Scrum,
Creating Great Products
& Critical Systems

What to Worry About, What's Missing, How to Fix it

Neil Potter
The Process Group
neil@processgroup.com
processgroup.com

Agenda

- **Scrum / Agile Overview**
- **What to Use From The Agile Manifesto**
  - All of it / some of it?
- **Definition of Scrum**
- **Scrum Risks to Look Out For and What to do About Them**
  - Mistaking speed for progress
  - 1-liner requirements (the devil is in the details)
  - Missing architecture / design
  - Missing final system test / validation
  - Missing configuration management
  - Missing risk management
  - Missing process assurance (known as ScrumBut)

Definition of Agile and Scrum

- **Agile**
  - "*Agile software development refers to a group of software development methodologies based on iterative development, where requirements and solutions evolve through collaboration between self-organizing cross-functional teams.*"

Wikipedia.

- **Scrum**
  - *Scrum is an agile process for software development. Scrum consists of predefined milestones and events that scope, estimate, plan and status the project.*

Agile Manifesto

"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- *Individuals and interactions* over *processes and tools*
- *Working software* over *comprehensive documentation*
- *Customer collaboration* over *contract negotiation*
- *Responding to change* over *following a plan*

That is, while there is value in the items on the right, we value the items on the left more."

Reference: http://www.agilemanifesto.org, 2001

What to Use?

"We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

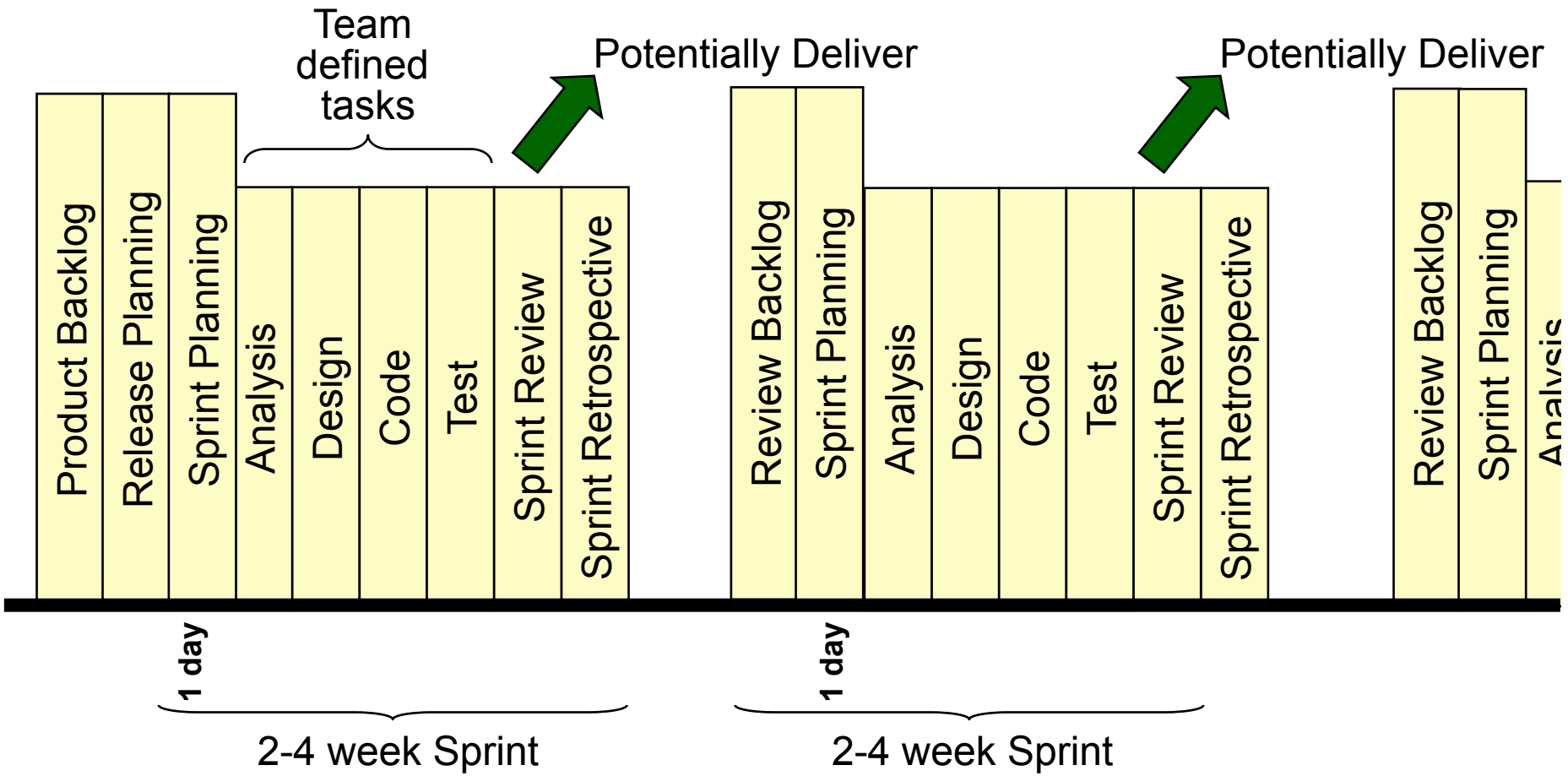
- *Individuals and interactions* over *processes and tools*
- *Working software* over *comprehensive documentation*
- *Customer collaboration* over *contract negotiation*
- *Responding to change* over *following a plan*

