Agile Metrics

Dave Nicolette
Senior Consultant, Valtech Technologies

Definition:
Agile software development

An approach to software development based on the values and principles expressed in the Agile Manifesto.

http://www.agilemanifesto.org

Definitions

Metric

A standard for measuring or evaluating something; basis for asessment

Measure

A proportion, quantity, or degree

Quantify

To express in quantitative terms or as a numerical equivalent

http://www.yourdictionary.com

What is this about?

Measuring and tracking agile software development projects.

- ✧ Why do we measure?
- ✧ What do we measure?
- ✧ How do we measure?
- ✧ Who cares?

Why do we write software?

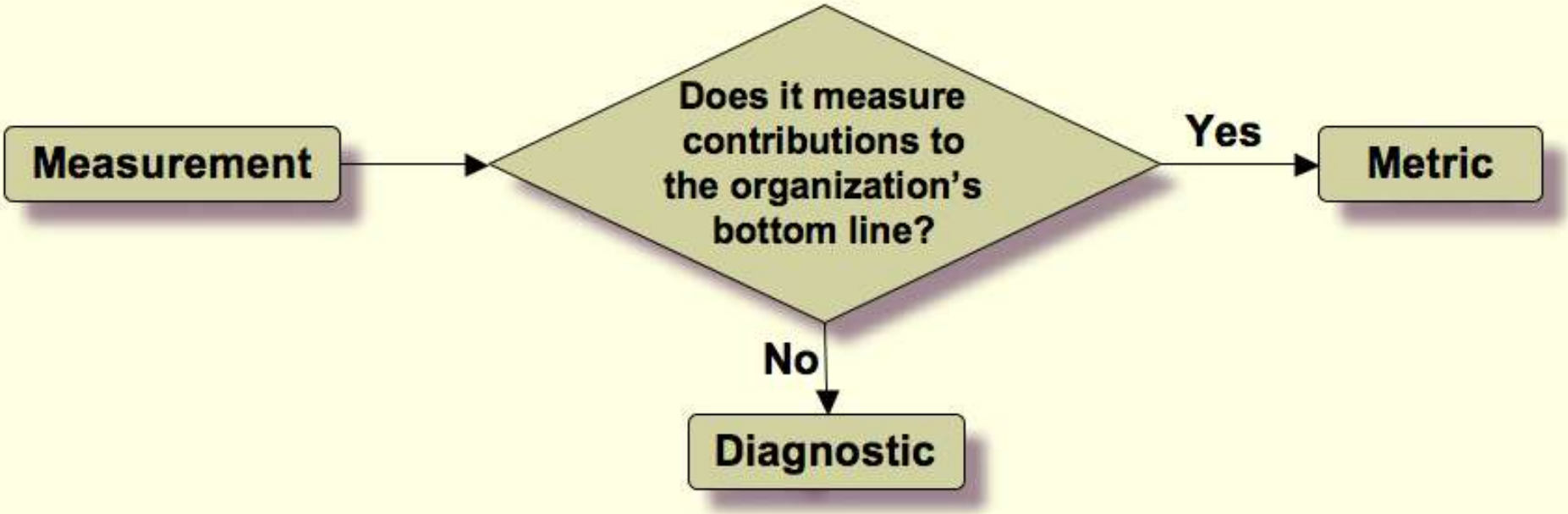
So, what do we want to measure?

Why measure anything else?

Metrics versus Diagnostics

A metric measures something of direct value to the business

A diagnostic measures something about our ability to produce the thing of value



Pete Behrens, Trail Ridge Consulting

How can we use metrics*?

* in the strict sense

Uses for metrics (measures of business value)

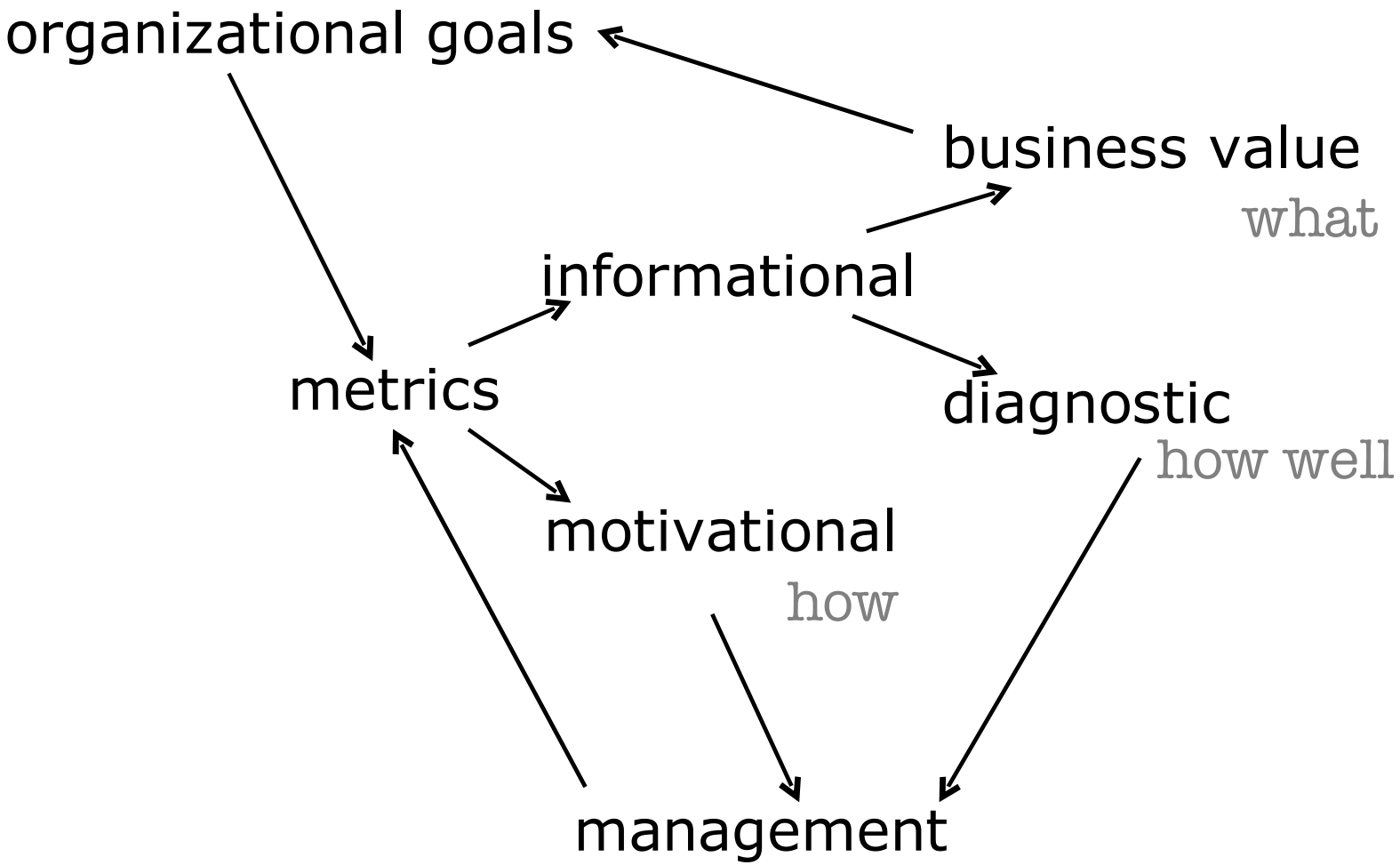
- ✧ Understand real return on investment
- ✧ Plan market strategies, product releases, and advertising campaigns
- ✧ Fail fast and minimize losses
- ✧ Identify and mitigate business risks
- ✧ Award bonuses to software developers

How can we use diagnostics?

Tell me how you are going to measure me and I will tell you how I will behave.

- Eliyahu Goldratt

Another view of metrics



Agile Thinking About Metrics

- Minimum necessary to stakeholders, and no more.

- Every measurement has a purpose. (Don't measure something just because you can.)

- Measure outcomes, not outputs

- Measure work items completed, not time spent per task.

- Assess trends, not snapshots

Who cares about software projects?

Team member

Product Owner

ScrumMaster

Project Manager

Executive

Production Support

Auditor

Process Improvement

Researcher

Team Member

Cares about…

technical quality
impediments

Scope of interest…

team

team
IT organization
business unit
enterprise

Timeframe of interest…

day - iteration

day - iteration
iteration - release
release - project
project - program
all

Product Owner

Cares about…

time to market
alignment with business needs

Scope of interest…

business unit
team
IT organization
enterprise

Time frame of interest…

iteration - release
day - iteration
release - project
project - program
all

business unit
iteration - release

Auditor

Cares about…

compliance with regulations
traceability

Scope of interest…

enterprise
team
IT organization
business unit
program
industry

Timeframe of interest…

enterprise
day - iteration
iteration - release
release - project
project - program
all

Who cares about software projects?

Team member

Product Owner

ScrumMaster

Project Manager

Executive

Production Support

Auditor

Process Improvement

Researcher

Basic metrics for agile project teams

Principle

Working software is the primary measure of progress.

Measure

Running Tested Features (RTF)

Metric

Direct measure of delivered results.

