How To Deliver Software On Time

(or know early that you'll be late)

Chris Nelson, PE

Professional Software Engineer

@ChrisNelsonPE

LinkedIn.com/in/ChrisNelsonPE

NoPerfectProgram.WordPress.com

Project Management Tools

- Traditional project management
  - Work Breakdown Structure (WBS)
  - Gantt Chart
- Agile software development
  - Workload chart
  - Burndown chart
- Ad hoc and custom reports

Open Source Tools

- Trac
  - Tickets
  - Wiki
- Plugins
  - Timing & Estimation
  - Estimation Tools
  - Master Tickets
  - Subtickets
  - Team Calendar
  - TracJSGantt (TracPM)

http://trac.edgewall.org/

https://trac-hacks.org/

Trac Requirements

- Linux, OS X, or Windows
- Python
- RDBMS
  - SQLite (Default)
  - MySQL
  - PostgreSQL

“On our project we're always 90% done!”

http://blog.codinghorror.com/on-our-project-were-always-90-done/

Create Small Tasks

- Smaller tasks = Better estimates
  - 100% done on 9 of 10 tasks is more accurate than 90% done on one task

Task	Estimate
Develop UI	80 hrs

Task	Estimate
1.0 Develop UI	90 hrs
1.1 Develop API	30 hrs
1.2 Develop Icons	10 hrs
1.3 Develop CLI	15 hrs
1.4 Develop Web UI	15 hrs
1.5 Develop App	20 hrs

Do High-Risk Tasks ASAP

Task	Estimate	Confidence
1.0 Develop UI		
1.1 Develop API	30 hrs	Medium
1.2 Develop Icons	10 hrs	High
1.3 Develop CLI	15 hrs	Very High
1.4 Develop Web UI	15 hrs	High
1.5 Develop App	20 hrs	Low

- High risk has many causes
  - High estimate
  - Low confidence

Choosing Confidence Levels

Confidence	Value
Unspecified	0
Low	1
Medium	2
High	3
Very High	4
Complete	5

Using Confidence Levels

Task	Estimate	Confidence	Value
1.0 Develop UI	90 hrs	*Med-High*	*2.39*
1.1 Develop API	30 hrs	Medium	2
1.2 Develop Icons	10 hrs	High	3
1.3 Develop CLI	15 hrs	Very High	4
1.4 Develop Web UI	15 hrs	High	3
1.5 Develop App	20 hrs	Low	1

((30*2) + (10*3) + (15*4) + (15*3) + (20*1)) / 90

Academic Scheduling

- RCPSP
  - Resource
  - Constrained
  - Project
  - Scheduling
  - Problem

Pragmatic Scheduling

- SSGS
  - Serial
  - Schedule
  - Generation
  - Scheme

Schedule Generation

- Find eligible tasks
  - No incomplete predecessors
  - Resource is available
- Sort eligible tasks
  - Bugs before enhancements
  - High Risk before Low
  - Fragile code base before stable
- Schedule one task
- Find more eligible tasks
- Repeat at Sort

Resource Availability

```
class IResourceCalendar(Interface):

    # Return the number of hours available for the resource

    # on the specified date.

    def hoursAvailable(self, date, resource = None):

        """Called to see how many hours are available on date"""
```

Task Comparison Function

```
class ITaskSorter(Interface):
    # Process task list to precompute keys or otherwise make
    # compareTasks() more efficient.
    def prepareTasks(self, ticketsByID):
        """Called to prepare tasks for sorting."""

    # Provide a compare function for sorting tasks.
    # May be used as cmp argument for sorted(), and list.sort().
    # Returns -1 if t1 < t2, 0 if they are equal, 1 if t1 > t2.
    def compareTasks(self, t1, t2):
        """Called to compare two tasks"""
```

