

Managing Technical Debt Mile High Agile 2013 Denver, Colorado April 19, 2013 by Kenny Rubin

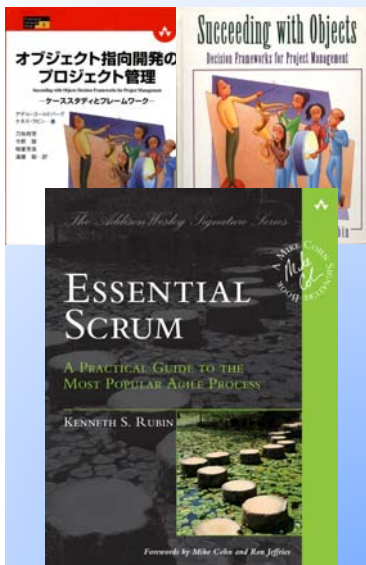
www.innovation.com

Copyright © 2013, Innolution, LLC. All Rights Reserved.

1

Background of Kenny Rubin

Author



Trainer/Coach

Trained more than
19,000 people in
Agile/Scrum, SW
dev and PM

Provide Agile/
Scrum coaching to
developers and
executives



Experience

Former Managing
Director



My first Scrum project was
in 2000 for bioinformatics

GENOMICA

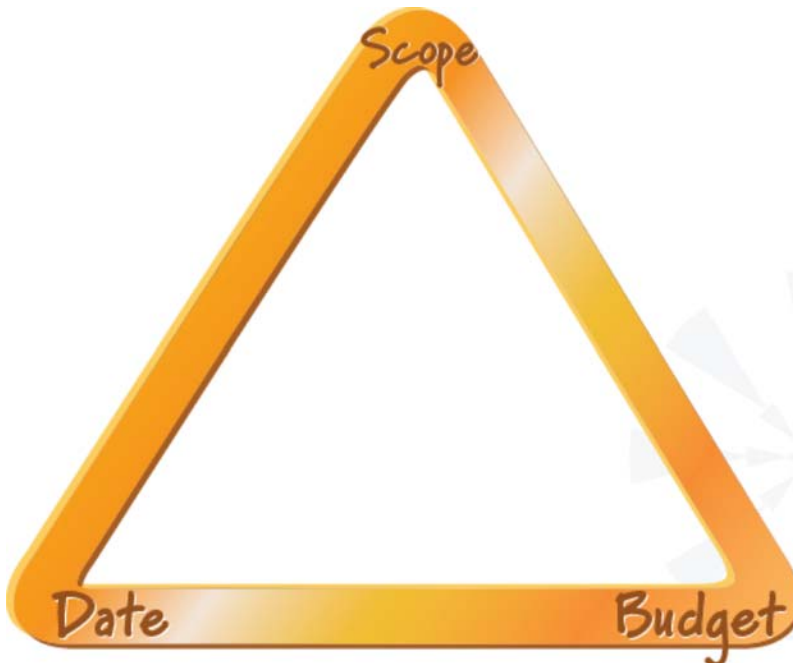


Executive

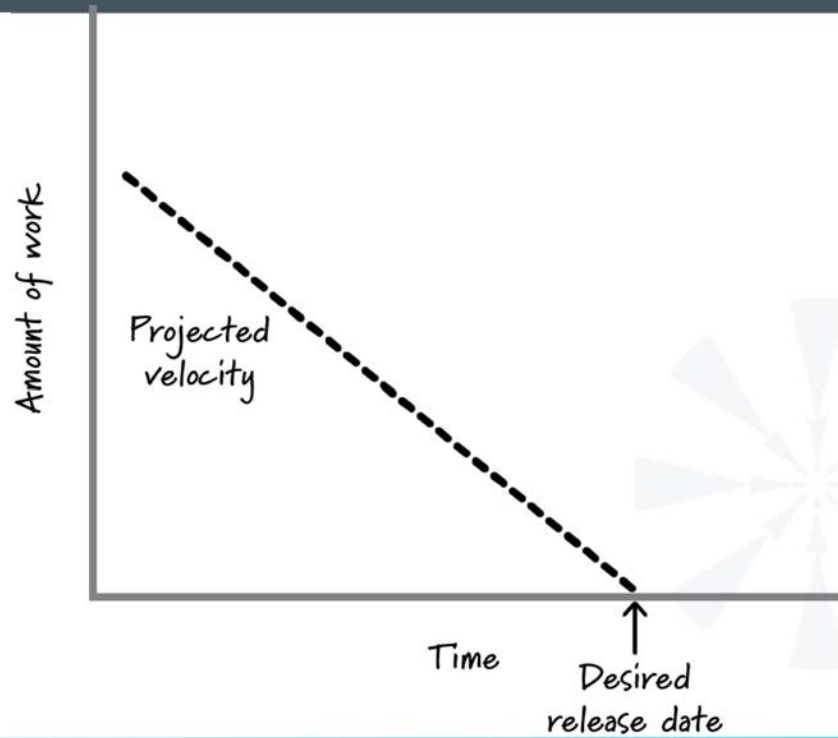




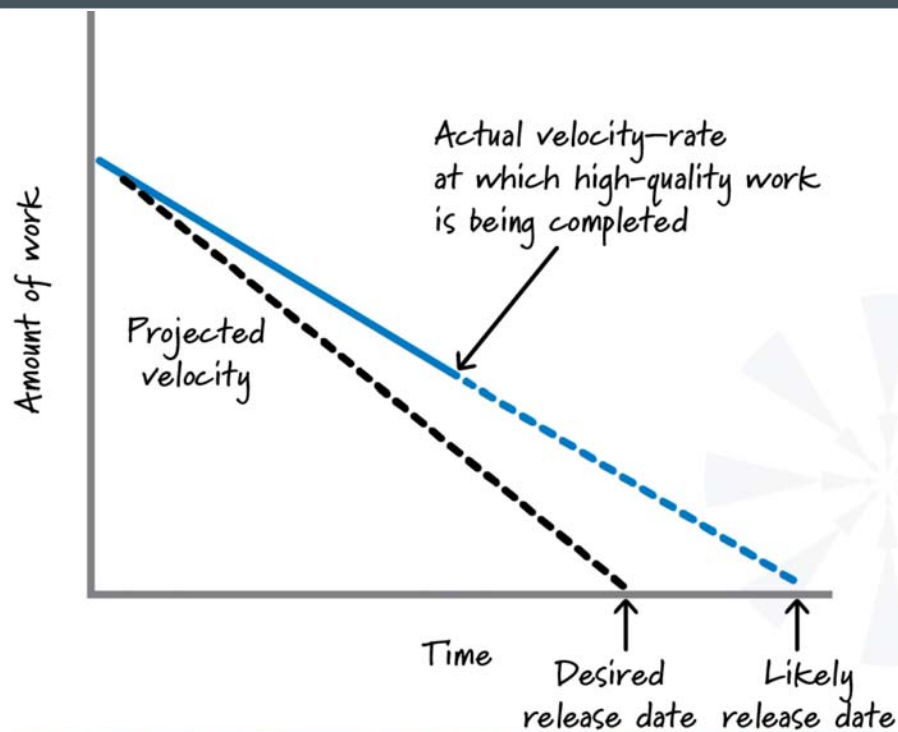
Over Constrained Development Effort



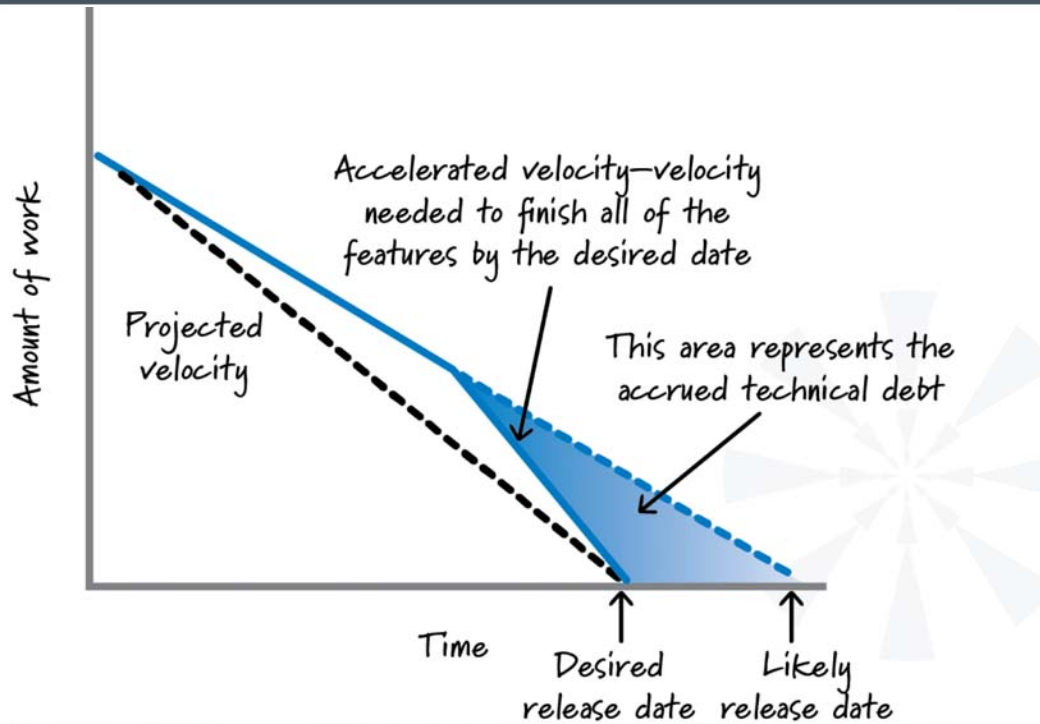
Desired Release Date



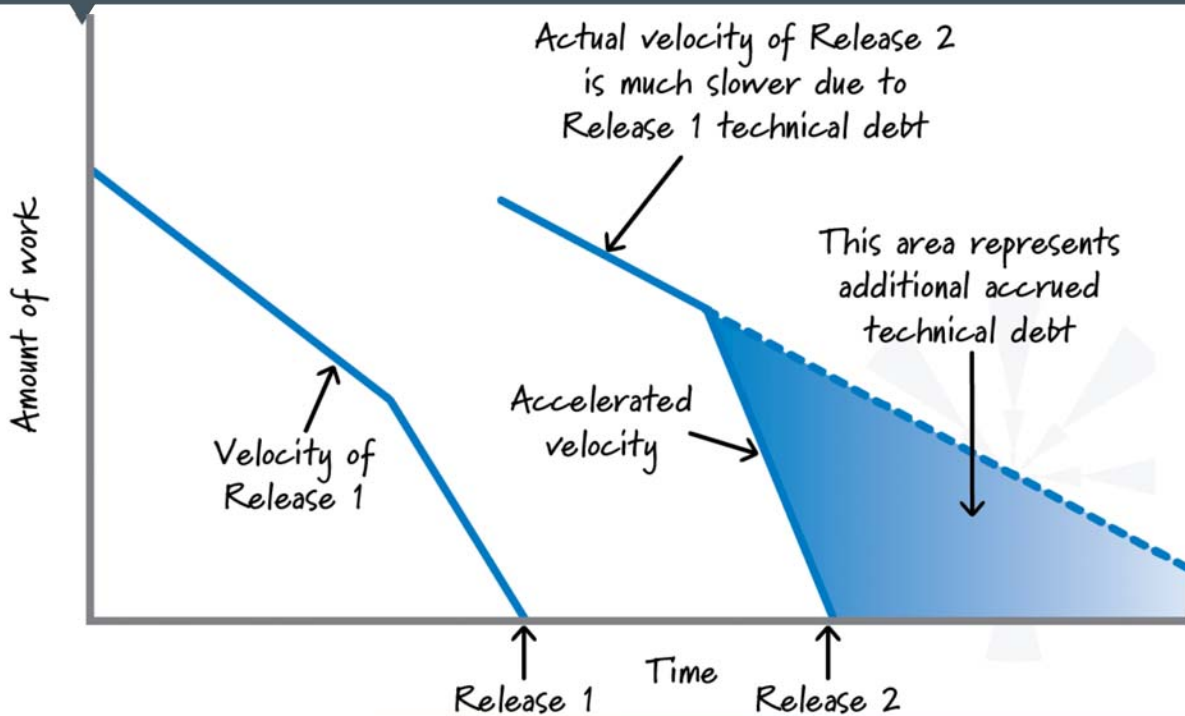
Likely Release Date



Compromised Release Date Leading to Technical Debt



More Compromises; More Technical Debt



✱ Definition of Technical Debt

Ward
Cunningham

Shipping first time code is like going into debt. A little debt speeds development so long as it is paid back promptly with a rewrite. . . .

The danger occurs when the debt is not repaid. Every minute spent on not-quite-right code counts as interest on that debt. Entire engineering organizations can be brought to a stand-still under the debt load of an unconsolidated implementation.

Common
Usage

Today, technical debt refers both to the shortcuts we purposely take and also to the many bad things that plague software systems.



✱ Categories of Technical Debt

Unfit (bad) design

Defects

Insufficient test coverage

Excessive manual testing

Poor integration and release management

Lack of platform experience

Etc.



