COMP 5700/6700/6706 Software Process

Spring 2016

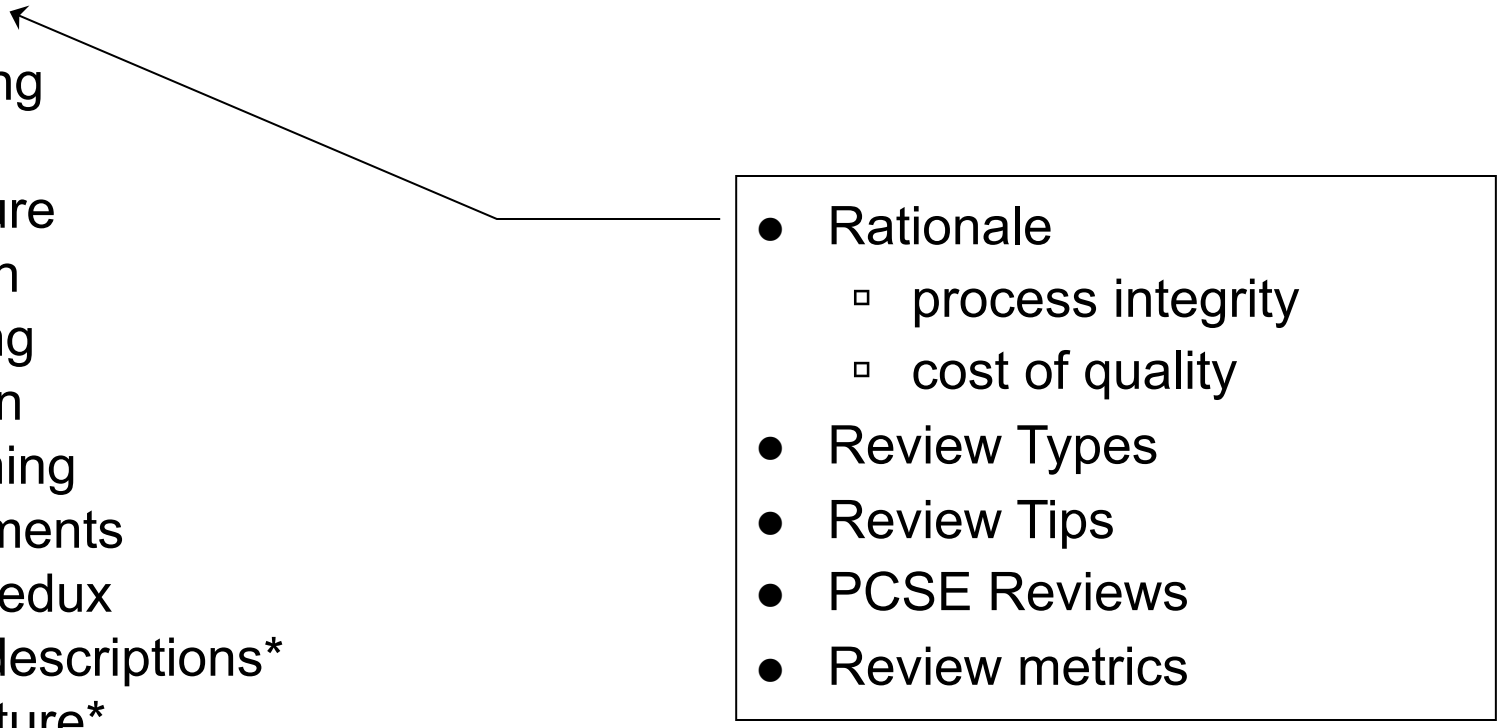
David Umphress

Reviews

- Lesson: Reviews
- Strategic Outcomes:
  - To understand the relevance of reviews
- Tactical Outcomes:
  - To know the types of common reviews techniques
  - To know the rationale of reviews
  - To understand how to conduct a personal review
  - To understand the common review metrics
- Readings:
  - "Code Reviews for Fun and Profit" http://www.aleax.it/osc08_crev.pdf
- Instant take-aways:
  - Personal reviews
- Bookshelf items
  - Fagan, M. 1986. Advances in Software Inspections. *IEEE Transactions on Software Engineering,* vol. SE-12, no. 7.

