2015 TOCICO International Conference

Improving Project Portfolio Performance with Buffer-Type Flexibility and Task-level DBR

Bahadir Inozu, Ph.D., NOVACES, LLC
Michael Hannan, Fortezza Consulting
Hilbert Robinson, Delta Airlines

2015 TOCICO International Conference

Part 1: Improving IT Project Portfolio Throughput With Task-level DBR, Lean, and Agile

Bahadir Inozu, Ph.D., NOVACES, LLC
Michael Hannan, Fortezza Consulting
Hilbert Robinson, Delta Airlines

Objective #1

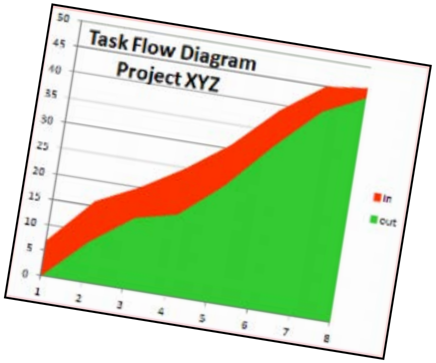
Higher Throughput of Project Completions

Objective #2

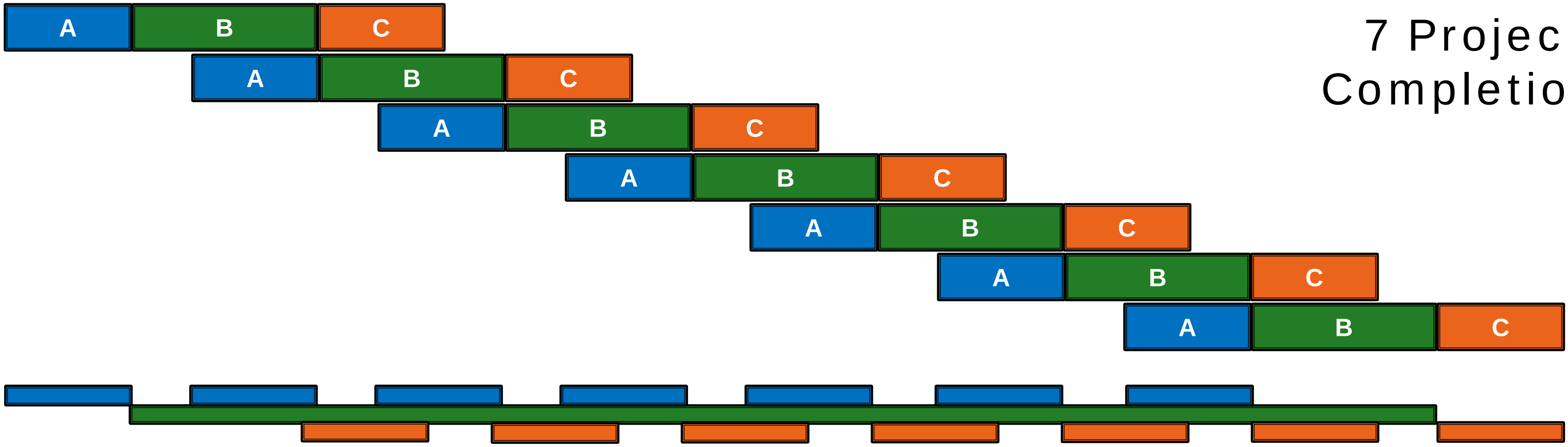
Stronger, More Enduring Discipline for Single-Tasking

Technique

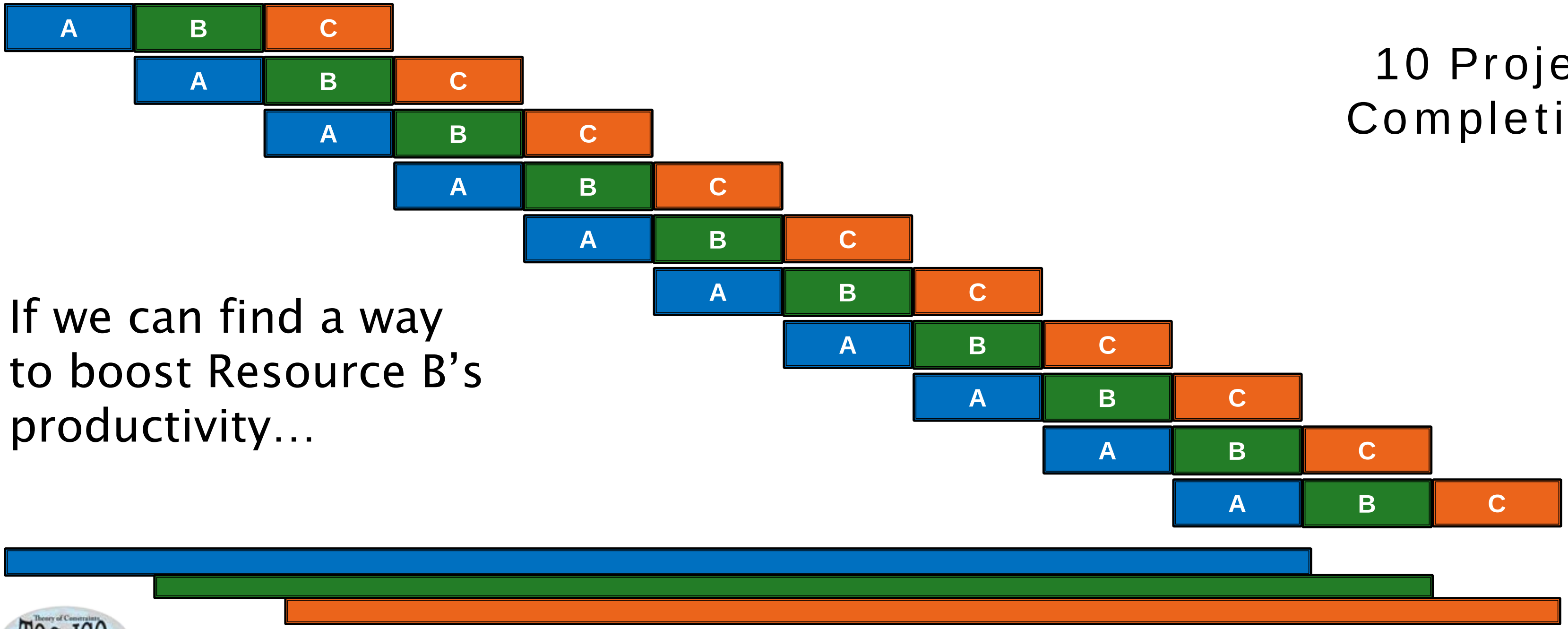
ACCLAIM™ Single-Tasking Method, with Task-level DBR