Critical Path

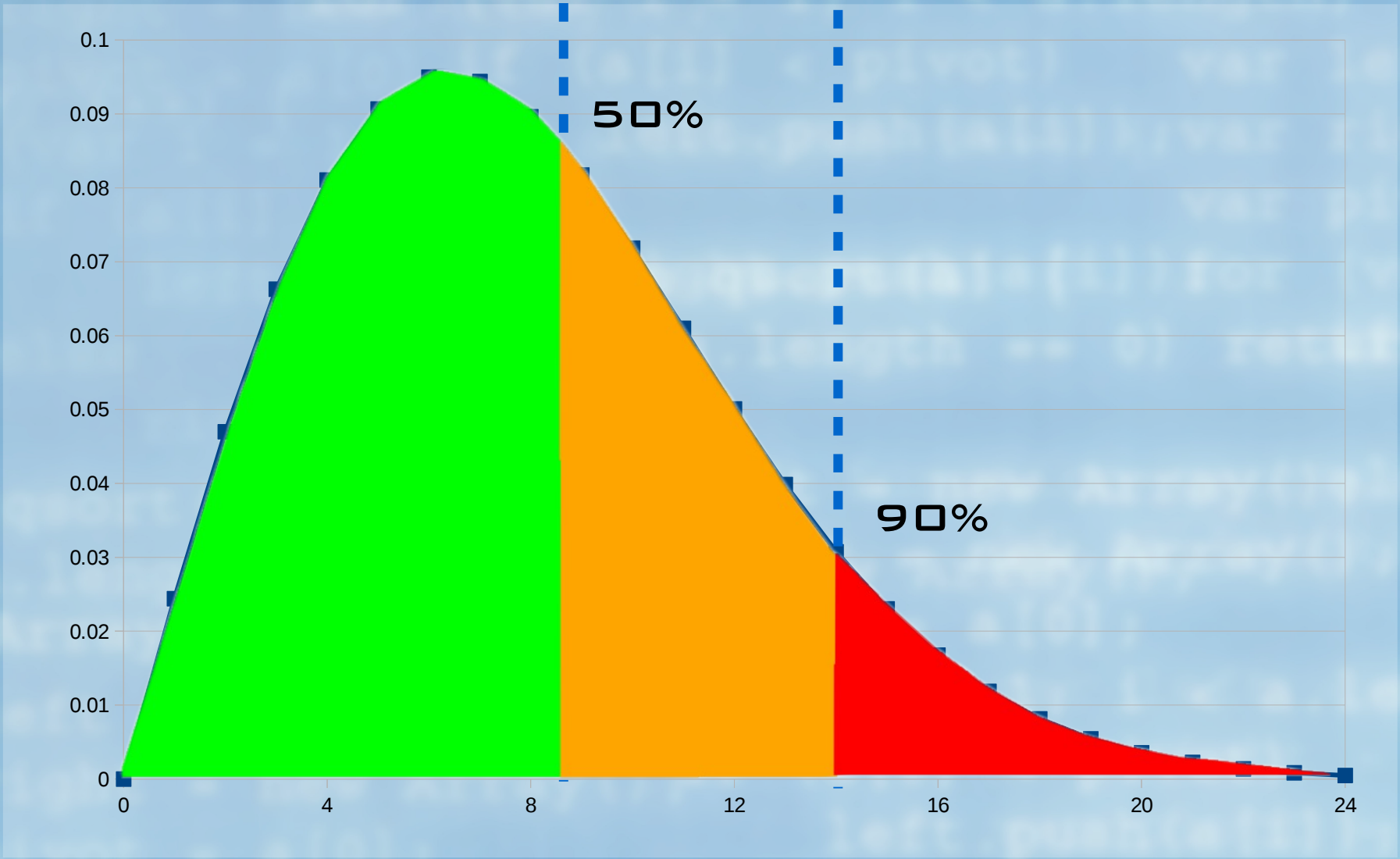
- Definition
  - Any delay in completing a task on the CP delays the completion of the project
- Computation
  - Compute ASAP schedule
  - Compute ALAP schedule
  - CP tasks have the same start and finish time in both schedules

