly significant for cases involving the use of COTS products developed for certain contexts, but applied in new contexts with newly expected capabilities, and ultimately found to lack the desired qualities as an outcome. This category is similar to the MacGyver TD and data TD from Clark’s taxonomy.

<table>
<thead>
<tr>
<th>TD Category</th>
<th>Description</th>
<th>Analogy to existing work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Function</td>
<td>The degree of functionality mismatch between COTS capabilities and system needs.</td>
<td>Local TD; Data TD</td>
</tr>
<tr>
<td>Performance</td>
<td>The degree of mismatches between COTS capabilities and system needs, w.r.t. performance properties.</td>
<td>MacGyver TD; Data TD</td>
</tr>
<tr>
<td>Interoperability</td>
<td>The degree of interface/ assumption mismatches among various interdependent COTS components, as well as among COTS and system custom components.</td>
<td>MacGyver TD; Data TD</td>
</tr>
<tr>
<td>Configuratio n Version</td>
<td>CPS configuration version planning needs to address solution availability plan. Greater tendency of COTS version upgrade/refresh may lead to more obsolete COTS.</td>
<td>Unavoidable TD; Local TD; MacGyver TD; Foundational TD; Data TD</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Documentat ion &amp; Support &amp;</td>
<td>Lack of documentation and vendor support will seriously impact on issue resolution related to obsolete COTS.</td>
<td>Unavoidable; Data TD</td>
</tr>
<tr>
<td>System Evolution Limitations</td>
<td>Requirements imposed by COTS may place great limitation on system evolution.</td>
<td>Unavoidable TD; Foundational TD; Data TD</td>
</tr>
<tr>
<td>Organic</td>
<td>People-centric perspective of TD focusing on organizational decision-making, behaviors, and practices associated with those personnel responsible for introductions of new technologies &amp; systems and/or the sustainment of existing systems</td>
<td>Local TD; Naïve TD; Strategic TD</td>
</tr>
</tbody>
</table>

Interoperability TD refers to incompatible interfaces/internal assumptions among multiple COTS and other components in the system. Each COTS comes with a set of assumptions with respect to other components or the physical environment, and insufficient COTS assessment/evaluation during acquisition phase may lead to pre-mature COTS commitment, which contains undiscovered assumption mismatches that may cause COTS integration difficulty and/or obsolete components in later phases. This category is similar to the MacGyver TD and data TD from Clark’s taxonomy.

Configuration Version TD refers to imposed requirements for upgrading COTS components with respect to its newly released versions by the commercial vendors. Unlike hardware suppliers, commercial software vendors typically version upgrade or update their products very frequently (i.e. every 6-12 months), forcing integrators to re-evaluate or reengineer various aspects of the CPS in order to maintain interoperability with current technologies. Keeping up with newer versions of every COTS component is often cumbersome, expensive, and infeasible for developing and maintaining CPS systems. Therefore, a major challenging issue is to strategically refresh COTS versions. If the decision is to skip certain newer COTS versions, a new type of versioning TD would be introduced, which can eventually lead to COTS /system obsolescence. This category represents the unavoidable TD in COTS-oriented CPS context, and may correspond to any of the four TD types in Clark’s taxonomy.

Documentation and Support TD refers to issues due to unavailable, insufficient, or obsolete documentation /vendor support, especially in the face of maintaining and supporting CPS during the operation lifecycle. As COTS become the primary focus of integration efforts for development and sustainment of CPS, such systems require maintenance and support that exceeds typical COTS vendor support [18]. This category represents another source of unavoidable TD in COTS-oriented CPS context, in particularly with respect to COTS obsolescence issues, and may correspond to data TD in Clark’s taxonomy.
System Evolution Limitations TD refers to the inflexibility for accommodating emerging new system functionalities that are required which are out of the initial scope. System requirements imposed by COTS products will place great limitation on system evolution over time. In COTS-based CPS systems, stakeholders may introduce some changes which may contribute to the problem of obsolescence. Candidate COTS-based solutions need to be thoroughly evaluated for imposed architectural sustainability and evolvability limitations. This category represents the third source of unavoidable TD in COTS-oriented CPS context, associated with limitations to system evolution imposed by COTS deficits or rigidity, and may correspond to Foundational TD and Data TD in Clark’s taxonomy.