Background



7 Project Completions



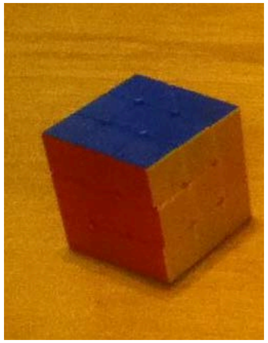
10 Project Completions

If we can find a way to boost Resource B's productivity…

What Do Leading Methods Teach Us About Flow?

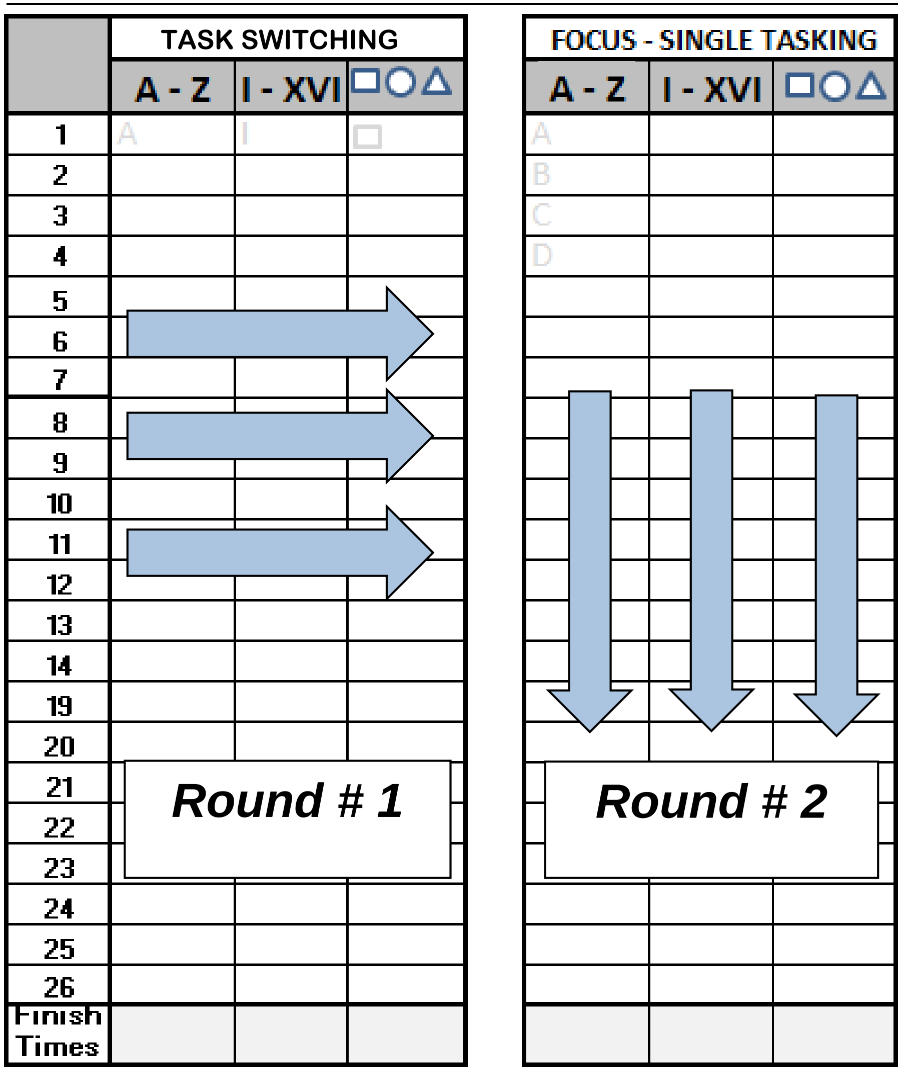
- Theory of Constraints (TOC)
  - Maximizing flow across an end-to-end process (system) can only be done once the **system constraint** is identified
- TOC, Lean
  - The less “work in process” (or **WIP**) in the system, the faster and more efficient the system—as long as lead times are shrinking faster than WIP.
- TOC, Psychology
  - Single-tasking is a highly effective way to minimize lead times for human-centric tasks

$$\text{Throughput} = \frac{\text{WIP}}{\text{Lead Time}}$$



$$\uparrow T = \frac{\downarrow W}{\downarrow LT \downarrow} \qquad \uparrow\uparrow T = \frac{\downarrow W}{\downarrow\downarrow LT \downarrow\downarrow} \qquad \uparrow\uparrow\uparrow T = \frac{\downarrow W}{\downarrow\downarrow\downarrow LT \downarrow\downarrow\downarrow}$$

