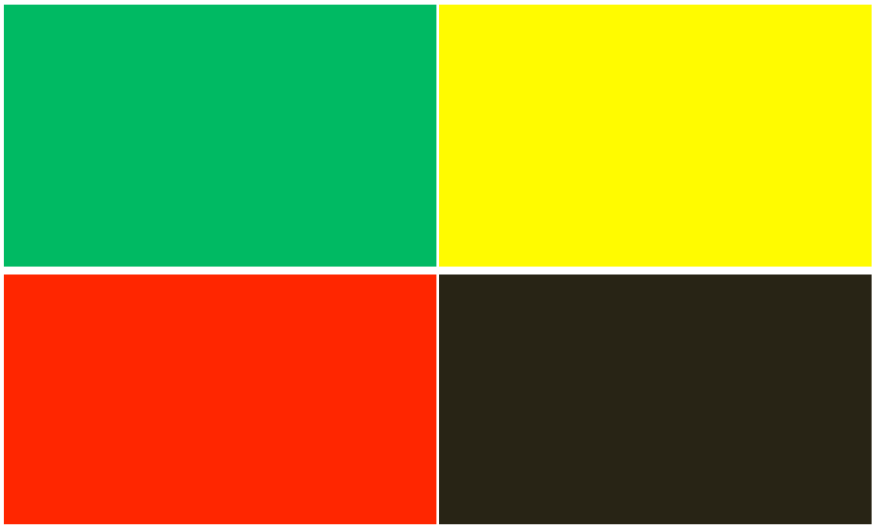


What colours is your backlog?

Philippe Kruchten

@pbpk

Philippe Kruchten, Ph.D., P.Eng., CSDP

Professor of Software Engineering
NSERC Chair in Design Engineering
Department of Electrical and Computer Engineering
University of British Columbia
Vancouver, BC Canada
pbk@ece.ubc.ca

Founder and president
Kruchten Engineering Services Ltd
Vancouver, BC Canada
philippe@kruchten.com

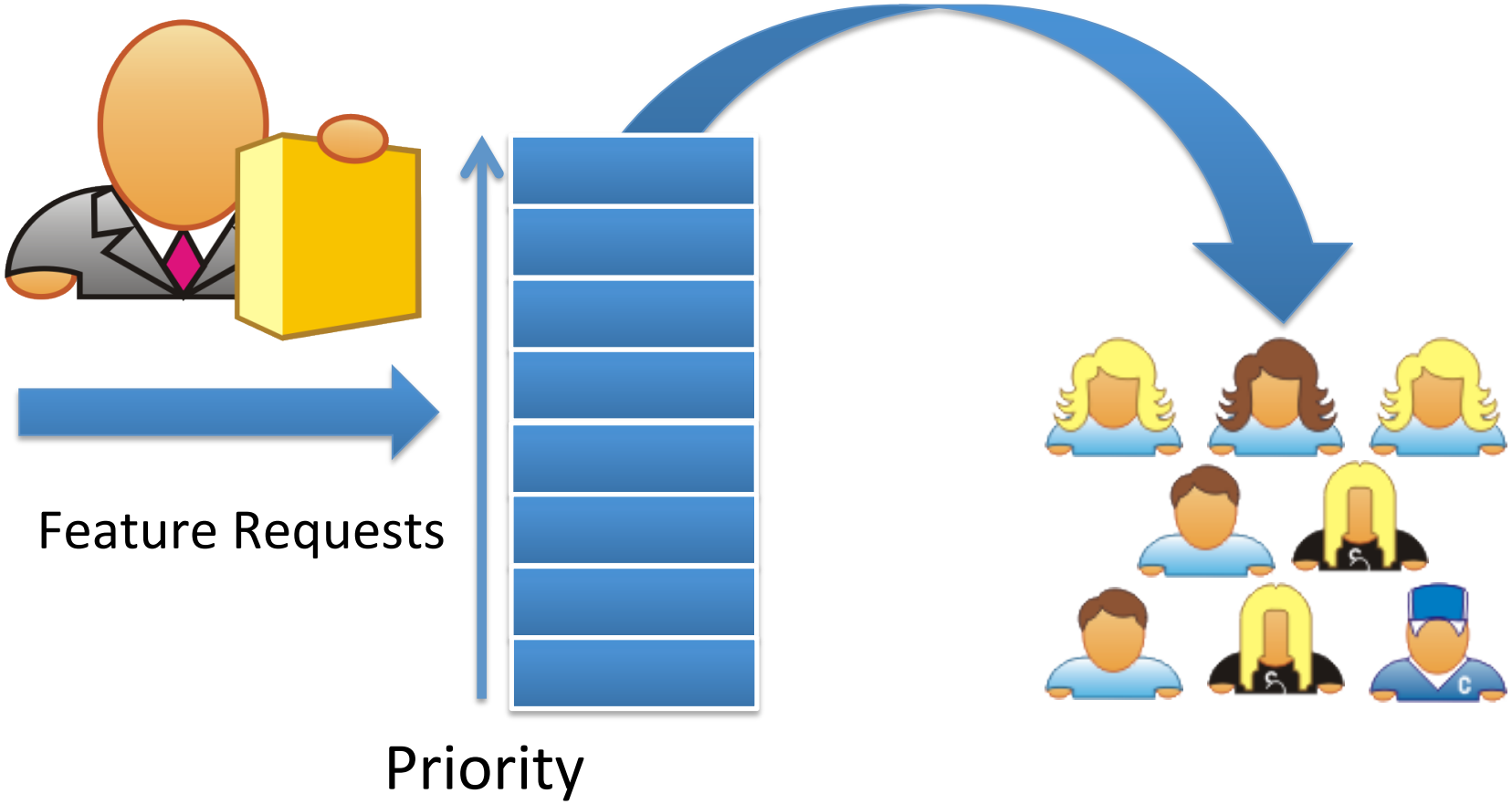
Agile Vancouver

IEEE

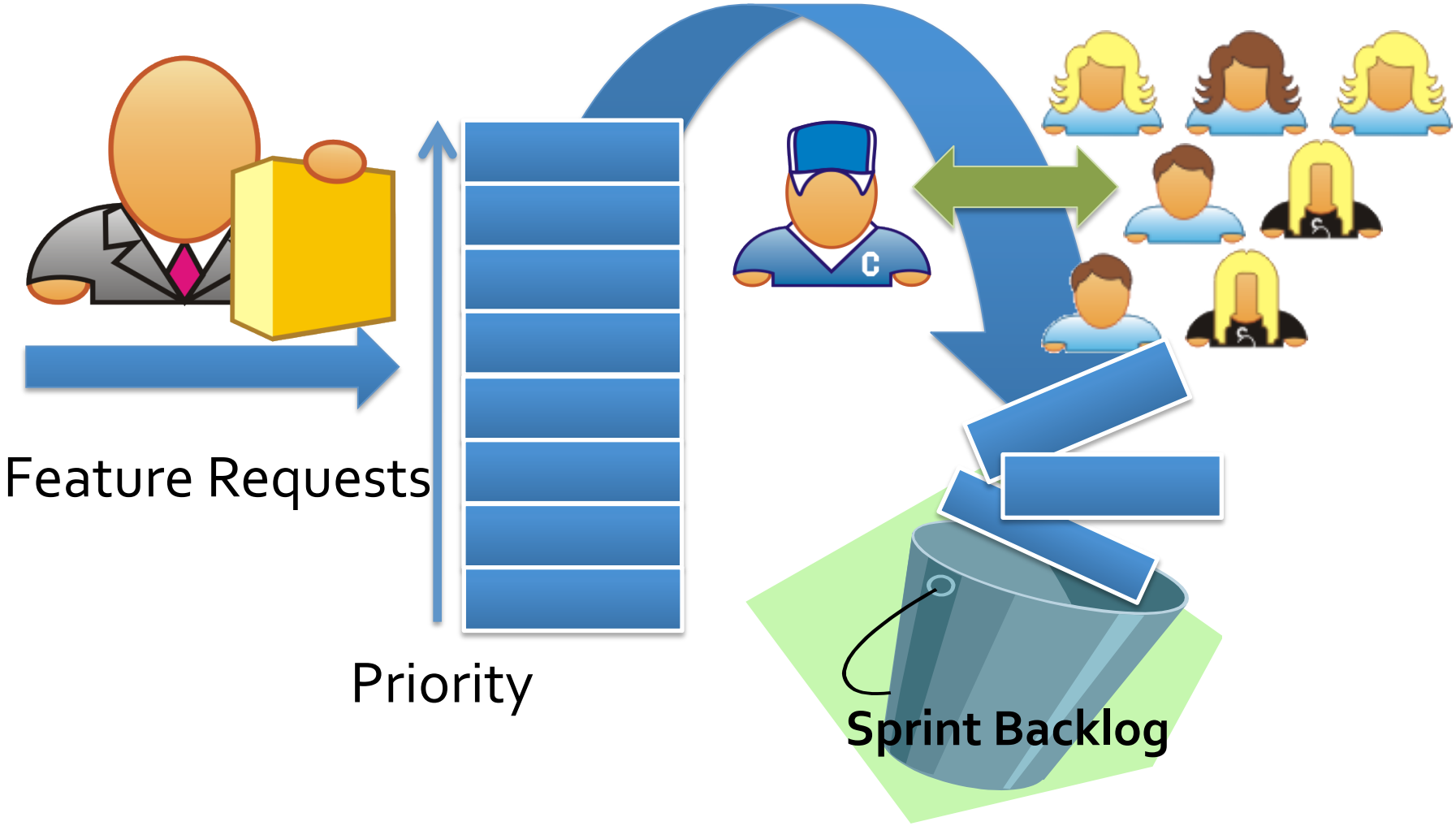
42

ifip

The Backlog

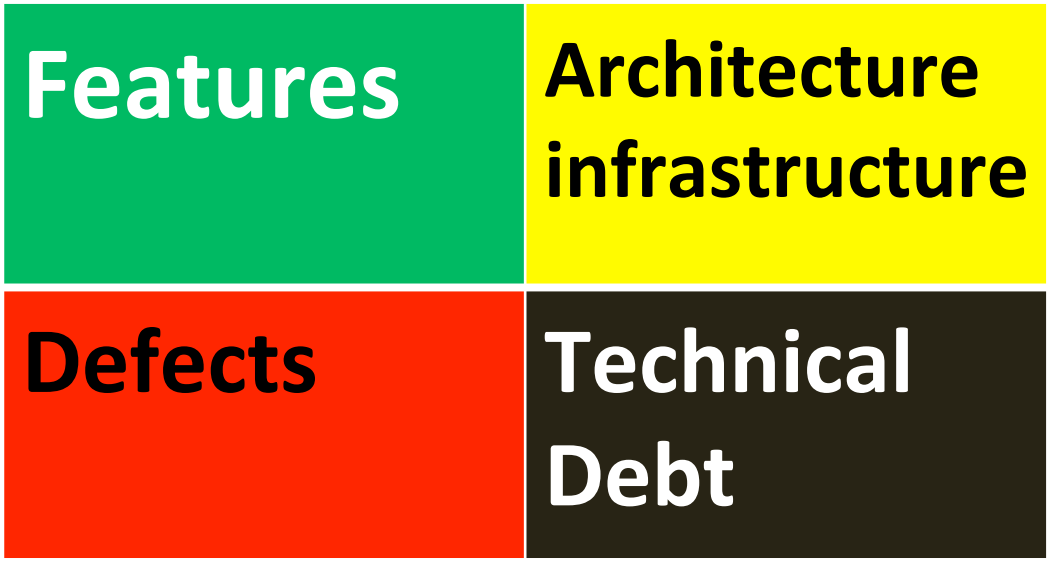


Sprint Planning



But what is in your backlog ?

