FY2022 NDAA Section 835 Independent Study on Technical Debt in Software-Intensive Systems

SEPTEMBER 14, 2023

Ipek Ozkaya, PhD
Technical Director, Engineering Intelligent Software Solutions

Brigid O'Hearn
Senior Software Acquisition Innovation Specialist

Carnegie Mellon University
Software Engineering Institute

Document Markings

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

DM23-0969

Agenda

- Background
- Sec 835 - FY2022 NDAA Extract
- Definition of Technical Debt
- Study Methodology
- Findings
- Recommendations

Background

- DoD lead: OUSD(A&S)/Acquisition Data and Analytics
  - Study results expected to be of key importance to the iterative and Agile acquisition approaches used by programs on the Software Acquisition Pathway (DoDI 5000.87)
- Study by: Carnegie Mellon University / Software Engineering Institute (SEI); Section 835 requires study to be performed by a federally funded research and development center
  - SEI is a recognized leader in the practice of understanding and addressing technical debt in software-intensive systems
  - Personnel assigned include globally recognized thought leaders in tech debt, and experts who are deeply familiar with DoD Software Modernization efforts
  - Study start date: May 1, 2022
  - Report due to Congress: November 1, 2023

Sec 835 - FY2022 NDAA Extract

(b) Study Elements – analyses and recommendations, including actionable and specific guidance and any recommendations for statutory or regulatory modifications, on the following:

(1) Qualitative and quantitative measures which can be used to identify a desired future state for software-intensive systems.

(2) Qualitative and quantitative measures that can be used to assess technical debt.

(3) Policies for data access to identify and assess technical debt and best practices for software intensive systems to make such data appropriately available for use.

(4) Forms of technical debt which are suitable for objective or subjective analysis.

Measures

Data Access

Forms of tech debt

(5) Current practices of Department of Defense software-intensive systems to track and use data related to technical debt.

(6) Appropriate individuals or organizations that should be responsible for the identification and assessment of technical debt, including the organization responsible for independent assessments.

(7) Scenarios, frequency, or program phases during which technical debt should be assessed.

(8) Best practices to identify, assess, and monitor the accumulating costs technical debt.

(9) Criteria to support decisions by appropriate officials on whether to incur, carry, or reduce technical debt.

(10) Practices for the Department of Defense to incrementally adopt to initiate practices for managing or reducing technical debt

Current DoD Practices

Roles

Scenarios

Decision Support Aids

Management Practices

Definition of Technical Debt

- Section 835 defines technical debt as: “an element of design or implementation that is expedient in the short term, but that would result in a technical context that can make a future change costlier or impossible”

- The DoD software pathway policy (DoDI 5000.87) requires technical debt management and defines it like Section 835. It also adds: “Technical debt may result from having code issues related to architecture, structure, duplication, test coverage, comments and documentation, potential bugs, complexity, coding practices, and style which may accrue at the level of overall system design or system architecture, even in systems with great code quality”

Study Methodology

- Literature review summarizing the state of the art
  - Particular focus areas include: Gap between automation (static analysis tools) and their ability to alert users to comprehensive symptoms of tech debt; recognized practices for managing tech debt
- Report for program stakeholders
  - State-of-the-practice report, discussion, and feedback from personnel on a large, complex, safety- and security-critical program
- Deep dives on program data
  - Examination of program artifacts, practices, data, and decision-making
- Interviews: 16 engagements included interviewing stakeholders from the Federal Government and industry, to gain a broad view of the state of the practice. The type of systems covered defense business systems, embedded safety-critical systems, embedded weapon systems, and command & control systems. The Section 835 10 study elements were used in the interviews. Of the interviews,
  - 11 were held with DoD organizations (Air Force, Army, Navy, Joint)
  - 4 were held with industry (DIB and commercial)
  - 1 was held with a program under another Federal Government agency, the National Aeronautics and Space Administration Jet Propulsion Laboratory (NASA JPL)

Findings

Findings with supporting details will be in the final report of the study due to Congress on November 1, 2023

Data gathering and analysis has focused on the 10 study elements specified in Sec 835(b), but are presented here in the aggregate for clarity. These findings came from interviews as well as deep dives on program data:

1. DoD programs are aware of technical debt as a concept and its presence in their systems.
2. DoD programs do not employ consistent technical debt management practices.
3. Managing technical debt is often deprioritized.
4. Use of metrics and reporting can help demonstrate that a program is taking technical debt seriously.
5. Use of tools can help prevent unintentional quality issues from turning into technical debt.
6. Categories of funding can hamper appropriate management of technical debt.
7. The Software Acquisition Pathway (SWP) explicitly highlights technical debt management, which creates a seamless entry point for DoD programs already on the SWP to start technical debt management practices.
8. The emergence of AI-augmented software development tools and their relationship to technical debt is still not fully understood.
9. Industry has established best practices in understanding and managing technical debt, which can inspire the DoD.
10. Educating stakeholders is key.
11. Analyzing data collected by tools allows for identifying technical debt correctly.