Task-switching Game



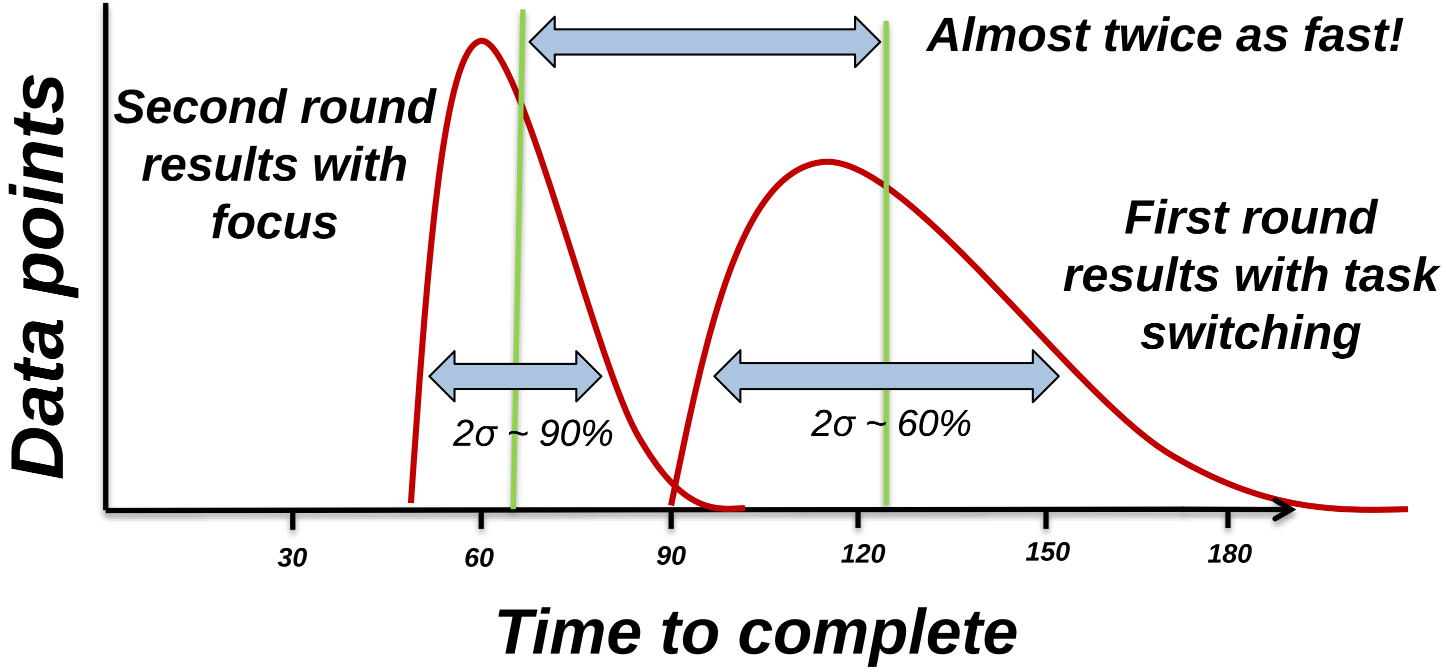
	TASK SWITCHING		
	A - Z	I - XVI	□○△
1	A	I	□
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			
13			
14			
19			
20			
21			
22			
23			
24			
25			
26			
Finish Times			

Round # 1

FOCUS - SINGLE TASKING		
A - Z	I - XVI	□○△
A		
B		
C		
D		

Round # 2

Typical Game Results



Almost twice as fast!

Second round results with focus

First round results with task switching

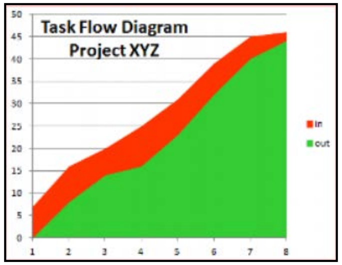
What Do Leading Methods Teach Us About Maximizing Flow? (cont'd)

- Lean/Kanban
  - **Visualizing** the actual flow of work—especially for workflows that aren't inherently visual—is critical for team members to identify impediments and to experiment with improvement ideas.
  - Enabling the system to "**pull**" work, vs. having work assigned or "pushed," tends to improve flow while empowering teams.
  - Minimizing batch sizes—ideally down to a batch size of one, or "**single-piece flow**"—can generate impressive flow improvements
- Agile/Scrum, Psychology
  - **The team** knows how to be **more productive** than the sum of its members
  - **The team** is much more motivated when working under a disciplined framework designed to foster **team autonomy**.

Crafting a Task-flow Maximization Approach

- If the team can break tasks on the project plan into fine-grained subtasks that take less than a week, and ideally about a day…

Ease the Flow

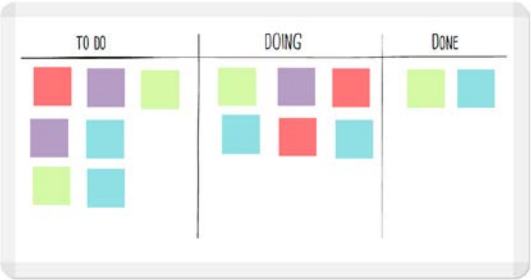


- If the team can maintain a ready supply of these fine-grained tasks to avoid starving the constraint…

Feed the Machine

- If these fine-grained tasks are visible, and can be pulled for execution by any team member, one at a time, without concern for sequence…