Organic TD refers to any combination and degree of technological, systemic, project, and program decisions, behaviors, and practices made by the workforce, management and/or senior/executive leadership of the organization responsible for introductions of new technologies and systems and/or the susttainment of existing systems. This category supports that it is possible to create a framework and leadership decision cycle to enable the capability to streamline potentially overbearing acquisition processes while focusing on core critical TD management, processes, and tools which affect systems sustainment supportability, reliability, availability, maintainability, and cost. This is a people-centric category with an organizational perspective as it relates to TD, and includes a category of Clark’s taxonomy and two categories from Rubin’s taxonomy.

5.2 Template for Representing COTS Technical Debt

Based on the literature review of COTS OM studies, we extend the current notion of software “technical debt” [21], and propose the following template for formally representing COTS technical debt in design and developing of CPS systems, as shown in Table 4.

Table 4. Summary of Attributes for Representing a COTS TD Item

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>A unique identifier for the COTS TD item.</td>
</tr>
<tr>
<td>Name</td>
<td>The name of a specific COTS TD item</td>
</tr>
<tr>
<td>Location</td>
<td>The location of the identified COTS TD item, e.g. the name of the COTS(s)</td>
</tr>
<tr>
<td></td>
<td>it is associated.</td>
</tr>
<tr>
<td>Accountable</td>
<td>The party responsible to repay the COTS TD item, e.g. COTS vendor, integration team</td>
</tr>
<tr>
<td>Party</td>
<td>program office, specific organization. This identifies the “accountable”</td>
</tr>
<tr>
<td></td>
<td>debt-holder for the liability. The Accountable Party is identified at the</td>
</tr>
<tr>
<td></td>
<td>start of a new design/development/modernization effort, and can assign TD</td>
</tr>
<tr>
<td></td>
<td>“tracking” and “maintenance of TD visibility” within its span of authority/</td>
</tr>
<tr>
<td></td>
<td>control.</td>
</tr>
<tr>
<td>Type</td>
<td>The COTS TD type that the COTS TD item is classified into (See the Preliminary</td>
</tr>
<tr>
<td></td>
<td>Taxonomy in Section 4.3).</td>
</tr>
<tr>
<td>Description</td>
<td>General information on the COTS TD item.</td>
</tr>
<tr>
<td>Open date/time</td>
<td>The specific date/time when the COTS TD is identified.</td>
</tr>
<tr>
<td>Principle</td>
<td>The estimated cost of repaying the COTS TD item.</td>
</tr>
</tbody>
</table>
**Interest amount**  The estimated extra cost of tolerating the COTS TD item.

**Interest probability**  The probability that the interest for the COTS TD item needs to be repaid.

**Contagion [26]**  The degree of spreading of the COTS TD item through the interfaces with other system components, if this TD is allowed to continue to exist.

**Context**  A certain implementation context of a specific COTS TD item

**Propagation rule**  How the COTS TD item impacts the related parts of the CPS system

**Intentionality**  Is the COTS TD item intentionally or unintentionally incurred?

---

### 5.3 COTS TECHNICAL DEBT MANAGEMENT (TDM) ACTIVITIES

TDM is composed of a set of activities that prevent potential TD from being incurred or deal with existing TD to keep it under a reasonable level [44]. In the field of software engineering, the main TDM activities consist of TD identification, measurement, prioritization, prevention, monitoring, repayment, representation/documentation, and communication, as described in the following Table 5.