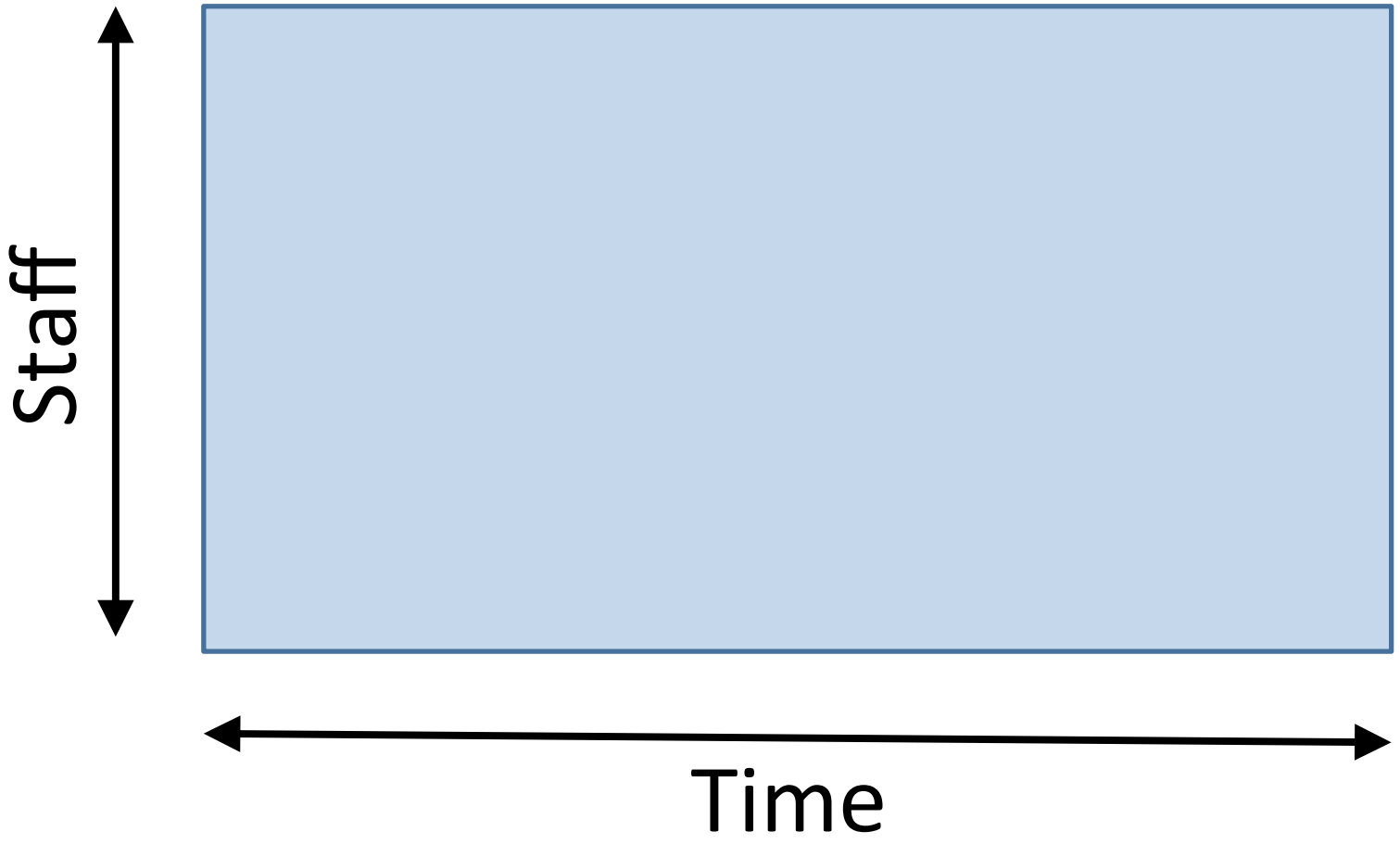
Painting your backlog



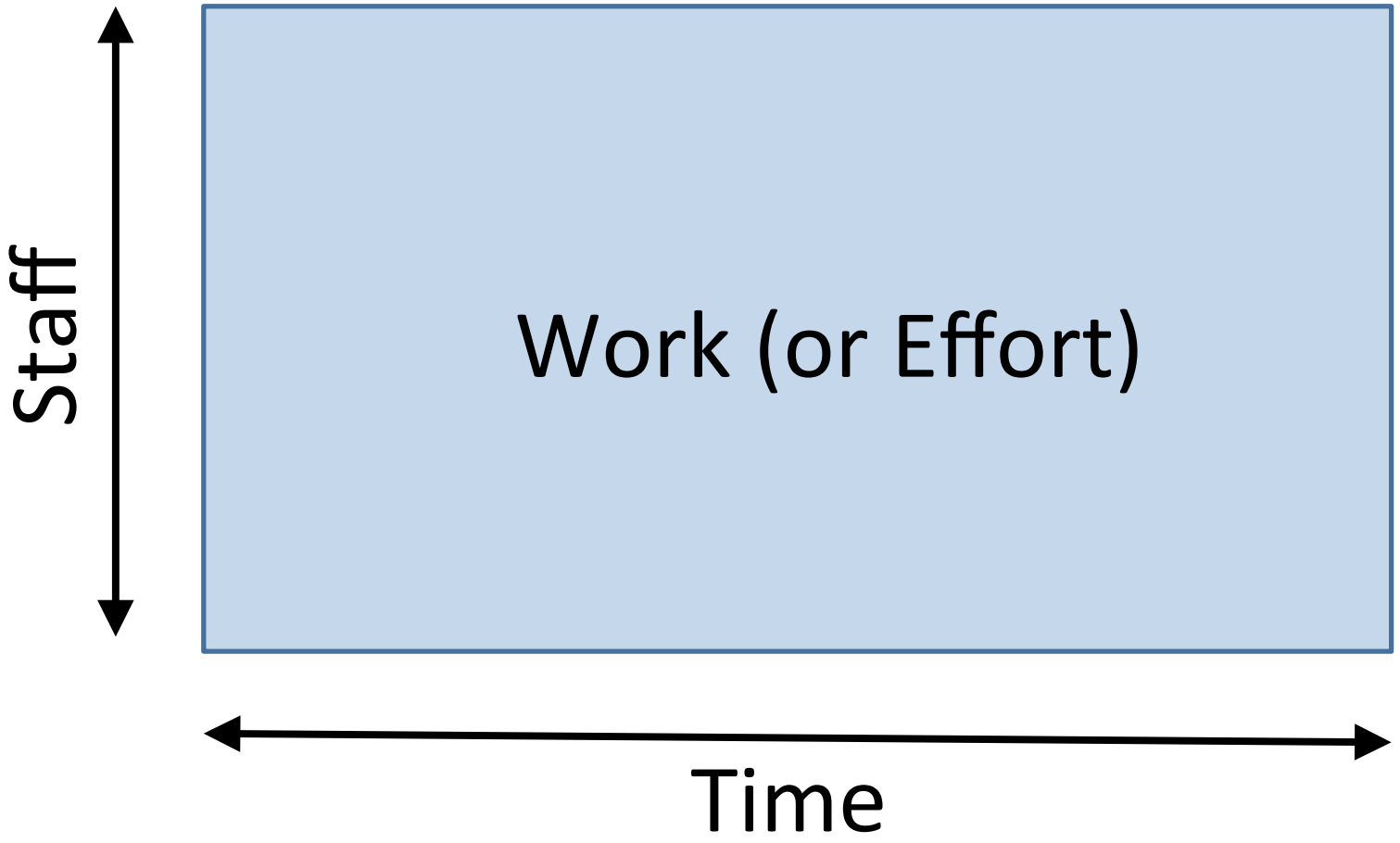
Time-box

DSDM 1995

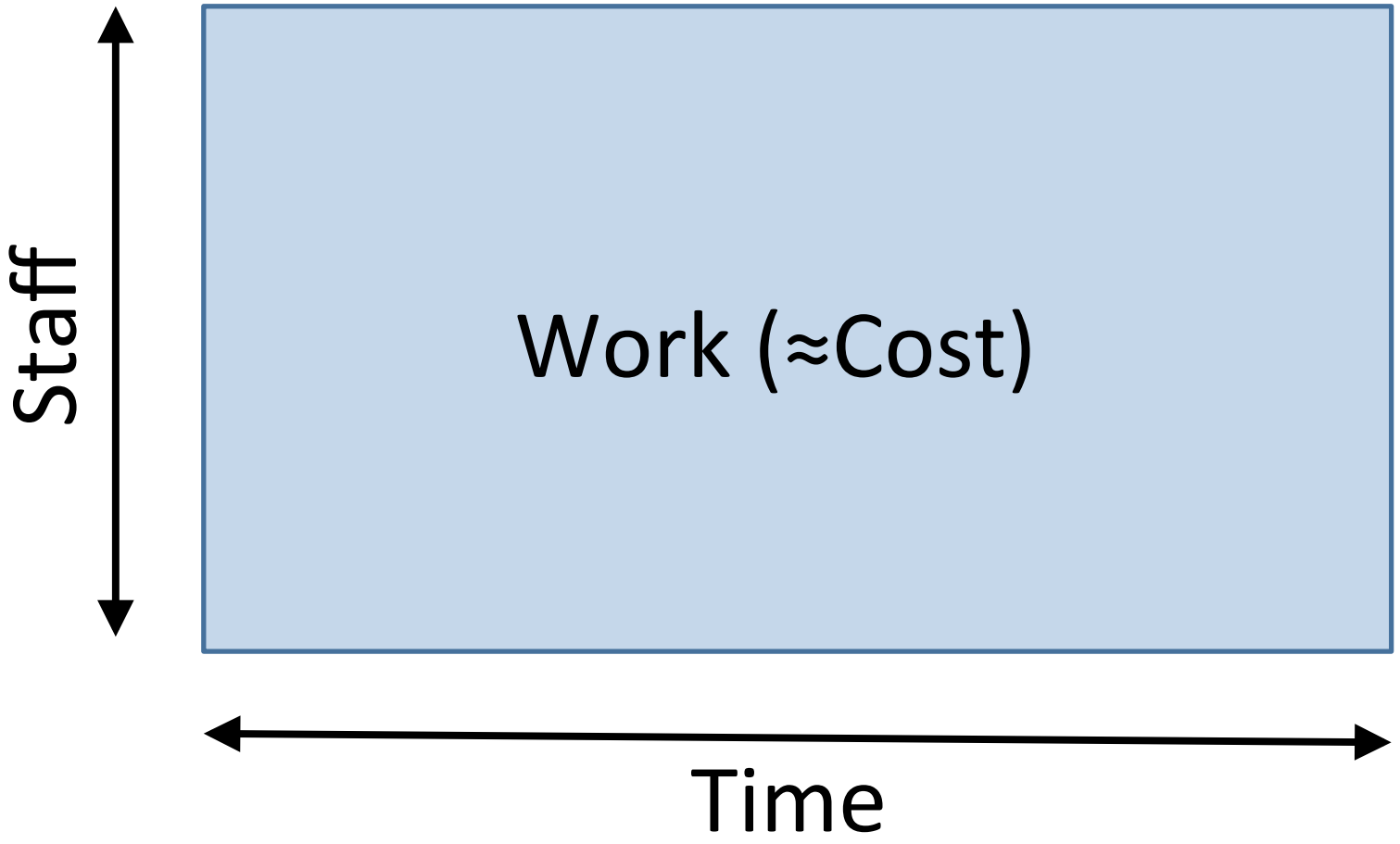
Time-box



Time-box

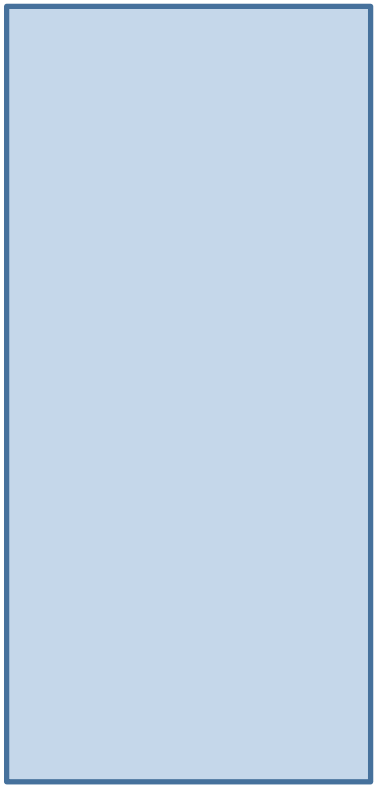


Time-box



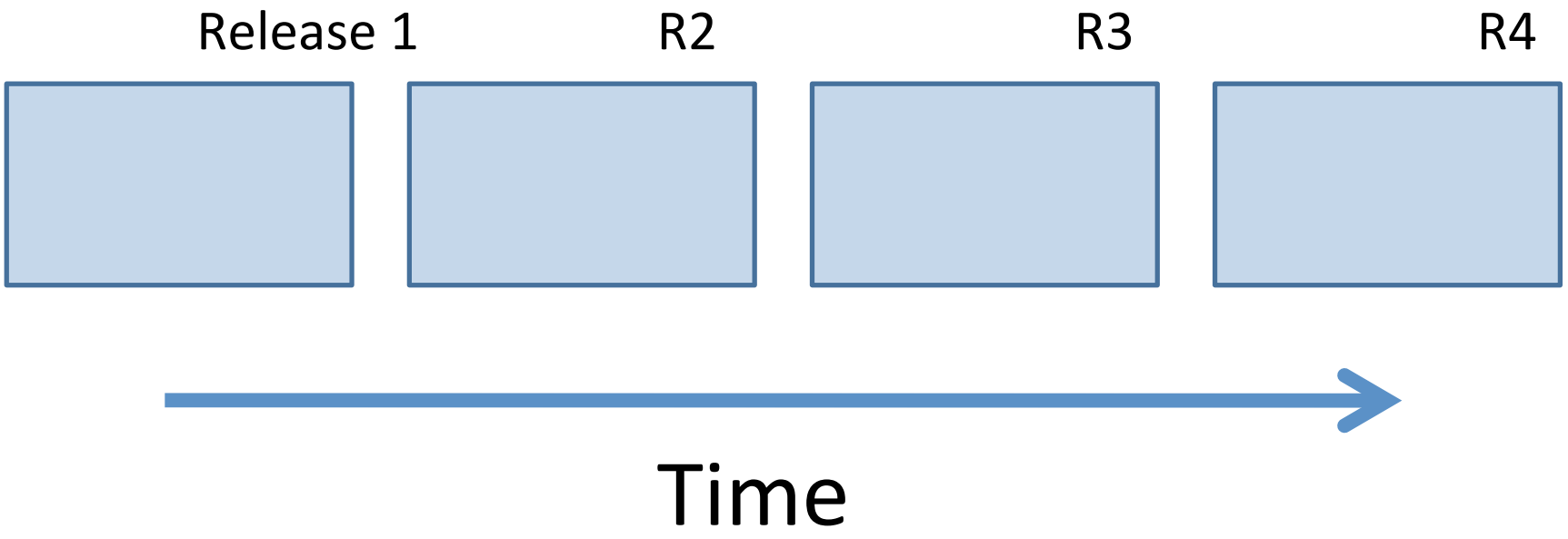
Time-box

better than

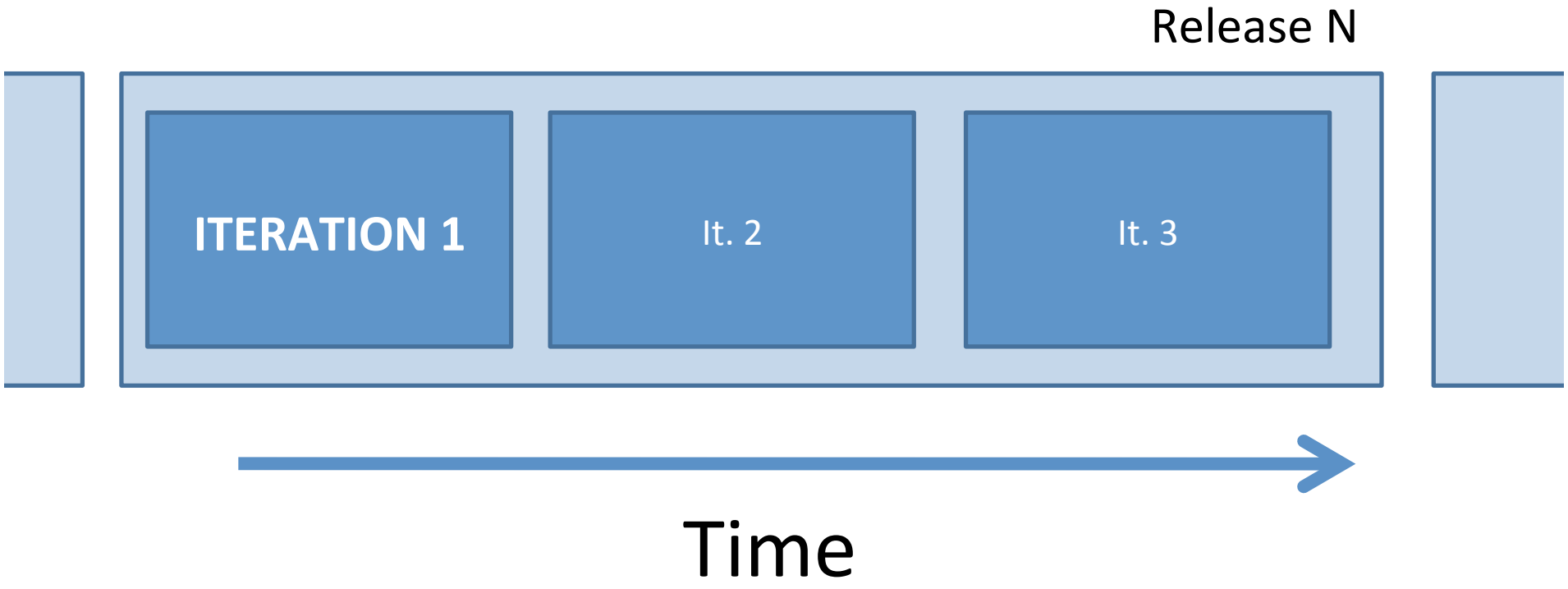


Brooks, Mythical Man-Month, 1975
Boehm, COCOMO, 1981

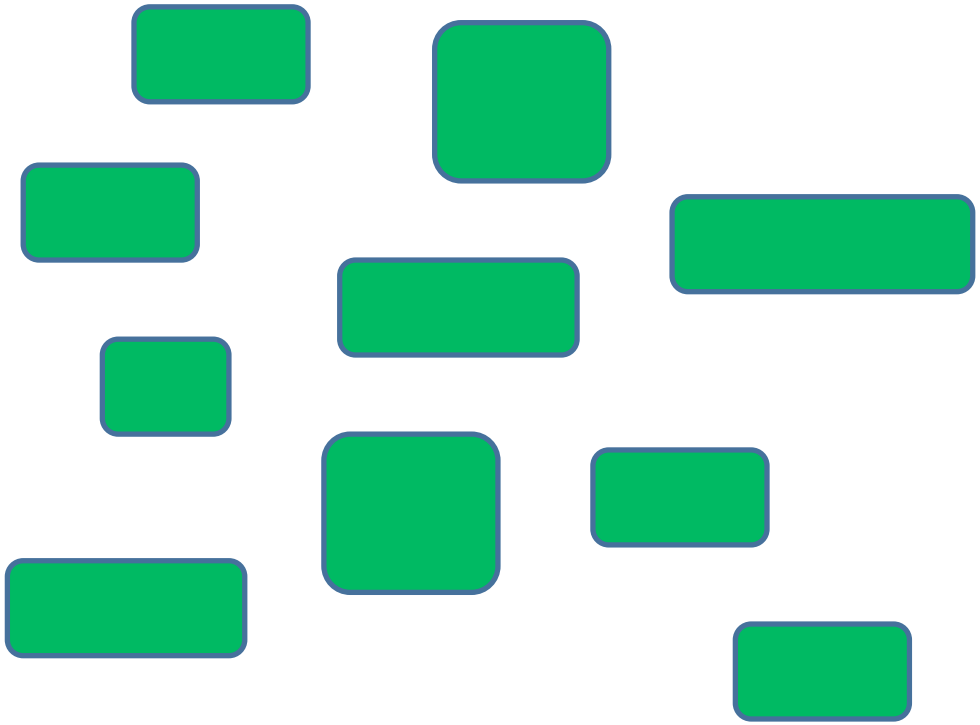
Time-boxes: Releases



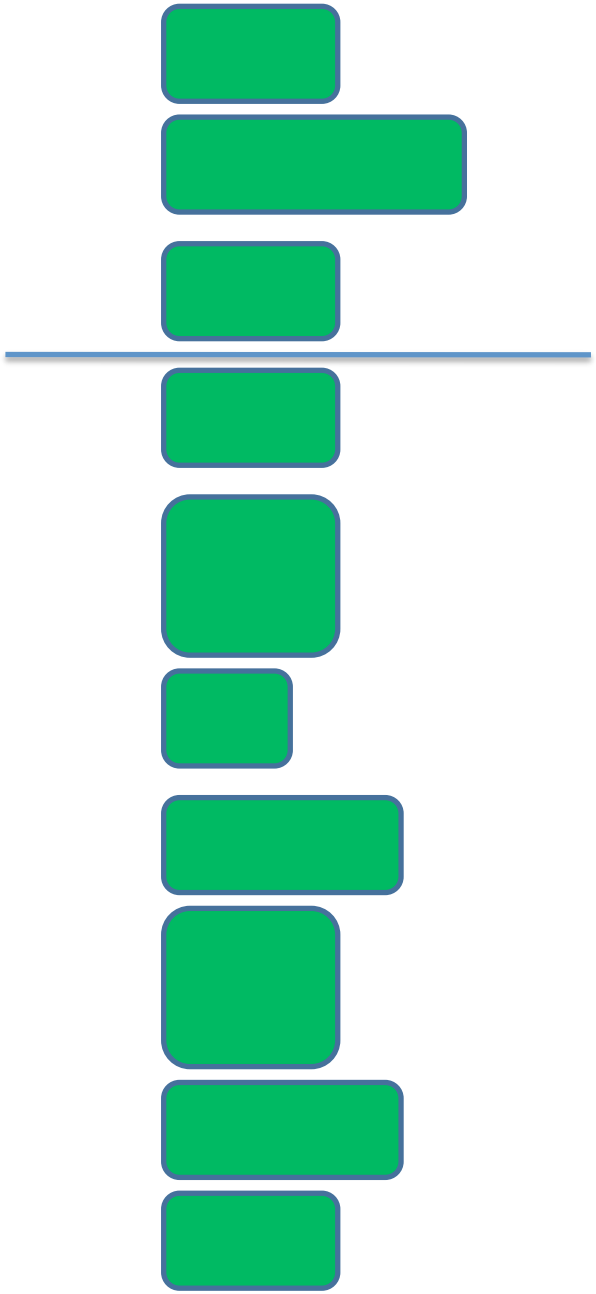
Time-boxes: Iterations (sprints)



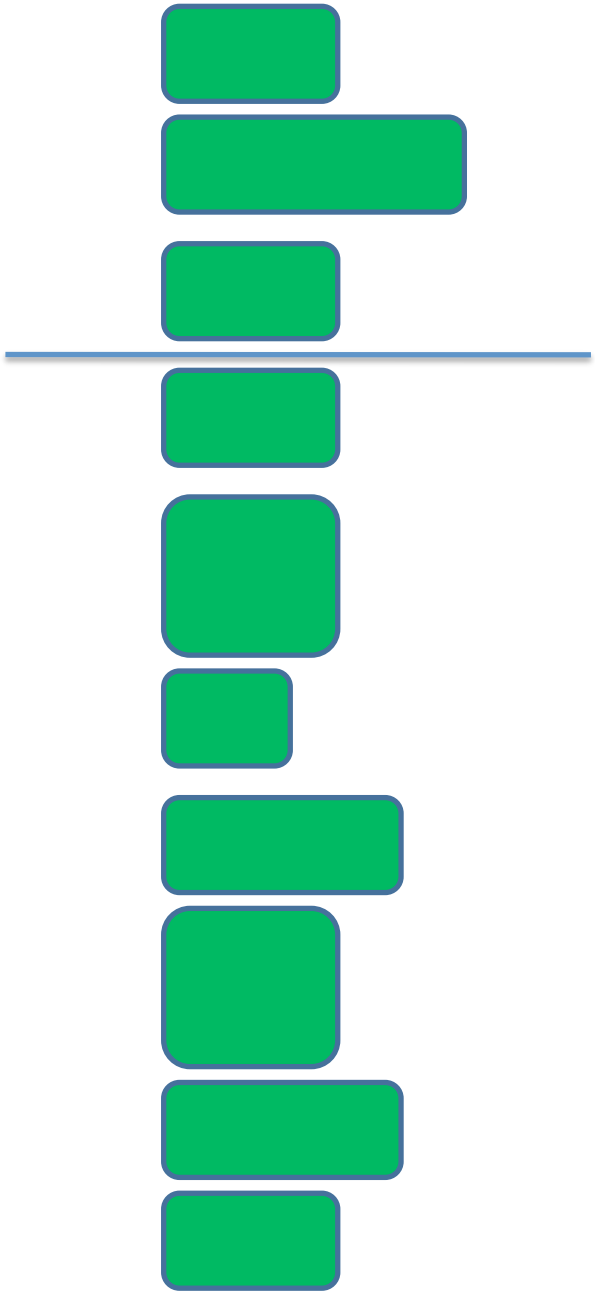
Features



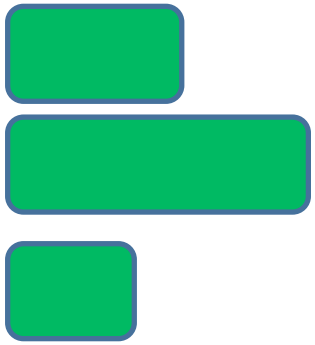
Intent



Features

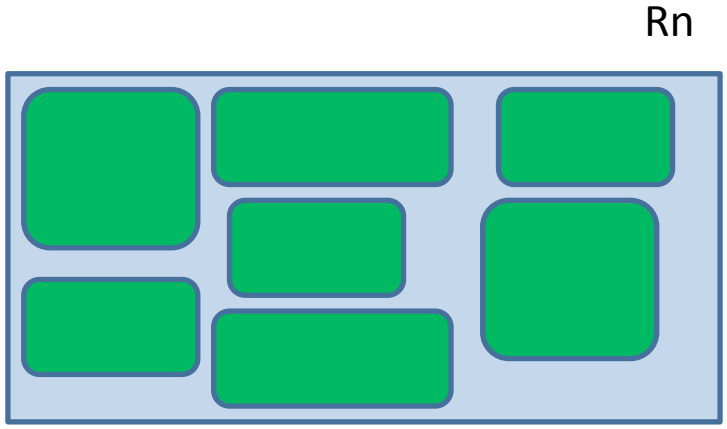


Features

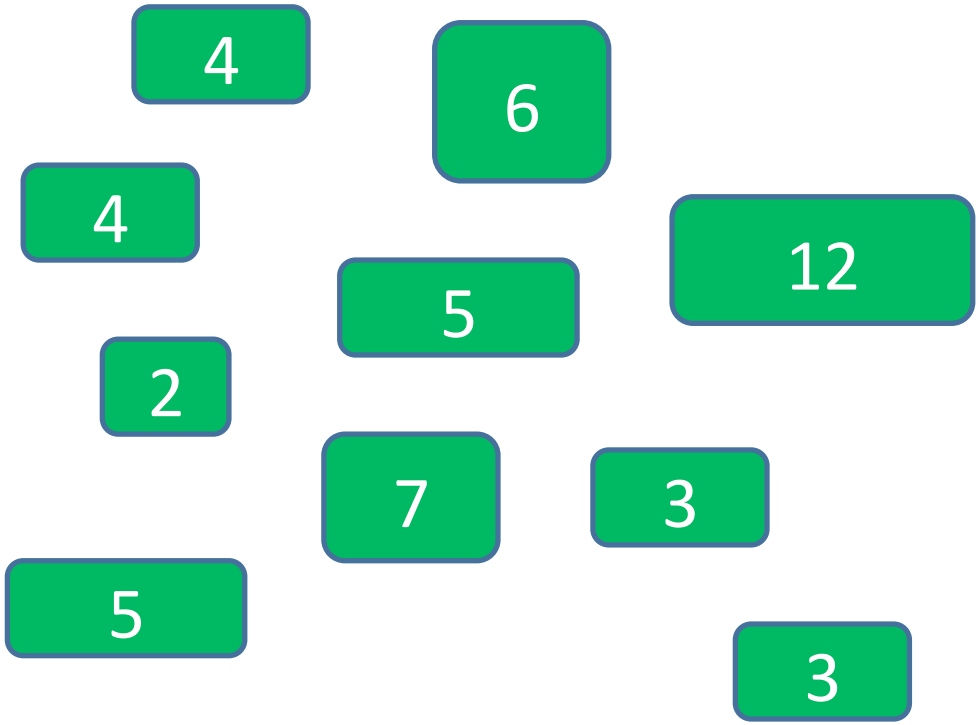


Features

?