Visual Mgmt + Pull System + Single-piece Flow

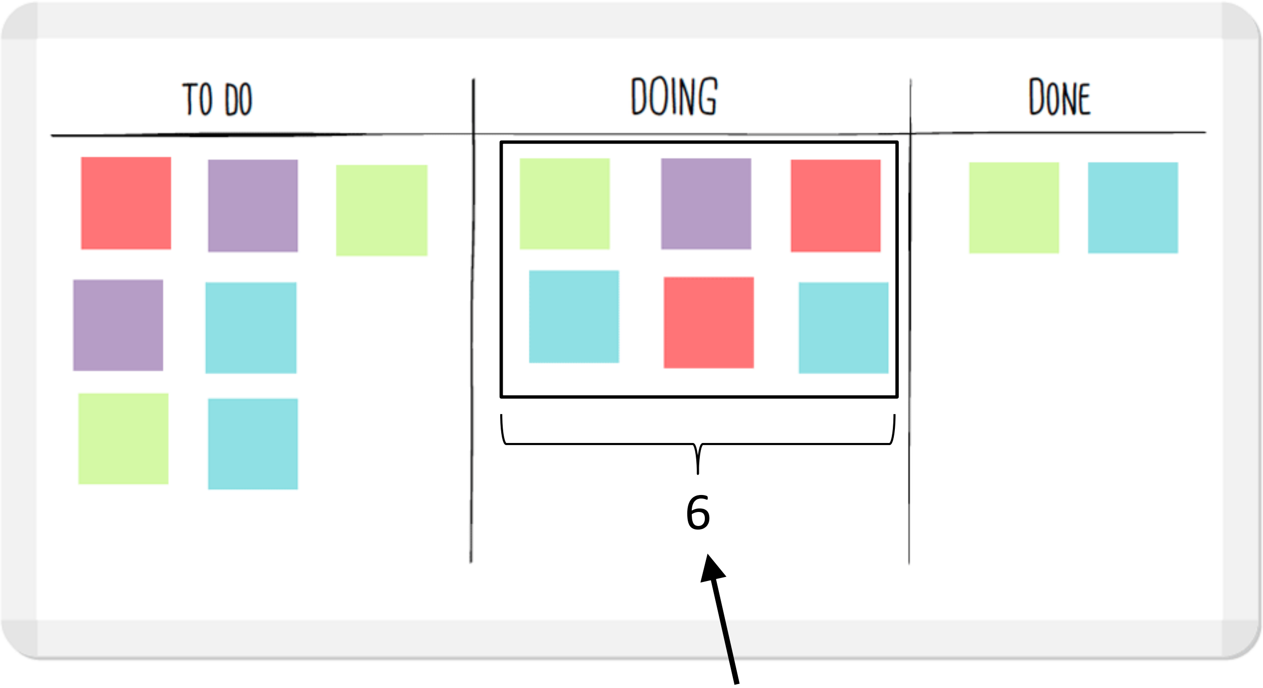


- If the system constraint can be identified, and the flow managed to minimize end-to-end WIP…

TOC WIP Minimization

- Then we can maximize flow, while protecting and enforcing single-task discipline at the constraint
- We have also eliminated task-level commitments, strengthened team autonomy, and aligned team behavior with portfolio-level throughput objectives
- For Agile/Scrum teams, we have eliminated sprints in favor of maximizing flow

How Do We Know Whether Single Tasking is Actually Happening?



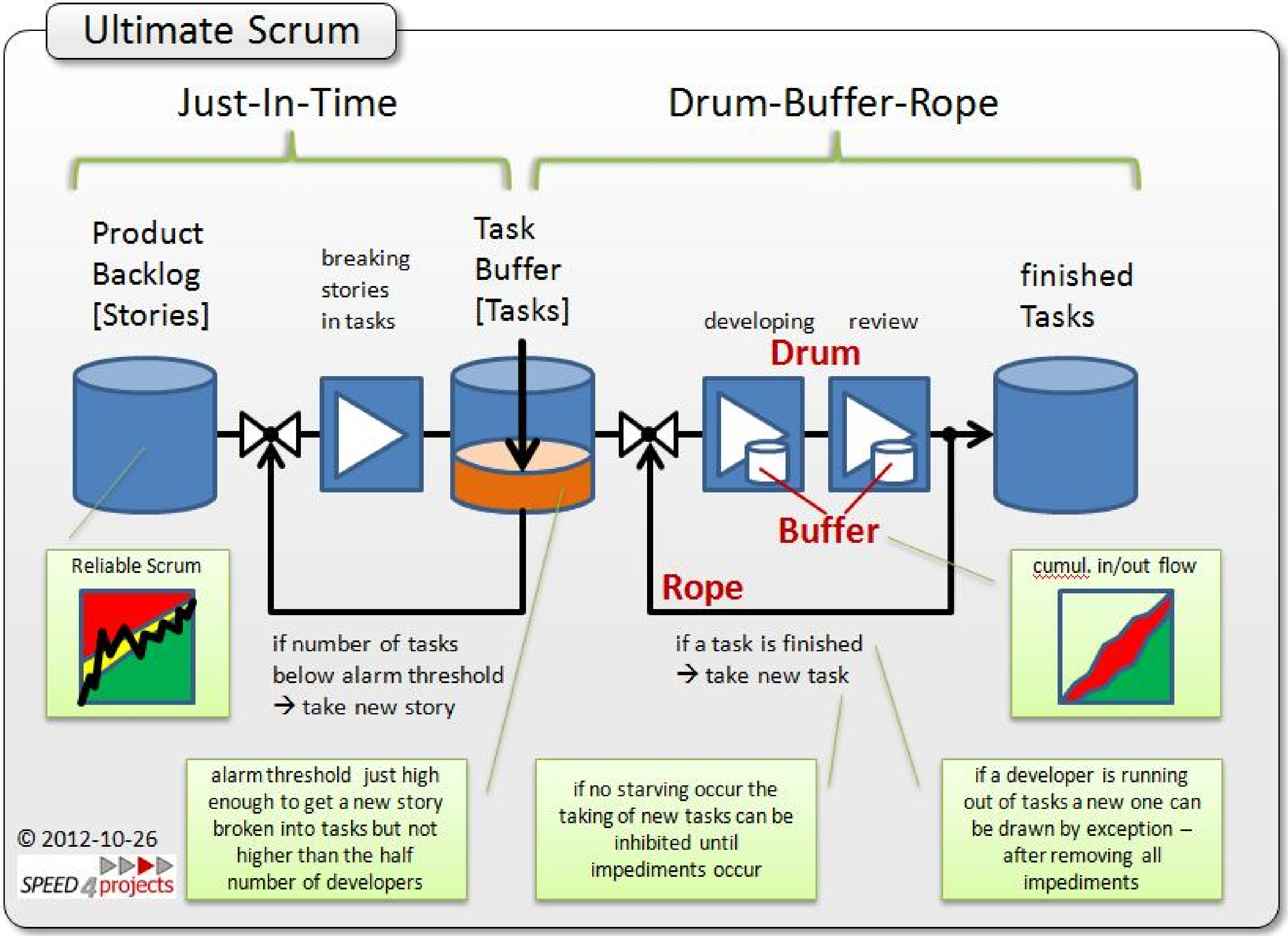
Number of Task Owners

8

Ensure that this number…

…is always slightly lower than this number

Wolfram Müller of VISTEM/Speed4Projects.net



Ultimate Scrum

Just-In-Time

Drum-Buffer-Rope

Product Backlog [Stories]

breaking stories in tasks

Task Buffer [Tasks]

developing

review

Drum

finished Tasks

Buffer

Rope

Reliable Scrum

cumul. in/out flow

if number of tasks below alarm threshold → take new story

if a task is finished → take new task

alarm threshold just high enough to get a new story broken into tasks but not higher than the half number of developers