Both, based on goals and challenges

That is, while there is value in the items on the right, we value the items on the left more."

Reference: http://www.agilemanifesto.org, 2001

Definition of Scrum



Scrum Positives

Positives
Work is chunked in 2-4 week increments (Sprints).
Scope changes are managed using the Backlog, iteration planning and the rule “No changes during a sprint.”
Small (1-4 week) iterations create team momentum and early feedback on progress and technical solutions.
Scrum process can be learned and used in less than 2 days.
Daily Standup meetings and Burndown charts provide quick and easy project status.
Engineering practices (often used in Agile) can reduce risk: • Automated unit tests. • Pair programming (2 developers working together). • Test driven development (write a test first). • Continuous integration (e.g., daily builds to find problems).

Scrum Risks to Look Out For and What to do About Them

Does 100% Scrum do What You Need?

- Plan for software aspects of certification
- ✔ Software development plan
- Software verification plan
- Software configuration management plan
- Software quality assurance plan
- System requirements
- Software requirements standards
- Software design standards
- Software code standards
- Software design description
- ✔ Source code / Executable object code
- Review of all requirements, design and code
- ✔ Testing of executable object code
- Code coverage analysis
- Unit testing
- Integration testing
- ✔ Black-box and acceptance testing
- Software quality assurance records
- Software conformity review
- Traceability
- Independent testing

If your team is full-Scrum – know what is missing

Mistaking Speed For Progress

Speed vs. Progress

With no:

- **Design:**
  - Architecture + design notes
- **Peer-reviews to find defects**
- **Checks for interface for errors**
- **System test**
- **Analysis of verification results:**
  - e.g., defect density, pass/ escape rate, cause

Great product?

Implement yourself into a corner?

1-Liner Requirements
(The Devil is in The Details)

Requirements / Backlog

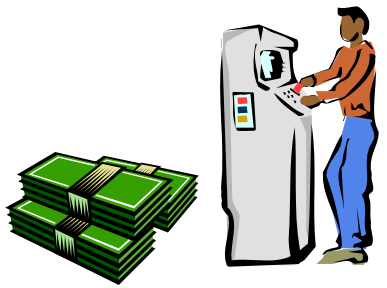
Priority	User Story	Estimate (Points)
	As an account owner, I want to withdraw cash	
	As an account owner, I want to deposit cash	
	As an account owner, I want to deposit check(s)	
	As an account owner, I want to deposit foreign check(s)	
	…	
	…	
	…	

Typical format: As a <user>, desired action

1-liners can be ambiguous and lead the developer to guess

Sample Use Case for an ATM – 1

Automated Teller Machine [Elicitation question]



Name: **Withdraw Cash** [What does the user want to do?]

Actor: **Account Owner** [Who?]

Description:

The user withdraws a specific amount of cash from a specified account.

Trigger: **Account Owner selects Withdrawal action** [What initiates it?]

Preconditions: [What state is the user / system before the event?]

1. The Account Owner is logged in to the ATM.
2. The Account Owner has at least 1 account with a positive balance.
3. The ATM contains cash.

Postconditions: [What state is the user / system after the event?]

1. The requested amount of cash has been dispensed.
2. The account balance is reduced by the amount withdrawn plus any fees.
3. The ATM cash balance is reduced by the amount withdrawn.