Diagnostic

If flat or declining over time, a problem is indicated.

Motivational

Team members naturally want to see RTF increase.

Basic metrics for agile project teams

Principle

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Measure

Earned Business Value (EBV)

Metric

Direct measure of delivered results.

Diagnostic

Trend should be an S curve; otherwise, problems in prioritization or valuation are indicated.

Motivational

Team members like to deliver value because it makes them feel they are contributing to the success of the organization. Stakeholders are motivated to pay attention to the business value of incremental releases.

Basic metrics for agile project teams

Principle

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Measure

Burn

Metric

Direct measure of work remaining.

Diagnostic

Trends show scope changes and likely completion dates.

Motivational

Team members are motivated by seeing clearly when they are likely to finish the project.

Basic metrics for agile project teams

Principle

Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

Measure

Velocity

Metric

Direct measure of delivered results.

Diagnostic

Patterns in trends in velocity indicate various problems.

Motivational

Team members are encouraged and inspired by positive feedback from the customer. They take pride in achieving a high velocity and keeping it stable.

Basic metrics for agile project teams

Principle

Continuous attention to technical excellence and good design enhances agility.

Measure

Static code analysis, test coverage

Metric

N/A

Diagnostic

Various direct and indirect indicators of code quality.

Motivational

Team members are motivated to keep the codebase clean so that static code analysis results will indicate high quality work.

Static Code Analysis Example

Clover Coverage Report - DAS Sonar(Aggregated)
Coverage timestamp: Wed Oct 15 2008 00:02:13 EDT
Overview Package File
FRAMES NO FRAMES SHOW HELP

Statistics for project Clover database Wed Oct 15 2008 00:02:13 EDT:

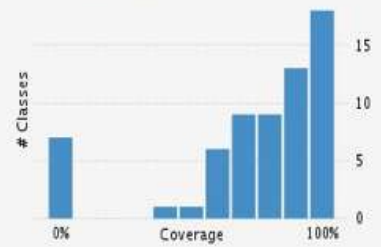
Stmts:	2,170	**LOC:**	6,685	**Total cmp:**	862
Branches:	508	**NCLOC:**	4,728	**Cmp density:**	0.4
Methods:	385	**Files:**	92	**Avg method cmp:**	2.24
Classes:	96	**Packages:**	18		

Stmts/Method: 5.64
Methods/Class: 4.01
Classes/Pkg: 5.33

Coverage 96 classes, 2,305 / 3,063 elements
75.3%

Test Results 111 / 112 tests 0.73 secs
99.1%

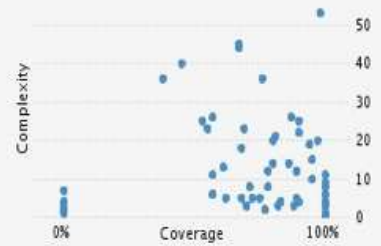
Class Coverage Distribution



Top 20 Project Risks

StateManagementFaultHandler DeviceServiceFaultHandler CustomerEligibilityDAODBImpl BeanUtil BeanUtil.AccessibleObjectComparator XMLBeanUtil ServicesServiceFaultHandler DtaOmServiceFaultHandler DeviceServicesAssembler AccountServiceFaultHandler XTrustProvider CustomerComponentImpl SetTopDeviceAssembler OrderManagementDAODBImpl FedexShippingStatusDAODBImpl LoggingAspect OIMDtaServiceAssembler XMLUtil XTrustProvider.TrustManagerFactoryImpl CustomerQueryDAODBImpl

Class Complexity



Least Tested Methods

1. 0% AccountServiceFaultHandler.handleServiceFaultForQueryDevices(ServiceFault) : void (8)
2. 0% AccountServiceFaultHandler.handleServiceFaultForGetExistingCustomerFinancialHistory(ServiceFault) : void (6)
3. 0% AccountInfo.equals(Object) : boolean (7)
4. 0% CustomerEligibilityDAODBImpl.queryEligibility(AccountId) : CustomerEligibility (4)
5. 0% CustomerComponentImpl.queryCustomerFinancialHealth(AccountId) : AccountFinancialHealth (5)
6. 0% CustomerComponentImpl.lookupAccountInfoByPhoneNumber(long) : List<AccountInfo> (4)
7. 0% CustomerComponentImpl.lookupAccountInfoByAltPhoneNumber(long) : List<AccountInfo> (4)
8. 0% CustomerQueryDAODBImpl.searchAccountByAccountNumber(AccountId) : AccountInfo (3)
9. 0% RoutingLookupResponse.RoutingLookupResponse(String,String,String,String,String,String,String,String,String,String,String,String) (1)
10. 0% CustomerComponentImpl.checkEligibility(AccountId) : CustomerEligibility (3)
11. 0% BeanUtil.toStringV2(Object) : String (4)
12. 0% LoggingAspect.logForDAO(ProceedingJoinPoint) : Object (1)
13. 0% LoggingAspect.logForService(ProceedingJoinPoint) : Object (1)
14. 0% OrderManagementDAODBImpl.submit(SubmitOrderRequest) : OrderId (2)
15. 0% AccountInfo.hashCode() : int (1)
16. 0% XTrustProvider.install() : void (2)
17. 0% OrderManagementComponentImpl.trackOrder(String) : CarrierTrackingInfo (2)
18. 0% AccountId.hashCode() : int (2)
19. 0% DasFixtures.createDasEventFixture() : DasEvent (1)
20. 0% DasFixtures.createAddDeviceRequest() : AddDeviceRequest (1)

Most Complex Packages

1. 84.7% com.comcast.ivr.das.business (219)
2. 65.2% com.comcast.ivr.das.integration.dao.espservices (178)
3. 66.1% com.comcast.ivr.das.util (117)
4. 75.5% com.comcast.ivr.das.integration.dao.customer (60)
5. 70.5% com.comcast.ivr.das.domain.account (54)

Static Code Analysis Example

warns of
large methods

Statistics for project Clover database Wed Oct 15 2008 00:02:13 EDT:

Stmts:	2,170	LOC:	6,685	Total cmp:	862	Stmts/Method:	5.64
Branches:	508	NCLOC:	4,728	Cmp density:	0.4	Methods/Class:	4.01
Methods:	385	Files:	92	Avg method cmp:	2.24	Classes/Pkg:	5.33
Classes:	96	Packages:	18				

not covered
by tests

cyclomatic
complexity

Financial Metrics for Agile Projects

Budget burn

✧ of interest to project managers, program managers, and CIO
✧ tracks the use of budgeted funds over time
✧ says nothing about the value of the project

ROI calculation (budget basis)

✧ compares cost of delivery with project budget
✧ not “real” business ROI
✧ may show that the project team delivered at lower cost than anticipated (or not)

What happened to...

- ...tracking team members' time?
- ...tracking bugs and bug fixes?

Projecting Completion Dates

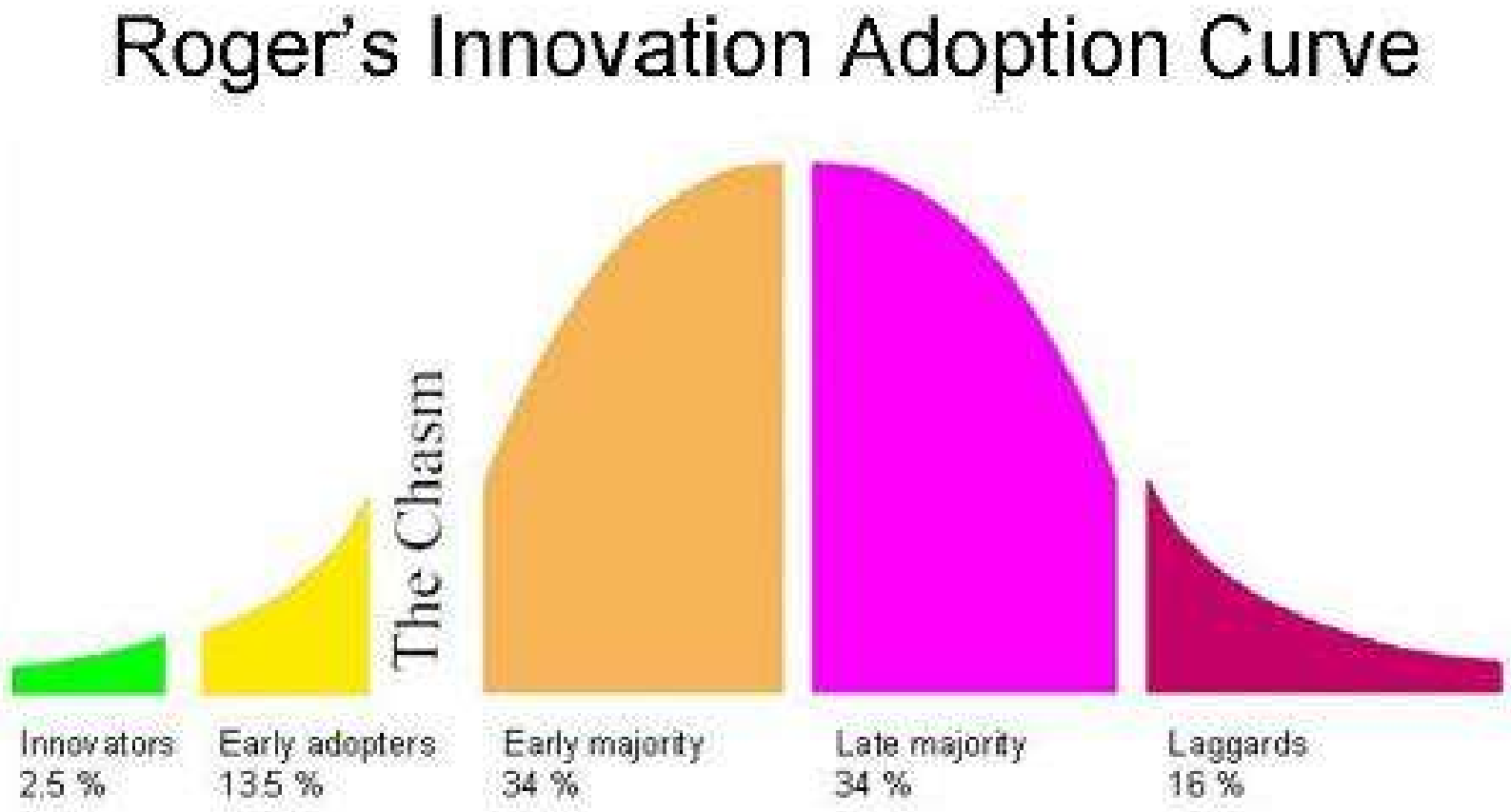
Initial projection based on:

- Team's high-level sizing of overall scope
- Scope growth buffer (judgment call, 20% - 100%)
- New team? Iter 1 velocity will be about 60% normal
- Unfamiliar domain? 20% velocity hit in Iter 1
- Unfamiliar technology? 20% velocity hit in Iter 1
- Assume 20% improvement per iteration
- Use observed velocity from Iter 1 as starting point
- Plug assumed values for a few iterations, draw trend

Refine projection in each iteration as more is learned:

- More observations of actual velocity
- Better understanding of scope
- Improved team cohesion and collaboration

An additional challenge

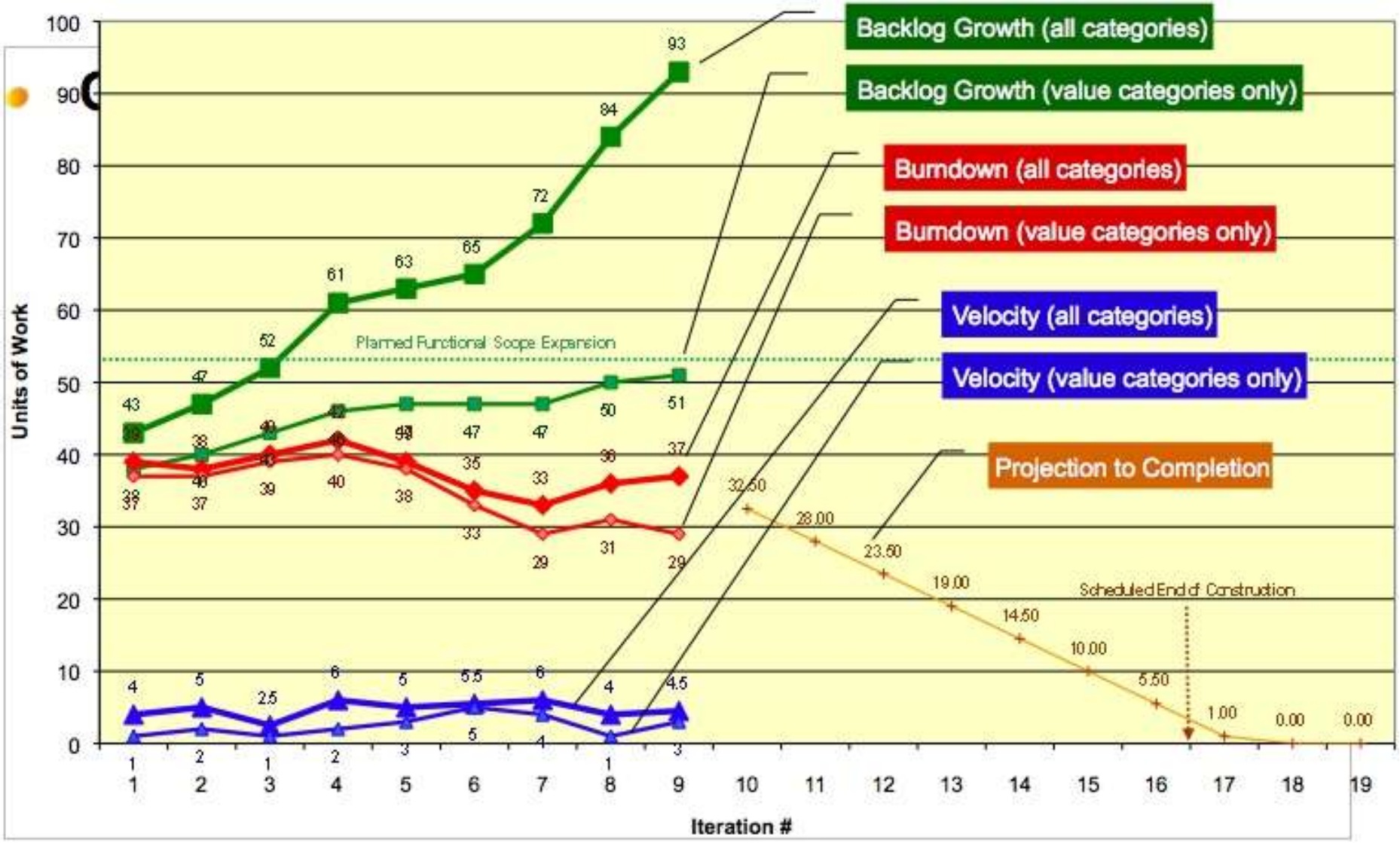


Everett Rogers, 1983. This version: http://suewaters.wikispaces.com/Rogers

Organizational Differences

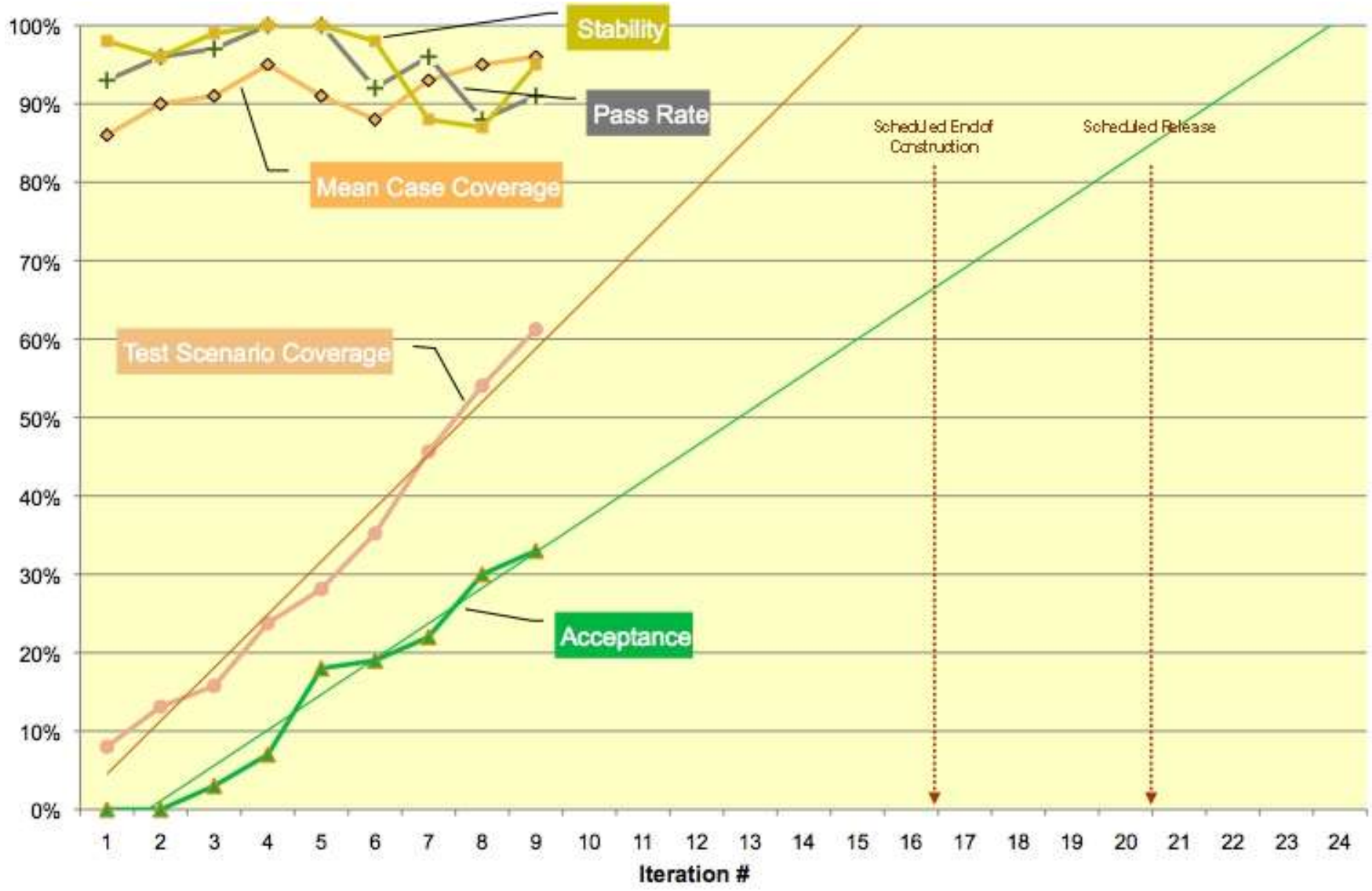
Traditional organization	Agile/Lean organization
Culture Risk aversion, blame-shifting, competition, zero-sum thinking, fear of failure	Risk management, trust, transparency, collaboration, failure as learning opportunity
Structure Administrative separation between application developers and their customers	Application developers work for the lines of business they serve; central IT is for central functions
Management philosophy Command-and-control, Theory X, crack the whip	Self-organizing teams, Theory Y, enable and support people
Teams Temporary assignment, multiple assignment, functional silos	Stable teams, dedicated teams, cross-functional teams
Financial management Cost-center mentality; Cost Accounting	Profit center mentality; Throughput Accounting