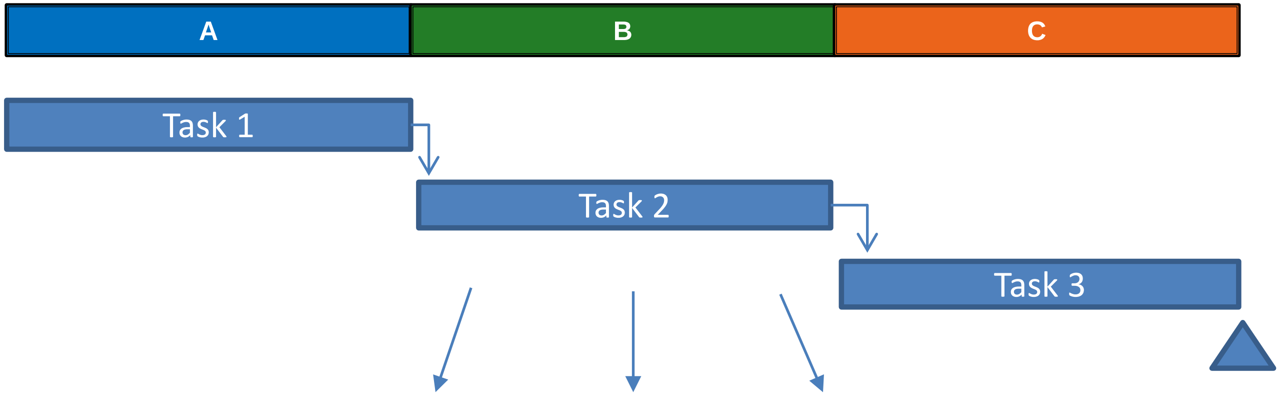
if no starving occur the taking of new tasks can be inhibited until impediments occur

if a developer is running out of tasks a new one can be drawn by exception – after removing all impediments

© 2012-10-26

SPEED 4 projects

How Do Granular Tasks on a Task Board Relate to the Tasks on a Gantt Chart?