Recommendations

SEI will provide recommendations with supporting details in the final report of the study due to Congress on November 1, 2023

A. ***Best Practices:*** DoD needs to share best practices to empower programs to incorporate technical debt management into software development lifecycle activities as a core software engineering practice
B. ***Policy & Guidance:*** Update OSD policy & guidance to include technical debt management practices
C. ***Training:*** Make available and encourage appropriate training
D. ***Metrics:*** Require continuous collection of technical-debt related data and metrics
E. ***Financial Management Regulation updates:*** Update the FMR to clarify "color of money" concerns with technical debt
F. ***Tools:*** Ensure that more programs have access to modern development, analysis, and CI/CD tools and practices
G. ***Research:*** OSD should invest in/utilize technical debt research areas

Technical Debt Management Best Practices

Recommended practices can be rolled out in three stages:

- Stage 1: Bring visibility to existing technical debt
- Stage 2: Establish goals
- Stage 3: Establish tooling and measurement environments

Technical Debt Management Best Practices (cont.)

Stage 1: Bring visibility to existing technical debt

- Configure existing issue tracking and management tools to include a technical debt category, so that these instances can be tracked and handled separately.
- During design and architecture reviews, explicitly capture technical debt, including remediation strategies. These reviews typically surface issues that are not clear-cut defects or vulnerabilities, but would require substantial rework to improve the code.
- During development, empower developers to manually document as technical debt any issues that are difficult to resolve and that require further tradeoff and root cause analysis.
- As part of regular release reviews, capture technical debt, including remediation strategies. This technical debt may include overarching concerns (e.g., end-of-life of software, hardware, operating systems) that will require substantial rework.

Technical Debt Management Best Practices (cont.)

Carnegie Mellon University Software Engineering Institute

Stage 2: Establish goals

- It is essential to allocate time (e.g., in Agile capacity planning) to address technical debt during each delivery increment
- Relate technical debt to Agile enabler stories
- Review technical debt enabler stories regularly during sprints and other reviews
- Prioritize these stories alongside other capability priorities

This approach allows teams to apply measures, such as:

- percentage of resources allocated to be spent on managing quality and technical debt per delivery increment (e.g., sprint, iteration, gate, release)
- percentage of technical debt items in the backlog, which enables the program to visualize the technical debt that is carried

Technical Debt Management Best Practices (cont.)

Stage 3: Establish tooling and measurement environments

- Use of *automation and tool support* through modern software engineering tools (e.g., configuration management, continuous integration, code analyzers, development tools, issue trackers) to ensure quality and prevent unintentional technical debt from creeping in
- The expectation that software developers are encouraged to *manually* record instances of technical debt items as they occur so that they can be paid down in the future
- The establishment of heuristics at the program level, such as establishing an overall threshold for the percentage of open technical debt items in the backlog
  - Technical debt items should make up no more than 10-15 percent of overall backlog items. If the program exceeds the technical debt threshold, a technical debt reduction sprint should be planned.
- The use of tools and procedures that allow both tool-discovered design issues and developer-reported technical debt instances to be tracked, prioritized against other work, and regularly addressed alongside new capability development

Technical Debt Management Best Practices (cont.)

Stage 3: Establish tooling and measurement environments (cont.)

This stage is also when system-specific quantifiable metrics can be more effectively addressed. These metrics must be considered along with both (1) other program metrics (e.g., backlog items, defect rates) and (2) system metrics (e.g., metrics for software quality, system complexity, vulnerability, and security assessment).

Candidate high-level metrics include

- reporting the *planned time* spent paying down technical debt
- reporting the *actual time* spent addressing technical debt
- reporting the *total* number of technical debt items in the backlog (noting increases/decreases)
- reporting *deployment frequency* with a mapping to technical debt instances that hinder delivery

Questions?

Contact

Ipek Ozkaya
Technical Director, Engineering Intelligent Software Solutions

Email: info@sei.cmu.edu

Brigid O'Hearn
Senior Software Acquisition Innovation Specialist

Email: info@sei.cmu.edu