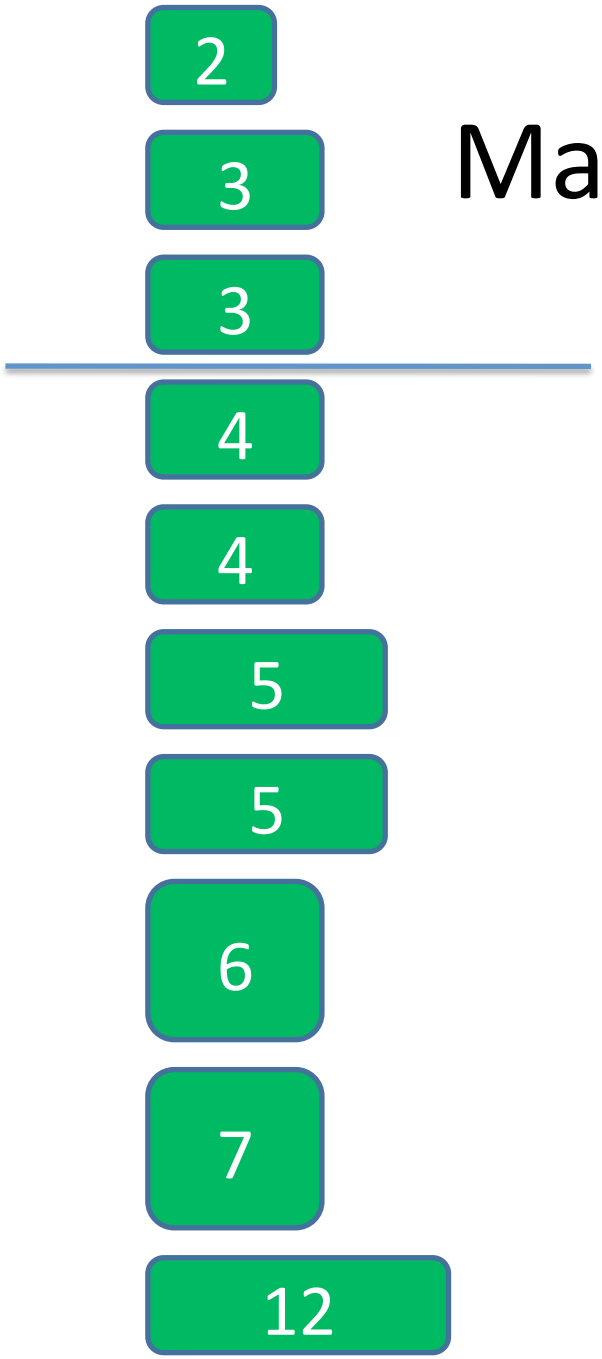


Features & Value



Utils

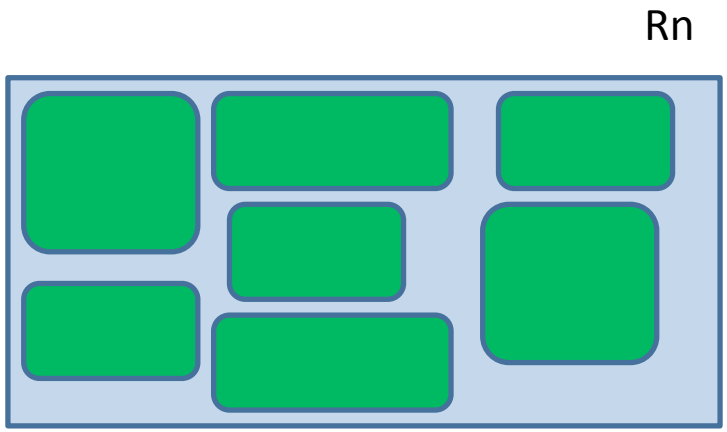
Maximizing value



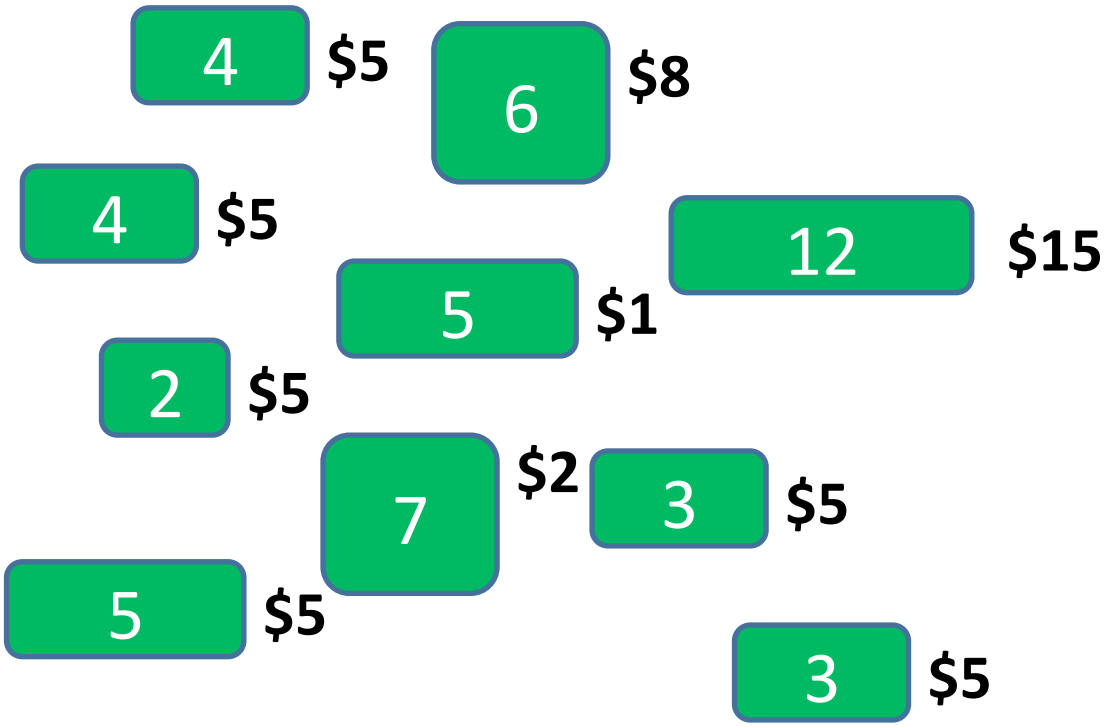
Highest value first
Ignore time

Cost?

?



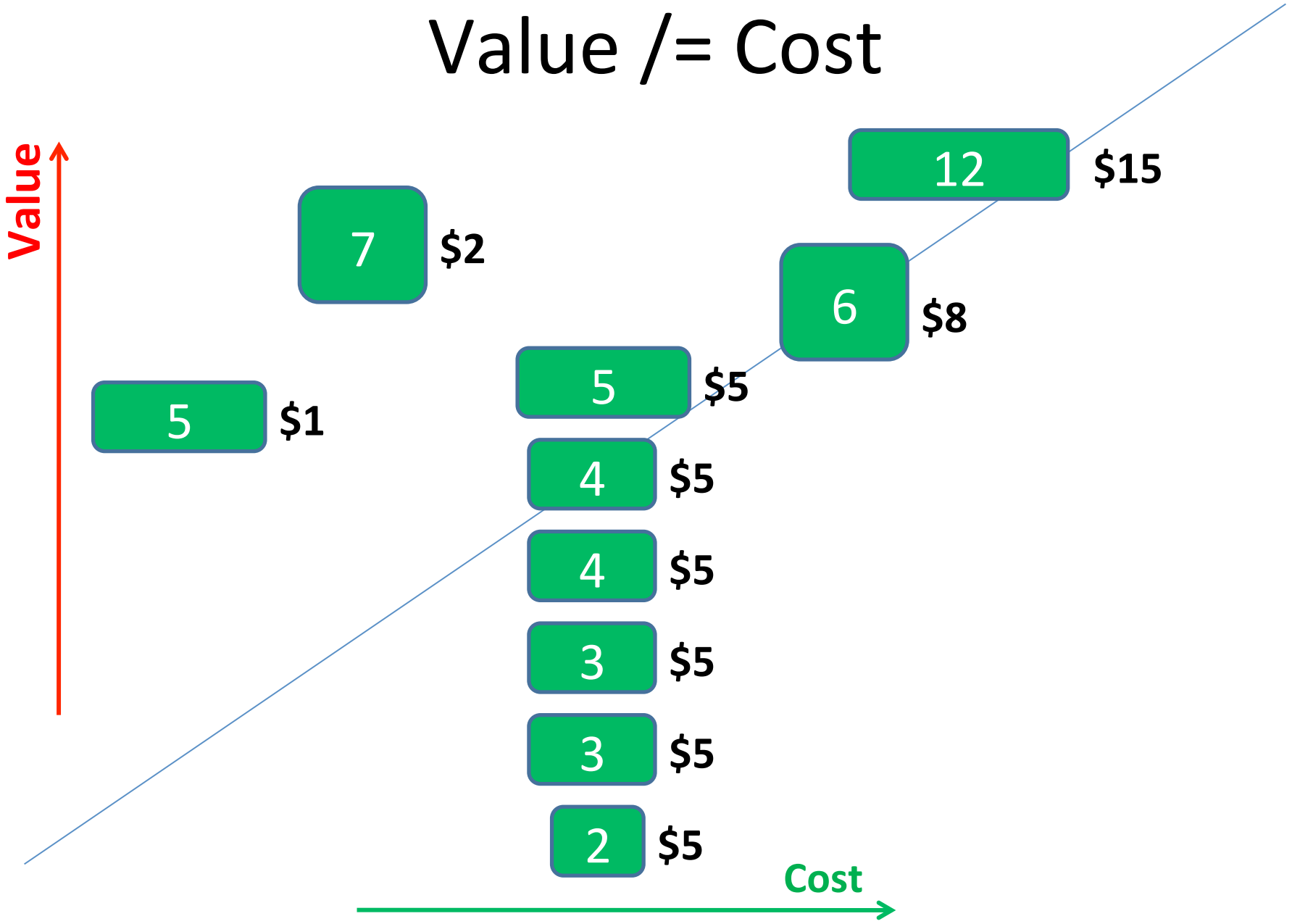
Value = Cost?



Points

Only for simplest cases

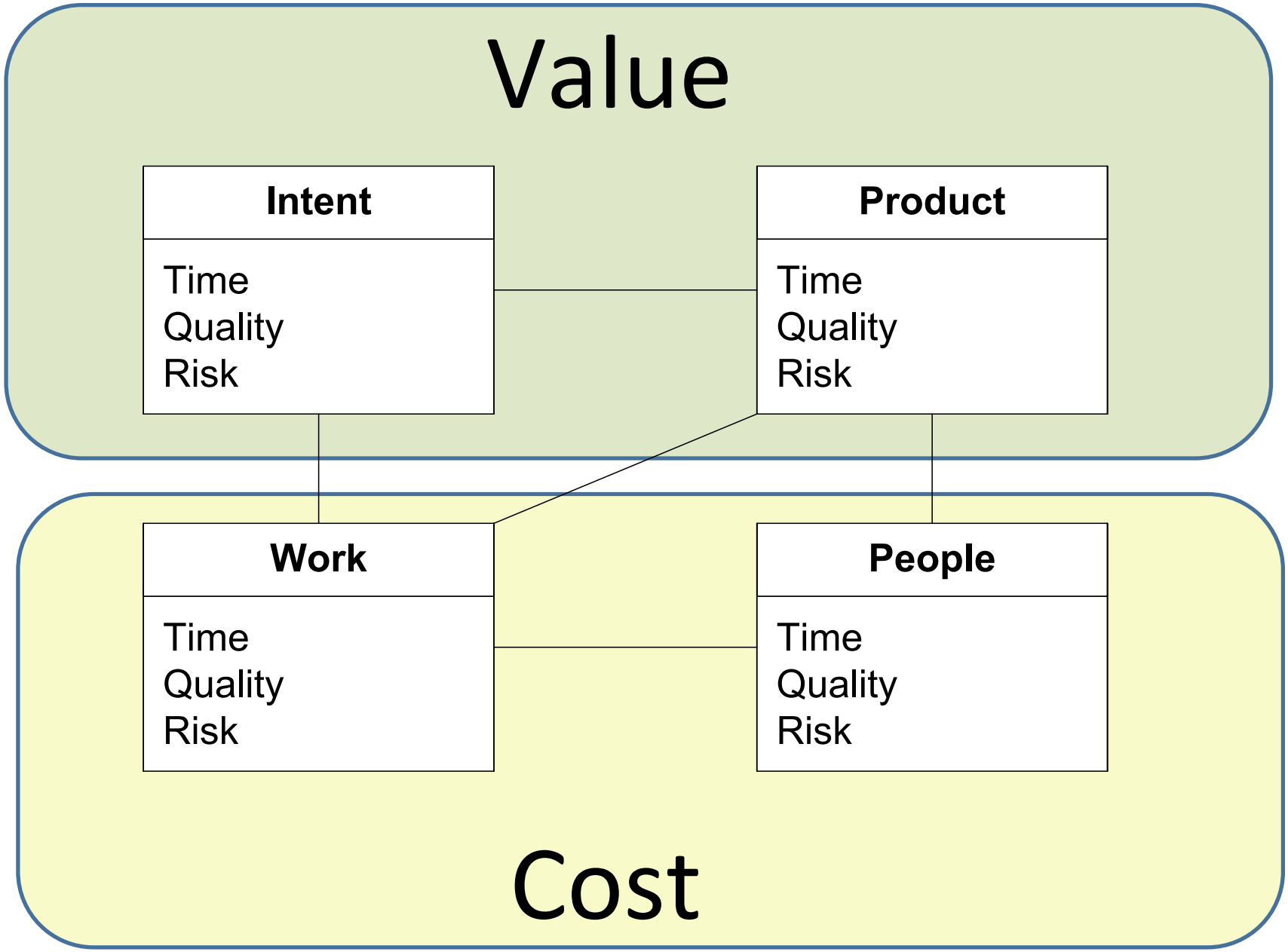
Value /= Cost



Value and Cost

- Value: to the business (the users, the customers, the public, etc.)
- Cost: to design, develop, manufacture, deploy, maintain

- Simple system, stable architecture, many small features:
  - Statistically value aligns to cost
- Large, complex, novel systems ?


Value
Intent
Time
Quality
Risk
Product
Time
Quality
Risk
Work
Time
Quality
Risk
People
Time
Quality
Risk
Cost

Efficiency *vs.* Effectiveness

Efficiency

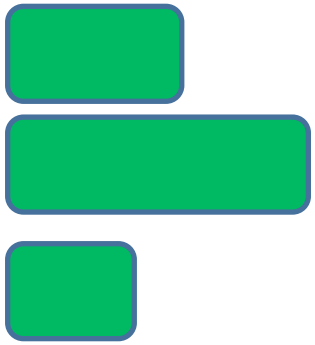
- relationship between the output in terms of goods, services or other results and the resources used to produce them

Effectiveness

- relationship between the intended impact and the actual impact of an activity

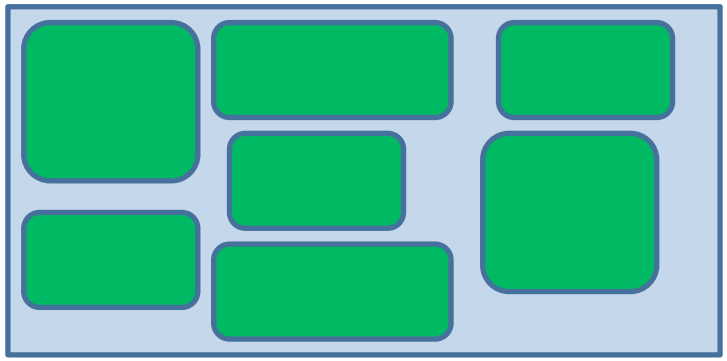
Cost

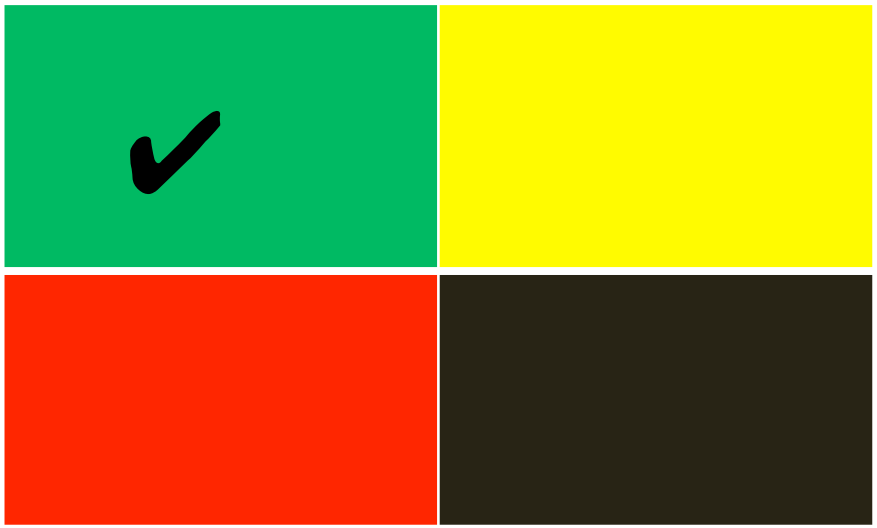
Value



Cost will impose the limit

Rn

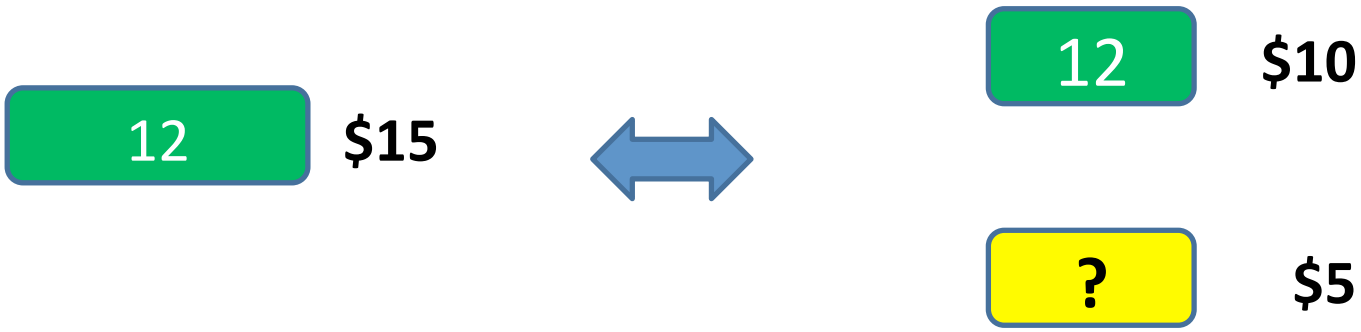




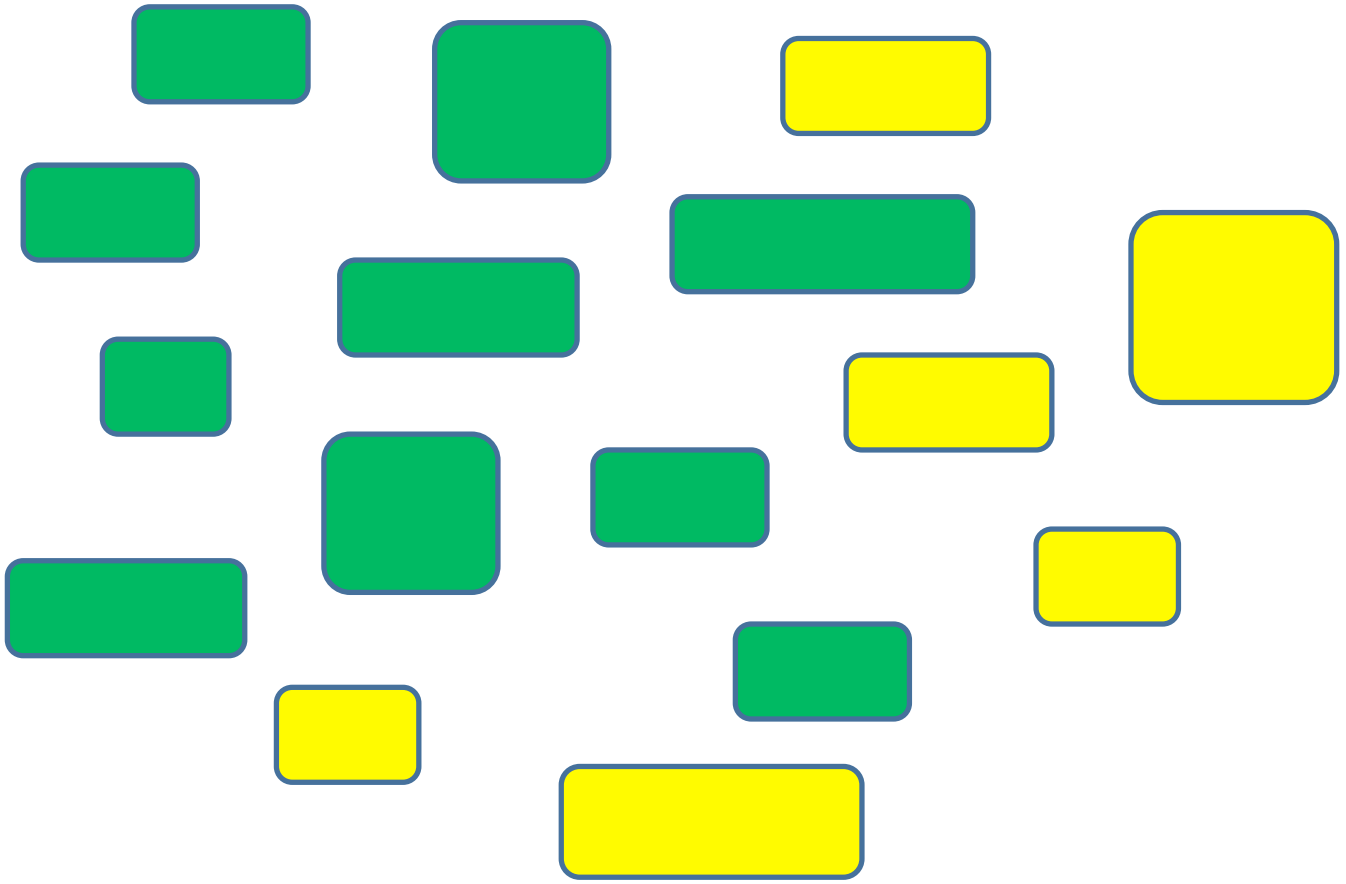
What colour is your backlog?