Release Progress Report Card



Credit: Al Goerner, Management Consultant

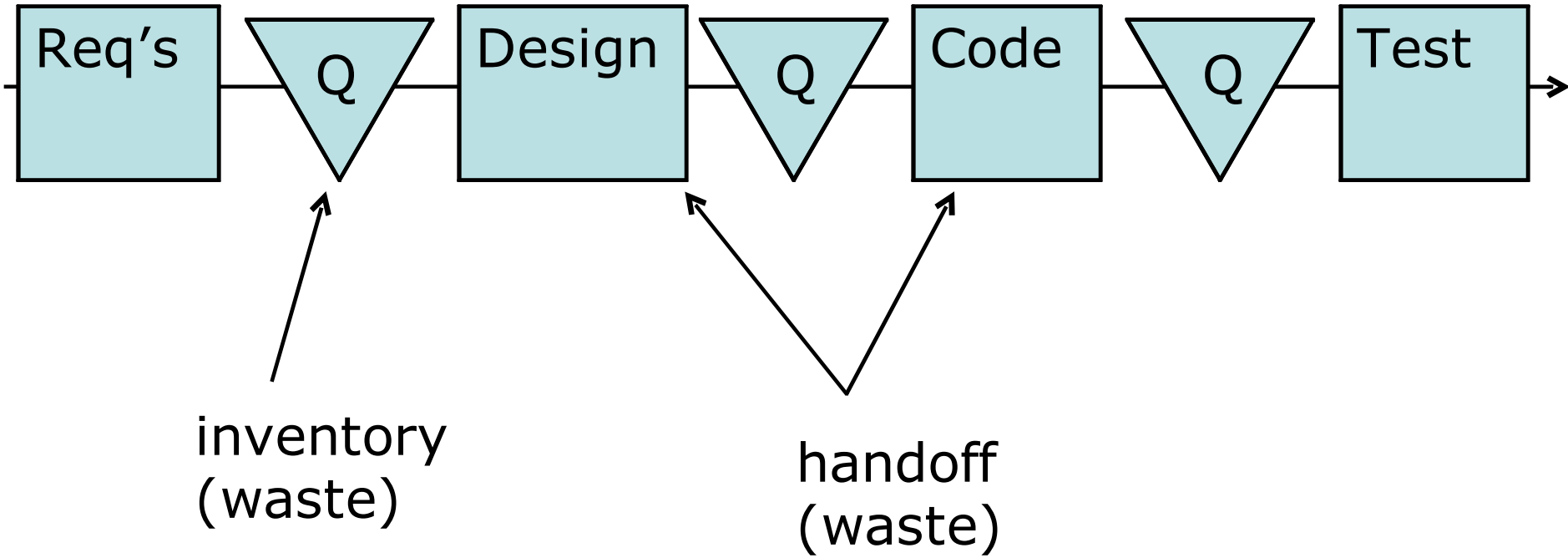
Release Quality Report Card



Credit: Al Goerner, Management Consultant

Iterative Waterfall

Repeat this pattern in each iteration:



Little's Law

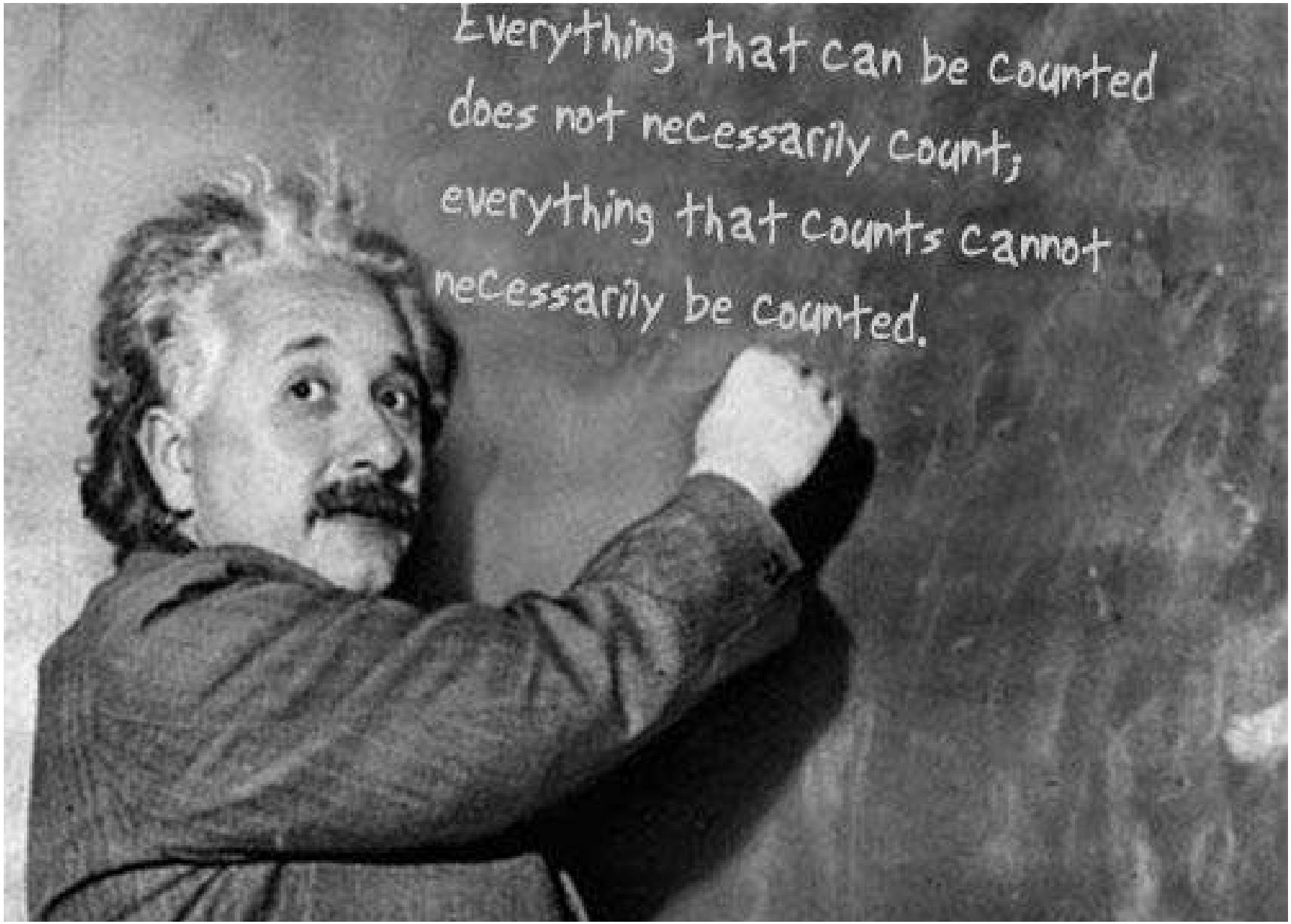
$$LT = WIP / ACR$$

LT = Lead Time
WIP = Work in Process (units)
ACR = Average Completion Rate (units per time period)

Story Cycle Time

The number of iterations it takes to complete a story.