✶ Technical Debt Accrual Modes

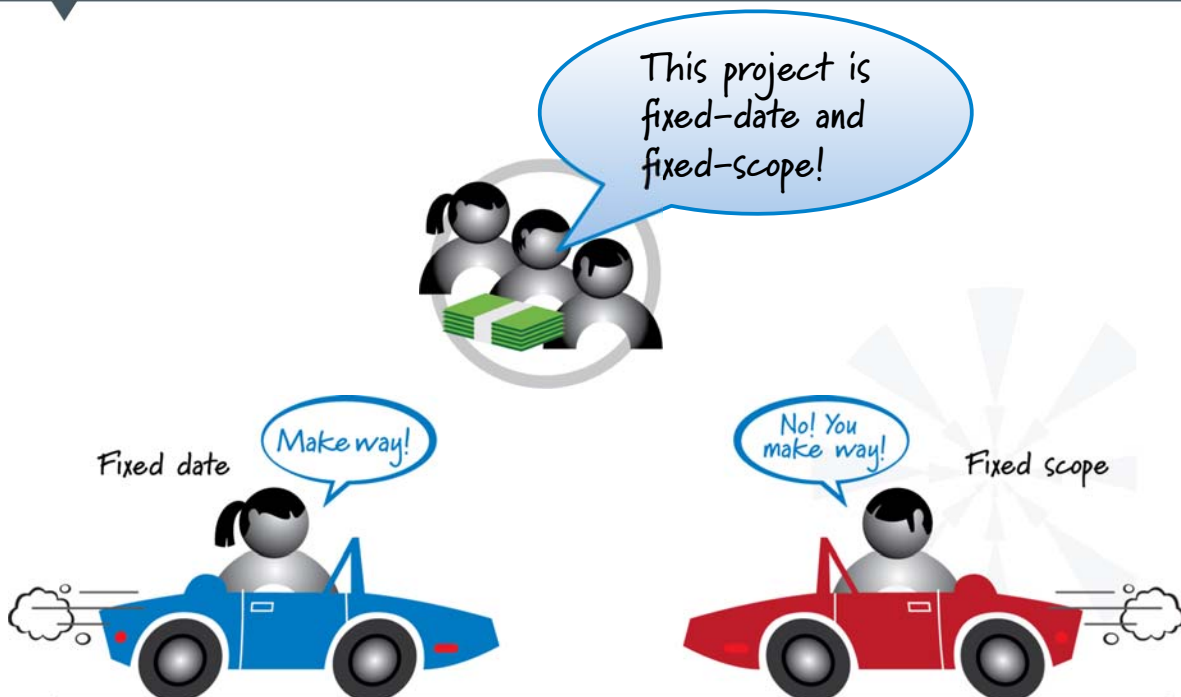
Naïve technical debt

Unavoidable technical debt

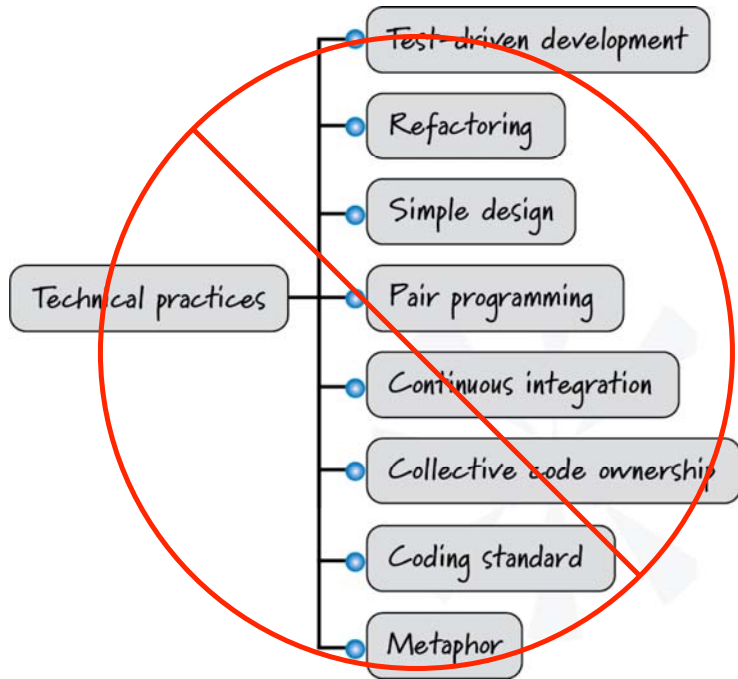
Strategic technical debt



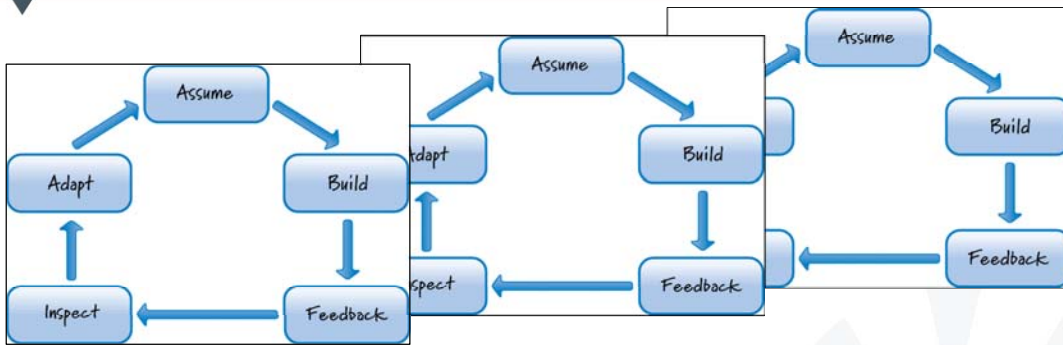
✶ Naïve Technical Debt – Business Immaturity



Naive Technical Debt – Technical Immaturity

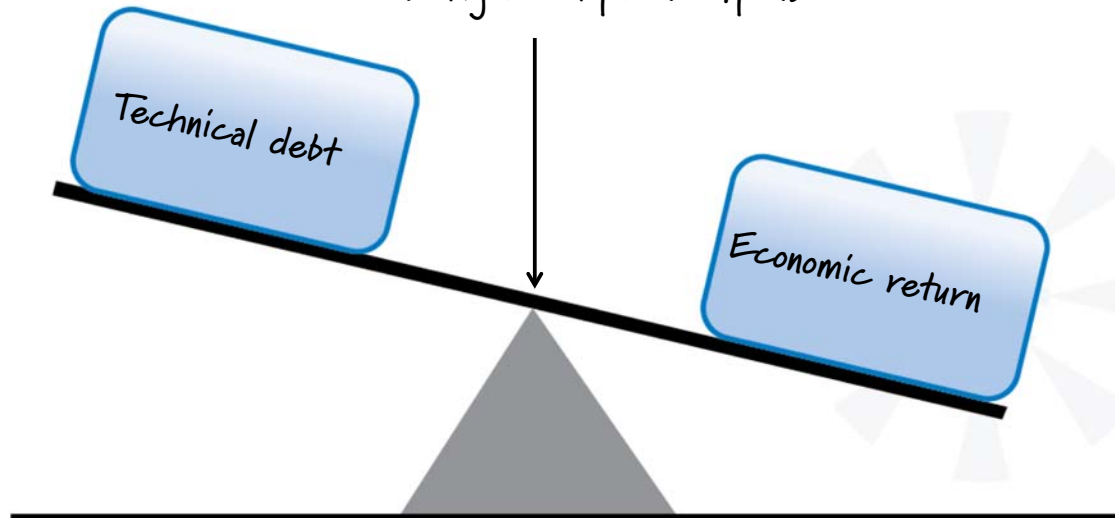


Unavoidable Technical Debt



✱ Strategic Technical Debt

- Shorten time to market
- Preserve startup capital
- Delay development expense



✱ Managing Technical Debt

Managing the accrual of technical debt

Making technical debt visible

Servicing (repaying) technical debt





Question – Managing the Accrual of Technical Debt

What approaches would you use to manage the accrual of technical debt in your organization?