Syllabus

- Software engineering raison d'être
- Process foundations
- Common process elements
- Construction
- Reviews
- Refactoring
- Analysis
- Architecture
- Estimation
- Scheduling
- Integration
- Repatterning
- Measurements
- Process redux
- Process descriptions*
- Infrastructure*
- Retrospective



COMP5700/6700/6706 Goal Process

Minimal Guiding Indicators

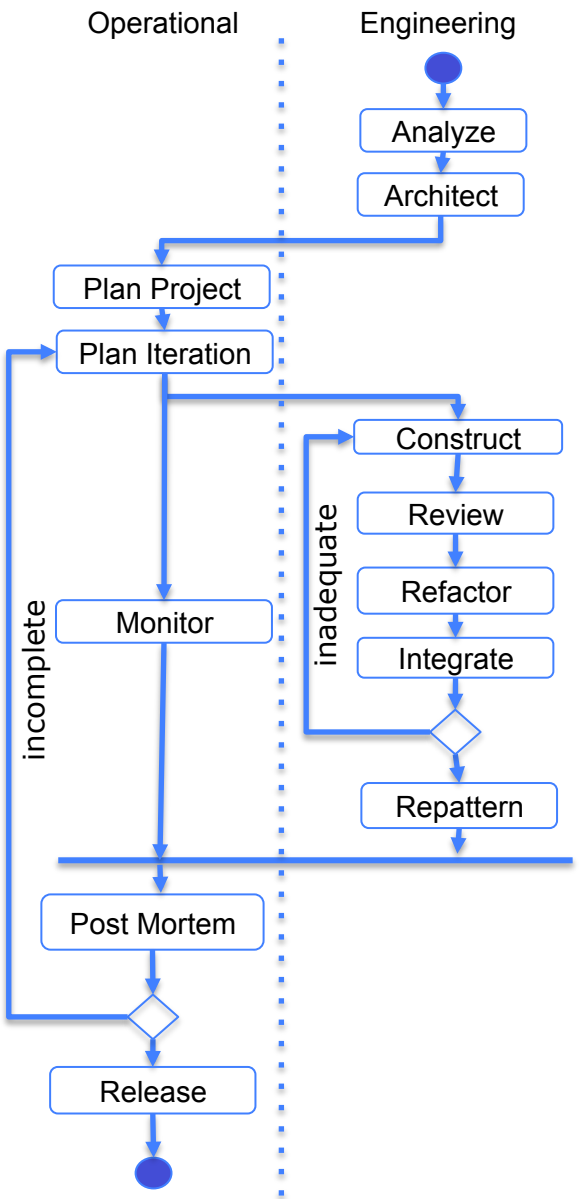
Goal	Indicator
Cost:	None
Schedule:	PV/EV >.75
Performance:	
Product:	
NFR:	none
FR:	100% BVA
Process:	pain < value

Minimal Sufficient Activities

Engineering Activities
- Envision
  - Analyze
- Synthesize
  - Architect
- Articulate
  - Construct
  - Refactor
- Interpret
  - Review
  - Integrate
  - Repattern

Operational Activities
- Plan
  - Plan project
  - Plan iteration
- Monitor
- Release

Minimal Viable Process



Minimal Effective Practice

MSA	MEP
Analyze	Scenarios
Architect	CRC
Plan Project	Component-based estimation
Plan Iteration	Component-iteration map
Construct	TDD
Review	Review checklist Test code coverage
Refactor	Ad hoc sniffing
Integrate	Ad hoc
Repattern	Ad hoc
Monitor	Time log Change log Burndown
Post Mortem	PV/EV
Release	Eclipse zip spreadsheets