(so far)

Invisible Features



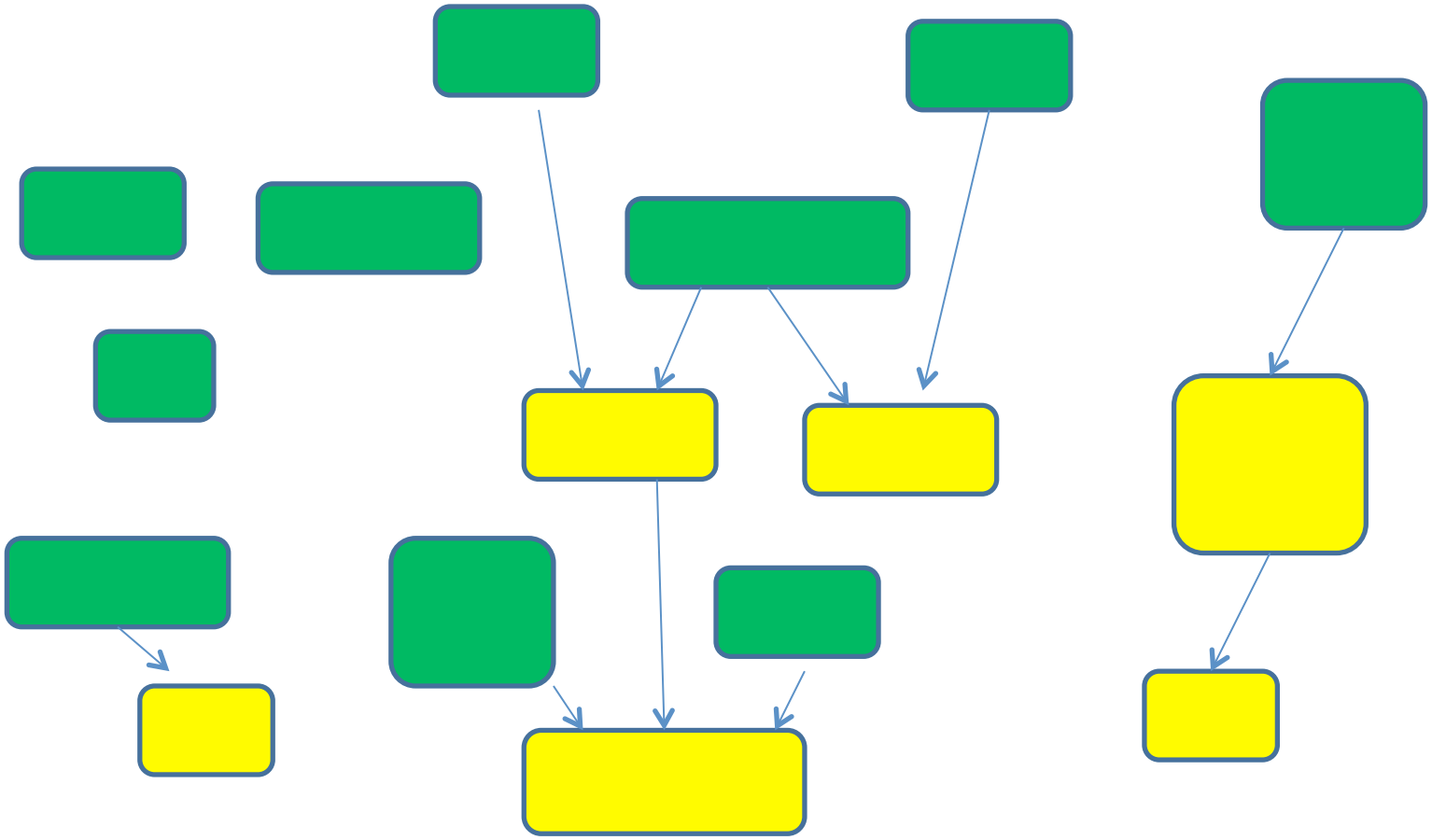
Features and …



Invisible features

- Architecture
- Infrastructure
- Frameworks
- ….

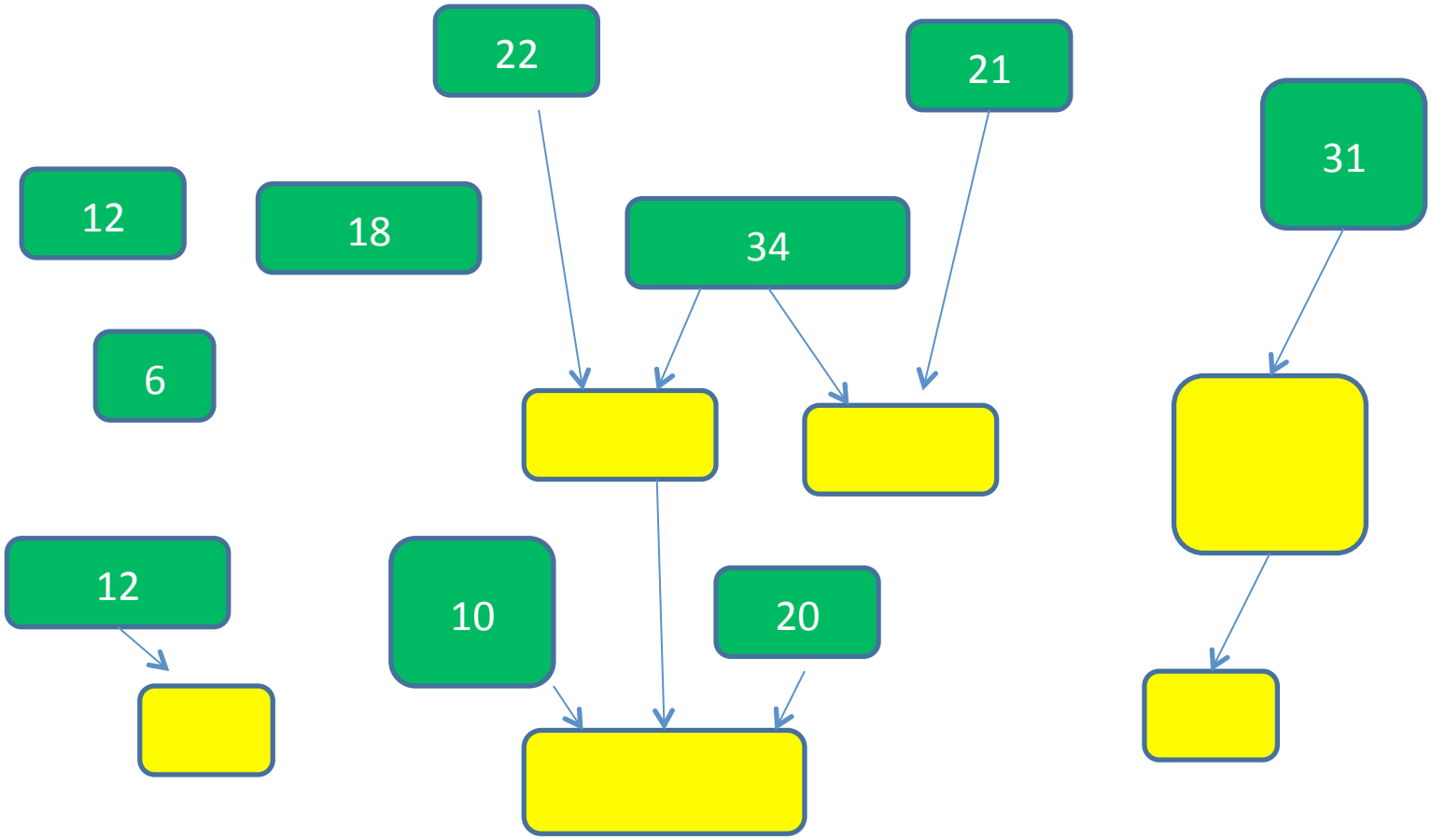
Dependencies



Architecture: Value and Cost

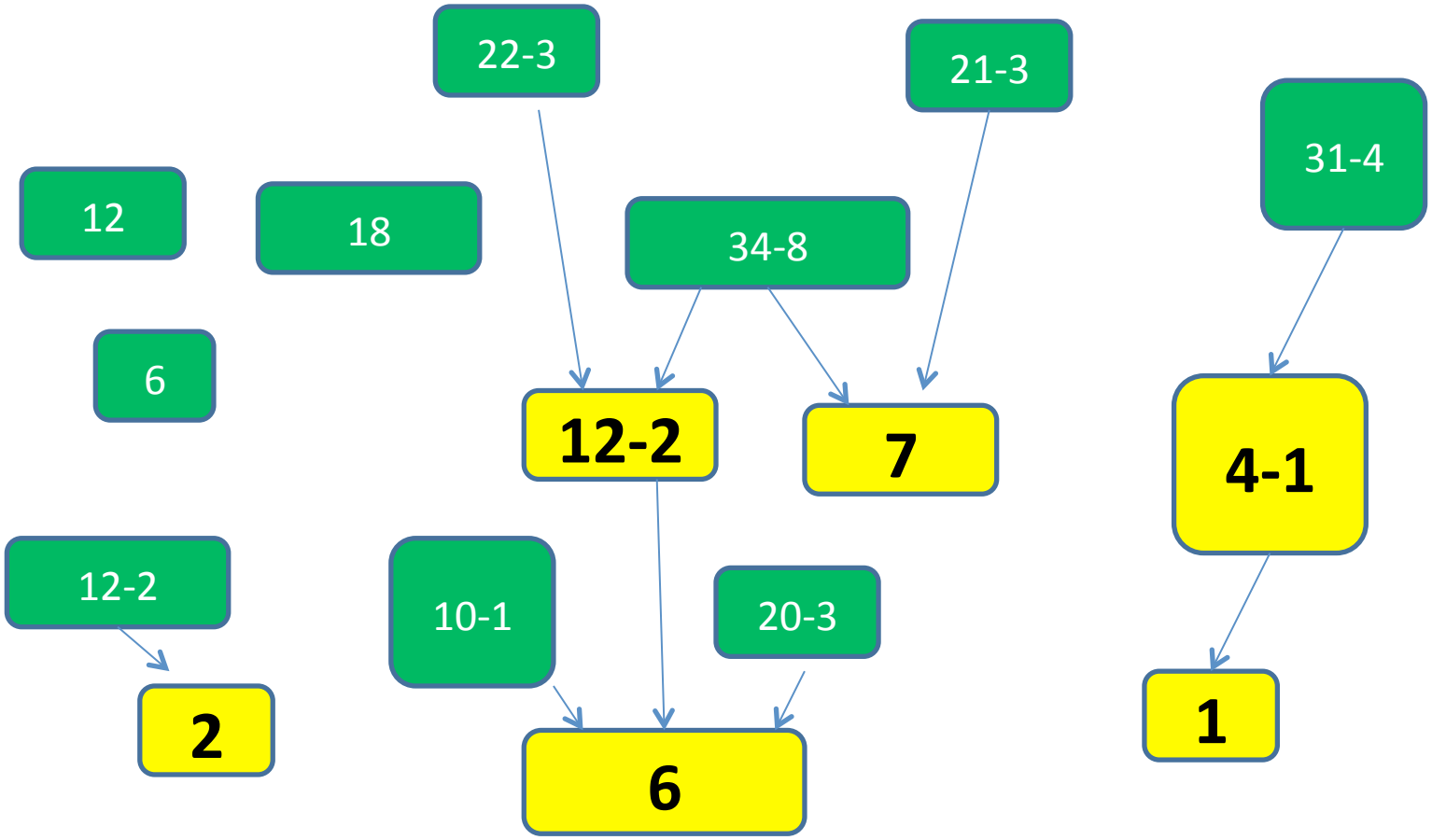
- Architecture has no (or little) externally visible "customer value"
- Iteration planning (backlog) is driven solely (?) by "customer value"
- YAGNI, BUFD, Metaphor,…
- "Last responsible moment!"
- We'll refactor it later!
- *Ergo:* architectural activities are not given proper attention
- *Ergo:* large technical debts

Value

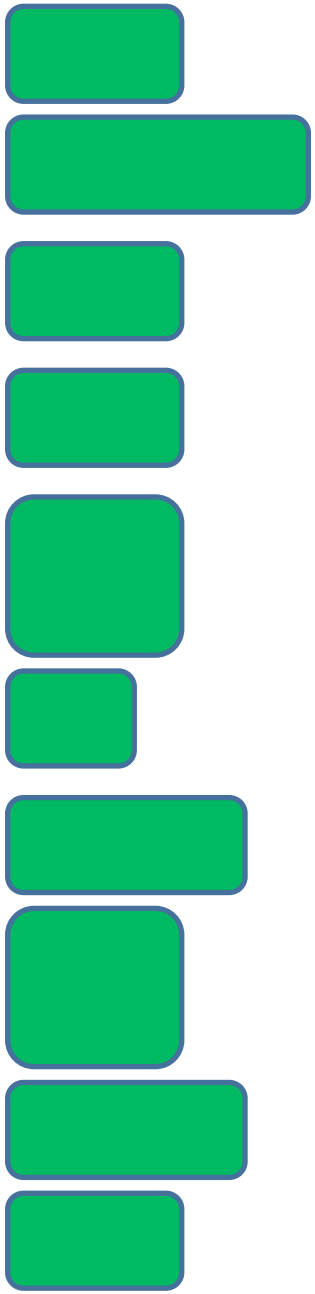


∑ = 186 utils

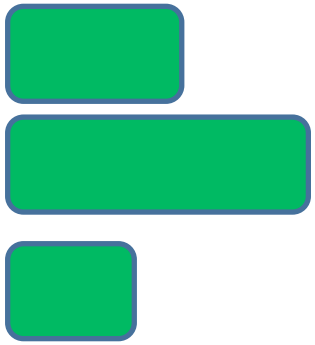
Value reallocated to architecture



∑ = 186 utils

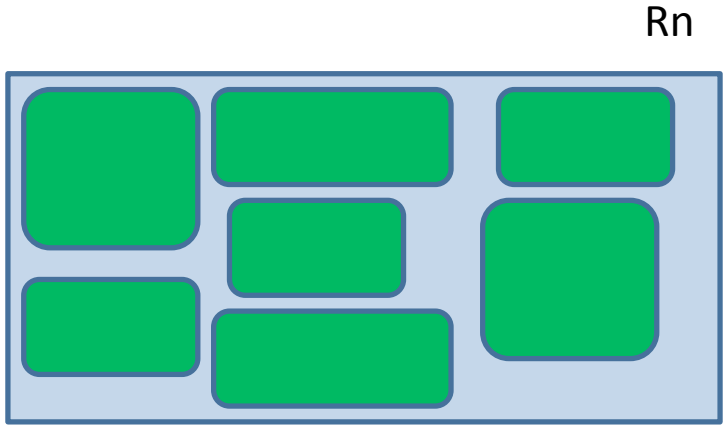


Features

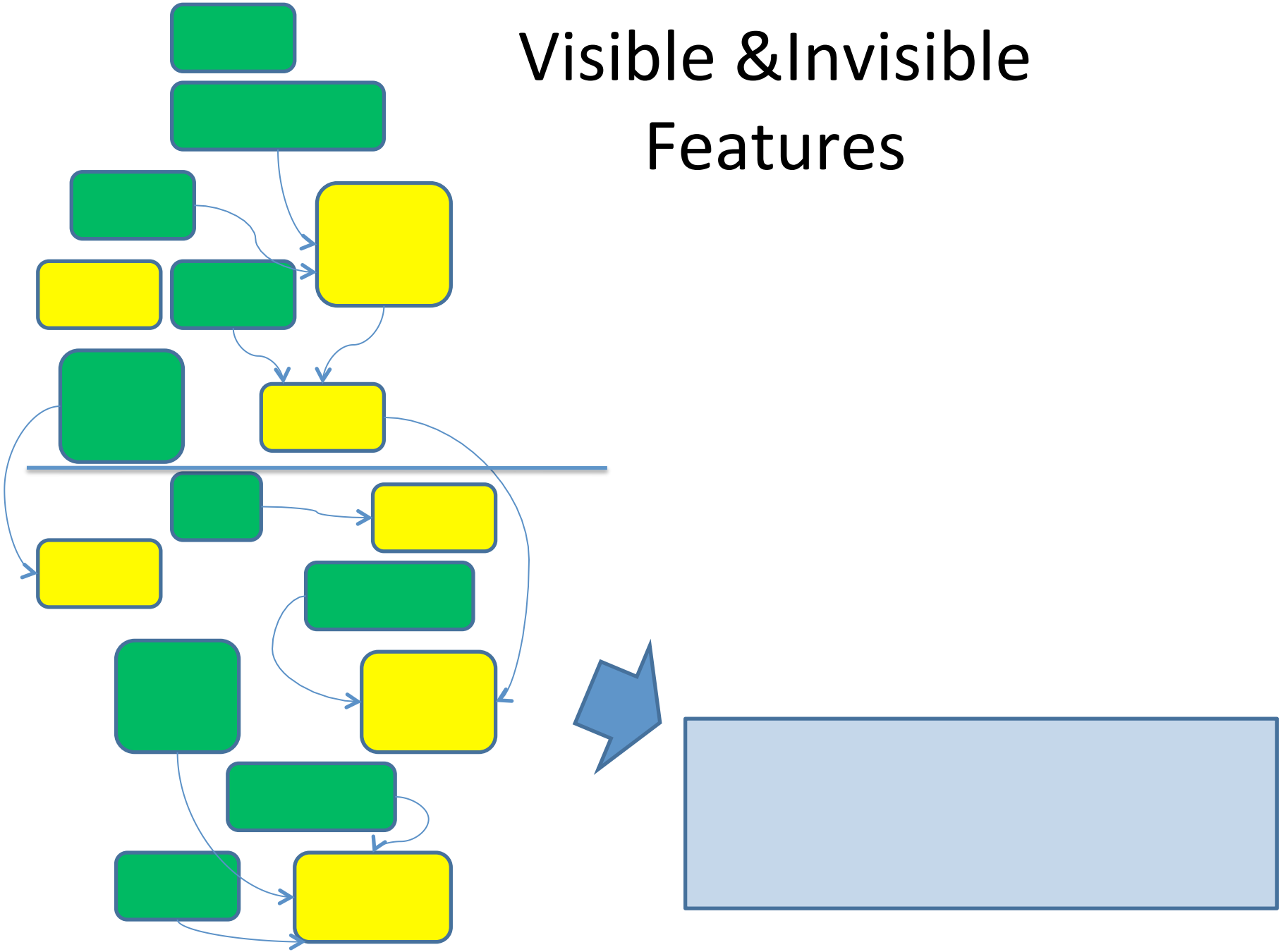


Features

?