Critical Chain Scheduling & Buffer Management

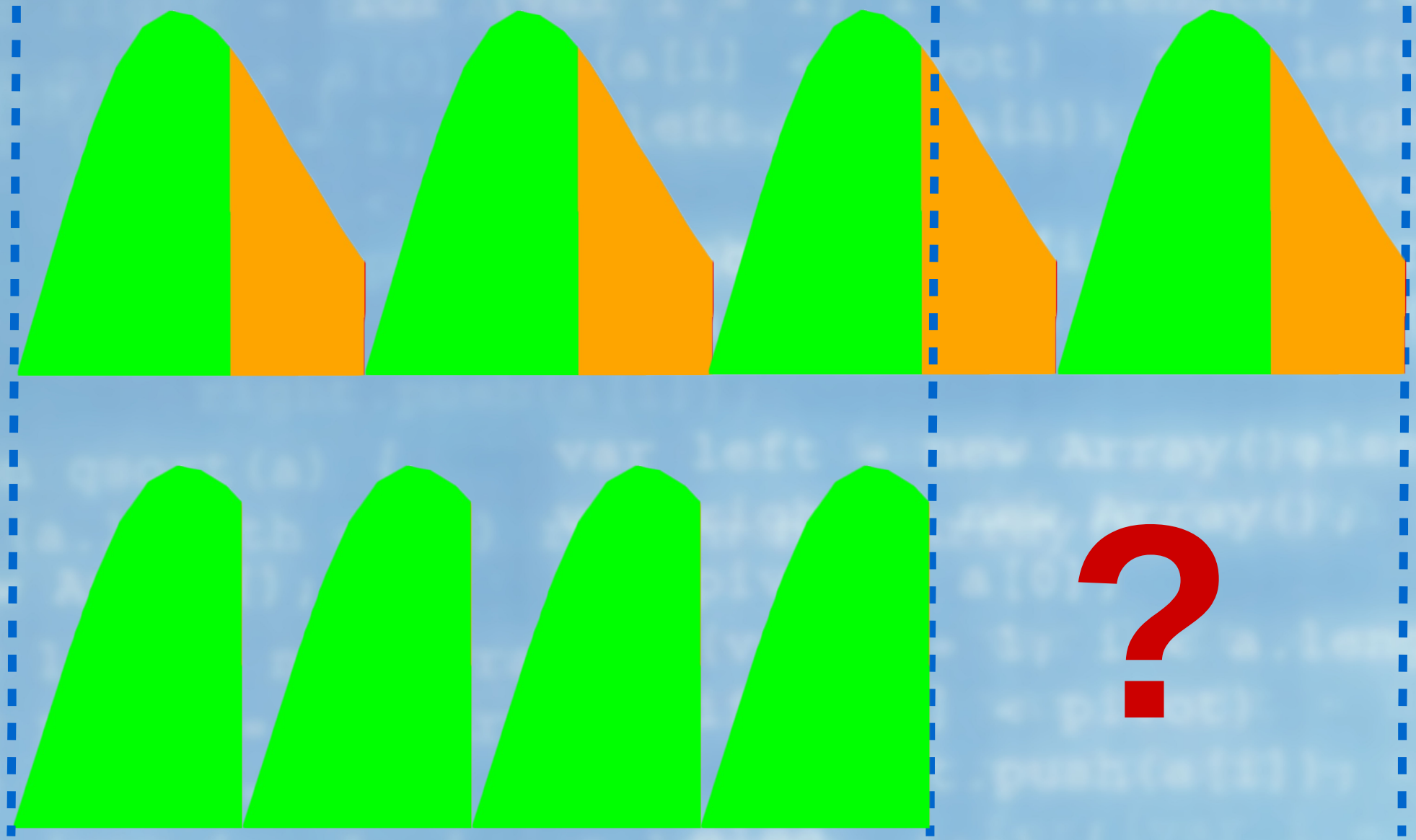
- Parkinson's Law
  - "Work Expands to fill the time alloted"
- Student Syndrome
  - Everything gets done at the last minute