Until Now

Minimal Guiding Indicators

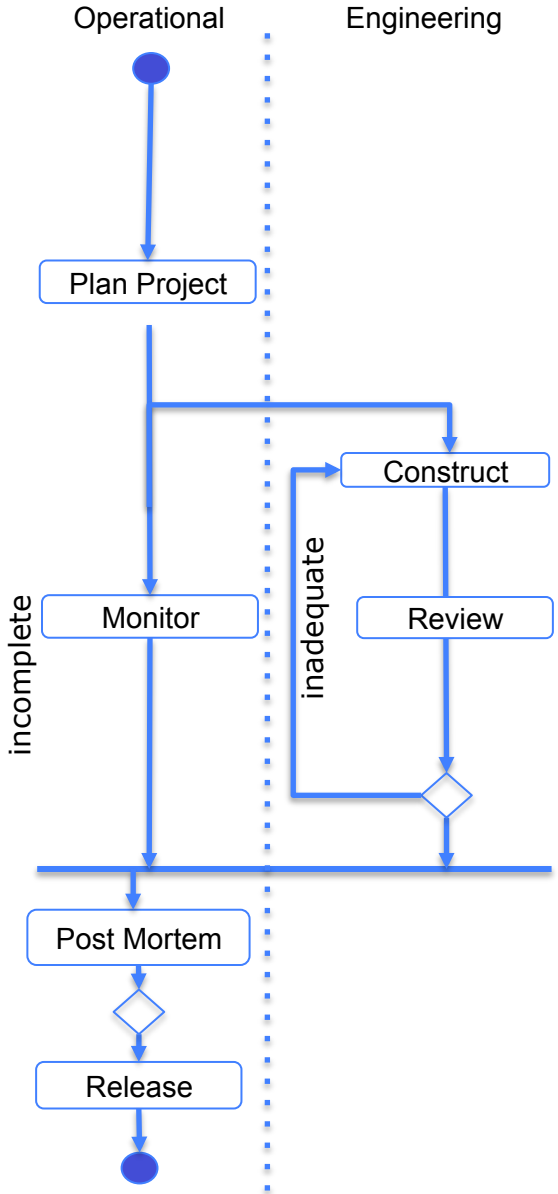
Goal	Indicator
Cost:	None
Schedule:	PV/EV >.75
Performance:	
Product:	
NFR:	none
FR:	100% BVA
Process:	pain < value

Minimal Sufficient Activities

Engineering Activities
- Envision
  - Analyze
- Synthesize
  - Architect
- Articulate
- Interpret

Operational Activities
- Plan
  - Plan project
  - Plan iteration
- Monitor
- Release

Minimal Viable Process



Minimal Effective Practice

MSA	MEP
Plan Project	• Guess projected LOC. • Guess projected effort.
Construct	• TDD • Commit the code to git.
Integration Test	• Run acceptance tests.
Monitor/Cntrl	• Record activities in Time Log. • Record defects/changes in Change Log.
Post Mortem	• Count and record LOC.
Release	• Uploaded zipped Eclipse project. • Upload spreadsheet.

Economics of Quality

50,000 LOC, no reviews

- 25+ defects/KLOC at test entry
- total of 1250 defects
- at typical 10+ hours per defect = 12,500+ programmer hours
- that is 6 programmer years
- if properly planned, these tests could be done in 12 to 15 months