Priority: **High**

Sample Use Case for an ATM - 2

Normal flow: [What is the ideal flow?]

Actor Actions	System Responses
1. Select Withdrawal action.	2. Display user's accounts.
3. Select desired account.	4. Ask user to enter amount.
5. Enter desired amount.	6. If amount is okay, dispense cash and debit account.
7. Remove cash from dispenser.	

- Alternative Flow: [Any alternatives?]
  - Step 4: display list of common amounts, let user select one
- Exceptions: [Conditions the system needs to deal with?]
  - Step 6: amount is not a multiple of $20.00
  - Step 6: amount exceeds $400
  - Step 6: amount exceeds account balance
  - Step 6: amount exceeds cash available in ATM

Missing Architecture / Design
Missing Final System Test / Validation

Design

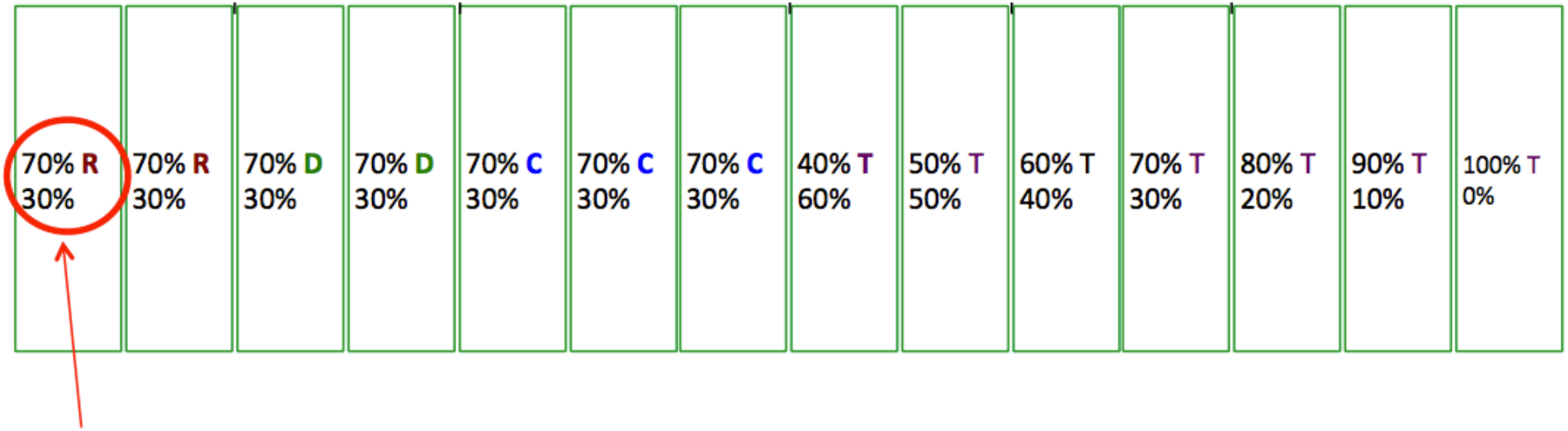
- **Why design?**
  - Identify potential **sticky areas** that need to be investigated.
  - Find **errors** earlier.
  - Clarify and **communicate** concepts and definitions to others.
- **Design example:**
  - **Architecture, interfaces, data definitions, constraints**
    - » e.g., design for testing, expansion, security, portability and technology.
    - » Textual design notes (in backlog or code header), modeling languages, event tables, flow/timing diagrams, pseudo code.

- **You might consider:**
  - Sprint N: 80% design, 20% coding/test.
  - Sprint N+1: 60% design, 40% coding/test.
  - Sprint N+2: 40% design, 60% coding/test.

processgroup.com/monthlytidbits.html#tidbit10

Final System Test / Validation

Sprint



- **70% Requirements**
- **30% Other (Requirements, Design, Code, unit Test, system Test, acceptance Test?)**