A

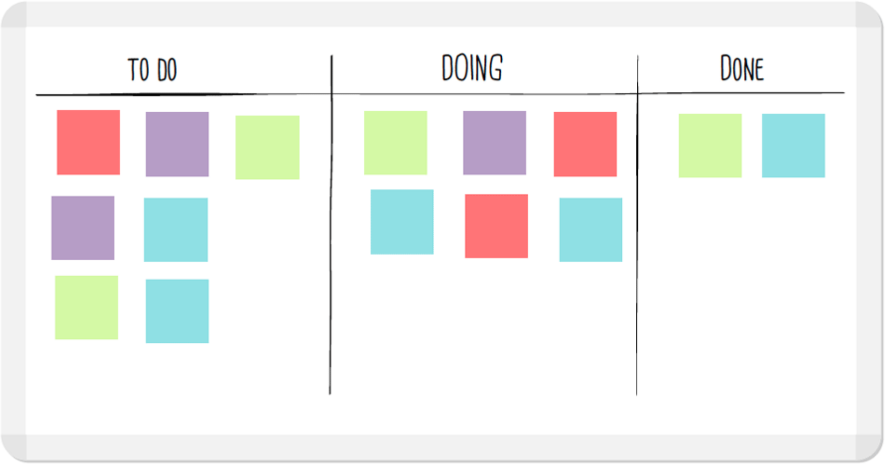
B

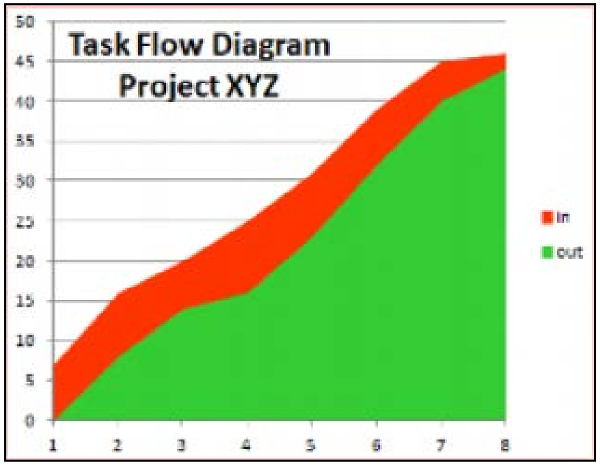
C

Task 1

Task 2

Task 3

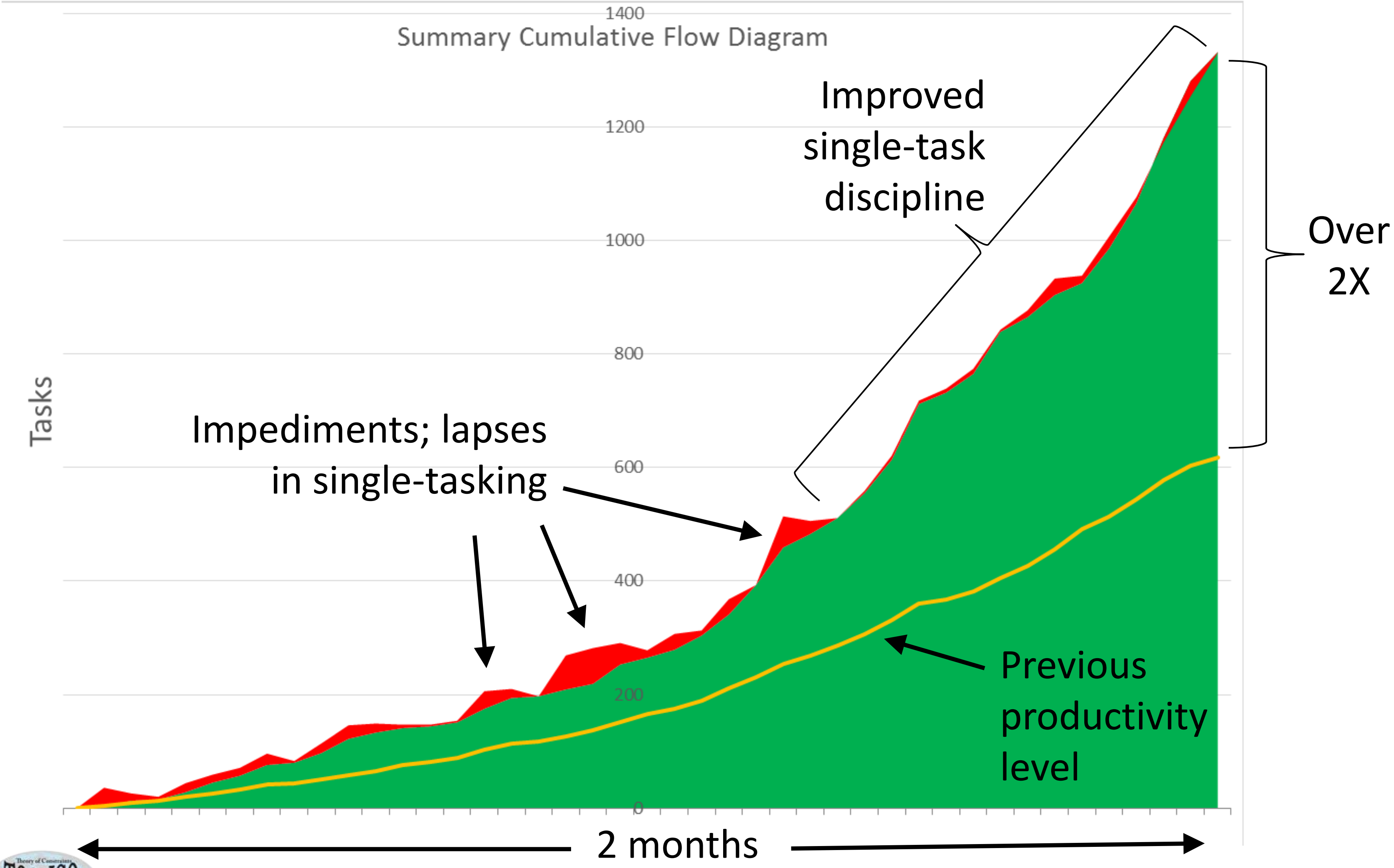




Initial Experience in a Relevant Context

- One large IT organization has been live for 2 months, with the following initial results
  - Immediate flow improvement of over 2X across a 30-person pool of software developers, architects, and visual designers
  - Multiple examples of individuals achieving 6X-8X productivity jumps—both senior top performers and average junior staff alike
  - Team morale and excitement were never low, but now greatly enhanced
  - Effective way to implement and enforce single tasking
  - Project schedules are accelerating, sometimes by 30% or more
  - Solving the “Resource B” problem
- In the beginning phases of being rolled out in a second large IT organization

ACCLAIM Single Tasking Method at Utah's IT agency



Real-life scenarios "from the front"

- The leadership commitment to rolling out CCPM at a large IT organization was tentative, but software-development teams were crying out for relief from being severely overloaded.
- An aircraft maintenance group sought our advice on how to help them address the backlog of maintenance tasks—some at the "project" level, some at the "program" level, and some in support of operational needs (vs. project/program needs).
- A large IT organization with strong discipline in use-case development sought advice on how to improve the flow of use-case completions.

ACCLAIM Single Tasking Method—Key Challenges

- “Just another way for micro-managers to micro-manage us.”
- Some may view it as “just a glorified to-do list,” without understanding that the primary purpose is to foster flow via single-tasking, WIP minimization, and team brainstorming
- Can still be a little challenging for some managers to avoid doing “drive-by status checks,” interrupting single-task focus.
- Without effective project staggering, will likely result in excessive “project switching,” even when there is no task switching.
- There can be a temptation to “game the system” by taking credit for higher throughput just by breaking down tasks.

ACCLAIM Single-Tasking Method: Key Takeaways

- Can be applied to any environment—not just software development—as long as four conditions exists:
  1) There is a pool of resources that can be managed as a group
  2) We can ensure that there is always a ready supply of fine-grained tasks
  3) We can ensure that there are always slightly fewer tasks than task owners
  4) Task owners can "pull" tasks when ready, one at a time, according to well-understood prioritization rules
- With single-tasking already in place, the flow-maximization features of this method can further boost team productivity by 50% or more
- If single-tasking is not yet in place, can be used as a disciplined way to implement and enforce it, while achieving dramatic jumps in productivity
- Has the potential to jack up throughput significantly when focused on the project portfolio's system constraint
- Can "grease the skids" nicely for CCPM adoption

Part 2: Improving Project Portfolio Reliability with Buffer-Type Flexibility

Michael Hannan, Fortezza Consulting

Bahadir Inozu, Ph.D., NOVACES, LLC

Hilbert Robinson, Delta Airlines

Objective #1

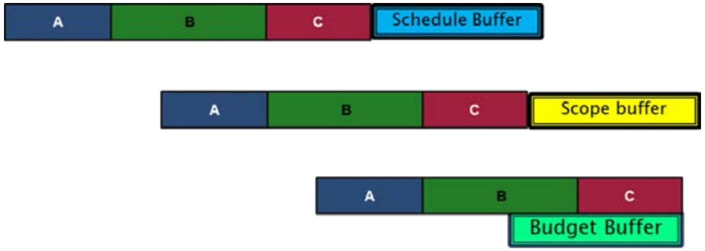
Enhanced Project Portfolio Reliability

Objective #2

Seamless Portfolio Integration of Project-level Methods (CCPM/Agile/Others)