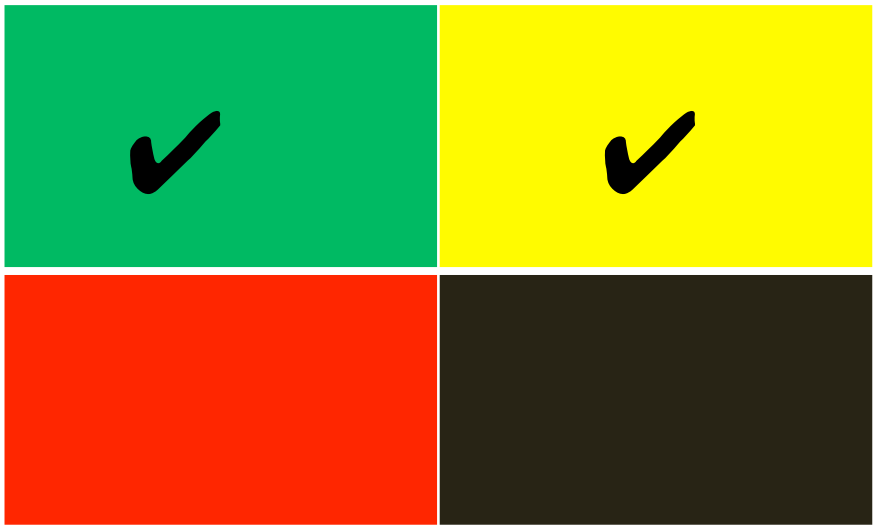


Visible &Invisible Features



Time-box

Time-box with Buffer



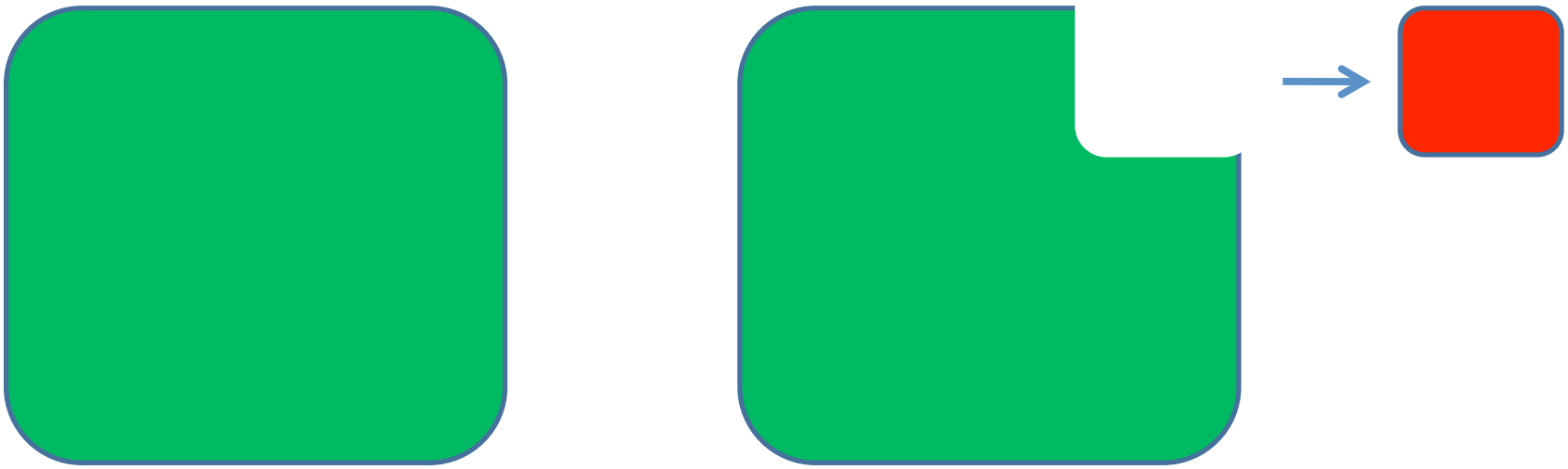
What colour is your backlog?

(so far)

Defects

- Defect = Feature with negative value
- Fix (defect) has a positive cost (= work)
- Time/place of discovery
  - Inside development (in-house, in process)
  - Outside development in a released product (escaped defects)

Escaped Defect has Negative Value

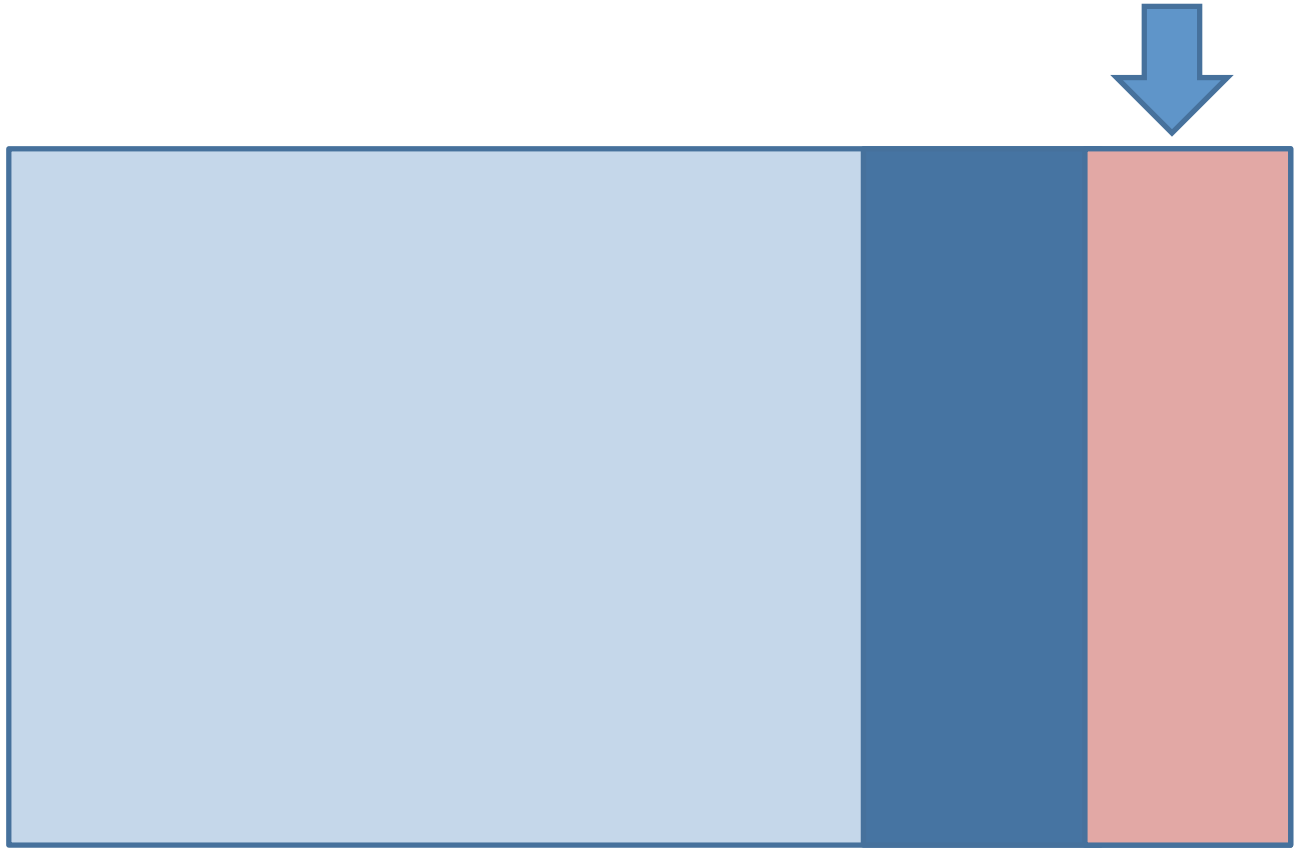


Perfect product

Imperfect product

Defect

Buffer for in-process defects

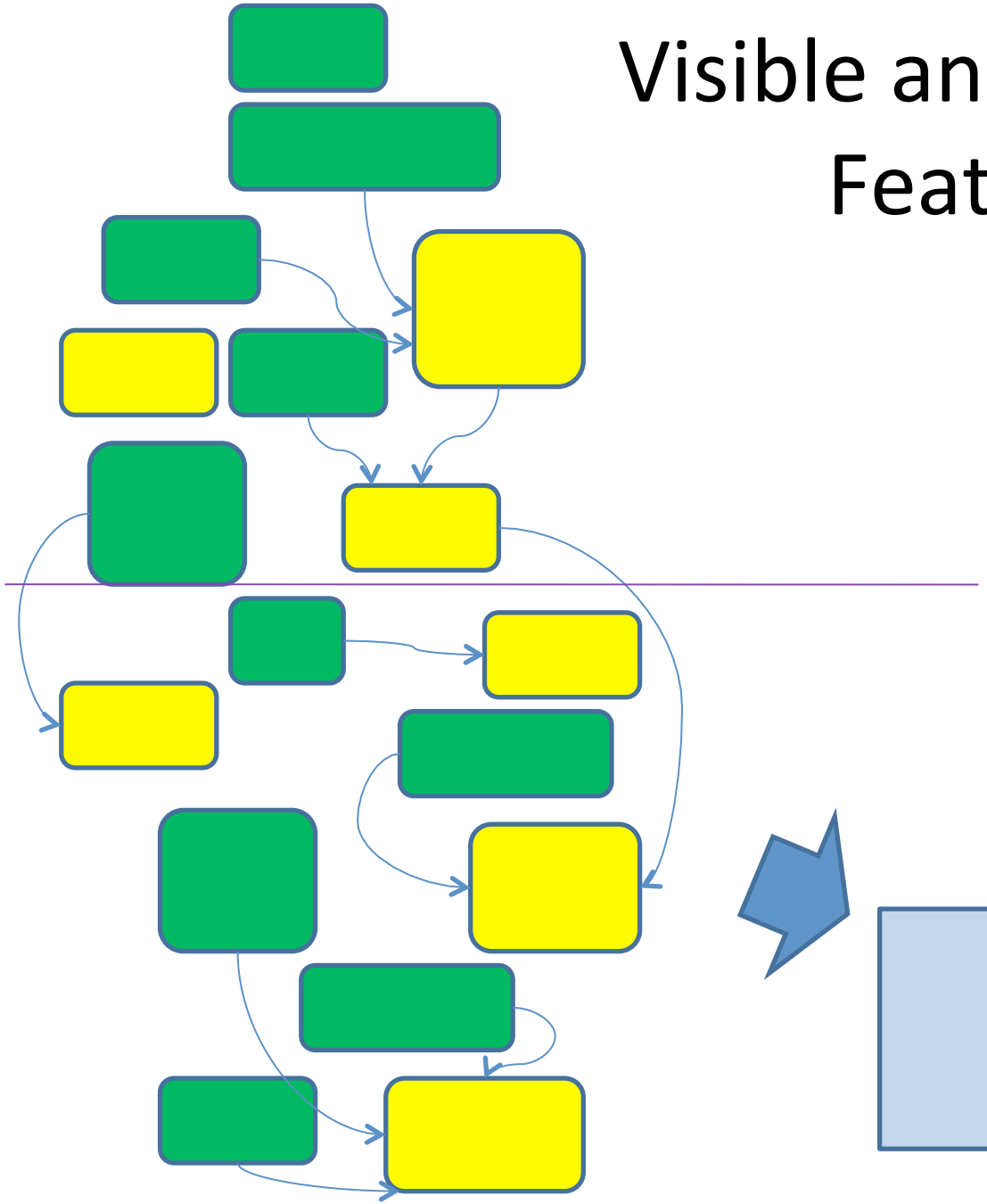


Fixing a Defect has a Cost

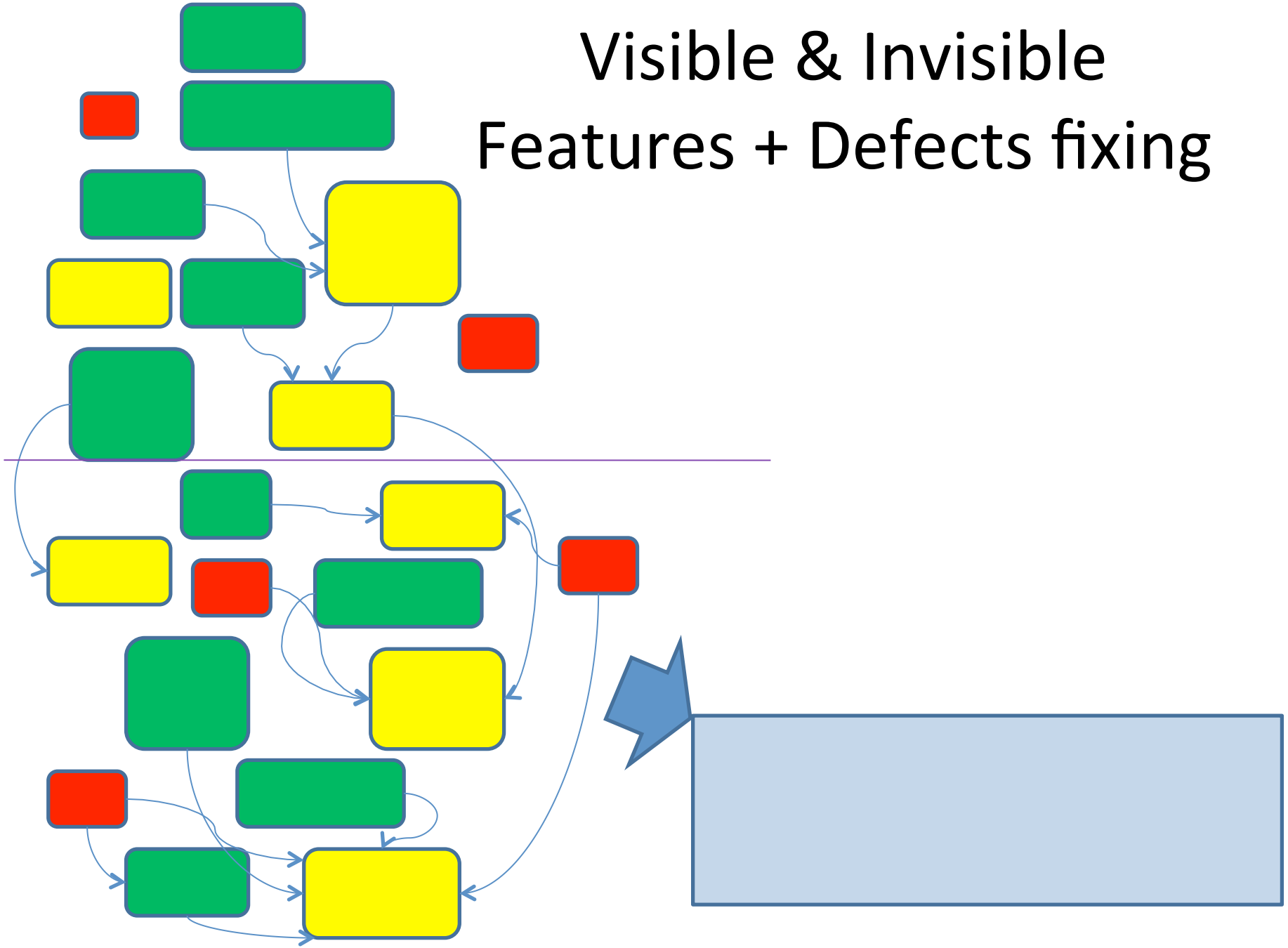
- Defects have both value and cost
- Value of fixing a defect = −Value of the defect
- Cost of fixing a defect (estimated)

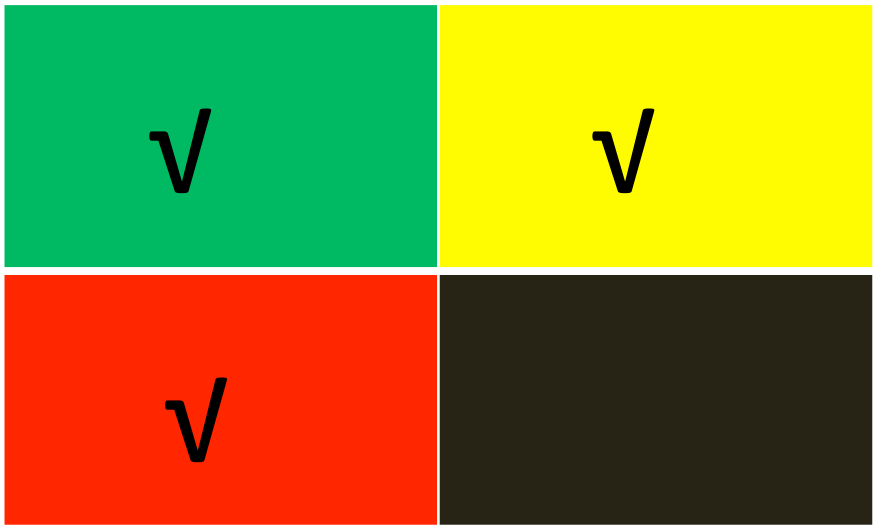
- Defects have dependencies
  - Defect fix depend on invisible feature
  - Visible feature depending on a fix

Visible and Invisible Features



Visible & Invisible
Features + Defects fixing





What colour is your backlog?

(so far)

Technical Debt

- Concept introduced by Ward Cunningham
- Often mentioned, rarely studied
- All experienced SW developers “feel” it.
- Drags long-lived projects and products down
- Friction

Cunningham, OOPSLA 1992

Origin of the metaphor

- Ward Cunningham, at OOPSLA 1992

"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, object-oriented or otherwise."

Cunningham, OOPSLA 1992

Technical Debt Definition (2013)

- A design or construction approach that is expedient in the short term, but that creates a technical context in which the same work will cost more to do later than it would cost to do now (including increased cost over time).