processgroup.com/monthlytidbits.html#tidbit12

Missing Configuration Management

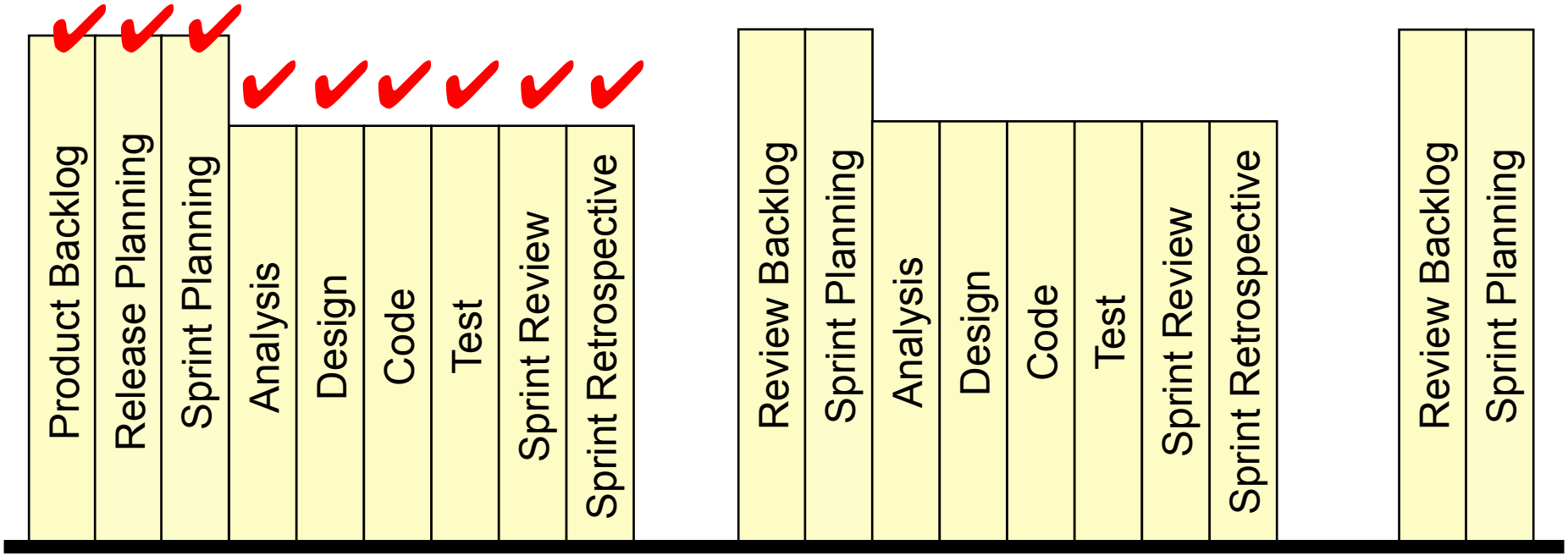
What Version of _ Are You Working On?

What outputs do you want to:

- Not lose
- Share (the same version)
- Know the history of
- Find easily with confidence

Consider:

- Versioning / labeling (baseline)
- Tracking changes after a baseline
- Making sure everyone has version X
- Tools: SVN, GIT, Windchill, SharePoint..

