Management of Technical Debt – A Lockheed Martin Experience Report

Robert Eisenberg
Lockheed Martin
Gaithersburg, MD
Robert.J.Eisenberg@lmco.com
301-240-5510

LOCKHEED MARTIN

Agenda

- Overview & Objectives
- Accomplishments
- Challenges
- Summary

Lockheed Martin - Overview

- World's largest Defense contractor
- Organized along five major business areas
  - Relatively independent processes
- Software is critical to our products, but we do not view ourselves as primarily a software company
- Many mature software intensive systems
  - Extensive use of metrics
  - CMMI 3-5

Technical Debt Objectives

- Develop standard approach to computing software technical debt
- Develop techniques to effectively utilize the software debt metrics
- Explore and apply the debt metaphor beyond software

Accomplishment – Software Guidebook and Calculator

- Developed Software Guidebook and Debt Calculator
  - Addresses measurement, analysis, management, remediation, and prevention of debt in code
  - Makes visible both intentional an unintentional debt
  - Is based on thresholds for specific quality attributes
    - Green/Yellow/Red Thresholds established.
    - Debt becomes the cost to return to green.

		Status		
		Green	Yellow	Red
Method Complexity > 10	<=	5%	10%	otherw
Unit Test Coverage	>=	80%	60%	otherw
...				

Debt Data Details	9/15/2013	10/1/2013	10/15/2013
Method Complexity			
Count - method >= 10	36	34	37
Count - total methods	600	680	745
% of methods >= 10	6.0%	5.0%	5.0%
Hours to "green"	12.0	0.0	0.0
Unit Test Coverage			
Count - uncovered lines	2,210	2,512	2,729
Count - uncovered conditions	1,108	1,255	1,522
% Covered	78.0%	78.5%	78.5%
Hours to "green"	60.4	52.6	59.4

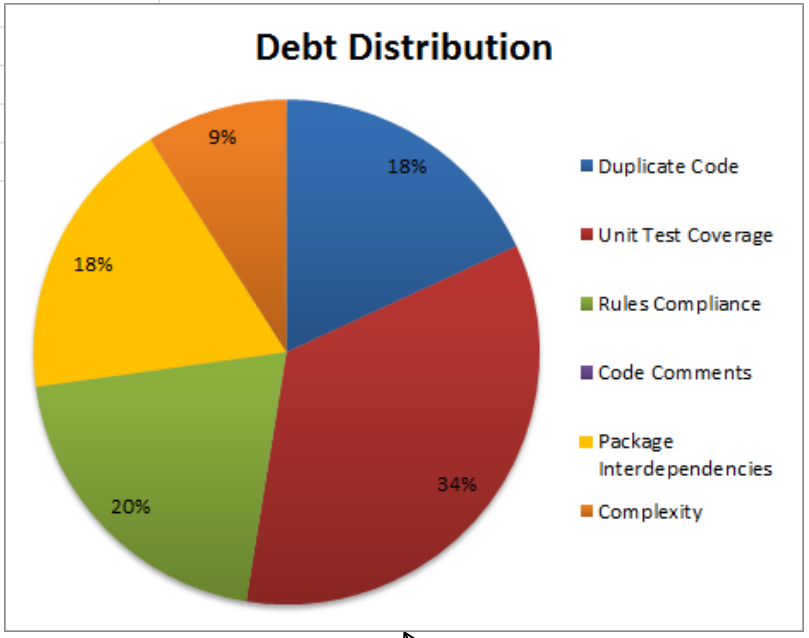
Note: Values shown do not represent LM thresholds or data

Accomplishment – Software Guidebook and Calculator (cont.)