McConnell 2013

Technical Debt (S. McConnell)

- Implemented features (visible and invisible) = assets = non-debt
- Type 1: unintentional, non-strategic; poor design decisions, poor coding
- Type 2: intentional and strategic: optimize for the present, not for the future.
  - 2.A short-term: paid off quickly (refactorings, etc.)
    - Large chunks: easy to track
    - Many small bits: cannot track
  - 2.B long-term

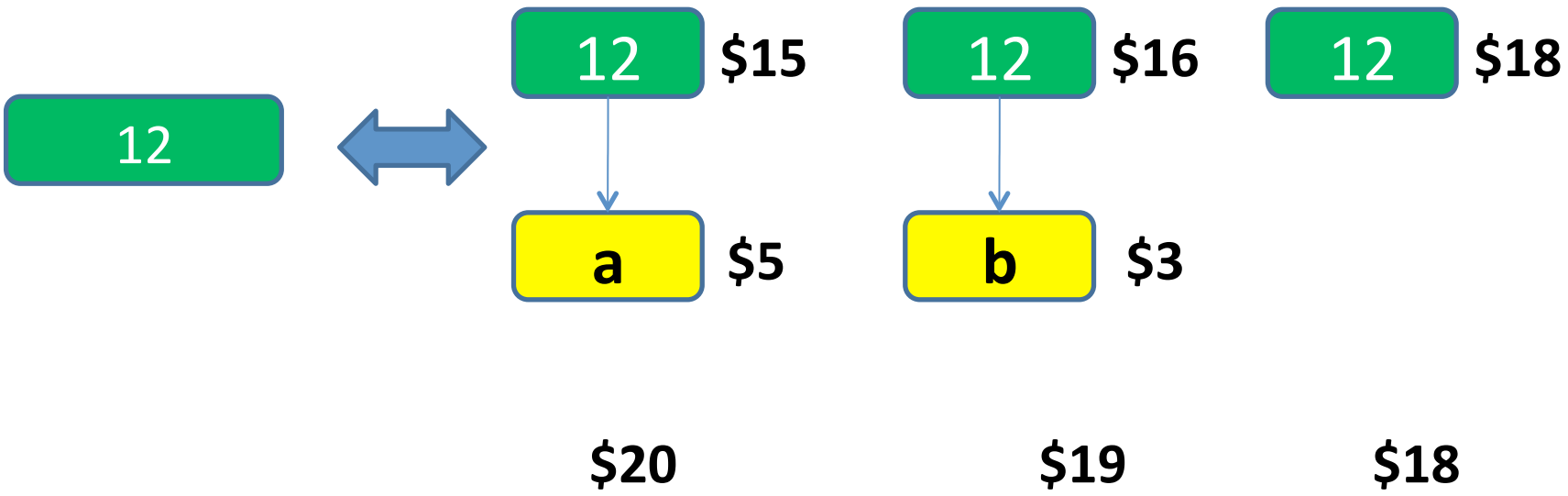
McConnell 2007

Technical Debt (M. Fowler)

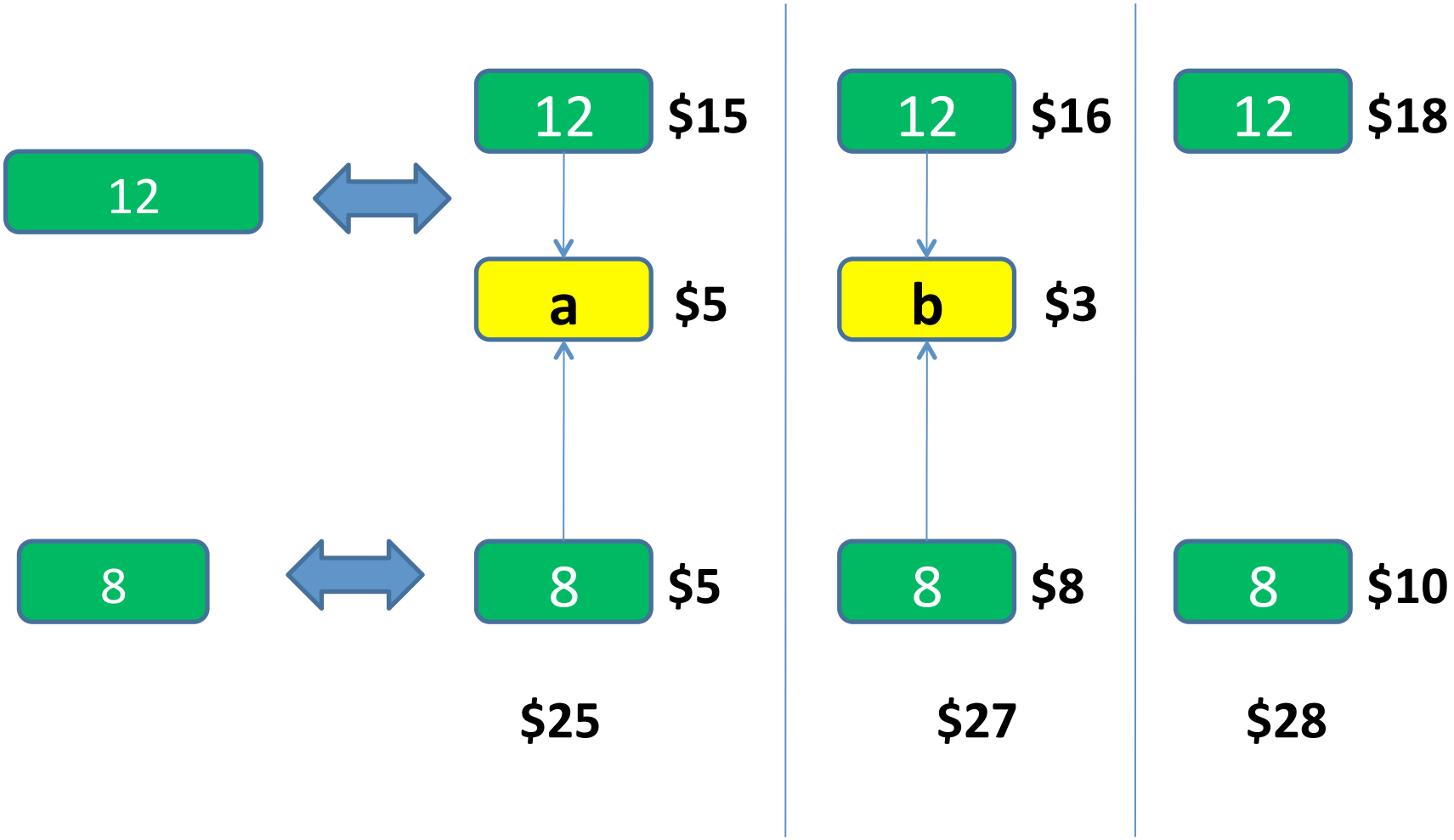
	Reckless	Prudent
Deliberate	*"We don't have time for design"*	*"We must ship now and deal with consequences"*
Inadvertent	*"What's Layering?"*	*"Now we know how we should have done it"*

Fowler 2009, 2010

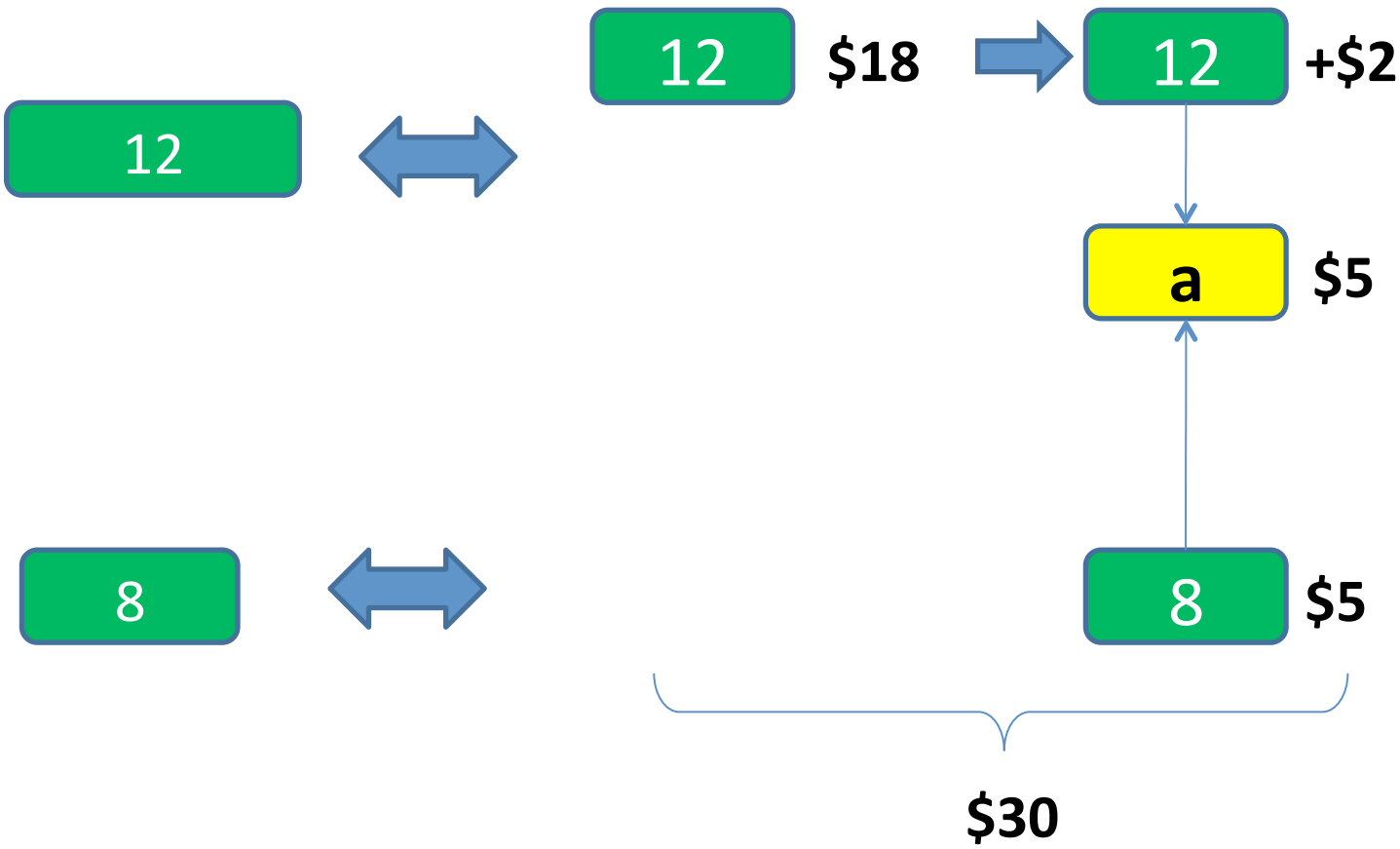
Technical Debt (1)



Technical Debt (2)



Technical Debt (3)



Interests

- In presence of technical debt:
  Cost of adding new features is higher

- When repaying (fixing), additional cost for retrofitting already implemented features

- Technical debt not repaid => lead to increased cost, forever
- Cost of fixing increases over time

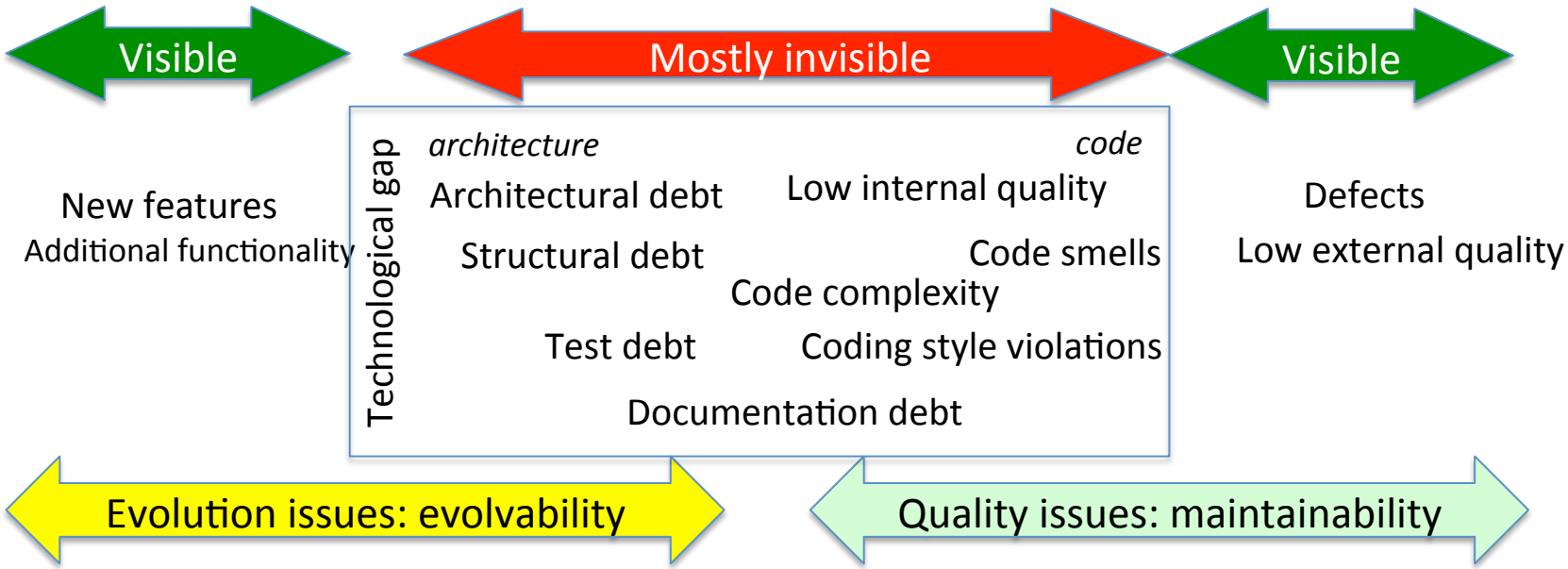
M. Fowler

TD litmus test

- If you are not incurring any interest, then it probably is not a debt

McConnell 2013

Technical debt landscape



Kruchten et al 2012

Technical debt

- Not just crappy code: wise investment
- Depends on the future

  “Technical futures”

Repaying debt

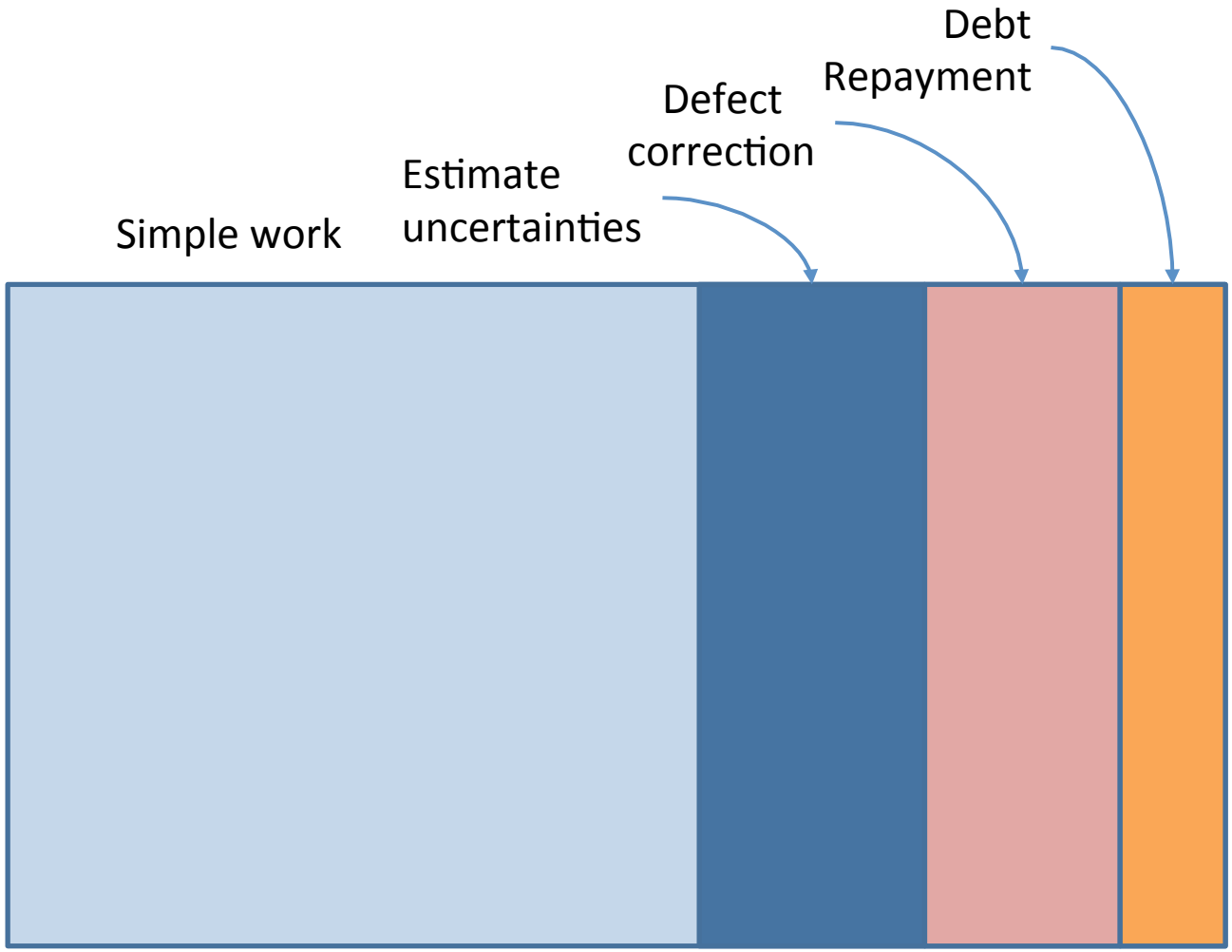
- What to repay ?
- When to repay ?

Tackling Technical Debt

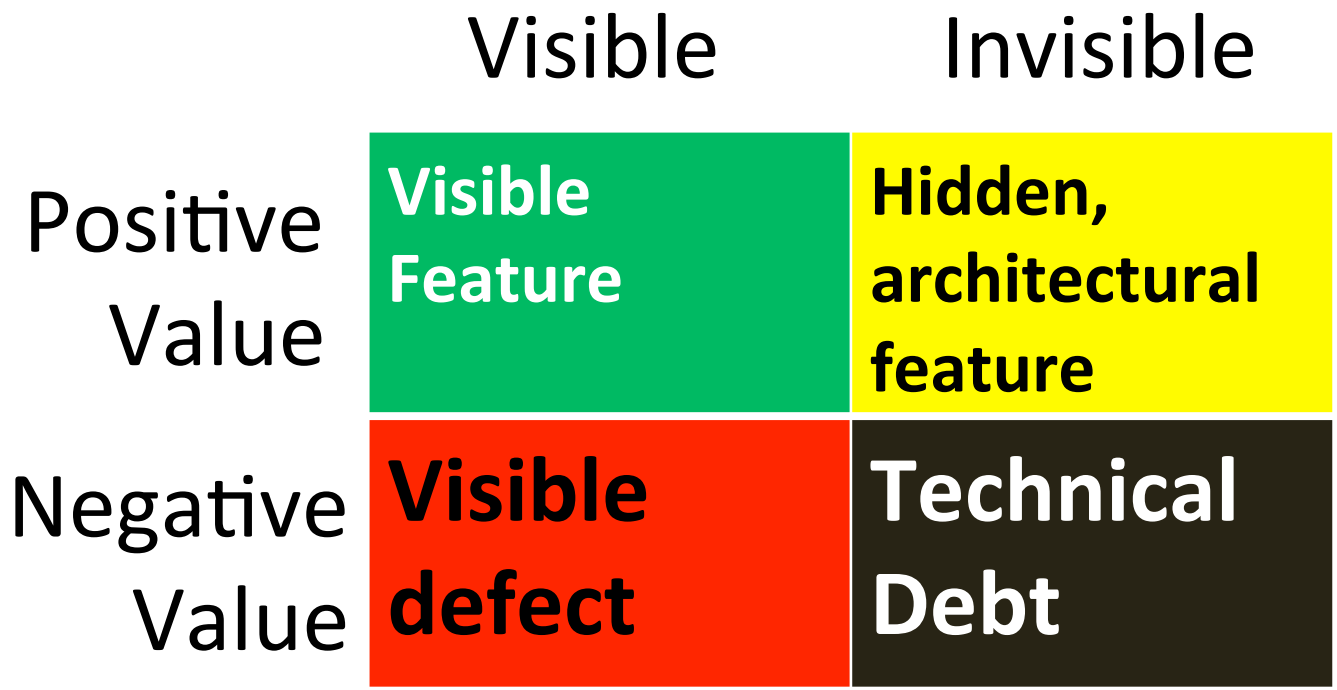
Attitudes and approaches found:

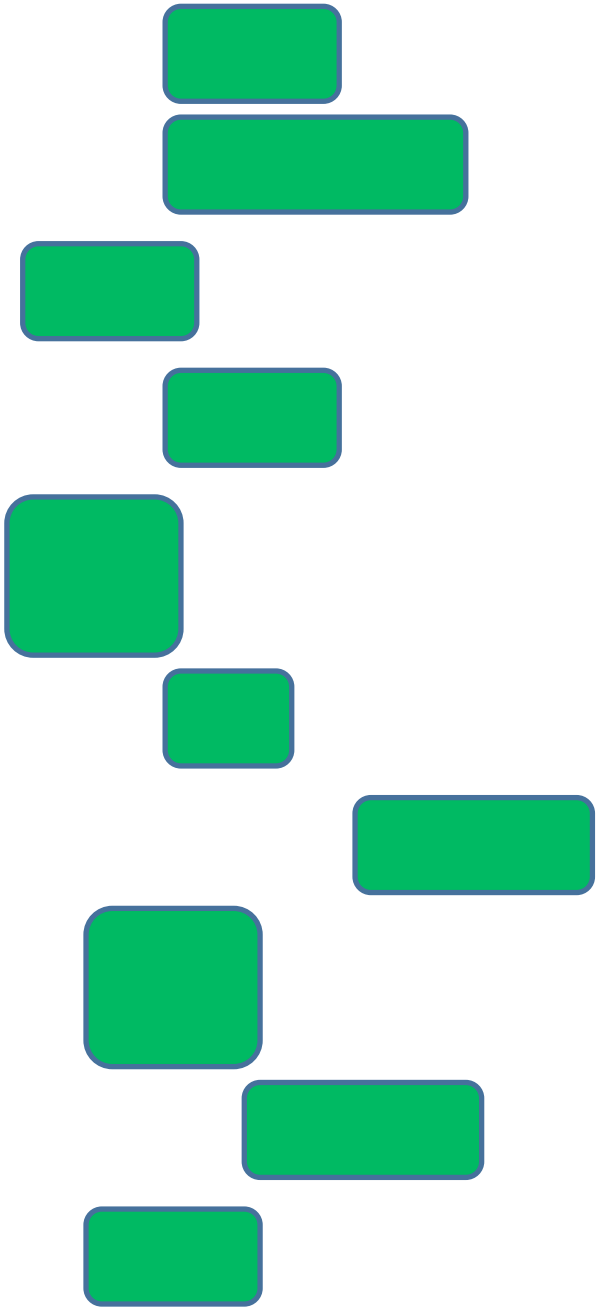
1. Ignorance is bliss
2. The elephant in the room
3. Big scary $$$$ numbers
4. Five star ranking
5. We're agile, so we are immune!
6. Constant reduction
7. Reduction iterations (sprints)