Technique

Buffer-type Flexibility

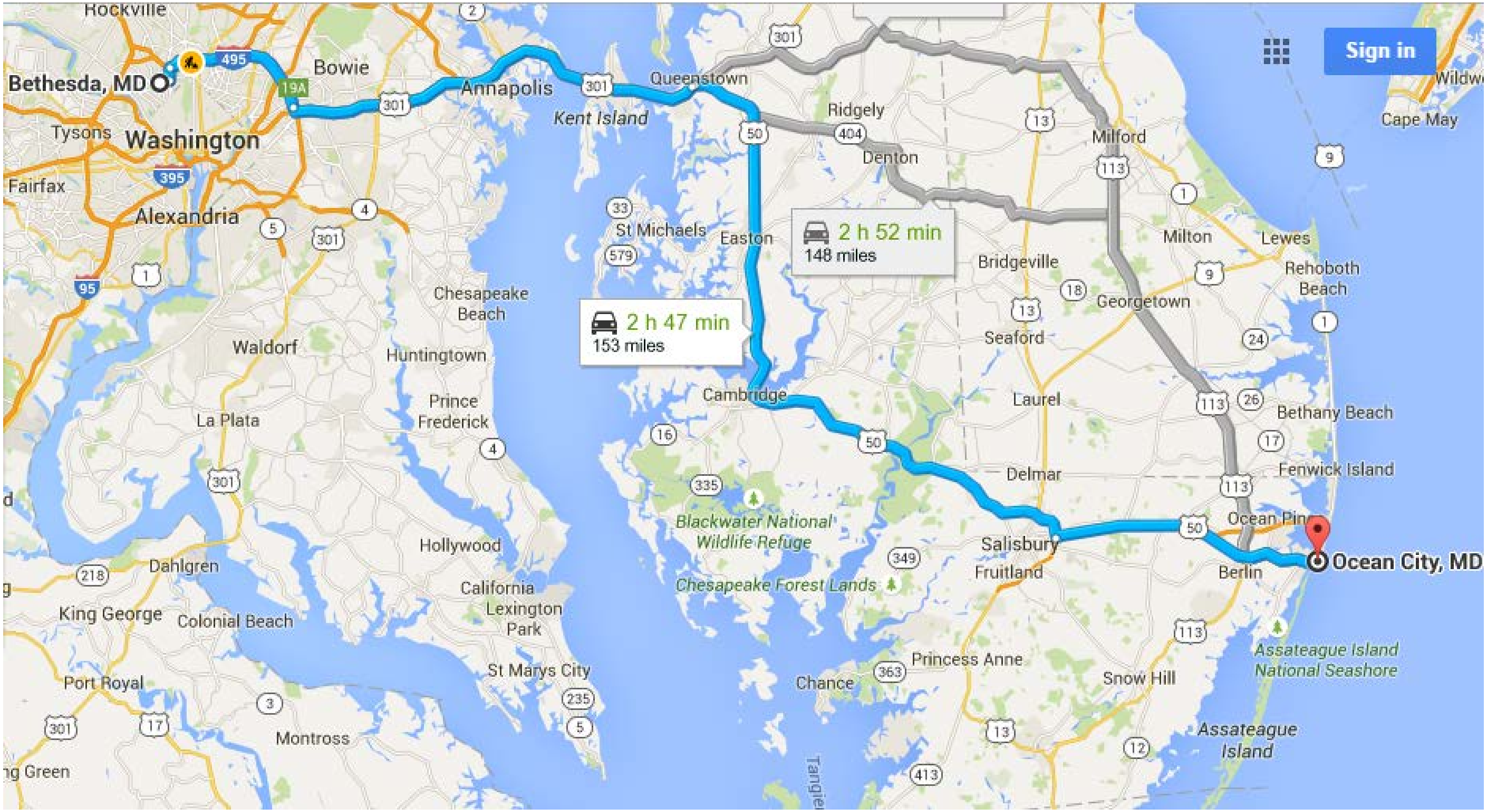


Background

- CCPM’s traditional emphasis on schedule buffers is well-placed
  - Schedule delays often drive budget overruns and create pressure to cut scope
  - Project ROI is often sensitive to schedule performance
- However, scope buffers and budget buffers are also reliability assets that can and should be exploited as well

  - When scope is fixed, and the schedule is already compressed aggressively, a budget buffer may be the only buffering option
  - It can sometimes be more valuable to add/change scope than to finish early