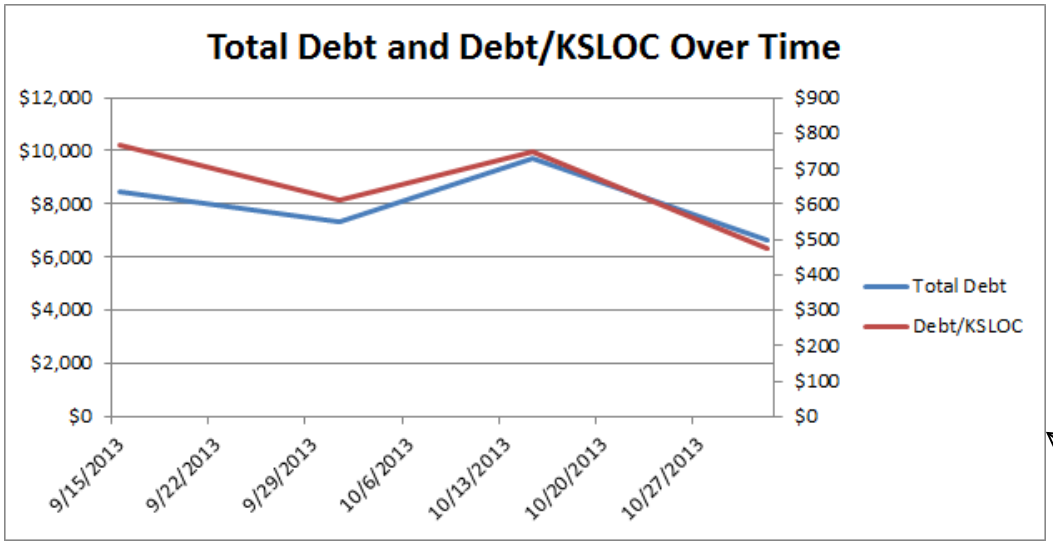
Using the Guidebook and Calculator technical debt can be measured, tracked, and managed over time

	9/15/2013	10/1/2013	10/15/2013	11/1/2013
Summary Level Info				
SLOC	11,000	12,000	13,000	
Total Debt (Hours)	84	73	97	
Total Debt $	$8,430	$7,320	$9,730	
Debt/KSLOC	$766	$610	$748	