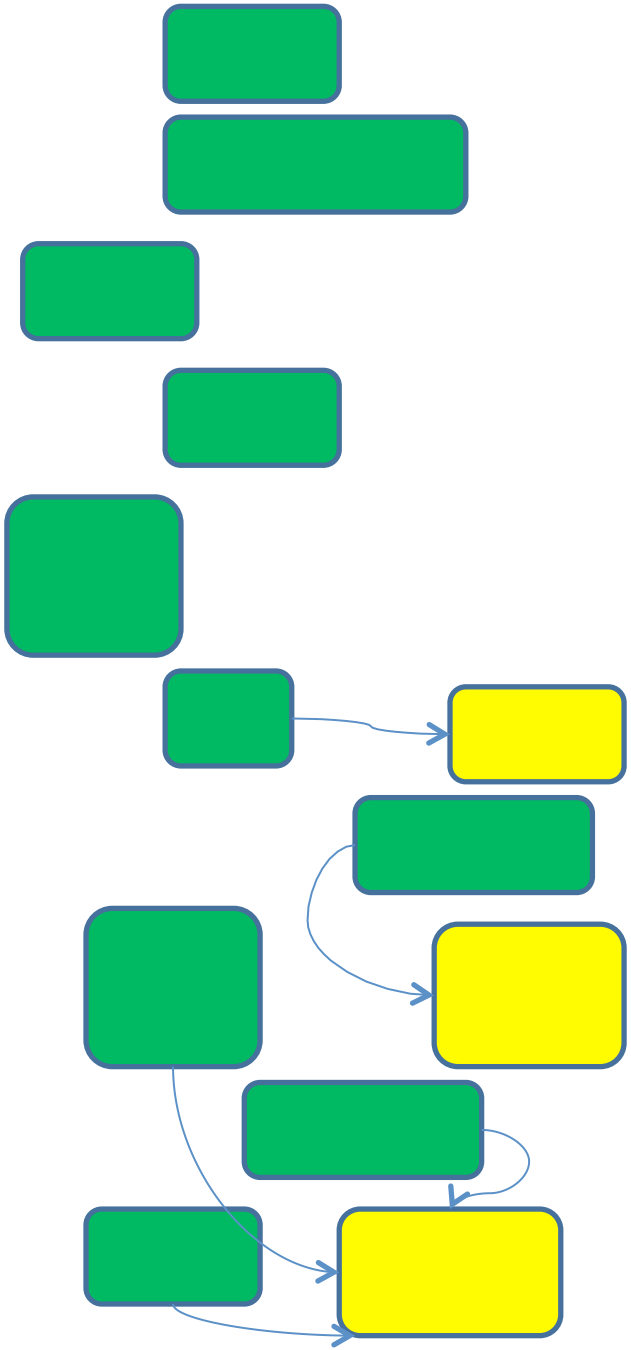
Buffer for debt repayment

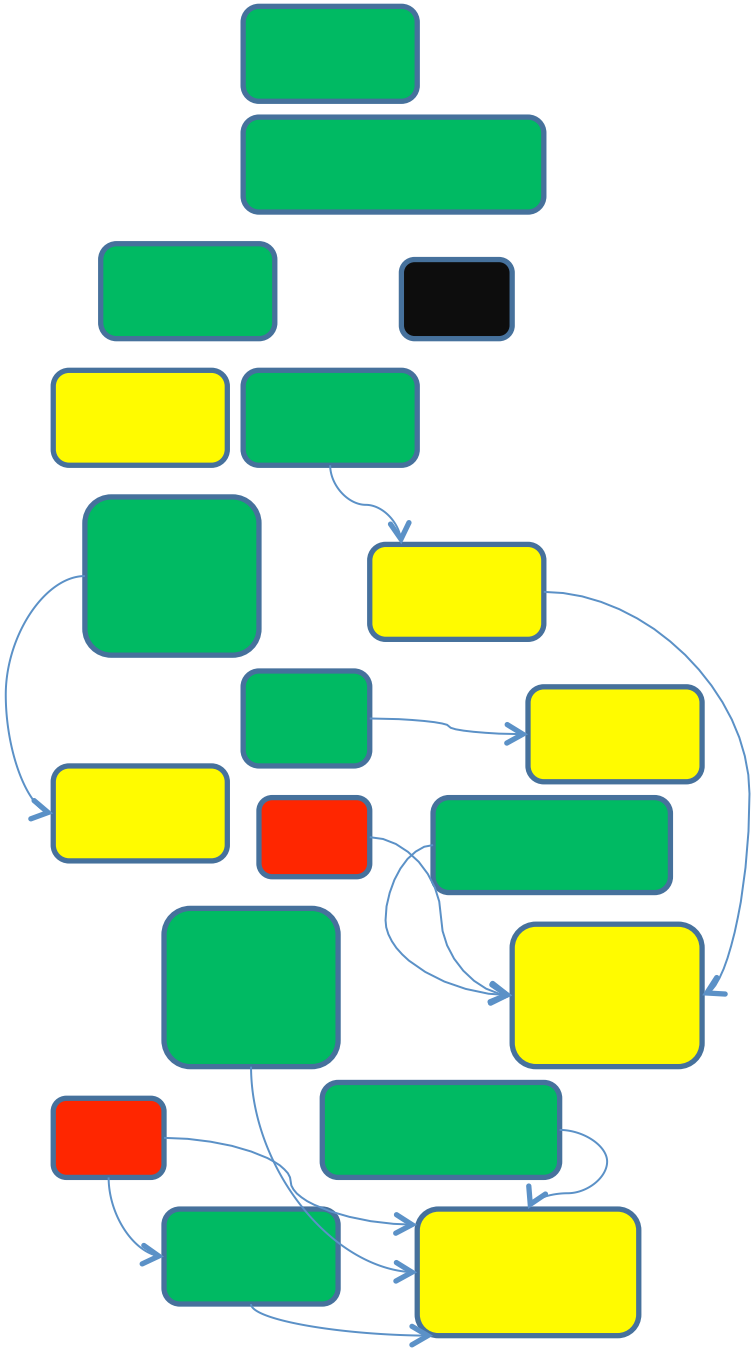


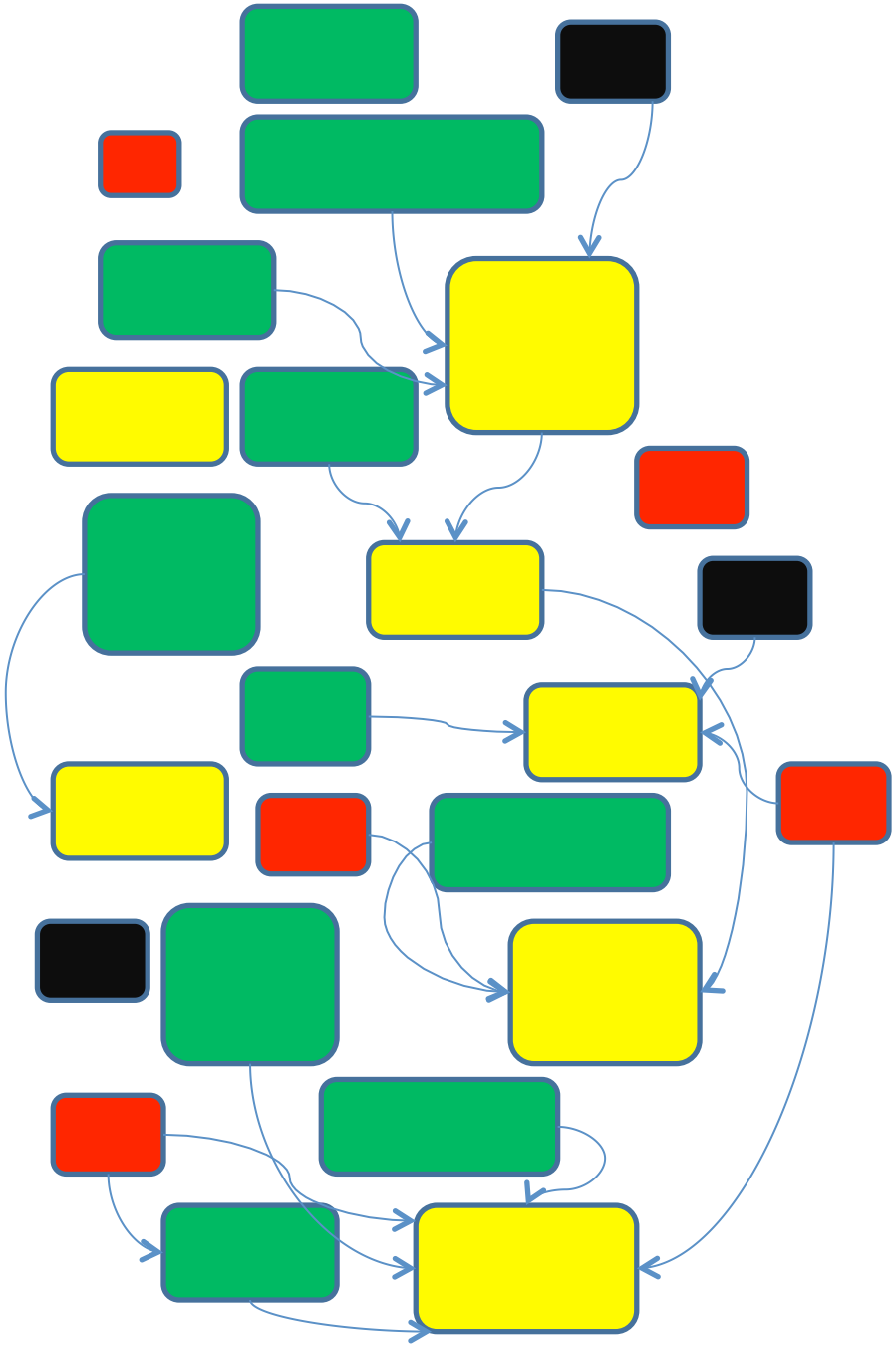
Colours in your Backlog

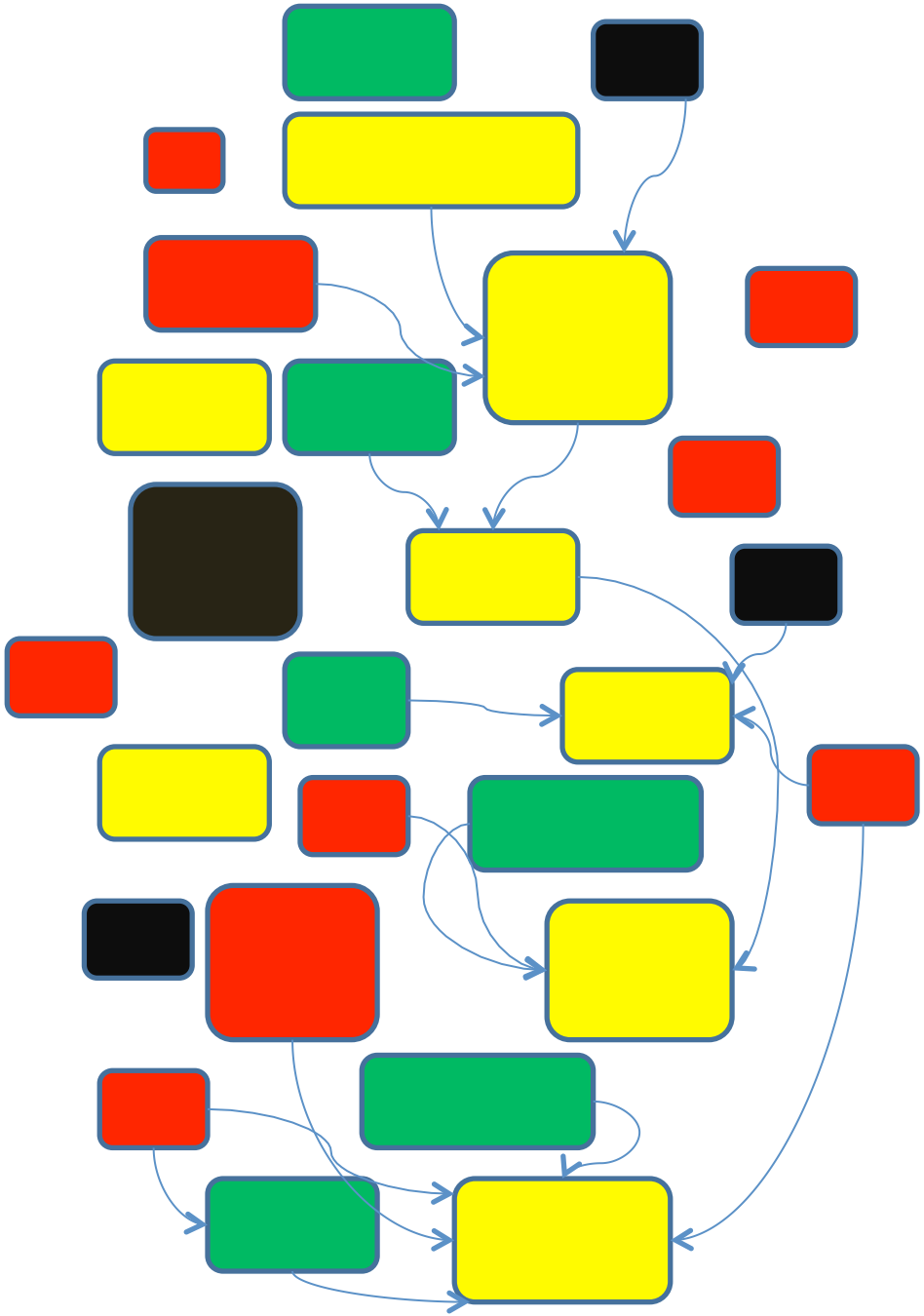


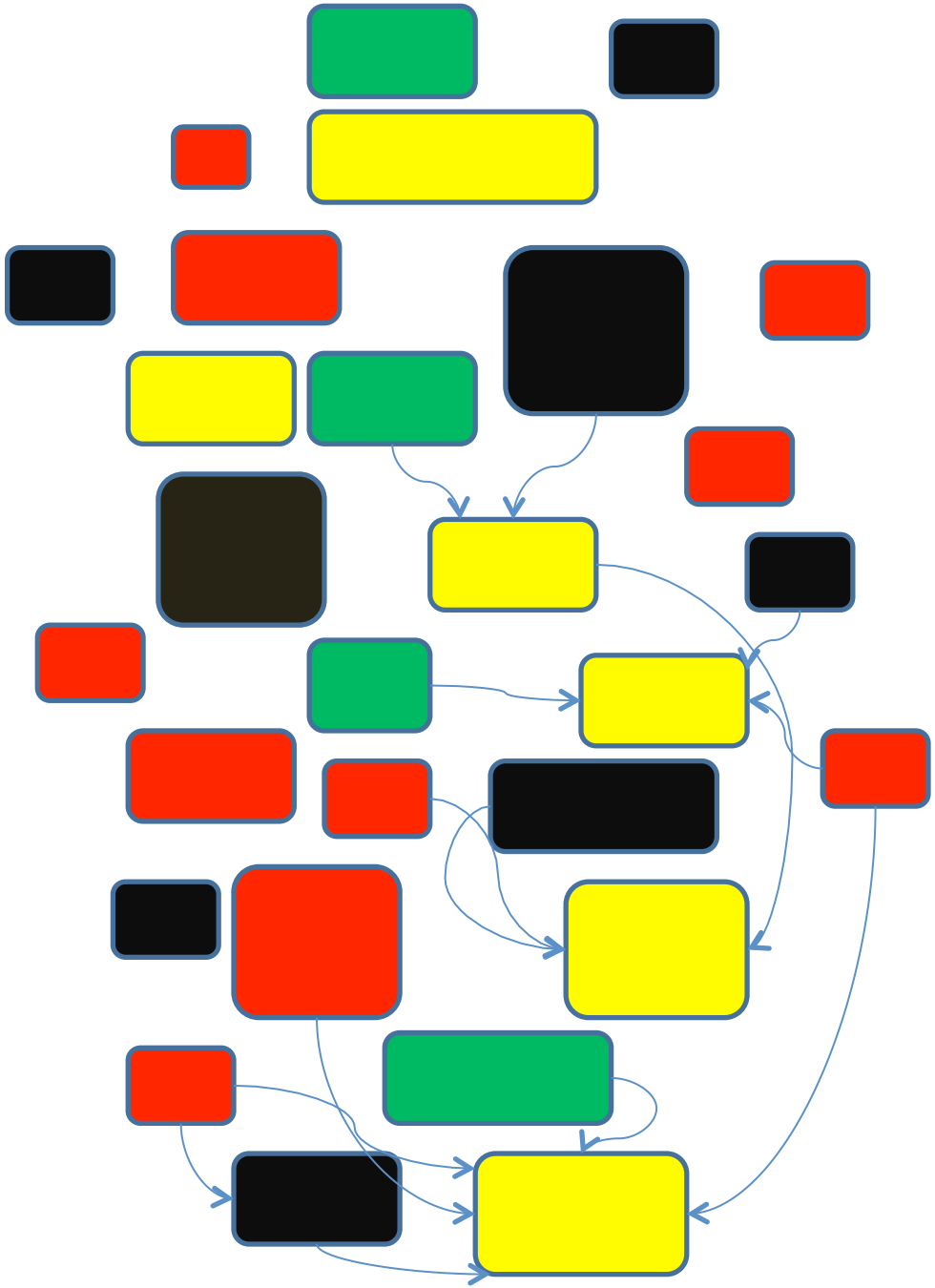


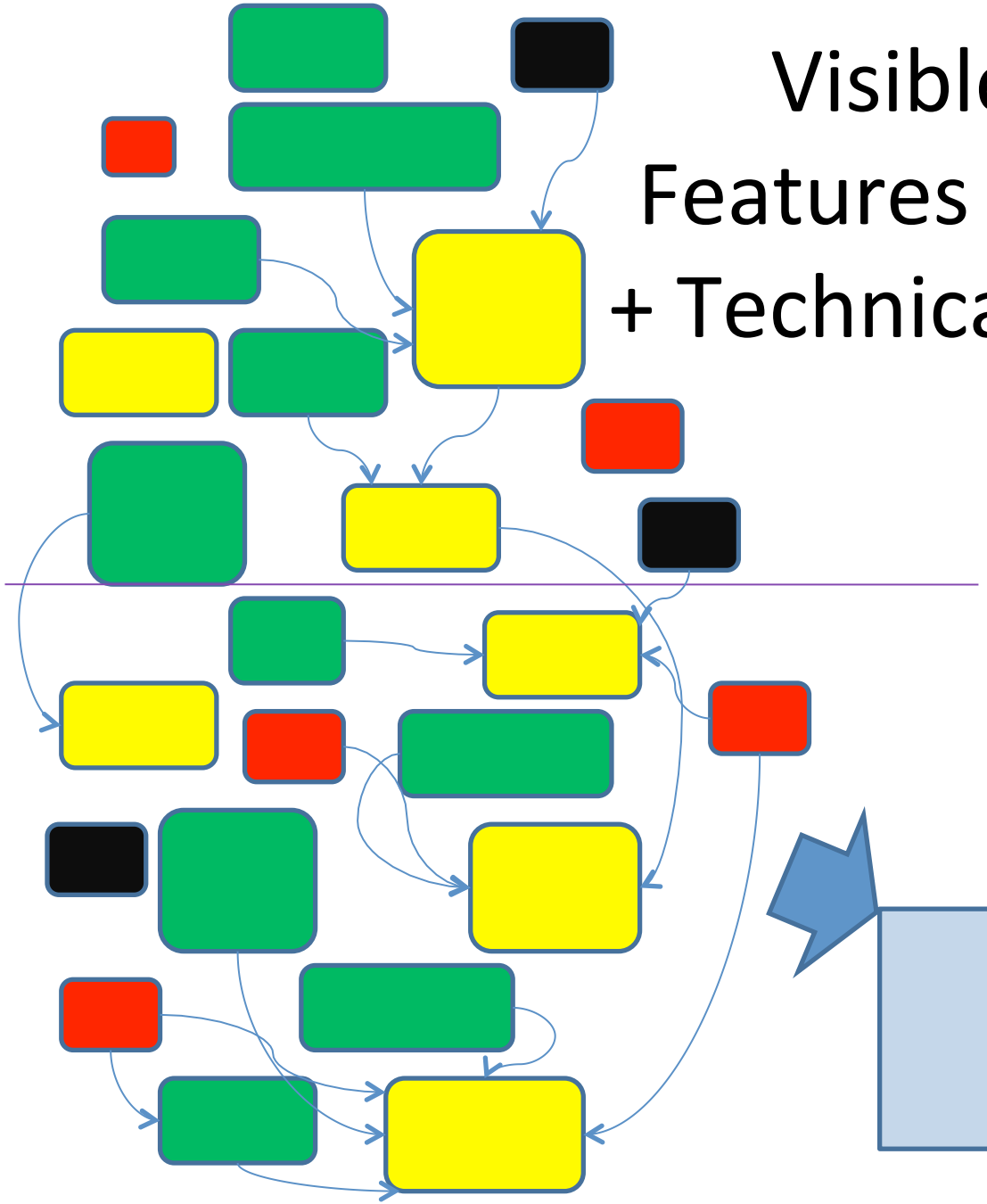












Visible & Invisible

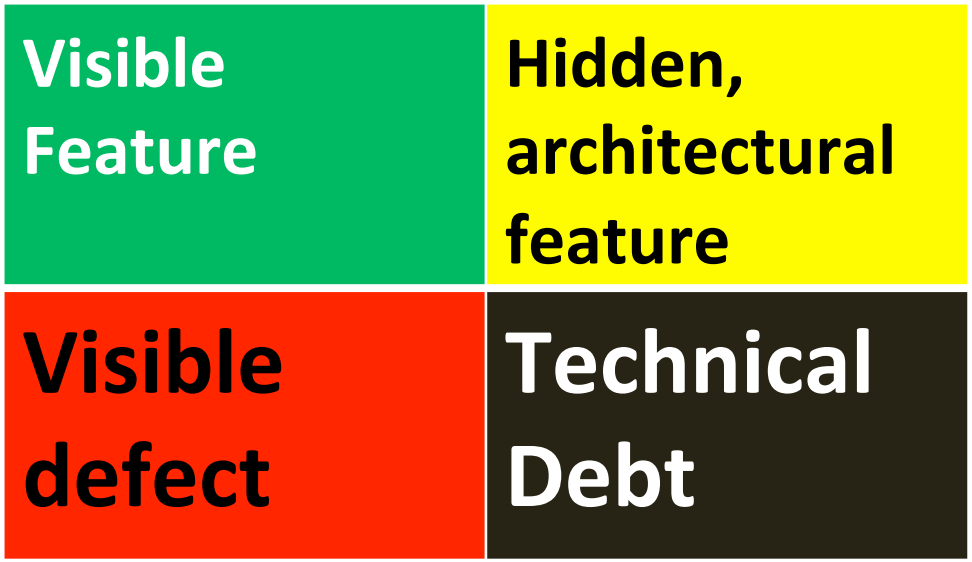
Features + Defects fixing

+ Technical Debt payment

Tensions

Product manager

Architects



Customer Support

Nobody?