Einstein's Wisdom

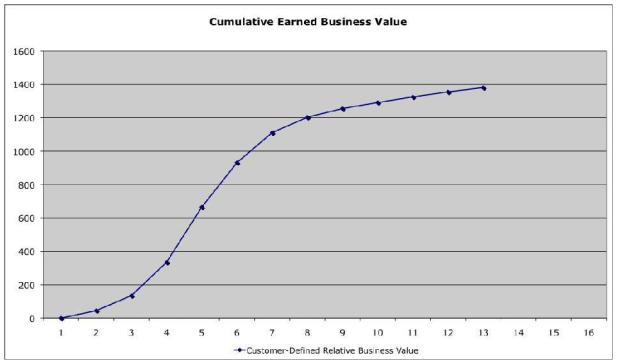


Metrics and Status Reporting

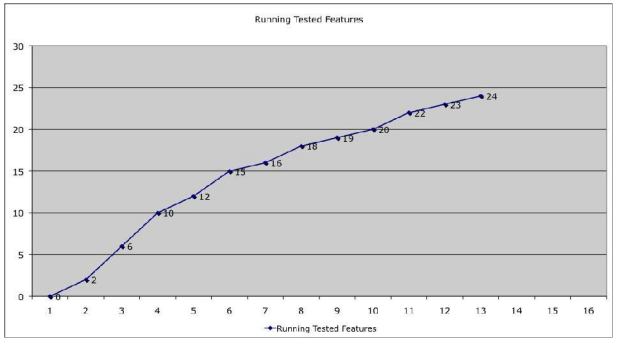
Don’t change slides yet!

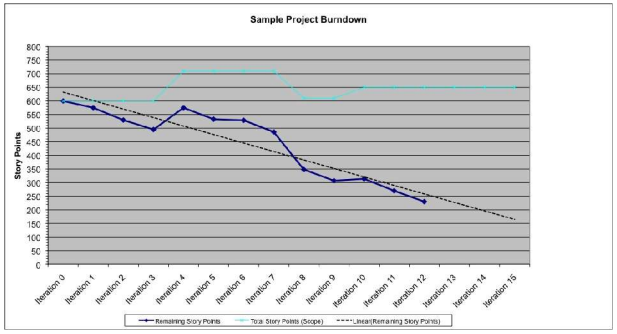
Sample Scorecard

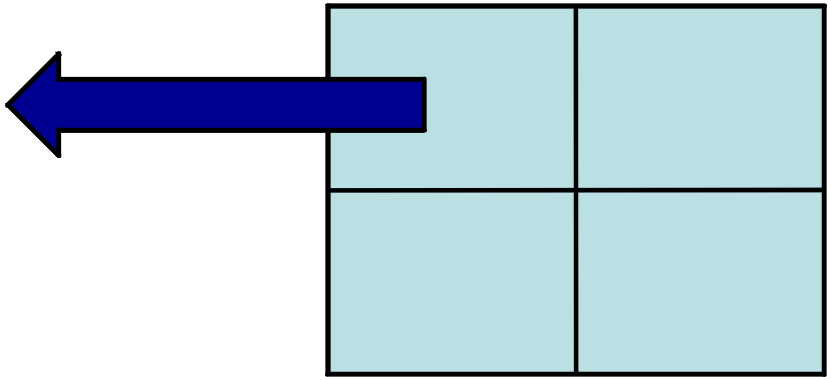
Value Delivery



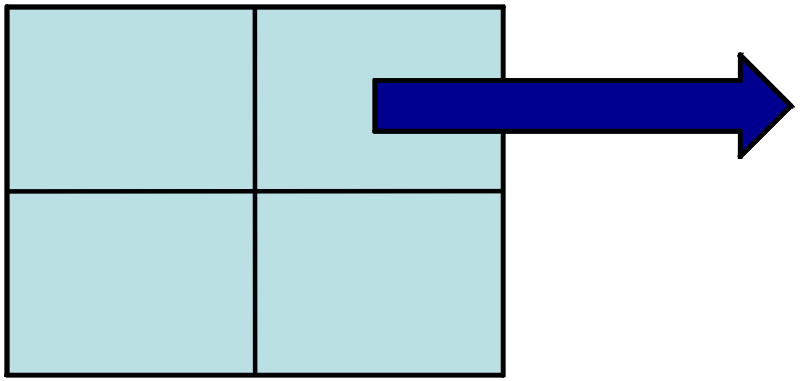
Risks



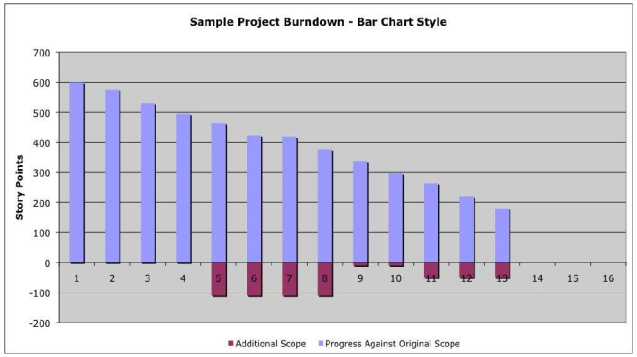


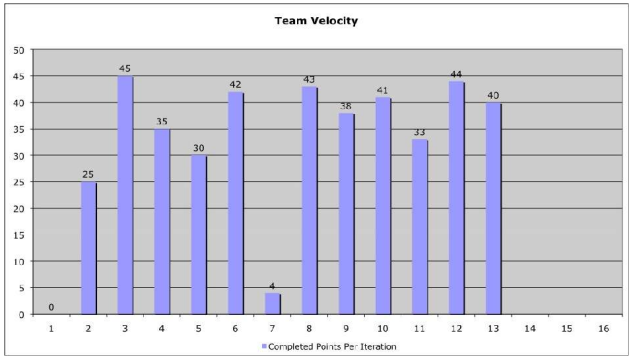


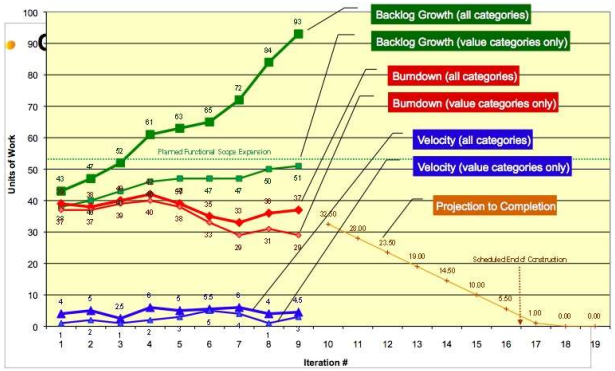
Sample Scorecard



Delivery Effectiveness





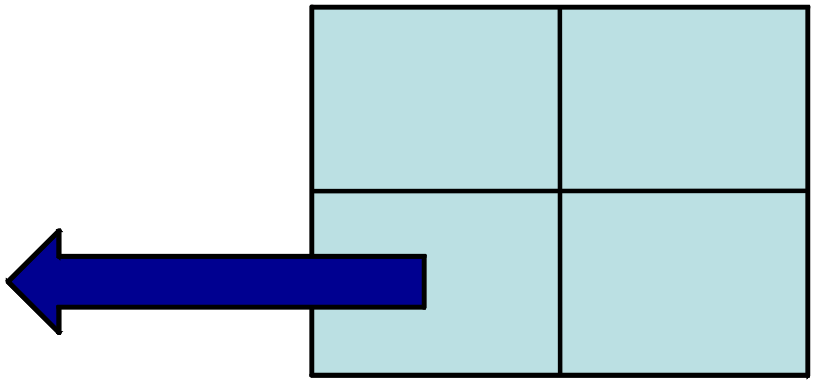


Story Cycle Time: 2

Sample Scorecard

Software Quality

- Customer satisfaction
- Non-functional requirements
- Testing metrics
  - Coverage
  - Tests passing
  - Least-tested components
- Static code analysis metrics
  - Cyclomatic complexity
  - Structural complexity
  - Cyclic dependencies
- Observational/calculated
  - Defect density



Throughput Accounting & Agile Development

Inventory (V)

raw materials — Product Backlog, Story List

Investment (I)

money sunk into Inventory — cost of backlog creation

Operating Expense (OE)

all costs other than Investment — all costs other than backlog management, including direct labor

Throughput (T)

sales price less OE — project budget less OE

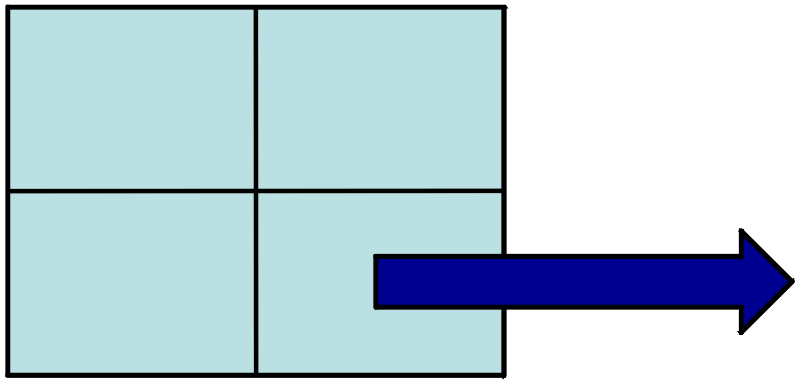
Net Profit (NP)

T – OE — T – OE

Return on Investment (ROI)

NP / I — NP / I

Sample Scorecard



Continuous Improvement

- Build frequency
- Escaped defects
- Use of TDD
- Big-bang refactorings
- Pairing time vs solo time
- Overtime
- Issues from Retrospective