Use Good Technical Practices

Naïve technical debt

Test-driven development

Refactoring

Test automation

Pair programming

Continuous Integration

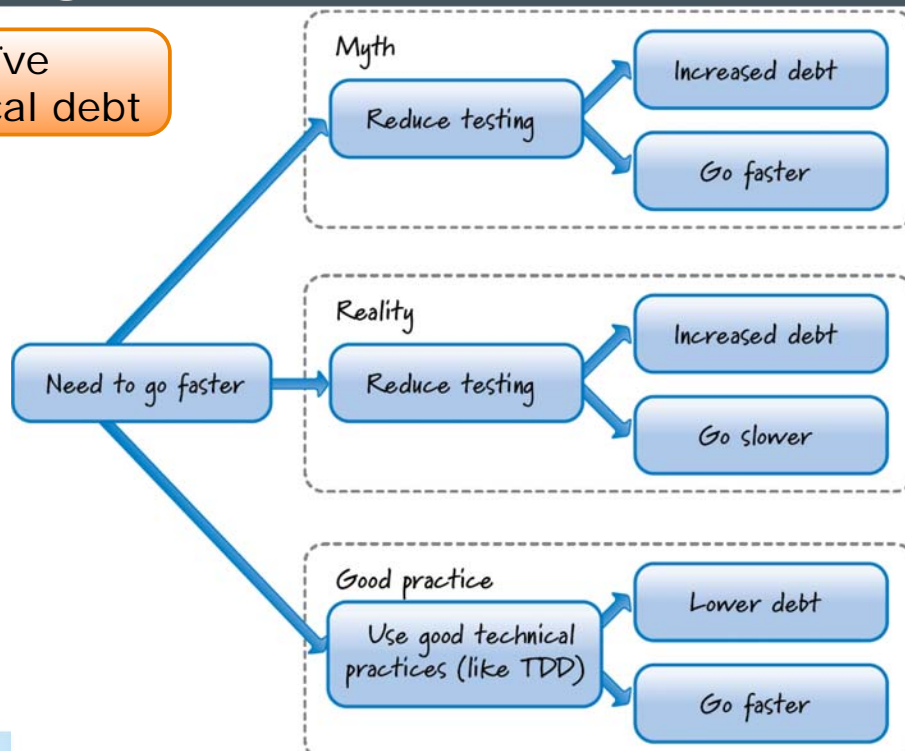
Simple design

Collective code ownership



Myth, Reality, and Good Practice of Testing

Naïve technical debt



Use Strong Definition of Done

Likely to accrue naïve technical debt

- Compiles
- Code is checked in

Less likely to accrue naïve technical debt

- Code commented
- Code reviews
- Code refactored
- Design reviewed
- Automated tests
- Zero known defects
- Acceptance tested

Weak

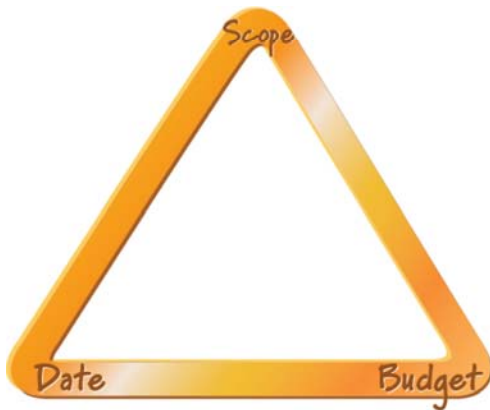
Strong

Definition of Done

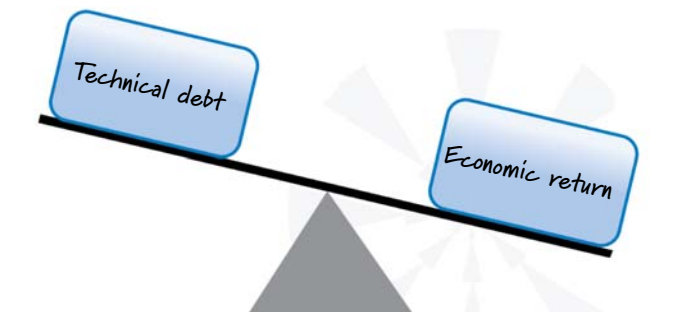


Properly Understand Technical Debt Economics

Naïve technical debt



Strategic technical debt



Example: Strategic Technical Debt

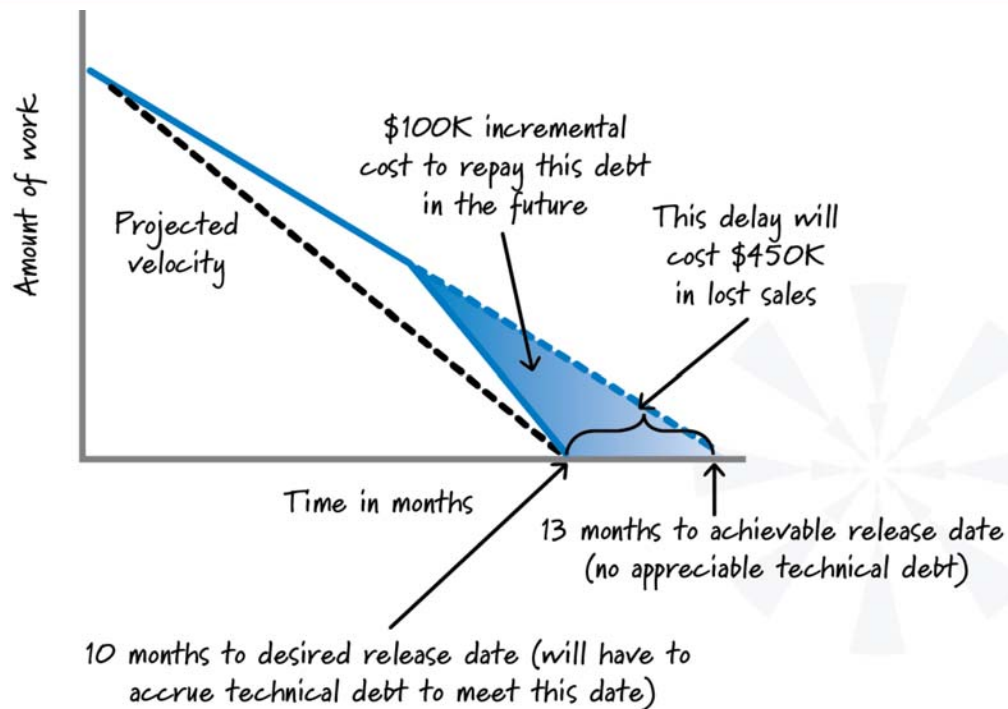
Each month of development costs \$100K

We cannot reasonably meet the target delivery date (at ten months) with all of the requested, must-have features