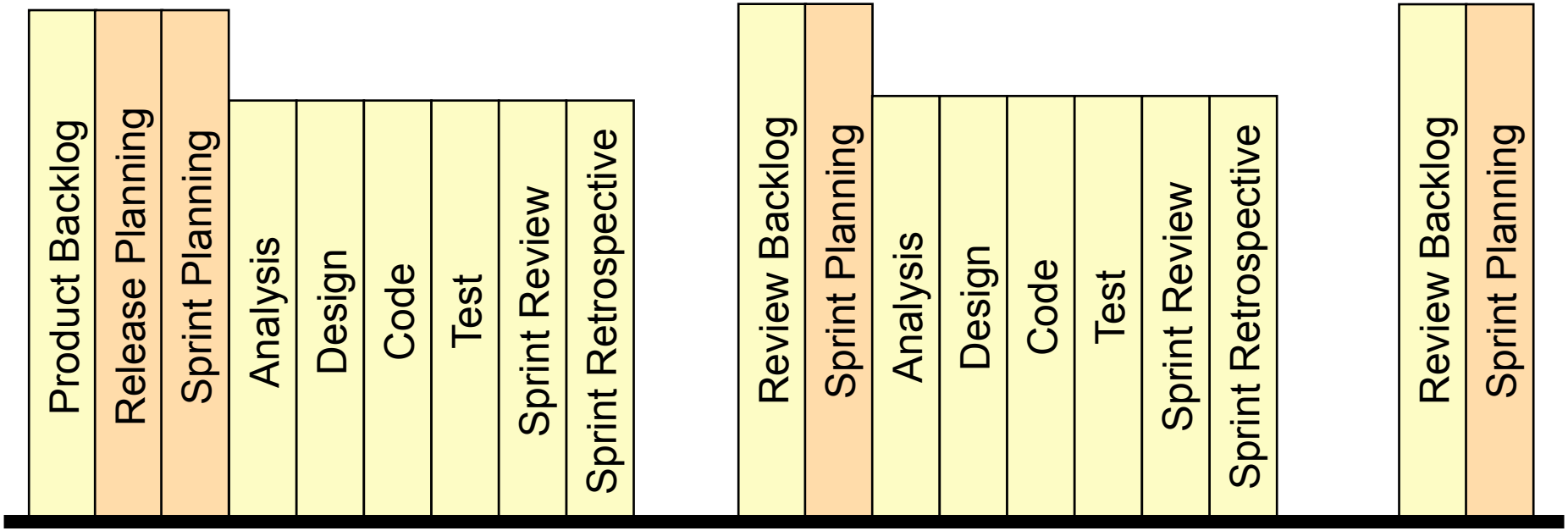


Missing Risk Management

Where to Add Risk Management

Agile perspective
High-risk features are worked on in earlier sprints.

Revision: Add risk management to:
- Release Planning
- Sprint Planning
- Daily Standups



Risk Management

Action items will likely be more numerous and detailed

Status of action items

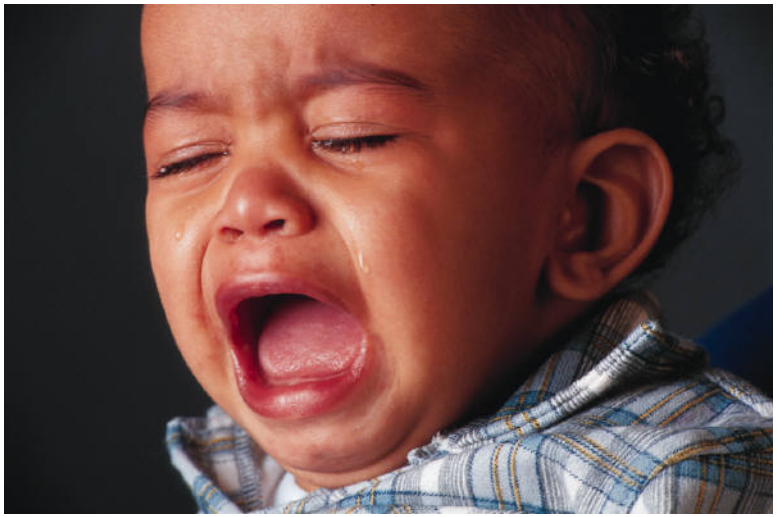
Risk Item (Potential Future Problem)	Consequence	Lkli.	Imp.	Pri.	Action-Likelihood	Action-Impact	Who	When	Status
New operating system might not be stable	Unable to ship	10	10	100	Test OS more	Identify 2nd OS	Joe		
System communication problems with XYZ	Feature x unavailable	8	9	72	Develop sys interface doc	Add replan milestone	Kim		
We may not have the requirements right	Delay next project + no usage of demo	9	6	54	Prototype of UI Use Case style requirements	Incremental release Limit distribution	Lois		
Requirements may change late in the cycle	New defects	7	7	49	Prototype top 10 requirements	Limit distribution	Joe		
Database S/W might be late	Revenue delay	4	8	32	Check with supplier	Help supplier	Pete		
Database expert might leave	Schedule delay	2	10	20	Make sure Jim is happy	Earmark Fred	Jane		
Total				327					

Keep track of the top 20% or top 2-3 risks

Missing Process Assurance (Known as ScrumBut)