Tools !?

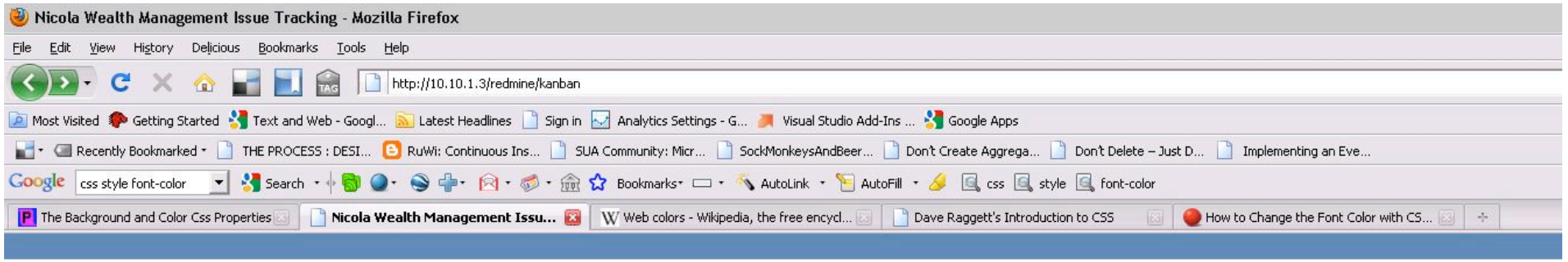
#94	Tell AnonymousUsers the benefits of registering	Low Cost	High Value	
#103	Attachments should be possible for all comments, including threaded replies	Medium Cost	High Value	
#97	Legal notices	Low Cost	Medium Value	
#100	Per-list unsubscribe for mailing-list-only users	Medium Cost	Medium Value	
#50	Group member list: "edit details" for someone's membership	Low Cost	Medium Value	
#96	Chapters (possibly networks?) map	Medium Cost	Low Value	
#102	Topics should show icon of some sort to indicate attachments	Low Cost	Low Value	

1.0 RC1 - Relase Candidate 1 (14 matches)

Ticket	Summary	Cost	Value	Owner
#49	switch to email-based usernames	High Cost	High Value	joshuagorner
#54	Who's Online listing	Medium Cost	High Value	francis
#55	National Office content	Medium Cost	High Value	
#58	chapter vs network	Medium Cost	High Value	joshuagorner
#61	Intuitive combinations of group visibility / privacy in UI	Medium Cost	High Value	
#93	Individual anonymous users should be able to sign up to mailing lists	Medium Cost	High Value	
#48	network_new_member cannot use dropdown to list members	Medium Cost	High Value	
#81	Prevent "private" networks	Medium Cost	High Value	
#98	Group creator (particularly networks) need not also be a member	Medium Cost	High Value	
#21	Verify email accounts automatically	Medium Cost	Medium Value	
#22	Multiple levels of membership in a group	High Cost	Medium Value	
#23	Groups should have "former members" to handle involvement history	Medium Cost	Medium Value	
#56	Suggested communities	Medium Cost	Medium Value	
#60	Notifications for group invitations / requests	Low Cost	Medium Value	

None (39 matches)

Ticket	Summary	Cost	Value	Owner
#64	fire and forget URL for signing up email addresses to the main list	Medium Cost	High Value	
#67	topic-creation preview	Medium Cost	High Value	
#99	feedback system	Low Cost	High Value	
#68	too many notices!	Low Cost	High Value	
#78	Clicking on a tag causes an error	Low Cost	High Value	
#90	private-messaging 'to' input very rough	Medium Cost	High Value	benbest
#91	password strength issue	Low Cost	High Value	



Kanban

Each list is a Pane of issues. The issues can be dragged and dropped onto other panes based on Roles and Permissions settings.

Incoming

- #16 - Feature - ISM Data Adapter - UI Can display and filter securities
- #23 - Feature - Accounting Manager - Accounting requires an accpac reconciliation view
- #24 - Feature - Accounting Manager - Autocomplete of memo fields
- #25 - Feature - Accounting Manager - Load views/viewmodels on demand
- #26 - Feature - Accounting Manager - Improve nHibernate session management
- #30 - Feature - Messaging - Implement point-to-point message queue
- #37 - Feature - General Tasks - Build Scripts

Backlog

Normal

- #17 - Feature - ISM Data Adapter - Show user securities that have incomplete required fields and allow them to update
- #28 - Feature - ISM Data Adapter - Can insert a new security into the database

Quick Tasks

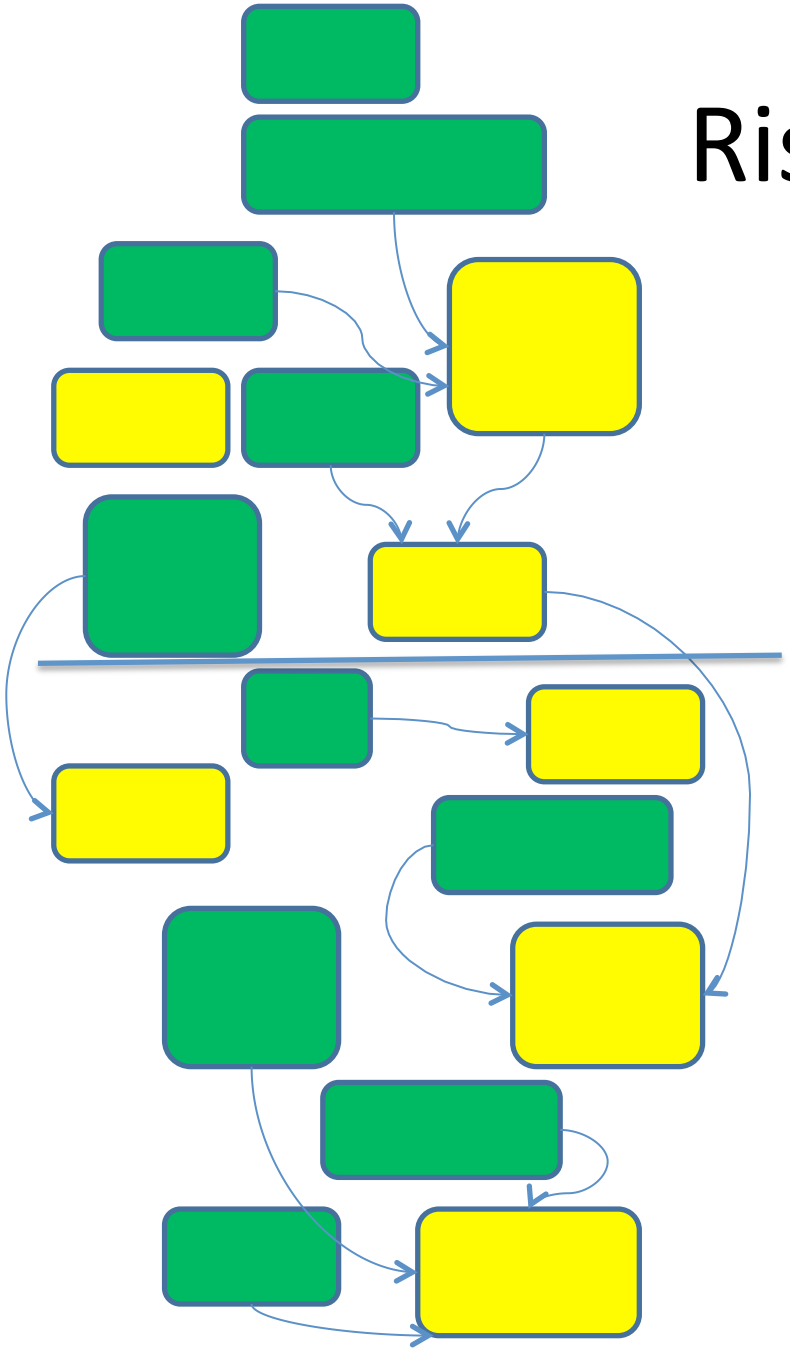
(No issues)

Selected Requests

- #44 - Feature - Insurance Manager - Add 10-8 split category for UL policies
- **#35 - Feature - ISM Data Adapter - FDP Gateway: Can add buy and sell transactions for holdings**

User	Active	Testing
Chris Nicola	**#39 - Feature - Accounting Manager - David's changes to commission report** **#43 - Feature - ISM Data Adapter - Create UI for loading cash transactions** **#8 - Feature - General Tasks - Environment Setup** **#45 - Feature - Insurance Manager - Display repcode field in policy list**	(No issues)
Adam Dymitruk	#46 - Feature - Dynamics CRM - Evaluate Scribe for FDP -> CRM data migration	(No issues)
Admin Istrator	#47 - Bug - Unsorted - Test Bug for Kanban Board	(No issues)
Jennifer Keates	#49 - Technical Debt - Unsorted - Test Technical Debt for Kanban Board	#27 - Feature - ISM Data Ac new cash transaction
Lee Tippetts-Aylmer	#48 - Architecture - Unsorted - Test Architecture for Kanban Board	(No issues)
Paul Newton	#22 - Feature - Database - Client List Table	(No issues)
Richard Haisinger	(No issues)	(No issues)
Tim Low	(No issues)	(No issues)

Based on Redmine, by Chris Nicola



Risks & Uncertainties

Rules of thumb:

… facing uncertainty:

- Green stuff: move up
  - Defer
- Yellow stuff: move down
  - Experiment now

Karl Wiegers, 1999
Kruchten, 1998

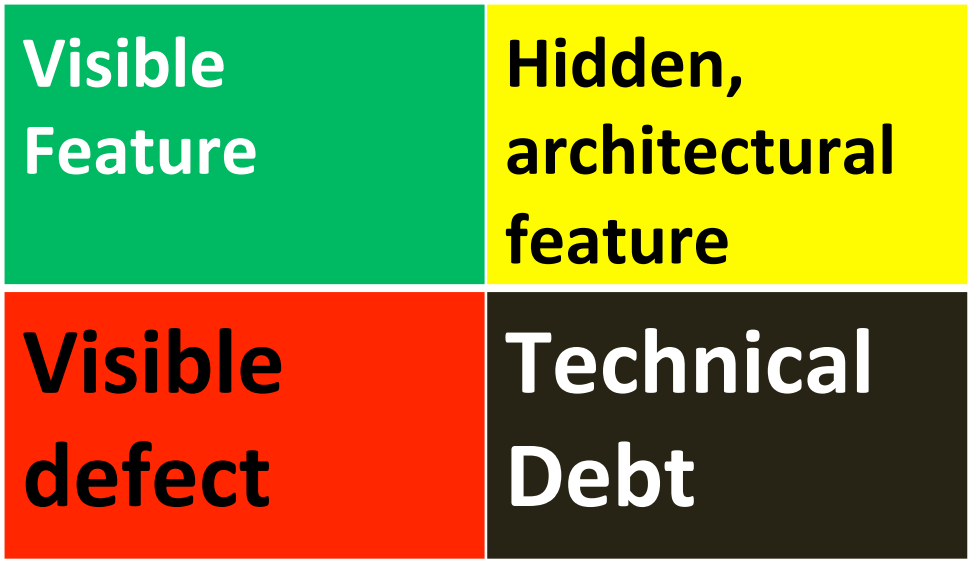
Key message(s)

- Having multiple repositories of things to do, managed by multiple or different people, based on different criteria is a bad idea.
- It leads to delays, frustrations, accumulated technical debt, reduced velocity, distrust….
- Manage all colours together
- Value is different than cost

Tensions

Product manager

Architects



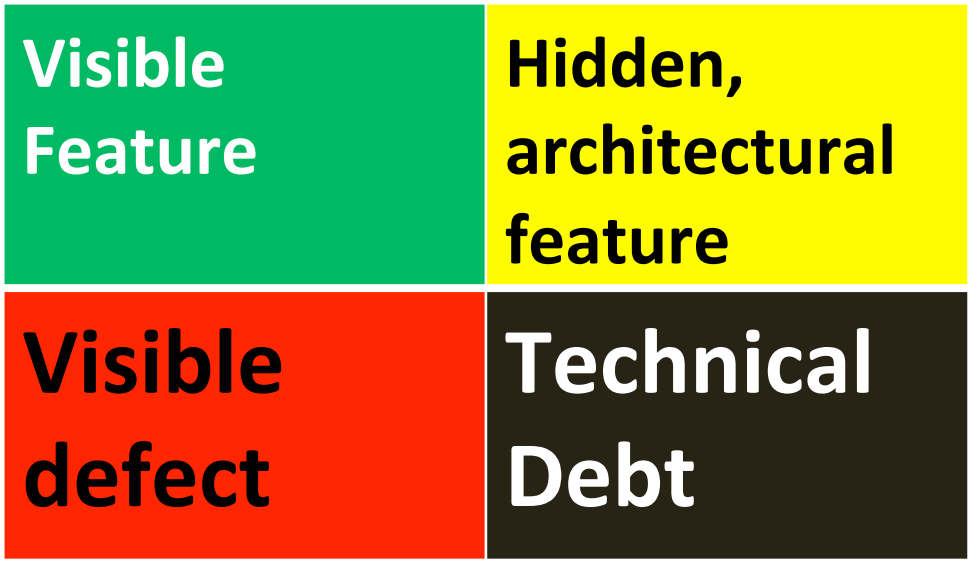
Customer Support

Nobody?

Agility

- Lead to a shared mental model of the real state of the project
- Common understanding of the nature and extent of commitments
- Scale

Manage them all together



Philippe Kruchten

@pbpk

philippe@kruchten.com

philippe.kruchten.com

References

- Brown, N., Cai, Y., Guo, Y., Kazman, R., Kim, M., Kruchten, P., et al. (2010). *Managing Technical Debt in Software-Intensive Systems*. Paper presented at the Future of software engineering research (FoSER) workshop, part of Foundations of Software Engineering (FSE 2010) conference.
- Brown, N., Nord, R., Ozkaya, I., Kruchten, P., & Lim, E. (2011). Hard Choice: A game for balancing strategy for agility. Paper presented at the 24th IEEE CS Conference on Software Engineering Education and Training (CSEE&T 2011), Honolulu, HI, USA.
- Cunningham, W. (1992). *The WyCash Portfolio Management System.* Paper presented at the OOPSLA'92 conference, ACM. Retrieved from http://c2.com/doc/oopsla92.html
- Curtis, B., Sappidi, J., & Szynkarski, A. (2012). Estimating the Principal of an Application's Technical Debt. IEEE Software, 29(6).
- Denne, M., & Cleland-Huang, J. (2004). *Software by Numbers: Low-Risk, High-Return Development,* Prentice Hall.
- Denne, M., & Cleland-Huang, J. (2004). The Incremental Funding Method: Data-Driven Software Development, *IEEE Software*, 21(3), 39-47.
- Fowler, M. (2009), *Technical debt quadrant,* Blog post at: http://www.martinfowler.com/bliki/TechnicalDebtQuadrant.html
- Gat, I. (ed.). (2010). *How to settle your technical debt--a manager's guide*. Arlington Mass: Cutter Consortium.
- Kruchten, Ph. (2010) Contextualizing Agile Software Development," Paper presented at the EuroSPI 2010 conference in Grenoble, Sept.1-3, 2010

References

- Kruchten, P., Nord, R., & Ozkaya, I. (2012). Technical debt: from metaphor to theory and practice. *IEEE Software*, 29(6).
- Kruchten, P., Nord, R., Ozkaya, I., & Visser, J. (2012). Technical Debt in Software Development: from Metaphor to Theory--Report on the Third International Workshop on Managing Technical Debt, held at ICSE 2012 *ACM SIGSOFT Software Engineering Notes*, 37(5).
- Li, Z., Madhavji, N., Murtaza, S., Gittens, M., Miranskyy, A., Godwin, D., & Cialini, E. (2011). Characteristics of multiple-component defects and architectural hotspots: a large system case study. *Empirical Software Engineering*, 16(5), 667-702. doi: 10.1007/s10664-011-9155-y
- Lim, E. (2012). *Technical Debt: What Software Practitioners Have to Say.* (Master's thesis), University of British Columbia, Vancouver, Canada.
- Lim, E., Taksande, N., & Seaman, C. B. (2012). A Balancing Act: What Software Practitioners Have to Say about Technical Debt. *IEEE Software*, 29(6).
- MacCormack, A., Rusnak, J., & Baldwin, C. Y. (2006). Exploring the structure of complex software designs: An empirical study of open source and proprietary code. *Management Science, 52*(7), 1015-1030.
- Nord, R., Ozkaya, I., Kruchten, P., & Gonzalez, M. (2012). In search of a metric for managing architectural technical debt. Paper presented at the *Working IEEE/IFIP Conference on Software Architecture (WICSA 2012*), Helsinki, Finland.
- McConnell, S. (2007) *Notes on Technical Debt*, Blog post at: http://blogs.construx.com/blogs/stevemcc/archive/2007/11/01/technical-debt-2.aspx
- Special issue of *Cutter IT Journal* on Technical Debt, edited by I. Gat (October 2010) Cutter IT Journal, 23 (10).
- Sterling, C. (2010) *Managing Software Debt*, Addison-Wesley.

References (cont.)

- R. O. Spinola, N. Zazworka, A. Vetrò, C. B. Seaman, and F. Shull, "Investigating Technical Debt Folklore: Shedding Some Light on Technical Debt Opinion," in Proceedings of the 4th Workshop on Managing Technical Debt, at ICSE 2013, P. Kruchten, I. Ozkaya, and R. Nord, Eds., IEEE, 2013.
- K. Schmid, "On the Limits of the Technical Debt Metaphor," in Proceedings of the 4th Workshop on Managing Technical Debt, at ICSE 2013, P. Kruchten, I. Ozkaya, and R. Nord, Eds., IEEE, 2013, pp. 63-66.
- K. Schmid, "A Formal Approach to Technical Debt Decision Making," in Proceedings of the Conference on Quality of Software Architecture QoSA'2013, Vancouver, 2013, ACM.

Other pointers

http://techdebt.org

http://www.ontechnicaldebt.com/

@OnTechnicalDebt