**Table 5. Main TD Management Activities**

<table>
<thead>
<tr>
<th>TDM Activity</th>
<th>Description/Example</th>
<th>Techniques</th>
<th>Example metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TD identification</strong></td>
<td>Detects TD caused by intentional or unintentional technical decisions</td>
<td>Static code analysis; dependency analysis; checklist</td>
<td>Violations of coding rules, lack of tests; static code metrics,</td>
</tr>
<tr>
<td><strong>TD measurement</strong></td>
<td>Quantifies the benefit and cost of known TD in a system through estimation techniques</td>
<td>Expert Estimation; estimation models; cost categorization; solution comparison</td>
<td>code metrics; operational metrics; ROI; Cost-benefit ratio; Real options</td>
</tr>
<tr>
<td><strong>TD prioritization</strong></td>
<td>Ranks identified TD items according to predefined rules, which is to be repaid first, and which can be tolerated until later releases.</td>
<td>Cost benefit analysis; High remediation cost first; Portfolio approach; High interest first</td>
<td>Portfolio approach considering TD items along with other new functionalities and bugs as risk and investment opportunities.</td>
</tr>
<tr>
<td><strong>TD prevention</strong></td>
<td>Aims to prevent certain TD from being incurred.</td>
<td>Development process improvement; design decision support; lifecycle cost planning; human factor analysis</td>
<td>Improve process to prevent certain type of TD; evaluate and choose candidate solutions with less potential TD</td>
</tr>
<tr>
<td><strong>TD monitoring</strong></td>
<td>Watches the change of cost and benefit of unresolved TD over time</td>
<td>Threshold-based; Planned check; TD propagation tracking; TD plot; TD monitor with quality attribute focus</td>
<td>Define threshold for quality metrics, and issue warnings if threshold is not met.</td>
</tr>
</tbody>
</table>
Among the above eight TDM activities, TD repayment, TD identification, and TD measurement gained the most attention in the software engineering community, as they correspond to the most fundamental questions in TDM: Where is the TD located? How much is the TD? How to repay the TD? We believe that these TDM definitions and techniques are largely adaptable for use in CPS context of TD, with slight adjustment to include more system engineering techniques such as design trade studies, modeling and simulation, etc. Further field study research in this direction is needed.

<table>
<thead>
<tr>
<th><strong>TD repayment</strong></th>
<th>Resolves or mitigates TD</th>
<th>Reengineering, rewriting; refactoring; bug fixing; fault tolerant; repackaging; automation</th>
<th>Make changes to the code, design, or architecture of the software system without altering external behavior, in order to improve internal quality.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>TD representation/documentation</strong></td>
<td>Provides ways to represent and codify TD in a uniform manner to address concerns of particular stakeholder</td>
<td>Various format of representing TD items.</td>
<td>Example TD data fields: ID, Location, Responsible / author, Type, Description, date /Time, principle, interest amount, interest probability, relation to other TD, context, propagation rule, intentionality</td>
</tr>
<tr>
<td><strong>TD communication</strong></td>
<td>Makes identified TD visible to stakeholders so that it can be discussed and further managed.</td>
<td>TD dashboard; backlog; dependency visualization; code metric visualization; TD list; TD propagation visualization</td>
<td>Dashboard or other visualization tool displaying undesirable dependencies, e.g. overly complex dependencies between system components</td>
</tr>
</tbody>
</table>
6. FUTURE DIRECTIONS - TOWARDS EFFECTIVE COTS TECHNICAL DEBT MANAGEMENT (TDM) FOR CPS SYSTEMS

TDM is composed of a set of activities that prevent potential TD from being incurred or deal with existing TD to keep it at an acceptable level [20]. In the field of software engineering, the main TDM activities consist of TD identification, measurement, prioritization, prevention, monitoring, repayment, representation/documentation, and communication. Among the above TDM activities, TD repayment, TD identification, and TD measurement gained the most attention in the software engineering community, as they correspond to the most fundamental what/where/how/how-much questions in TDM.

We believe that existing TDM definitions and techniques are largely adaptable to CPS, with appropriate adjustment to include more system engineering techniques such as design trade studies, modelling and simulation, etc. For example, unlike software systems where almost all artifacts are automatically represented and documented in electronic formats, many CPS components and process artifacts are mechanical or physical in nature. Intensive TD items in COTS-based CPS systems may come from the complex interdependencies among COTS hardware and software components.