ScrumBut

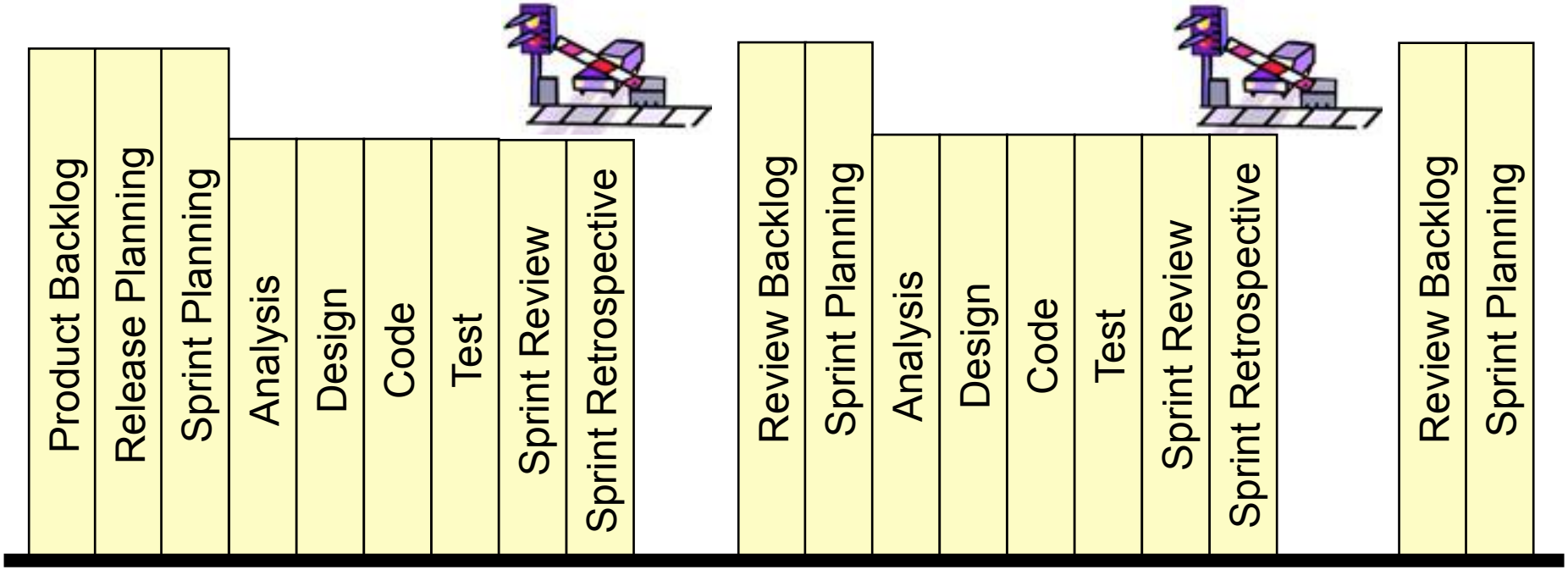
- **"I do Scrum, but I don't do:**
  - User stories
  - Automated testing
  - Sprints
  - Burndown charts
  - Retrospectives



What does your team do?
Are they Agile or Agile-declared?

Adding Gates and Governance

- **Establish gates at the end of each, or several, sprints:**
  - Negotiate upfront what will be available at the gates (some design, some code, some tests).
  - Work with process QA staff early so that there is agreement as to what will be audited and when.



Summary

- **Scrum was initially designed for small co-located teams:**
  - It does not automatically contain everything you want.
- **Not all Agile/Scrum teams actually do Agile/Scrum:**
  - Ask them what they do!
- **Scrum teams might see little value in "documentation":**
  - Use documentation to address communication and memory challenges.
  - Capture information in current tools or team workflow tools (e.g., RallyDev, Jira).
- **Don't be scared to add practices (and not break Scrum):**
  - **Add practices to sprints** to increase quality & reduce risk:
    - » e.g., Requirements, architecture, design – WHILE creating *working code*.