http://www.focusedperformance.com/articles/ccpm.html

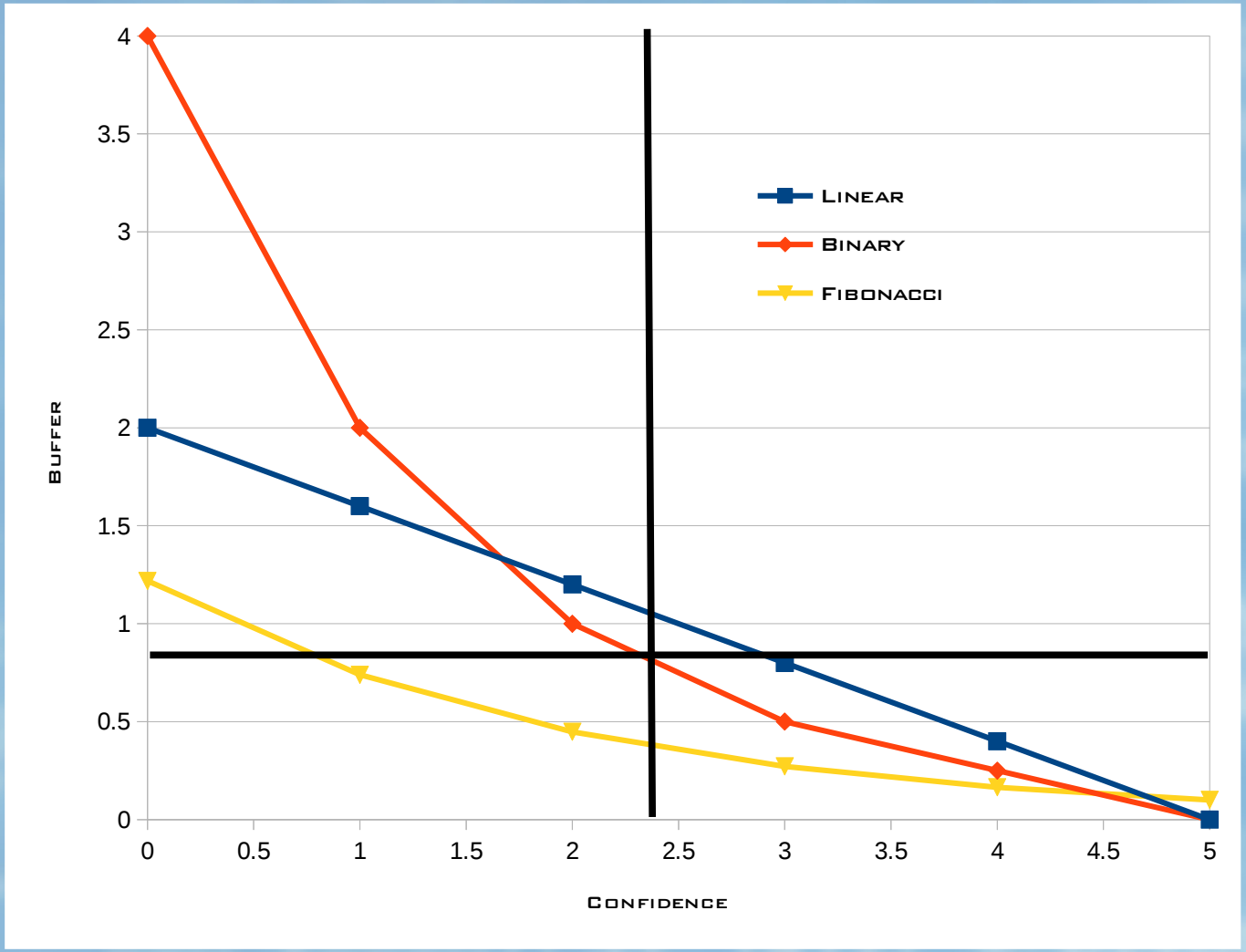
Estimate Distribution



50% vs 90% Confidence



Continuous Buffer Function



Task	Estimate	Confidence	Value	Buffer
1.0 Develop UI	90 hrs	Med-High	2.39	72 hrs

Setting the Due Date

Task	Estimate	Confidence	Value	Buffer
1.0 Develop UI	90 hrs	Med-High	2.39	72 hrs

- Compute the buffer
  - Find critical path
  - Total estimates
  - Compute weighted confidence
  - Total estimate * Weighted confidence = Buffer
- Compute project makespan
  - estimate + buffer = Makespan
  - 90 + 72 = 162 hours
- Figure out when you'll finish
  - Project Start + Makespan
  - June 17, 2015 + (162 / 30) = July 24, 2015
- Set due date for project NOT for tasks

Monitoring Buffer

- As tasks finish, confidence rises
  - Higher confidence requires less buffer
- As project proceeds, scope may change

Subject **Project projections**

To swmgr

```
13083: Armstrong
    0.89 weeks work (buffered) will end 2015-06-15, due 2015-06-30 (d:/>)
    https://trac.net/trac/ticket/13083

Performance is in the form (p:tc)
  p is period: d=day, w=week, m=month
  t is trend: \=late, /=early, -=on time
  c is change (since last period): v=worse, >=even, ^=better
```

Retrospective

57 tickets processed; 49 (85.96%) have estimates

Per-ticket accuracy ← (ACT - EST) / EST

min: -1 max: 378.6 avg: 15.43 sd: 73.28 median: 0.03

under: 25 over: 24 exact: 0 u:o ratio: 1.04

< 0 : under estimate
> 0 : over estimate
Magnitude is error

Overall accuracy (tickets with estimates):

est: 685.57 act: 1084.7 overall acc: 0.58

Overall accuracy (with default estimate, 8):

est: 749.57 act: 1094.78 overall acc: 0.46

Project duration: 28 days.

Ticket open offset from start of project: ← Ideally 0

Average: 0.00 (0.00% of dur.), Weighted: 0.00 (0.00%).

Ticket start offset from start of project: ← Ideally 50%

Average: 0.05 (0.19% of dur.), Weighted: 0.03 (0.10%).

Take Aways

- Use Data-based Tools
- Make your tasks small
- Schedule high-risk tasks early
- Put uncertainty in a project buffer
- Monitor progress
- Use experience to fine tune

Thank You!

Stay in touch.

Chris Nelson, PE
Professional Software Engineer

@ChrisNelsonPE
LinkedIn.com/in/ChrisNelsonPE
NoPerfectProgram.WordPress.com