As future work, it is essential to conduct further empirical studies, and develop modelling and simulation techniques for capturing the COTS TD accumulation process with respect to the CPS life cycle, to explore the following critical management decisions:

(1) How and when a COTS TD was introduced? Example TD items may be events such as a decision of skipping a new COTS version and sticking with using the old COTS version, or a new system requirement which invalidates some assumptions for a COTS to be functioning properly. More work is necessary to understand the distribution and frequency of dominant COTS TD items in CPS context.

(2) What is the impact of a COTS TD? Impact or effect of COTS TD may be multi-faceted including degraded functionality, poorer performance, additional work/rework needed, or delayed schedule. More work is needed for developing algorithmic, regression or machine learning based models for predicting the impact of dominant COTS TD factors.

(3) What was done to manage the COTS TD? From a CPS system life cycle point of view, we can simplify the management activities using three categories, i.e. to ignore, do component level maintenance to address local debt, or system level maintenance to address organizational debt. Potential data sources for extracting such information are system problem report repositories, operational logs, or user incident repositories.

(4) How much COTS TD has been accumulated at system level? Simulation models can be built to explore frequency of COTS changes and complex interdependencies within CPS, and investigate on the relationship between various COTS TD management activities and the consequence in CPS life cycle cost and/or risk.

(5) What is the baseline from which one should approach increasing an organization’s understanding of the context of TD, particularly within CPS containing COTS? The intent is to create a useful framework and leadership decision cycle to enable the capability to efficiently, effectively, and successfully introduce new technologies and systems with capabilities that surpass existing systems and technologies. Data sources may include past program experiences,
in terms of cost and schedule for delivered capabilities, and research into the factors considered by the organization in realizing the corresponding degree of success of that new technology or system introduction.

It is also our intention to provide decision support for early COTS decision making in the acquisition phase, such as evaluating competing alternative system solutions involving different COTS combination. It would be critical for program managers to have the analysis capabilities to maintain visibility of potential obsolescence issues and costs so that they can make informed COTS commitment decisions and have the ability to expeditiously adjust plans along the way across the entire systems life cycle with minimal impacts to the time of new systems introductions, to new and legacy systems readiness, and to new and legacy systems sustainment costs.

Leveraging existing TD work in software engineering field, further study will involve more empirical data analysis in CPS context, developing models, methods and tools for COTS TD management at CPS system level, possibly with utilization of hardware obsolescence management tools and methods [13, 14, 24], as well as COTS integration cost/risk analysis tool [17, 27].
7. CONCLUSIONS

As systems engineering competencies and practices grow, the compelling and critical need for a Systems Engineering technical debt metaphor grows as well. Introducing COTS products may contribute to increase Technical Debt (TD) due to their often shorter commodity life spans, and the consumer environments within which they may operate during that life span. Offsetting “obsolescence” to reduce performance risk and maintain systems reliability, availability, maintainability, and cost-control is an underpinning competency of systems engineering, with importance across all system life cycle phases.

To that end, this study proposed a taxonomy of COTS-related technical debts in order to support early identification, communication, and assessment of obsolescence risks in CPS system engineering life cycles. The seven technical debt types providing the greatest insight into obsolescence mitigation include COTS functionality mismatch, performance mismatch, interoperability difficulty, versioning frequency, documentation and support readiness, and limitation on system evolution. This provides the foundation, yet leaves flexibility for promising future work along several dimensions. It is expected that such notions of COTS technical debts will help to increase the efficiency, effectiveness, and accountability of COTS-based CPS design, development, maintenance, and sustainment, through the use of better-informed and more-considerate COTS decision making practices in the acquisition process to avoid presently unforeseen, expensive and unaffordable obsolescence issues later in the system’s life cycle, particularly in the sustainment phase.

Future directions include: (1) Develop methods and tools to model and simulate the lifecycle TD accumulation process for COTS-intensive CPS at system level, and analyze the optimal strategies for TD repayment at system level and/or COTS level; (2) Further elaborate and empirically validate the COTS TD taxonomy with more COTS-oriented CPS data collection and analysis, particularly with the focus of early identification and measurement of TD at COTS/NDI procurement time; (3) determine effective ways to represent and document COTS TD to meet stakeholders’ needs of browsing, communicating and managing TD despite the black-box nature of many COTS. Exploring visualization techniques will be an integral part of research along this direction as well.