Dropping features is just not an option



Example: Strategic Technical Debt (Cont)



Example: Strategy Technical Debt (Cont)

	Avoid Debt	Take on Debt
Monthly development cost	\$100K	\$100K
Total development months	13	10
Total development cost	\$1.3M	\$1M
Delay in months (to release product)	3	0
Delay cost per month	\$150K	\$150K
Total delay cost	\$450K	0
Debt-servicing months	0	4
Debt-servicing cost	\$0	\$400K
Total cost in lifecycle profits	\$1.75M	\$1.4M
Delay cost of incremental time to repay debt	\$0	X
Lifetime interest payments on technical debt	\$0	Y
Other debt-related costs	\$0	Z
Real cost in lifecycle profits	\$1.75M	\$1.4M+X+Y+Z





Question – Making Technical Debt Visible

How do you make technical debt visible to the people who need to see it?



Making Technical Debt Visible at the Business Level

Financial reporting

Cost of change

Velocity

Gross quantification

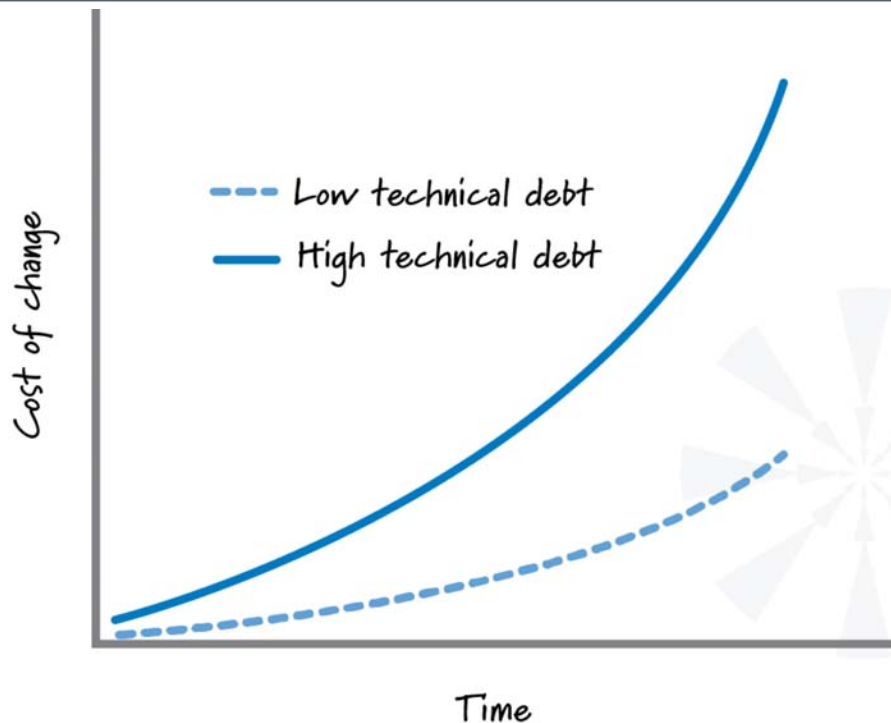


Business Visibility – Financial Reporting (Balance Sheet)

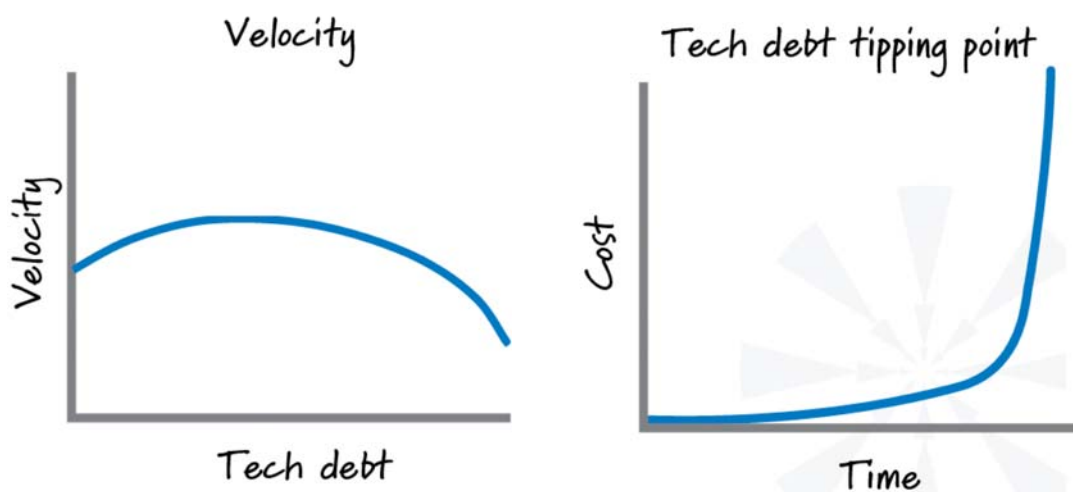
Assets		Liabilities	
Cash	\$600K	Current Liabilities	
Accounts Receivables	\$450K	Notes Payable	\$100K
		Accounts Payable	\$75K
		Short-Term Technical Debt	\$90K
Tools and Equipment	\$250K	Long-Term Liabilities	
		Notes Payable	\$300K
		Long-Term Technical Debt	\$650K
...