Summary Information

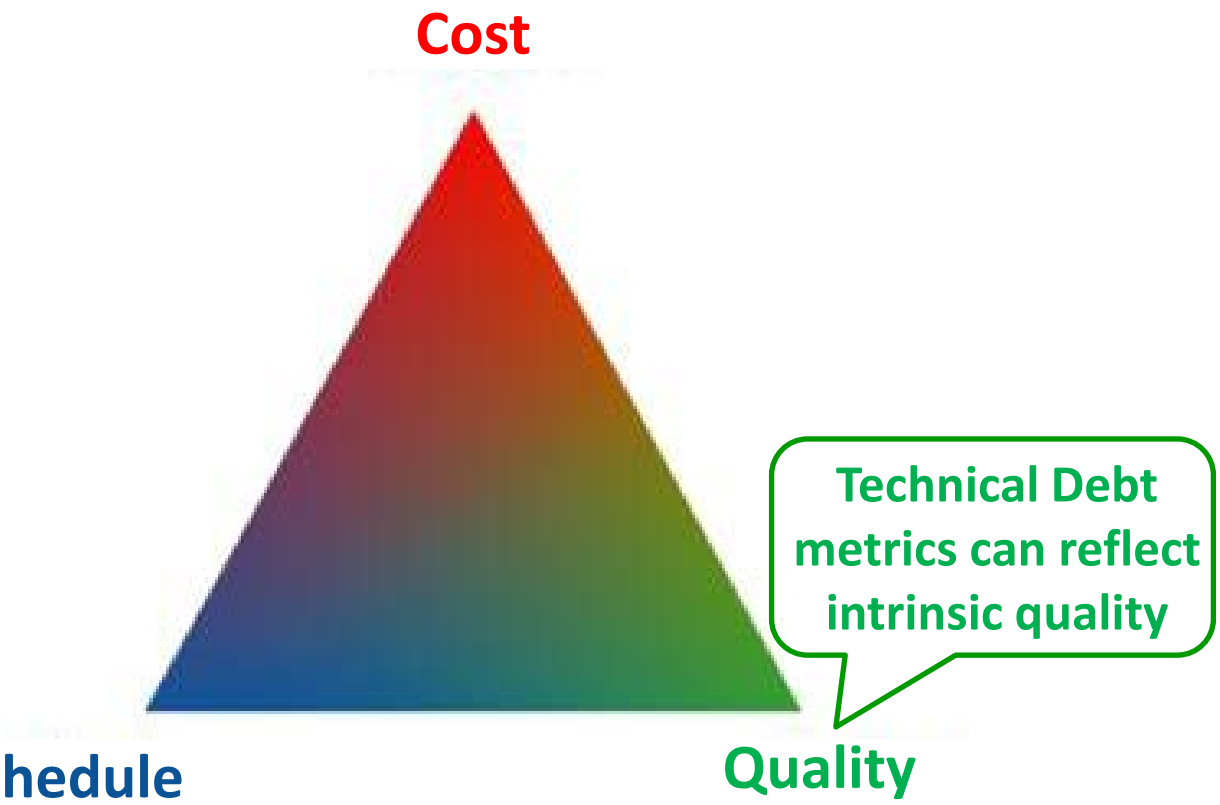


Sources of Program Debt



Graphs for trending

Accomplishment – Technical Debt as a Software Quality Metric



Raised awareness that technical debt can be utilized as a quality metric

Accomplishment – Product Line Analysis

- Debt centric approach used to analyze challenges within a product line
- Debt injection points were identified
- Developed checklist and safeguards to prevent inadvertent debt incursion
- “Cultural awareness” of technical debt identified as significant

Other Accomplishments

- Awareness of Technical Debt has expanded
  - Internal conferences
  - Internal and external webinars
  - Includes behavior aspects, “patterns of technical debt”
- Systematic use of technical debt code metrics is expanding
- Some senior managers are now requesting technical debt metrics
- Technical debt management is an integral part of our Agile approaches
- Concept has been applied to FPGA development

Challenges

- Expanding our usage of the debt metaphor
- Challenges will be posed as questions
- Lay the foundation for future discussion beyond the scope of this briefing