50,000 LOC, reviews

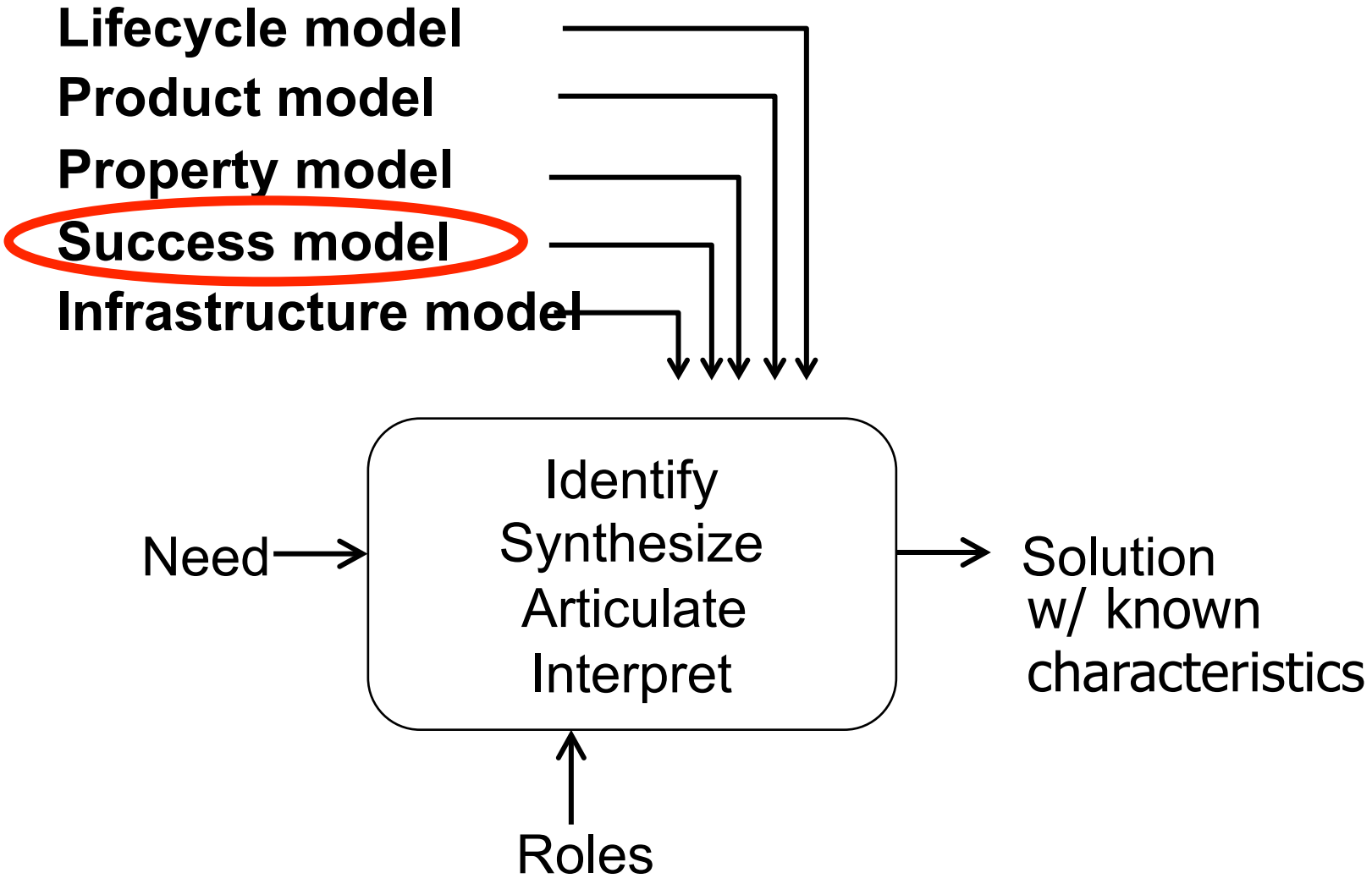
- inspections take about 10 programmer hours per 250 LOC, or about 2,000 hours
- = 1 programmer year
- if done well, inspections can remove ~ 80% of defects
- 250 defects would be left for test = 2,500 hours
- savings = 8,000 hours, or 4 programmer years

Humphrey. 1996. "A Disipline for Software Engineering" Addison Wesley

Economics of Quality

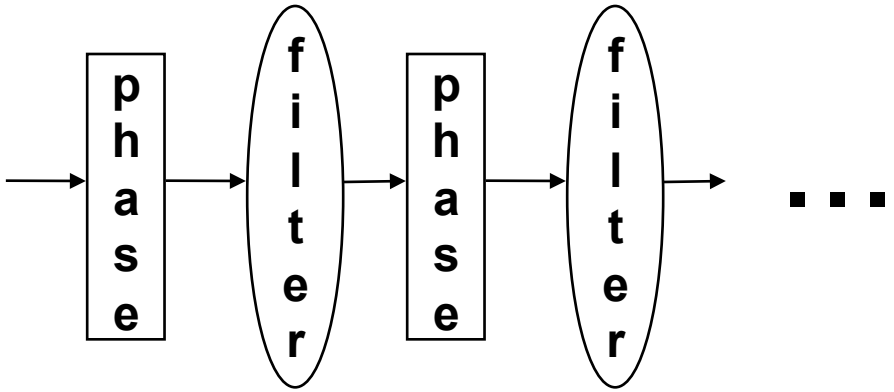
- Issue: Cost of Quality (COQ)
  - failure costs
    - repair, rework, and scrap
    - in PCSE, failure costs include debug and refactoring time
  - appraisal costs
    - costs of inspecting for defects
    - in PCSE, appraisal costs include all review time
  - prevention costs
    - finding and resolving defect causes
    - generally handled before projects start
    - should typically be a process and not a project activity
    - PCSE is intended to be a prevention cost