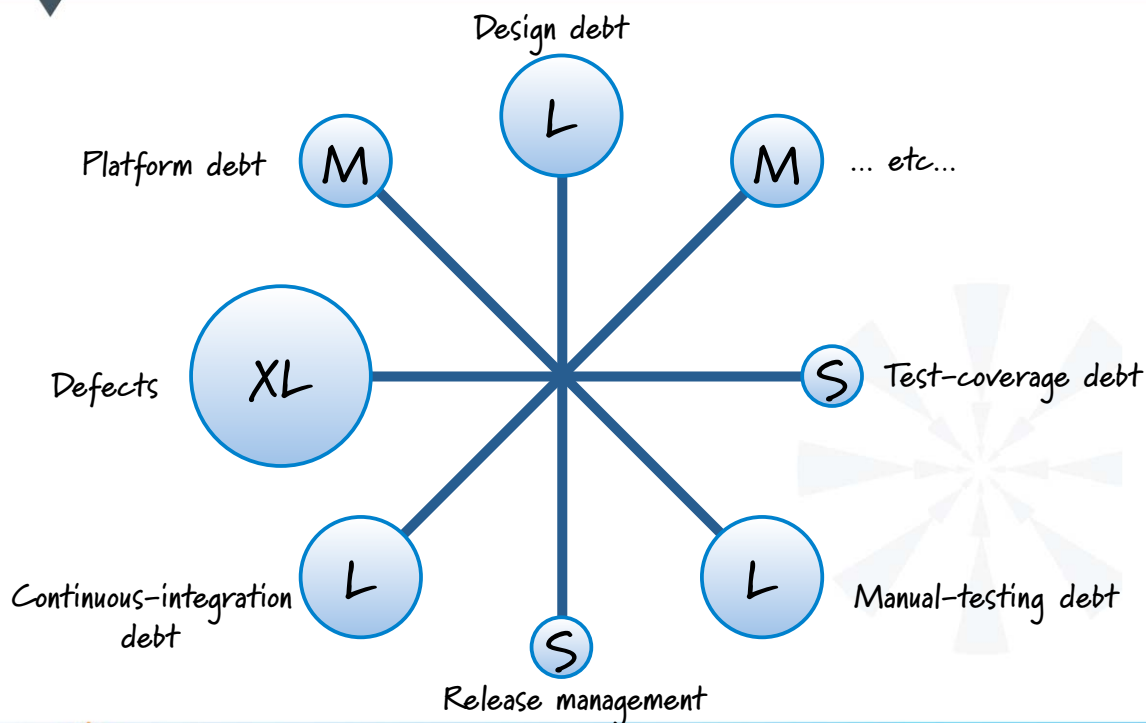
Business Visibility – Cost of Change



Business Visibility – Velocity (Cost)

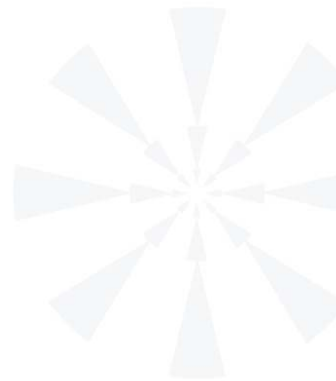
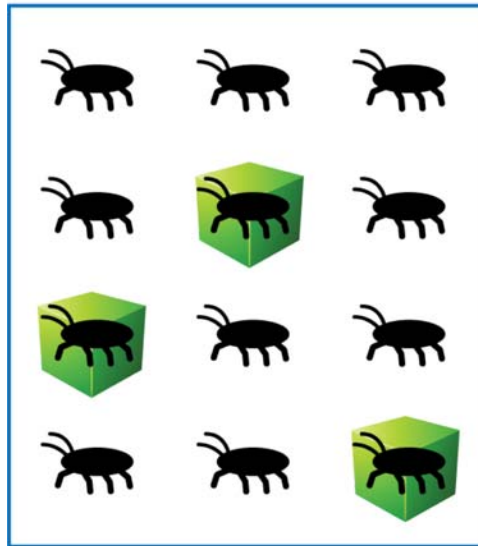


Business Visibility – T-shirt Sizes

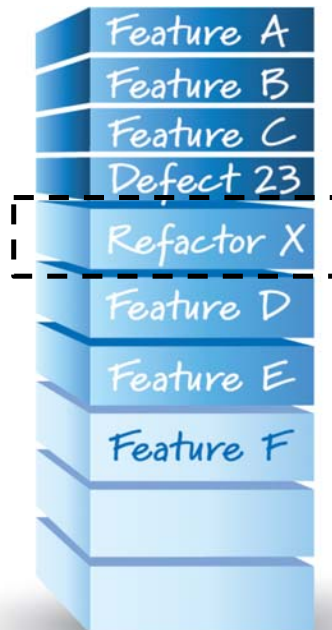


Technical Visibility – Defect Tracking System

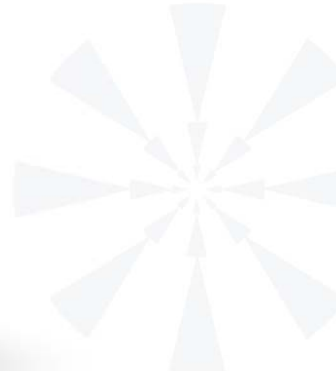
Technical debt in defect-tracking system



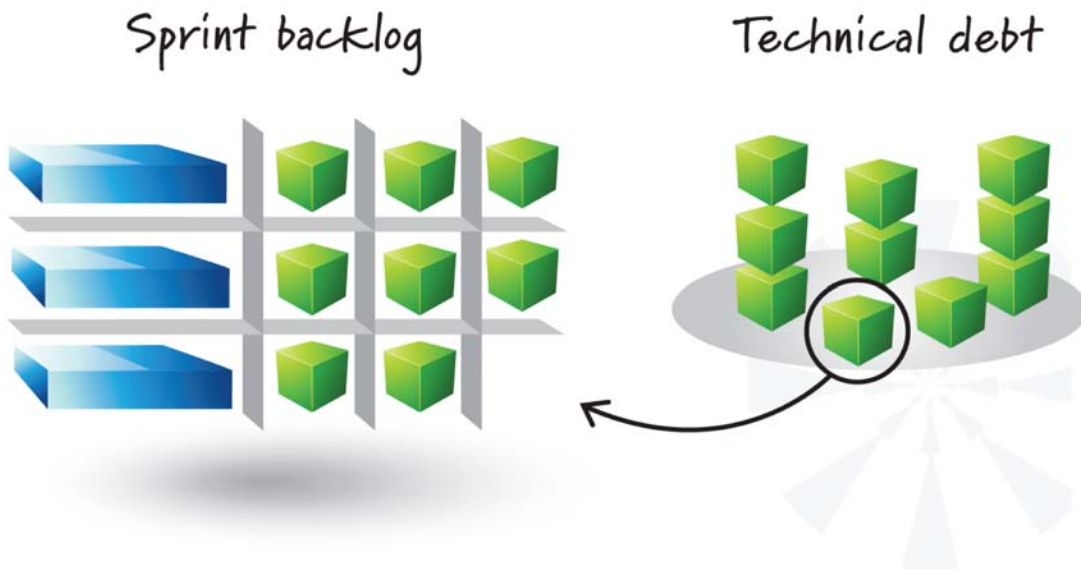
Technical Visibility – Product Backlog



Technical debt PBI



Technical Visibility – Technical Debt Backlog





Question – Servicing Technical Debt

What strategies will you use to service your technical debt?