3. A Literature Review on Obsolescence Management in COTS-Centric Cyber Physical Systems
   Turki Alelyani, Ronald Michel, Ye Yang, Jon Wade, Dinesh Verma, Martin Törngren. Submitted to CSER 2019, under review.

## APPENDIX A: SUMMARY OF INCLUDED EXISTING STUDIES IN THE LITERATURE REVIEW

<table>
<thead>
<tr>
<th>Ref. #</th>
<th>Paper Title</th>
<th>Year</th>
<th>Venue</th>
<th>MPTs?</th>
<th>Sector</th>
<th>Phase</th>
<th>Granularity</th>
</tr>
</thead>
<tbody>
<tr>
<td>[31]</td>
<td>Configurable Obsolescence Mitigation Methodologies</td>
<td>2013</td>
<td>Conference</td>
<td>Method</td>
<td>Industry</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
</tr>
<tr>
<td>[34]</td>
<td>Assessing COTS Integration Risk Using Cost Estimation Inputs</td>
<td>2006</td>
<td>Conference</td>
<td>Tool</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
</tr>
<tr>
<td>[38]</td>
<td>TOMCAT: An Obsolescence Management</td>
<td>2012</td>
<td>Journal</td>
<td>Tool</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
</tr>
<tr>
<td>[63]</td>
<td>A Study on Component Obsolescence Mitigation Strategies and Their Impact on R&amp;M</td>
<td>2003</td>
<td>Conference</td>
<td>Other</td>
<td>Industry</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
</tr>
<tr>
<td>[64]</td>
<td>The COTS Software Obsolescence Threat</td>
<td>2006</td>
<td>Conference</td>
<td>Method</td>
<td>Industry</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
</tr>
<tr>
<td>[65]</td>
<td>Designing Engineering Systems for Sustainability</td>
<td>2008</td>
<td>Journal</td>
<td>Other</td>
<td>Academia</td>
<td>Technology maturation and risk reduction</td>
<td>System</td>
</tr>
<tr>
<td>[66]</td>
<td>DESIGN REFRESH PLANNING MODELS FOR MANAGING OBsolescence</td>
<td>2012</td>
<td>Conference</td>
<td>Method</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>System</td>
</tr>
<tr>
<td>#</td>
<td>Title</td>
<td>Year</td>
<td>Type</td>
<td>Source</td>
<td>Methodology and Risk Reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>----</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
<td>--------</td>
<td>------------------</td>
<td>---------------------------------------------------</td>
<td>-------------------------</td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>Switch or Struggle: Risk Assessment for Late Integration of COTS Components</td>
<td>2007</td>
<td>Conference</td>
<td>Academia</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>A Framework for Identification and Resolution of Interoperability Mismatches in COTS-Based Systems</td>
<td>2007</td>
<td>Conference</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>Risk Management of COTS Based Systems Development</td>
<td>2008</td>
<td>Journal</td>
<td>Industry</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>Calculating the Application Criticality and Business Risk from Technology Obsolescence</td>
<td>2012</td>
<td>Journal</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>Implementing Large-Scale COTS Reengineering within the United States Department of Defense</td>
<td>2002</td>
<td>Journal</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>Analysing the Impact of Change in COTS-Based Systems</td>
<td>2005</td>
<td>Conference</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>A Taxonomy and Evaluation Criteria for DMSMS Tools, Databases and Services</td>
<td>2005</td>
<td>Conference</td>
<td>Academia</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>Mitigating COTS Obsolescence In Military Test</td>
<td>2001</td>
<td>Conference</td>
<td>Government</td>
<td>Operations and support</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>Forecasting Technology Insertion Concurrent with Design Refresh Planning for COTS-Based Electronic Systems</td>
<td>2005</td>
<td>Conference</td>
<td>Academia</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>A Methodology for Addressing Support Equipment Obsolescence</td>
<td>2000</td>
<td>Journal</td>
<td>Industry</td>
<td>Operations and support</td>
<td>Other</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>ATE Obsolescence Solutions: Cost and Benefits</td>
<td>2002</td>
<td>Journal</td>
<td>Industry</td>
<td>Production and deployment</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>Component Obsolescence Risk Assessment</td>
<td>2003</td>