Q&A

Additional Material

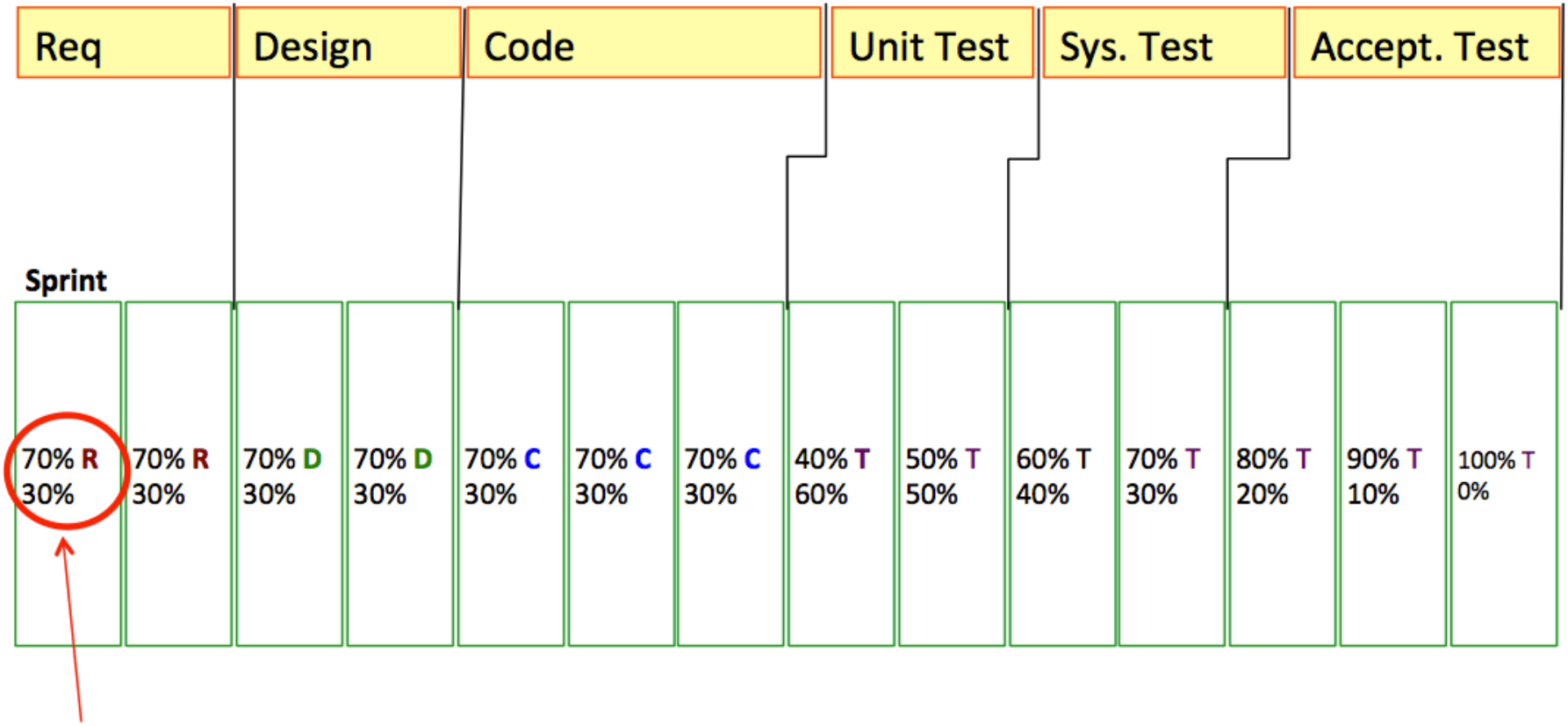
Example Risk Mitigation Actions -1

Potential Problem Using Scrum	Bring Your Own ____	Example
Ambiguous 1-line user stories.	Requirements Analysis	Elicit requirements. Use Cases. Peer review requirements.
Little/no design. Design flaws. Architecture difficult to modify.	Design	Architecture diagram. Larger % of design activity in initial sprints: – Sprint 1: 80% design, 20% code. – Sprint N: 5% design, 95% code.
No system-level testing in sprints leading to system defects.	System testing	Add system testing as a % of later sprints. Add a test/cleanup sprint.

Example Risk Mitigation Actions -2

Potential Problem	Bring Your Own ____	Example
No version control of documents or code leading to loss or incorrect versions being used.	Configuration management	Naming conventions. CM tool / storage definition. Access / edit controls.
Scrum But: "We do Scrum, but we don't have a backlog, Product Owner, sprint backlog or do daily standup meetings."	Process Assurance	Independent member of one Scrum team audits the practices of another Scrum team. Monthly management review of Scrum activities and progress.
Surprises (technical, personnel, supplier) that could have been foreseen and mitigated.	Risk Management	Add a risk assessment to the Sprint planning meeting. Track risks in daily standup meeting.
Scaling for large teams, or many small teams that are dependent upon each other.	Program management	Add system engineering / program management.