Helpful Terminology

Type of Tech Debt	Description
Happened-upon	Debt that the development team was unaware existed until it was exposed during the normal course of performing work on the product
Known	Debt that is known to the development team and has been made visible
Targeted	Debt that is known and has been targeted for servicing by the development team





Basic Algorithm for Managing Technical Debt

1. Determine if the known technical debt should be serviced. If it should be serviced, go to step 2.
2. If you are in the code doing work:
 1. And you discover happened-upon technical debt, clean it up.
 2. If the amount of happened-upon technical debt exceeds some reasonable threshold, clean it up until you reach that threshold.
 3. Classify the non-serviced, happened-upon technical debt as known technical debt.
3. Every sprint:
 1. Consider designating some amount of known technical debt as targeted technical debt to be serviced during the sprint.
 2. Favor servicing known technical debt with a high interest rate that is aligned with customer-valuable work.



Approaches for Servicing Technical Debt

Not all technical debt should be repaid

*Apply the Boy Scout rule
(service debt when you happen upon it)*

Repay high-interest technical debt first

Repay technical debt incrementally

Repay technical debt while performing customer-valuable work





Not All Technical Debt Should Be Repaid

We don't repay when the economics don't justify doing so

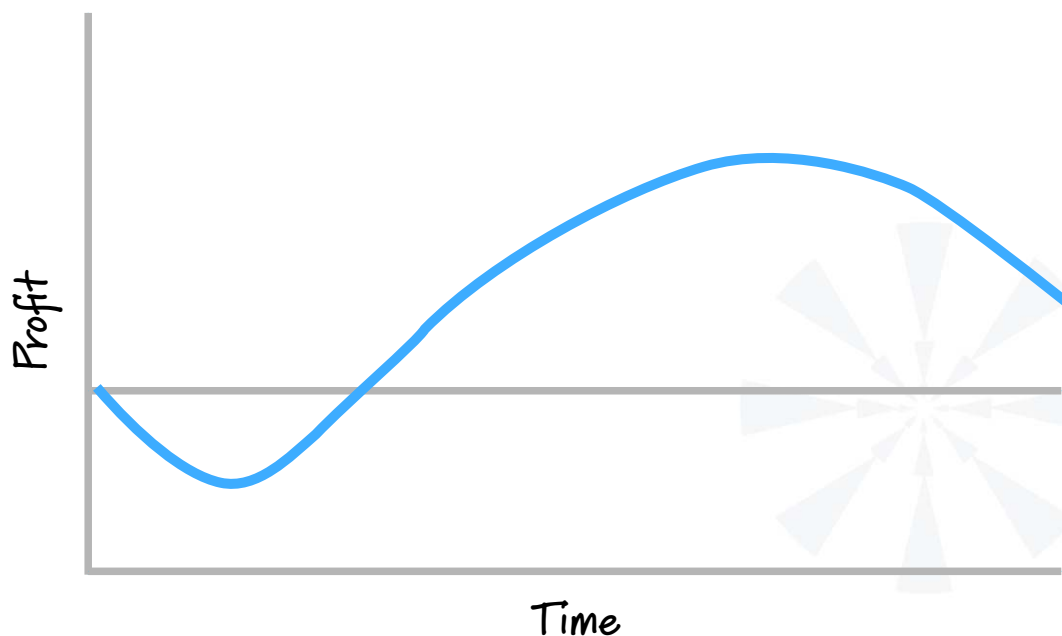
Product nearing end of life

Throwaway prototype

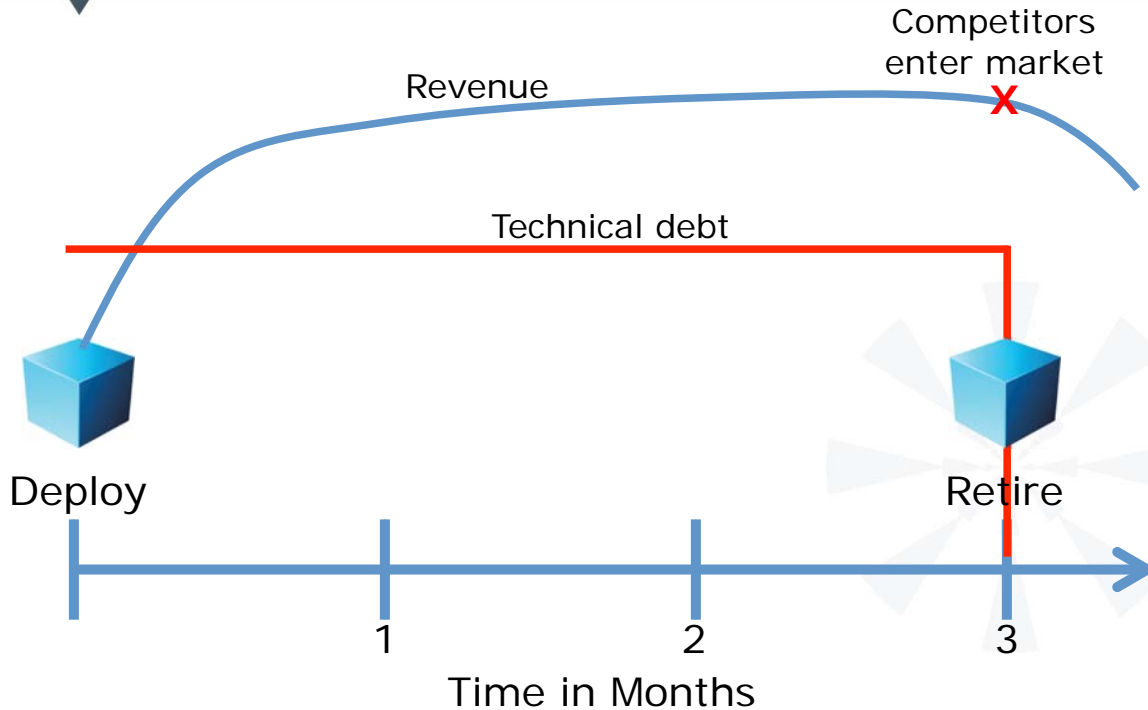
Product built for a short life



Don't Repay Near End of Life



Don't Repay when Built for a Short Life



Don't Repay with Throwaway Prototype

Knowledge acquisition

Filtering Engine Architecture Eval

As a developer I want to prototype two alternatives for the new filtering engine so that I know which is a better long-term choice.

Conditions of Satisfaction

Run speed test on both prototypes.
Run scale test on both prototypes.
Run type test on both prototypes.
Write short memo describing experiments, results, and recommendations.