Agile Balanced Metrics (Forrester)

Operational Excellence

- Project Management
- Productivity
- Organizational Effectiveness
- Quality

User Orientation

- User Satisfaction
- Responsiveness to needs
- Service Level Performance
- IT Partnership

Business Value

- Business value of projects
- Alignment with strategy
- Synergies across business units

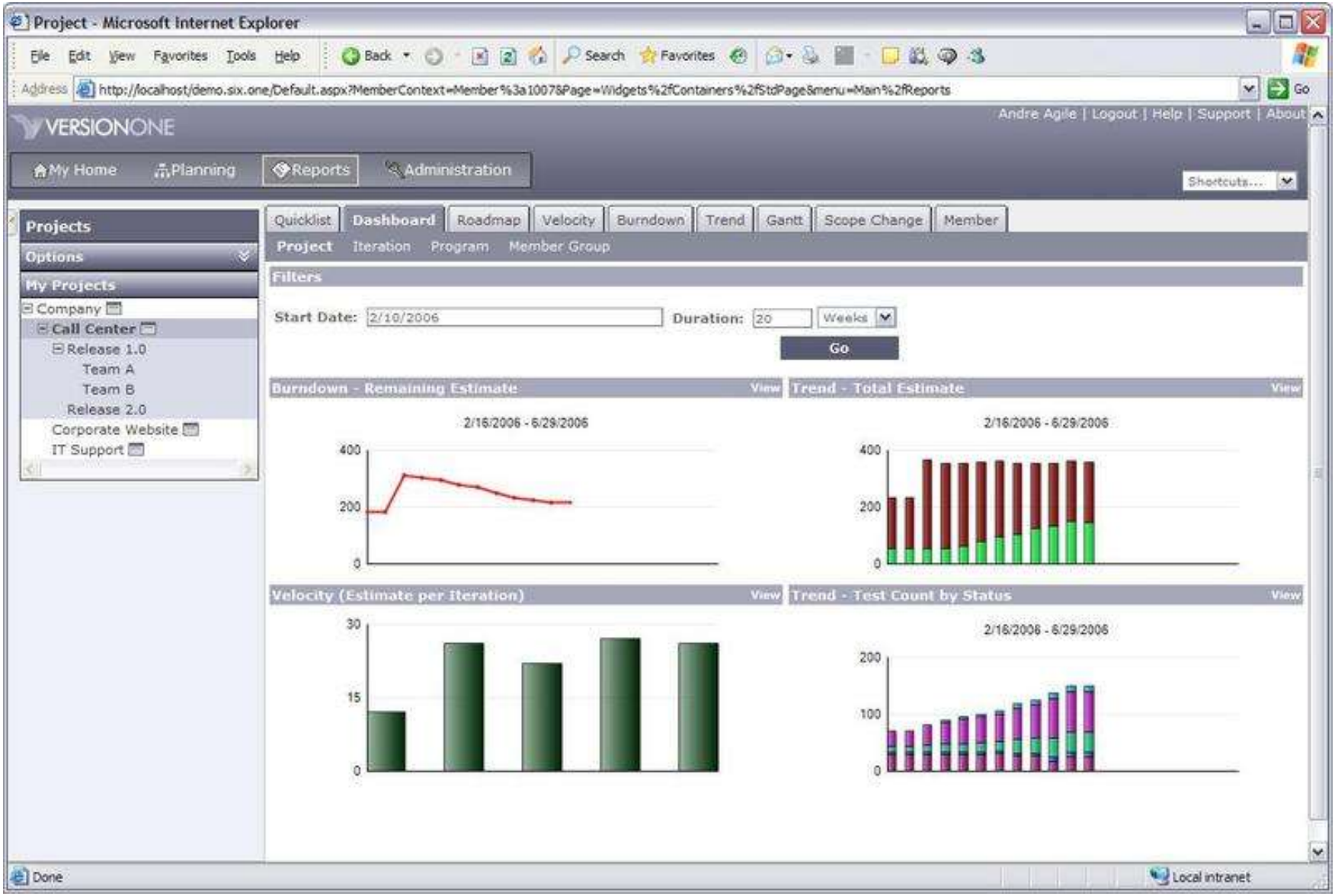
Future Orientation

- Development capability improvement
- Use of emerging processes and methodologies
- Skills for future needs

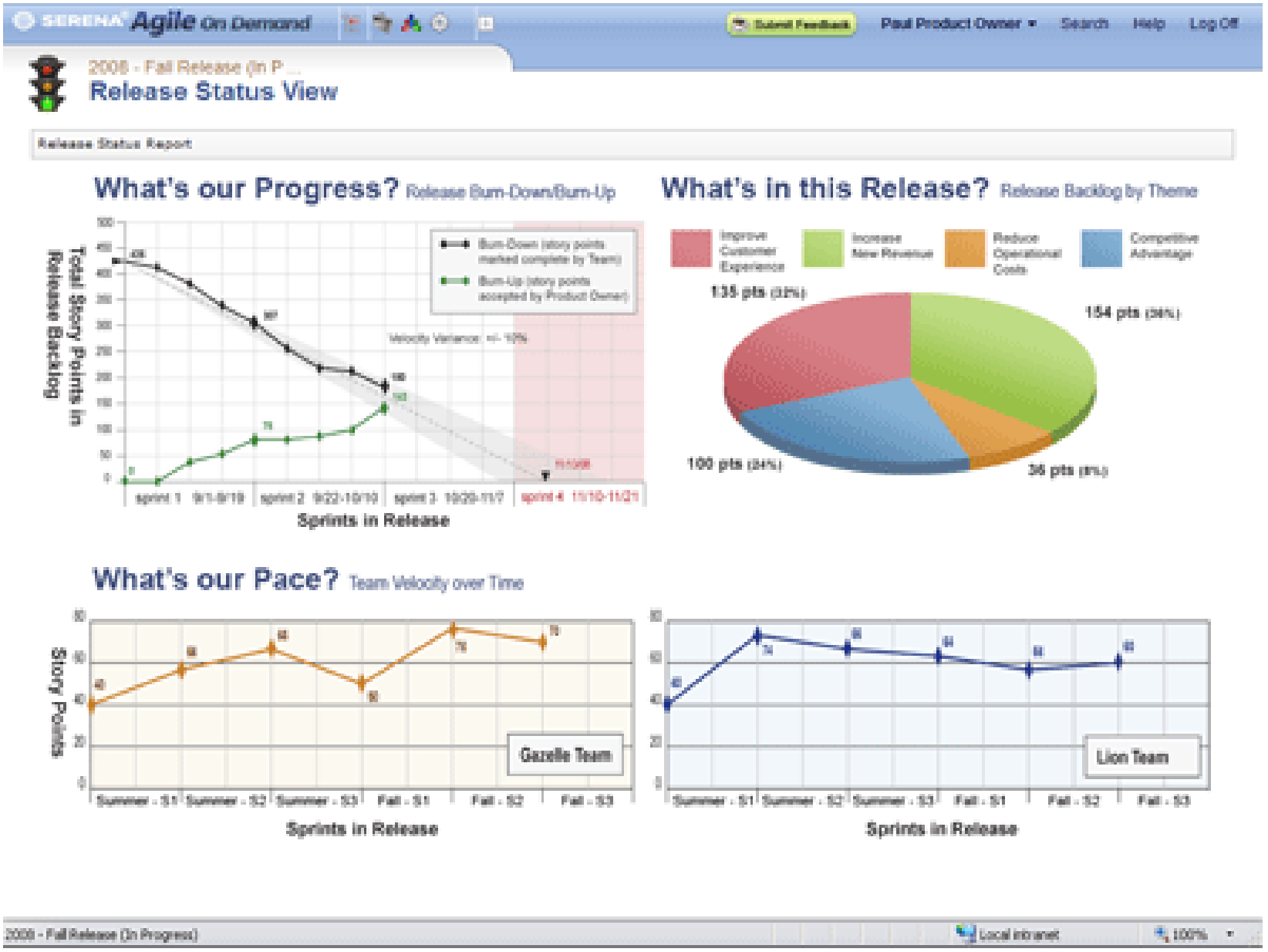
Agile Project Scorecard (Ross Pettit)

Delivery: Sample Project	This Project	Past Projects			Score
		Average	Best	Worst	
Development Excellence					
Code Toxicity					
Average Violations / Class	0	1.11	0.82	1.31	2
Duplication	0%	15%	10%	24%	
Barat	ND	ND	ND	ND	
Jdepend	ND	ND	ND	ND	
Code Hygienity					
Code Coverage	78%	18%	39%	0%	1
OO Attributes	ND	ND	ND	ND	
Quality Metrics					
Defects Delivered to QA					
Total	4	187	49	432	2
Critical + High	1	138	33	314	
Defects Delivered to Production					
Total	ND	31	0	176	ND
Critical + High	ND	20	0	113	
Business Value Metrics					
Story Value	77%	ND	ND	ND	NA
Story Lifetime	0	ND	ND	ND	
Lost Time	0%	10%	0%	150%	2
Project Management Metrics					
Schedule Variance, Project	ND	26%	0%	67%	ND
Schedule Variance, Development	4%	ND	ND	ND	
Performance Variance (Transparency)					
Iteration 1	31%	NA	NA	NA	2
Iteration 3	0%	NA	NA	NA	
Iteration 5	6%	NA	NA	NA	
Iteration 7	3%	NA	NA	NA	
Iteration 9	3%	NA	NA	NA	
Iteration 11	0%	NA	NA	NA	
Total					9
Score					Above Average

Sample Agile Dashboard (VersionOne)



Sample Agile Dashboard (Serena)



Using Metrics as Process Diagnostics

Mature agile teams notice problems early;
they are alert to "process smells." As for the rest...