<td>Journal</td>
<td>Government</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>A Model for Comparing Sourcing Strategies for Parts in Long Life Cycle Products Subject to Long-Term Supply Chain Disruptions</td>
<td>2013</td>
<td>Journal</td>
<td>Academia</td>
<td>Operations and support</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>Key Challenges in Software Application Complexity and Obsolescence Management within Aerospace Industry</td>
<td>2015</td>
<td>Journal</td>
<td>Process</td>
<td>Industry</td>
<td>Operations and support</td>
<td>Other</td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>Forecasting Electronic Part Procurement Lifetimes to Enable the Management of DMSMS Obsolescence</td>
<td>2011</td>
<td>Journal</td>
<td>Method</td>
<td>Industry</td>
<td>Operations and support</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>COTS-based System's Obsolescence Risk Evaluation</td>
<td>2006</td>
<td>Journal</td>
<td>Method</td>
<td>Industry</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>The COTS Software Obsolescence Threat</td>
<td>2006</td>
<td>Journal</td>
<td>Process</td>
<td>Industry</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>Proactively Managing Obsolescence with Test System Architecture</td>
<td>2015</td>
<td>Journal</td>
<td>Method</td>
<td>Industry</td>
<td>Operations and support</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Approaches to Reduce Impact of Electronic Component Obsolescence</td>
<td>1998</td>
<td>Conference</td>
<td>Tool</td>
<td>Industry</td>
<td>Operations and support</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>COTS Score: An Acceptance Methodology</td>
<td>2000</td>
<td>Conference</td>
<td>Method</td>
<td>Government</td>
<td>Production and deployment</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>Computer Upgrade and TPS Re-Host of Military ATE w COTS Hardware and Software</td>
<td>1995</td>
<td>Conference</td>
<td>Method</td>
<td>Government</td>
<td>Technology maturation and risk reduction</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>Software Obsolescence in Defence</td>
<td>2014</td>
<td>Conference</td>
<td>Method</td>
<td>Government</td>
<td>Operations and support</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>The ANTARES Offshore Data Acquisition: A Highly Distributed, Embedded and COTS-based System</td>
<td>2000</td>
<td>Conference</td>
<td>Other</td>
<td>Industry</td>
<td>Operations and support</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Software Obsolescence – Complicating the Part and Technology Obsolescence Management Problem</td>
<td>2007</td>
<td>Journal</td>
<td>Other</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>Metrics-Based Framework for Decision Making in COTS-Based Software Systems</td>
<td>2002</td>
<td>Conference</td>
<td>Other</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>Key Challenges in Managing Software Obsolescence for Industrial Product-Service Systems (IPS2)</td>
<td>2010</td>
<td>Conference</td>
<td>Other</td>
<td>Academia</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
</tr>
<tr>
<td>An obsolescence forecasting methodology to support strategic system support decision making</td>
<td>2007</td>
<td>Thesis</td>
<td>Method</td>
<td>Academia</td>
<td>Operations and support</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>Cyber-physical acquisition strategy for COTS-based Agility-Driven Engineering</td>
<td>2016</td>
<td>Thesis</td>
<td>Method</td>
<td>Academia</td>
<td>Materiel solution analysis</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>OBSOLESCENCE: A SYSTEMS ENGINEERING AND MANAGEMENT</td>
<td>2010</td>
<td>Thesis</td>
<td>Method</td>
<td>Academia</td>
<td>Operations and support</td>
<td>System</td>
<td></td>
</tr>
<tr>
<td>APPROACH FOR COMPLEX SYSTEMS</td>
<td>2008</td>
<td>Other</td>
<td>Engineering and manufacturing development</td>
<td>Component</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------------------------</td>
<td>------</td>
<td>-------</td>
<td>------------------------------------------</td>
<td>-----------</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Commercial-off-the-shelf (cots): Doing it right</td>
<td>DMSMS</td>
<td>Government</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix B: Other Cited and Related References