Throwaway Prototype



Apply the Boy Scout Rule

"Always leave the campground cleaner than you found it"



Remove happened-upon technical debt up to some reasonable threshold

Record any happened-upon technical debt that isn't serviced as known technical debt

Repay Technical Debt Incrementally

Period	Payment	Interest	Principal	Balance
0				\$200,000.00
1	\$1,297.20	\$ 1,125.00	\$172.20	\$199,827.80
2	\$1,297.20	\$ 1,124.03	\$173.16	\$199,654.64
3	\$1,297.20	\$ 1,123.06	\$174.14	\$199,480.50
4	\$1,297.20	\$ 1,122.08	\$175.12	\$199,305.38
5	\$1,297.20	\$ 1,121.09	\$176.10	\$199,129.28
6	\$1,297.20	\$ 1,120.10	\$177.09	\$198,952.18

Paying back incrementally is like making a monthly mortgage payment



=



Repay the High Interest Rate Technical Debt First

19% interest

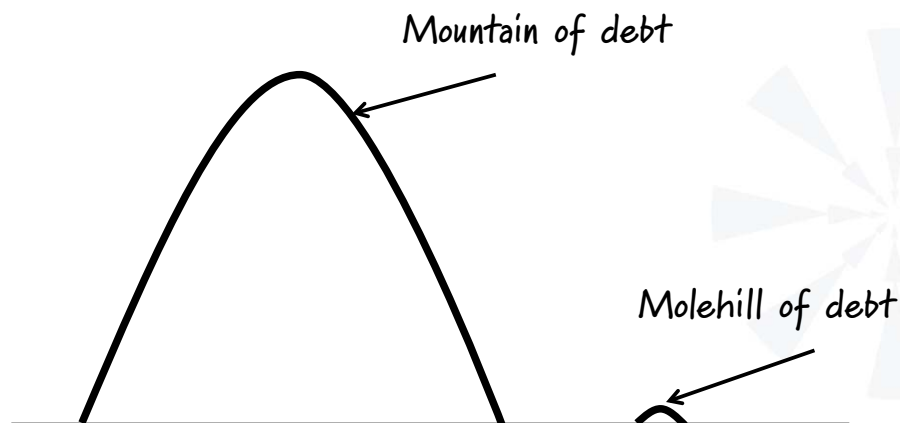


4.5% interest



Repay Smallest Technical Debt First

Prime the pump of debt reduction by repaying smallest/easiest debt first



Repay Debt While Performing Customer Valuable Work



Aligns debt-reduction work with customer-valuable work that PO can properly prioritize

Clarifies technical debt reduction is a shared responsibility and not something to defer and delegate

Reinforces technical debt prevention and removal skills; everyone gets to practice them all the time

Helps identify high-interest areas for debt servicing

Avoids repaying technical debt in unnecessary areas



Summary

Technical debt accrues when we take shortcuts today at tomorrow's expense

Three ways of type of debt: naïve, unavoidable, and strategic

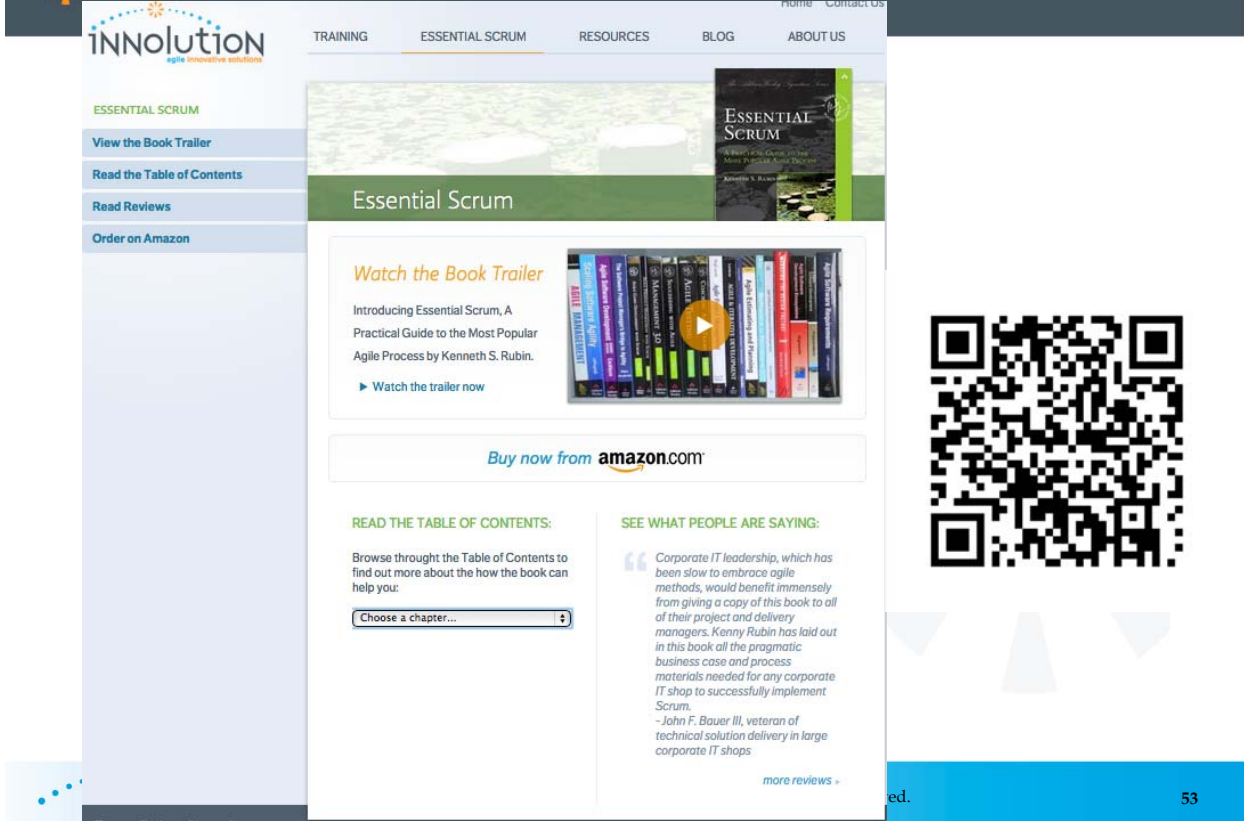
There are many consequences of poorly managed levels of technical debt

Managing accrual of technical debt

Making technical debt visible

Servicing technical debt





The screenshot shows the website for 'Essential Scrum'. The navigation bar includes 'TRAINING', 'ESSENTIAL SCRUM', 'RESOURCES', 'BLOG', and 'ABOUT US'. The main content area features a book cover for 'Essential Scrum' and a 'Watch the Book Trailer' section with a play button icon. Below this is a 'Buy now from amazon.com' button. There are also sections for 'READ THE TABLE OF CONTENTS' and 'SEE WHAT PEOPLE ARE SAYING' with a quote from John F. Bauer III. A QR code is located on the right side of the page.

Contact Info for Kenny Rubin



Email:	krubin@innolution.com
Website:	www.innolution.com
Phone:	(303) 827-3333
LinkedIn:	www.linkedin.com/in/kennethrubin
Twitter:	www.twitter.com/krubinagile
Essential Scrum: A Practical Guide to the Most Popular Agile Process	www.essentialscrum.com
Comparative Agility Website	www.comparativeagility.com