- ✧ Don't overreact to any single data point. Stuff happens (usually only once).
- ✧ Look for trends and for tell-tale patterns in the trends (it takes 3 data points to make a trend).
- ✧ Cross-reference more than one metric before drawing any conclusions.
- ✧ Use root cause analysis techniques to be sure you are fixing the right problem before you act.
- ✧Learn to recognize process smells. It's quicker than relying on metrics.

Problematic Measures

Not relevant to agile methods:

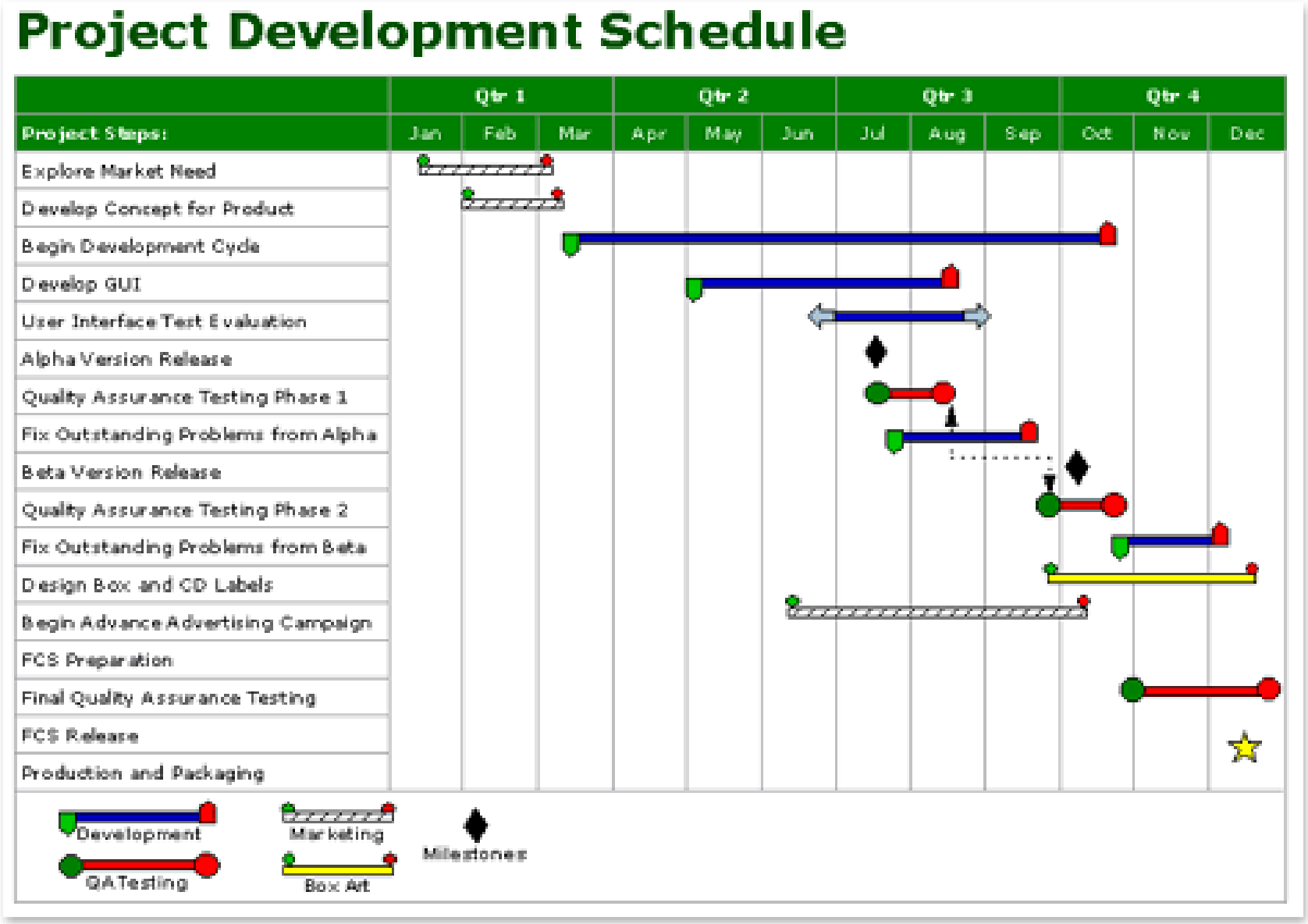
- ✧ Gantt chart
- ✧ Percent complete

Gantt Charts and Agile Projects

Differences between traditional and agile methods:

- ✧ Work Breakdown Structure (WBS)
- ✧ Approach to project planning
- ✧ Iron Triangle management
- ✧ Big bang vs. incremental delivery

Example of a Gantt Chart



Traditional Development Approach

1. Define technical architecture

2. Define database

3. Develop persistence layer

4. Develop business logic layer

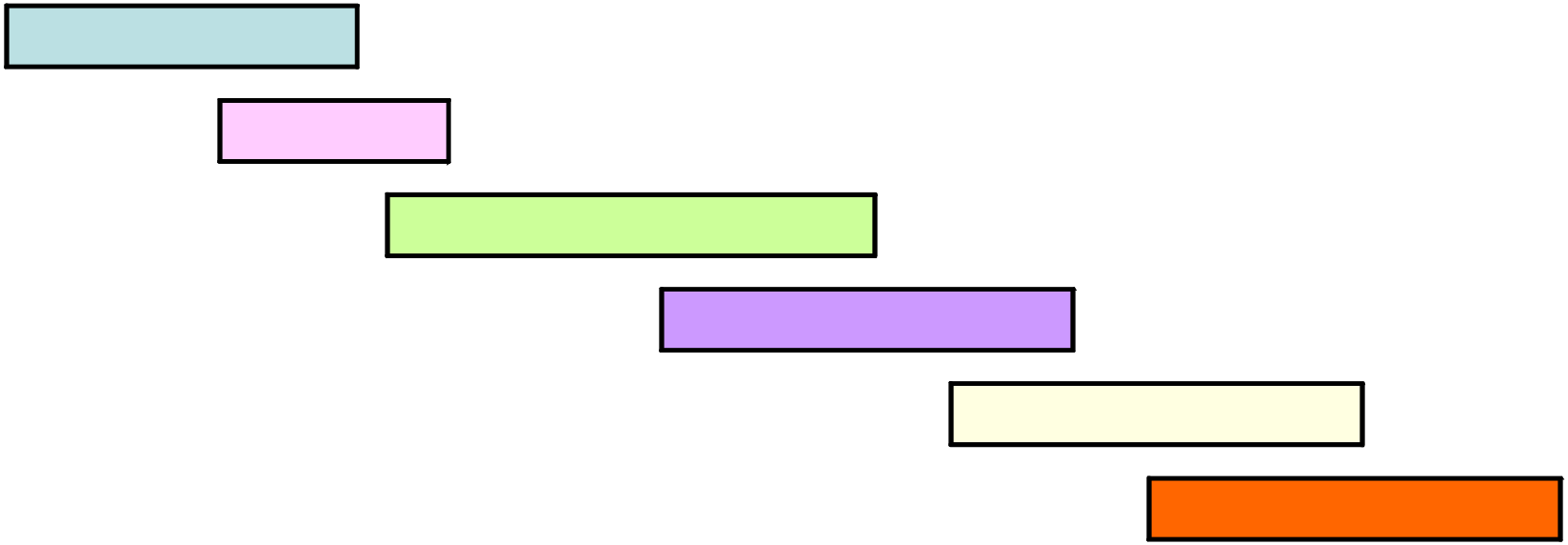
5. Develop presentation layer...etc.

Traditional WBS

1. Define technical architecture
    1.1. Task
    1.2. Task (depends on 1.1)
        1.2.1. Task
        1.2.2. Task (depends on 1.2.1)
    1.3. Task (depends on 1.2)
2. Define database
    2.1. Task
    2.2. Task (depends on 2.1)
3. Develop persistence layer (depends on 2)
    3.1. Task
    3.2. Task (depends on 3.1)
    etc.