Scrum - Under the Hood



- 70% Requirements
- 30% Other (Requirements, Design, Code, unit Test, system Test, acceptance Test?)

processgroup.com/monthlytidbits.html#tidbit12

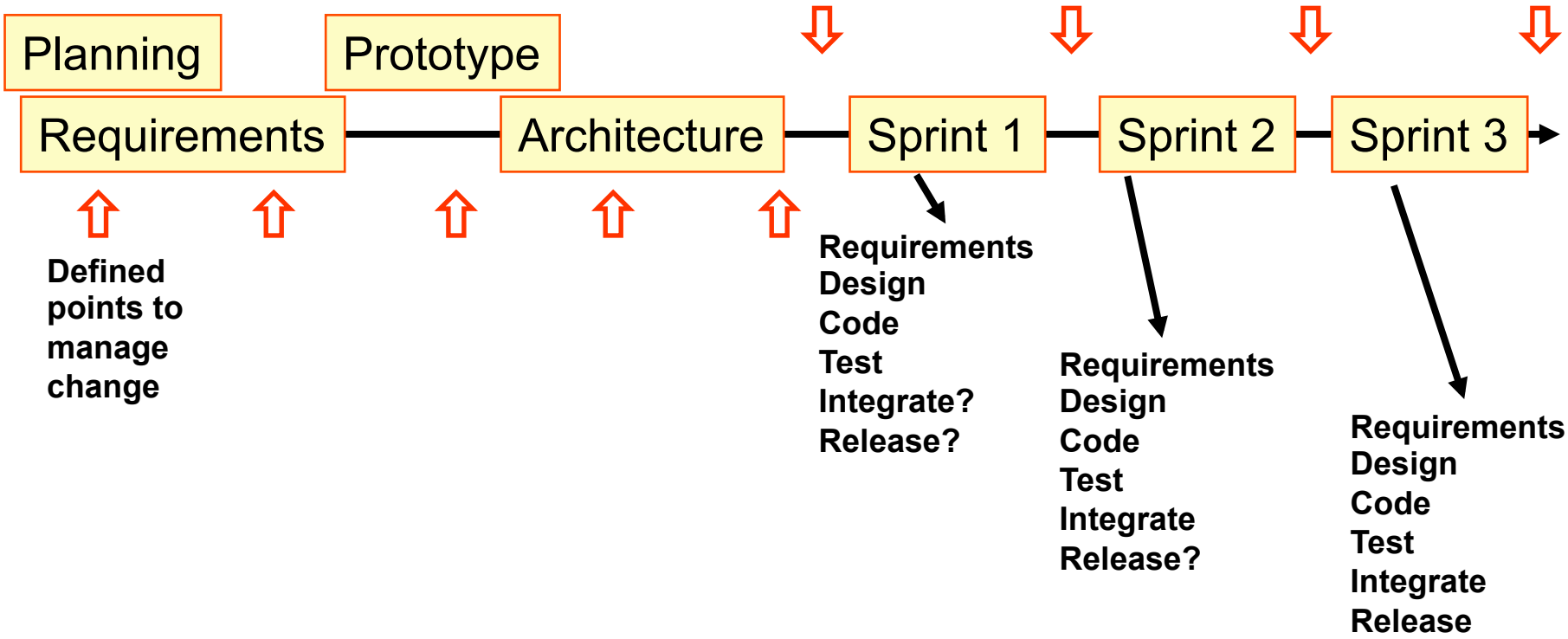
Hybrid or Separate

- **All teams adopt some Agile / Scrum practices:**
  - Increments of 2-4 weeks (for whatever the work is).
  - Potentially deliverable components early on.
  - Burndown charts (based on points and effort).
  - Continuous integration.
  - Pair programming.
  - Daily standup meetings.
- **Alternative - keep traditional & Scrum alive:**
  - Define an architecture that compartmentalizes Scrum and non-Scrum approaches
    - » Scrum for high-risk, unknown sections.
    - » Existing approach for everything else.
    - » The two approaches meet at the system testing phase

[Mike Cohn - Succeeding with Agile]

Incremental Life Cycle

- **Hybrid option:**
  - Simplify existing deliverables from Waterfall.
  - Move to an incremental life cycle (a balance between Scrum and Waterfall).



References

- Implementing Scrum (Agile) and CMMI Together. Process Group Post newsletter, March 2009.
  - http://www.processgroup.com/pgpostmar09.pdf
- Adding Practices to Scrum to Achieve Your Goals (and comparison with CMMI Level 3).
  - http://www.processgroup.com/pgpostapr2013.pdf
- Balancing Agility and Discipline: A Guide for the Perplexed, Boehm and Turner, Addison-Wesley, 2003.
- Agile Methods And Safety-critical Software, Peter Gardner, Silver Atena (YouTube).