Process Commonalities -- revisited



Product Integrity

- Determine effective quality methods.
  - Choices:
    - change defect removal
    - change defect injection
  - Concept
    - compound multiple defect removal phases to achieve desired combined yield
    - monitor cost and efficiency of removal phases, achieve most economical combination
    - monitor removal phases for low yields

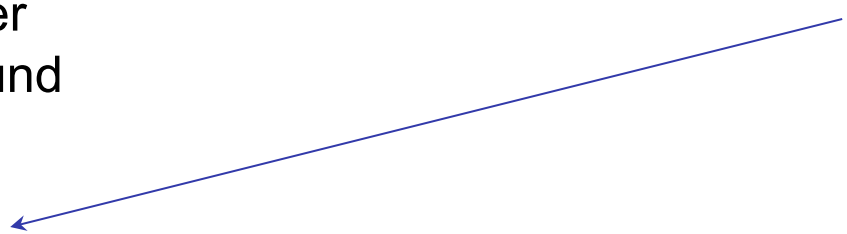


Product Integrity

- Goals
  - ... to seek the best possible match between requirements and end product
  - ... to act as a tool for
    - ensuring all requirements are addressed
    - examining intermediate products for quality
    - gaining visibility into process
  - ... via audits and reviews

Product Integrity

- Reviews are in-process evaluations of work
  - Heavyweight
    - formal reviews
      - conducted in meeting format
      - objective: communicate, obtain approval
    - inspections
      - accomplished in a structured group setting of peers
      - objective: find problems, not solve them
  - Lightweight
    - objective: find obvious defects quickly and cheaply
    - Team
      - over-the-shoulder
      - e-mail pass-around
      - pair
    - Personal
      - introspective



Review Rationale

- Reviews are efficient cuz
  - In testing
    - you start with a problem (which is unexpected and unplanned)
    - then you must search for the bug
    - then you must figure out what defect could cause such behavior
    - next, you devise a fix
    - finally, you implement and test the fix
- With reviews and inspections
  - you see the defect
  - then you devise a fix
  - finally, you implement and review the fix

Reviews

- ... are conducted at end of artifact cycle(s)
- Keys to successful reviews
  - produce an artifact that can be reviewed (where "artifact" = design, code, etc.)
    - a "clean" representation
    - a consistent and clear structure
    - the design's purpose and function be explicitly stated
    - you have criteria for artifact completeness
    - the artifact is structured in logical elements
  - follow an explicit review strategy
    - notes when to perform the review
    - specifies the order in which to review the artifact
    - uses a review checklist
    - establishes accountability

Tips

- LOC under review should be less than 200 and not exceed 400. Larger LOCs tend to overwhelm reviewers.
- Total review time should be less than 60 minutes and not exceed 90. Defect detection rates plummet after 90 minutes.
- Inspection rates less than 300 LOC/hour result in best defect detection. Expect to miss a significant percentage of defects if faster than 500 LOC/hour.
- Expect defect rates of around 15 per hour.
- Authors who prepare for the review with annotations and explanations have far fewer defects than those that do not.