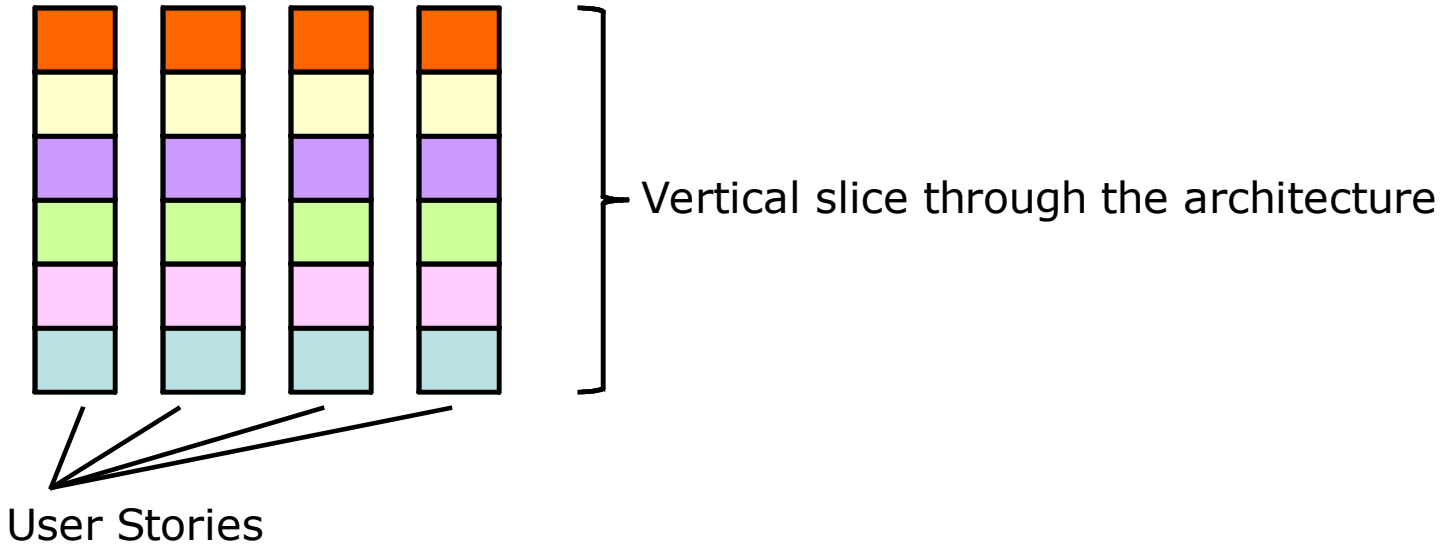
Gantt for Traditional Project

1. Arch
1. DB
1. Pers.
1. Bus.
1. Pres.
1. UI

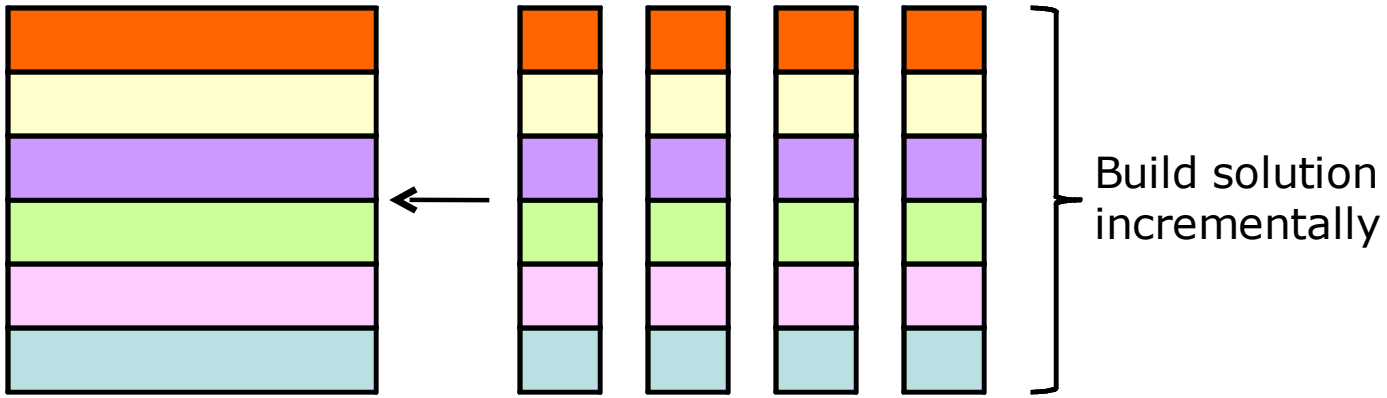


Agile Development Approach

1. Develop most important feature (per customer)



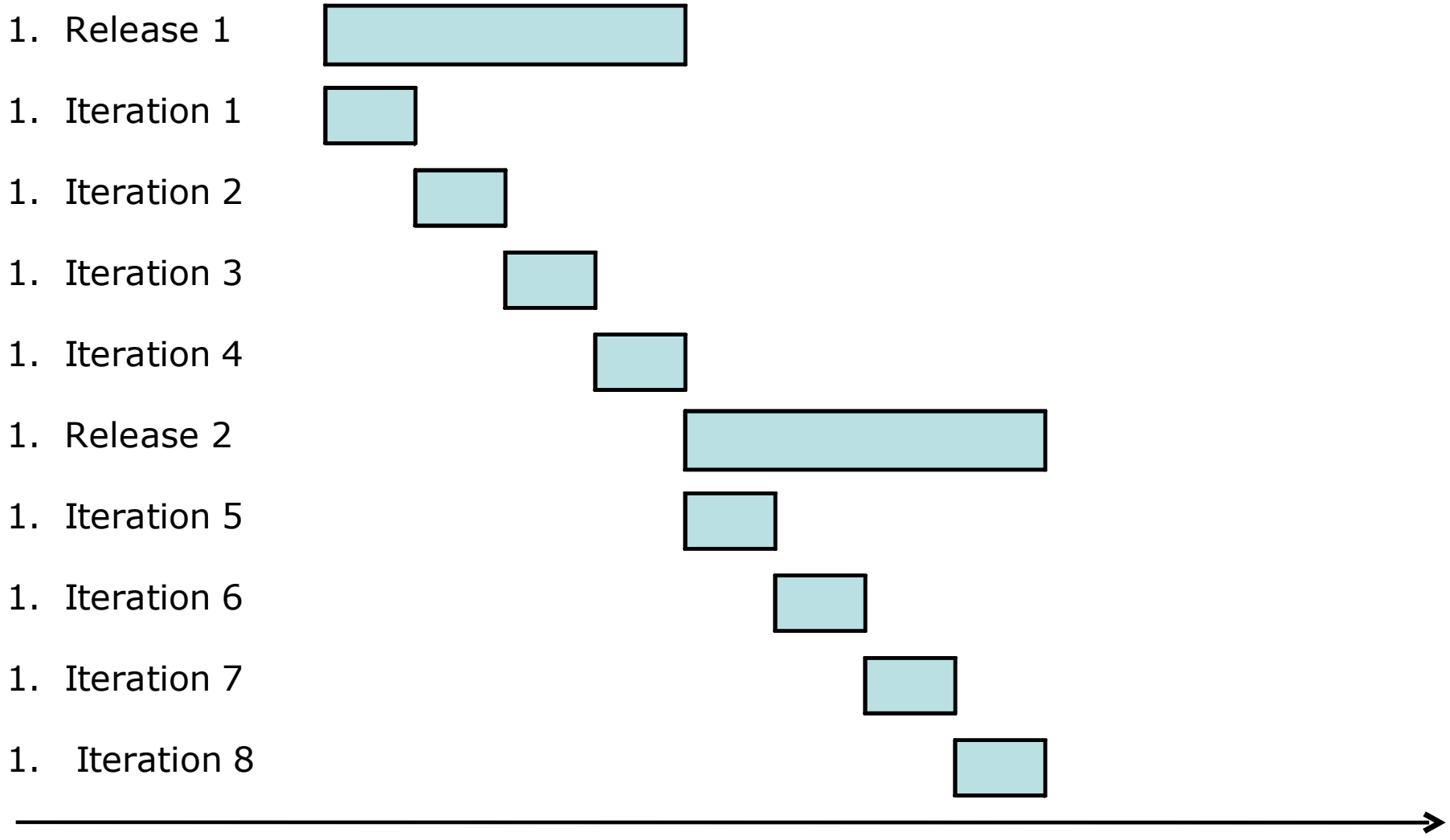
2. Develop next most important feature



Gantt for Agile Project

1. User Story 1 
1. User Story 2 
1. User Story 3 
1. User Story 4 
1. User Story 5 
1. User Story 6 
1. User Story 7 
1. User Story 8 
1. User Story 9 
1. User Story 10 
1. User Story 11 

Gantt Chart That Looks Reasonable…



…but doesn’t tell us anything useful.

Percent Complete (Traditional)

Task Estimate: 10 days

After 1 day: 10%
After 2 days: 20%
After 5 days: 50%
After 8 days: 80%
After 9 days: 90%
After 10 days: 90%
After 12 days: 90%
After 24 days: 90%
After 32 days: 100% *Whew*!
After 34 days: Bug report. Percent complete???

Percent Complete (Agile)

Known scope as of April 15: 500 points

Percent complete as of May 15: 38%

Percent complete as of June 15: 52%

Known scope after release 2 planning, July 15: 900 points

Percent complete as of August 15: 35%

What? How can percent complete go *down*?

Percent Complete

For any methodology, you can always determine the percent complete by looking at a calendar.

It is the percentage of the project schedule that has passed as of the current date.

Percent complete is equally meaningless for traditional and agile projects.