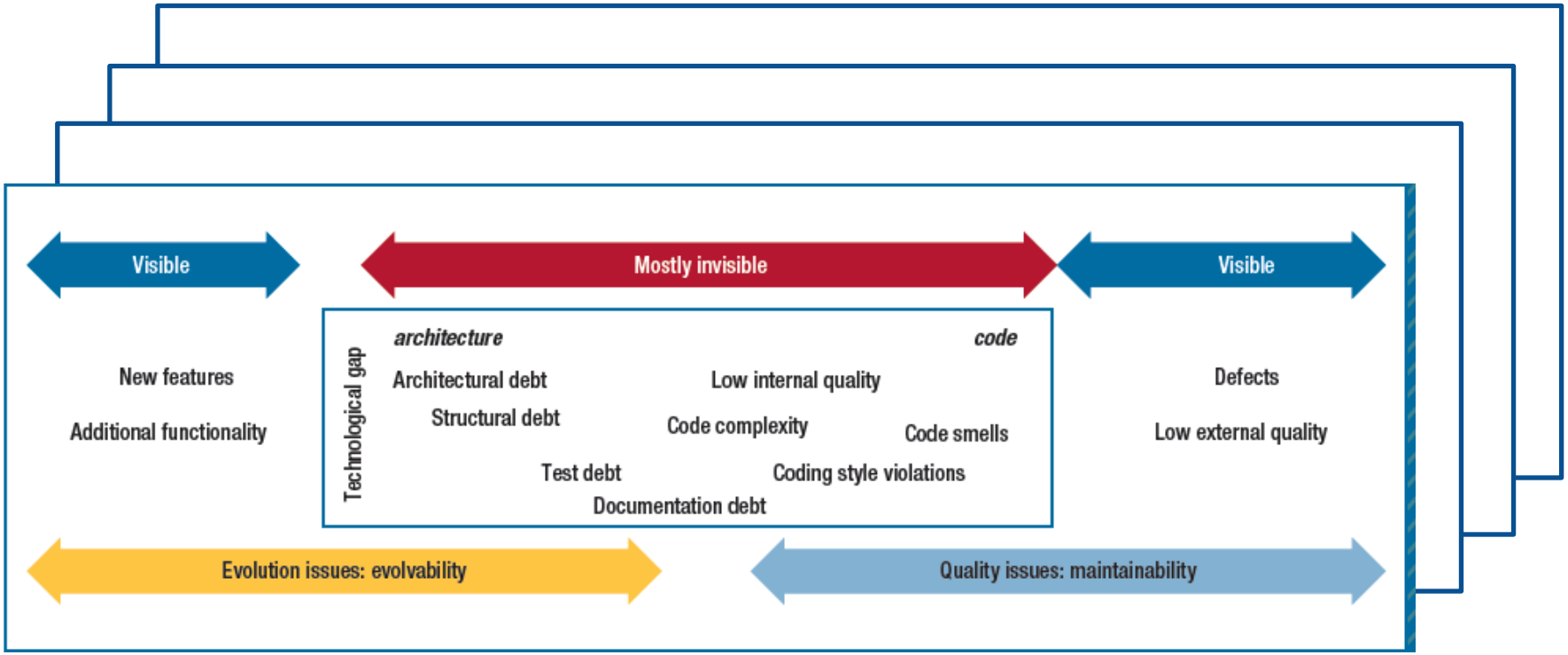
Challenge - Quantifying the Benefits

- Difficult to quantify the benefit of tracking aggregate metrics or other visible debt
  - How much debt is too much?
  - How do I utilize a dollarized metric on existing programs?
  - What is the ROI?
  - Is it a useful predictive measure?
  - How do we demonstrate the full benefits with only partial lifecycle visibility?

Business Leaders want Proof!

Challenge – Expanding the Technical Debt Landscape

- Should we expand this?
  - Not “left” or “right”, but in “depth”



From IEEE Software Nov/Dec 2012, Kruchten, Nord, & Ozkaya

Challenge - Usage as a Quality Metric beyond Software

- Technical debt is often described as a measurement of the *intrinsic* quality of the software.
- Can we expand this to other products, including intermediate products?
  - The intrinsic quality of requirements artifacts
  - The intrinsic quality of test artifacts
- Do we need new tools to enable this?

Is the debt metaphor appropriate to describe the *intrinsic* quality of other products?

Challenge -
Expanded Definition beyond Design/Construction

- Common definition:

… a **design or construction approach** that's expedient in the short term but increases complexity and cost in the long term

- Should we generalize across engineering such that debt is the **result of any suboptimal choice** for short term benefit with long term implications?
  - Inefficient processes
  - Proposal assumptions
  - Product rushed to meet an intermediate milestone or artifact deliverable

Is the debt metaphor appropriate
for any suboptimal short term decision?

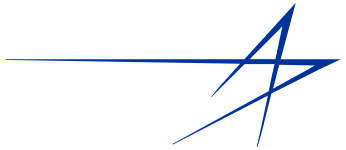
Pros and Cons of Expanding the Debt Landscape

- **Cons**
  - Muddies the definition
  - Creates confusion
  - Provides an excuse for bad behavior

- **Pros**
  - Helps identify, quantify, and address real problems
  - Makes inadvertent debt more visible
  - Makes bad behavior more visible

Summary

- Substantial progress in applying technical debt metrics to Software
- Steady progress toward adoption of software technical debt across the enterprise
- In early stages of expanding debt metaphor to wider array of products and disciplines
- The debt metaphor is powerful and resonates with business and technical leaders



Discussion