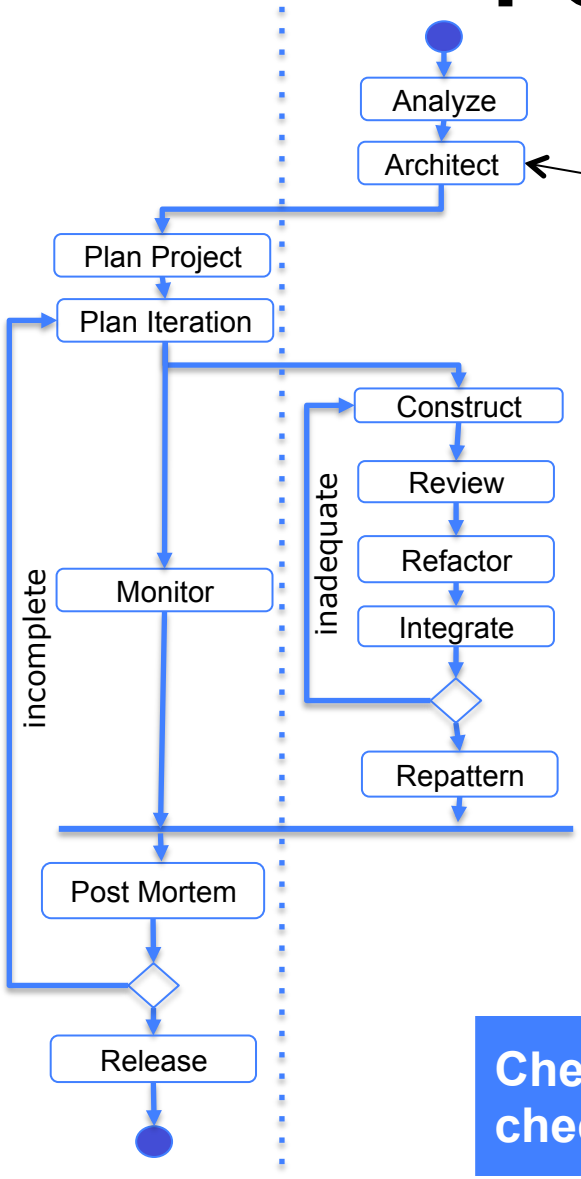
Best Kept Secrets of Peer Code Review. http://www.smartbear.com/codecollab-code-review-book.php

Deadly Sins of Reviews

- Participants don't understand the review process.
- Reviewers critique the producer, not the product.
- Reviews are not planned.
- Review meetings drift into problem-solving.
- Reviewers are not prepared.
- The wrong people participate.
- Reviewers focus on style, not substance.

Karl Wiegers, the Seven Deadly Sins of Software Review.
http://www.processimpact.com/articles/revu_sins.html

PCSE Reviews



Design review

Code review

- "code" = unit tests
- rationale:
  - > premise: if tests are solid, code is likely to be solid
  - > more likely to spot omissions
  - > focus on interface and simplicity
- process: examine test cases for
  - > requirements verification
  - > coverage
  - > completeness

Checklists constructed generic checklists + personal defect log

PCSE Review Checklist

Type	Category	% Occurrence
10	Documentation	1%
20	Build	0%
30	Product syntax	1%
40	Product logic	40%
50	Product interface	25%
60	Product checking	18%
70	Test syntax	5%
80	Test logic	10%
90	Test interface	0%
100	Test checking	0%

PCSE Review Checklist

Type	Category	% Occurrence
81	Algorithm deficiency	5%
82	Lack of understanding	20%
83	Requirements change	10%
84	Design inconsistency	5%

DAU's

Review Metrics

How would you measure a review?

Measure

Summary

Topics

- Rationale
  - process integrity
  - cost of quality
- Review Types
- Review Tips
- PCSE Reviews
- Review metrics

Key Points

- Quality should be viewed as an economic issue: defect removal, defect prevention are facets of quality improvement
- Reviews = proactive debugging; unit test = reactive debugging
- Must do's for reviews:
  - establish review goals
  - follow disciplined process
  - measure & improve results
- Common reviews:
  - heavyweight
  - lightweight
    - PCSE is lightweight with checklist tailored to personal defects

DAU's Review Checklist

Design

#	Item	Reviews	Hits
1	Are CRC cards semantically complete?	15	3
2	Are abnormal scenarios articulated?	18	10
3	Are acceptance tests identified?	10	4

Construction

#	Item	Reviews	Hits
1	Does a test case exist for each method?	18	4
2	Do test cases provide .GE. EQP coverage?	18	3
3	Does setup code avoid assumptions?	12	5
4	Do all tests pass after clean up?	10	5
5	Do all tests trace to an acceptance test?	10	0
6	Are tests named with a story?	10	4
7	Do *callable* tests use correct parms?	4	2

return

Review Metrics

- Explicit measures
  - the size of the program being reviewed
  - the review time
  - the numbers of defects found
  - the numbers of defects not found: “escapes”
- Derived measures
  - review yield: %found
  - LOC reviewed per hour
  - defects found per KLOC
  - defects found per review hour

return