Simple Example: Drive to the Beach



Additional Examples

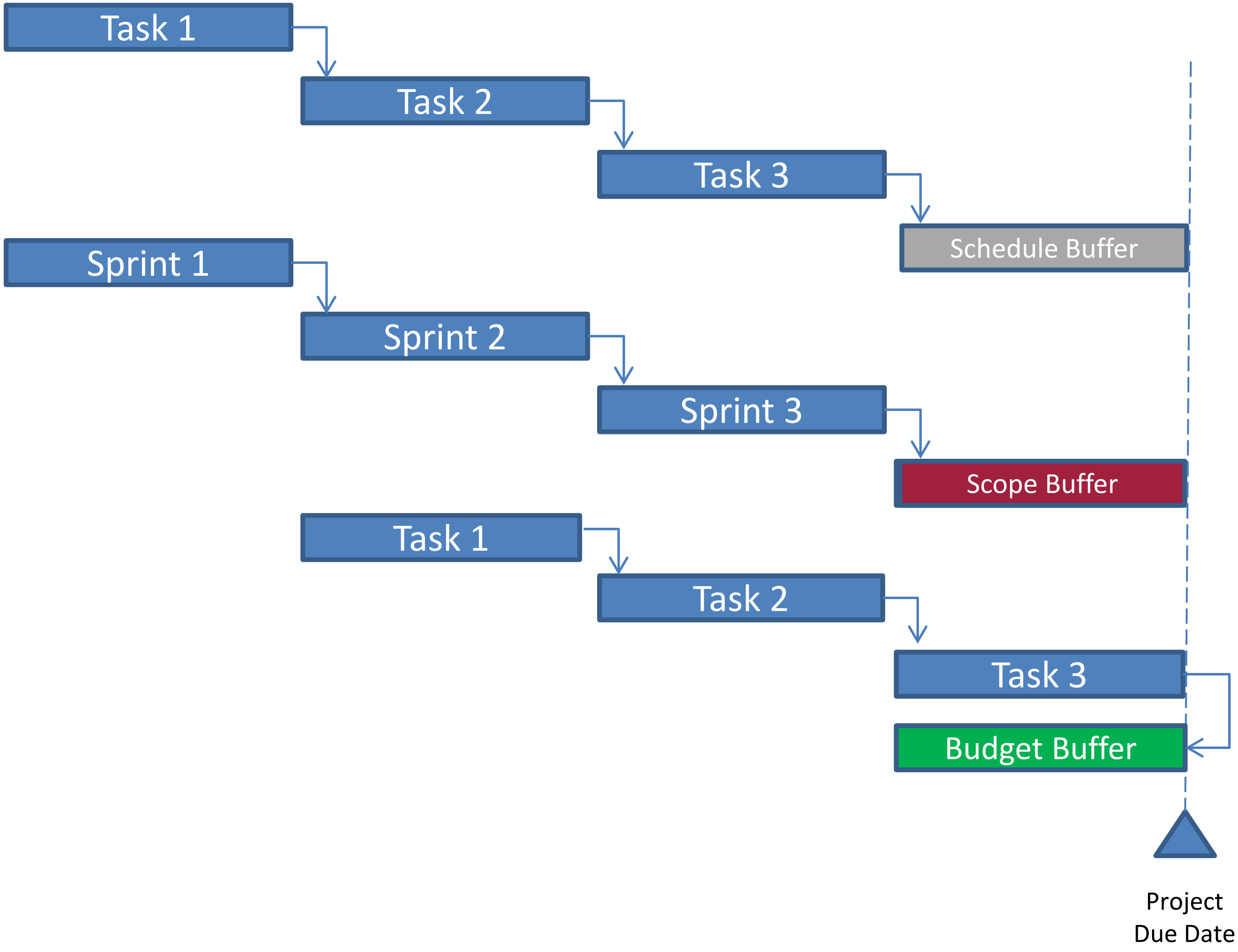
- A software-development project that must deploy within a target schedule window in which minimum scope is highly achievable, and with as many identified “nice-to-have” features as possible
- An aircraft carrier mission that must deploy on a given date, with all aircraft mission-ready
  - If the schedule buffer ends up getting exhausted before all aircraft are ready, can the mission succeed with fewer aircraft?
  - If it’s a training mission, due-date performance may still be critical, but what if budget adherence is even more critical?
  - If it’s a humanitarian mission with fixed budget (”Here’s $20M…go save as many lives as you can.”), we must trade off readiness level (scope) with schedule acceleration/delay, calling for an optimal mix of scope/schedule buffers
- An immunization program to roll out a new vaccine for a virus that may become a pandemic
  - Initial schedule-buffered plan may seem optimal during planning phase, but during execution, the pandemic strikes—should we cut the training project to accelerate schedule?

First, We Need an "Apples-to-Apples" View of All Buffer Types

- PMs tend to prefer a schedule view—such as a Gantt chart—to depict execution over a defined project duration
- CCPM Tools already support schedule buffers
- Most experienced project managers intuitively know how to translate between budget, scope, and schedule
  - For example, we might "buy schedule" by increasing budget or sacrificing scope

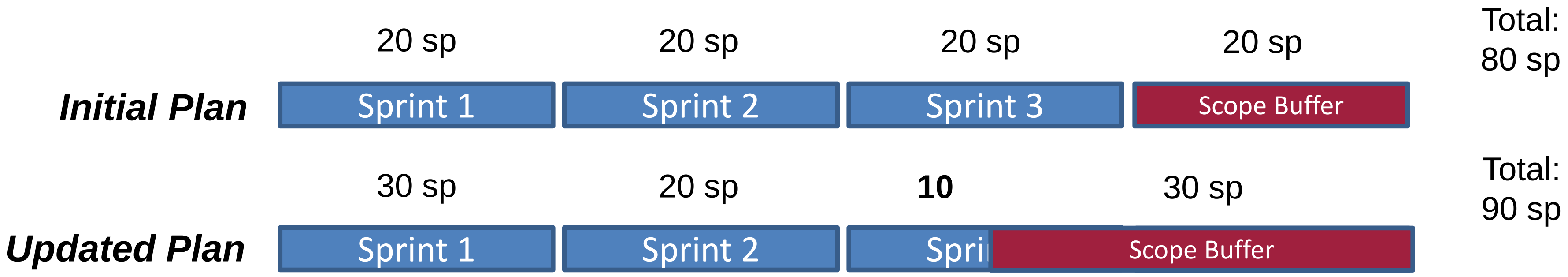
- Therefore, showing scope buffers and budget buffers as time-based offers the most common basis of understanding

Showing All Buffers As Time-Based



Time-Based Buffering for Scope-Buffered (Agile) Projects

- Agile/Scrum practitioners often prefer to think in terms of velocity, or "story points per sprint."
  - If actual velocity is <u>higher</u> than planned, it means we have added story points to our time-based scope buffer



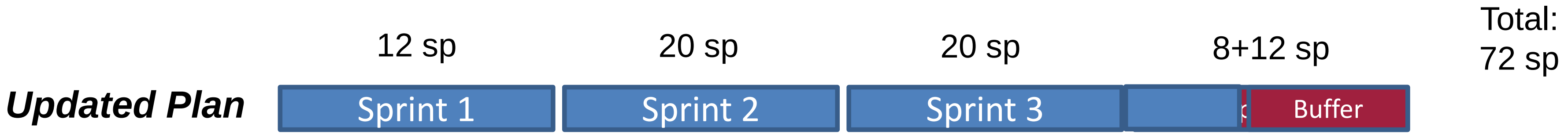
20 sp 20 sp 20 sp 20 sp Total: 80 sp

Initial Plan Sprint 1 Sprint 2 Sprint 3 Scope Buffer

30 sp 20 sp **10** 30 sp Total: 90 sp

Updated Plan Sprint 1 Sprint 2 Spri Scope Buffer

  - If actual velocity is <u>lower</u> than planned, it means we have consumed story points from our time-based scope buffer.



12 sp 20 sp 20 sp 8+12 sp Total: 72 sp

Updated Plan Sprint 1 Sprint 2 Sprint 3 Buffer

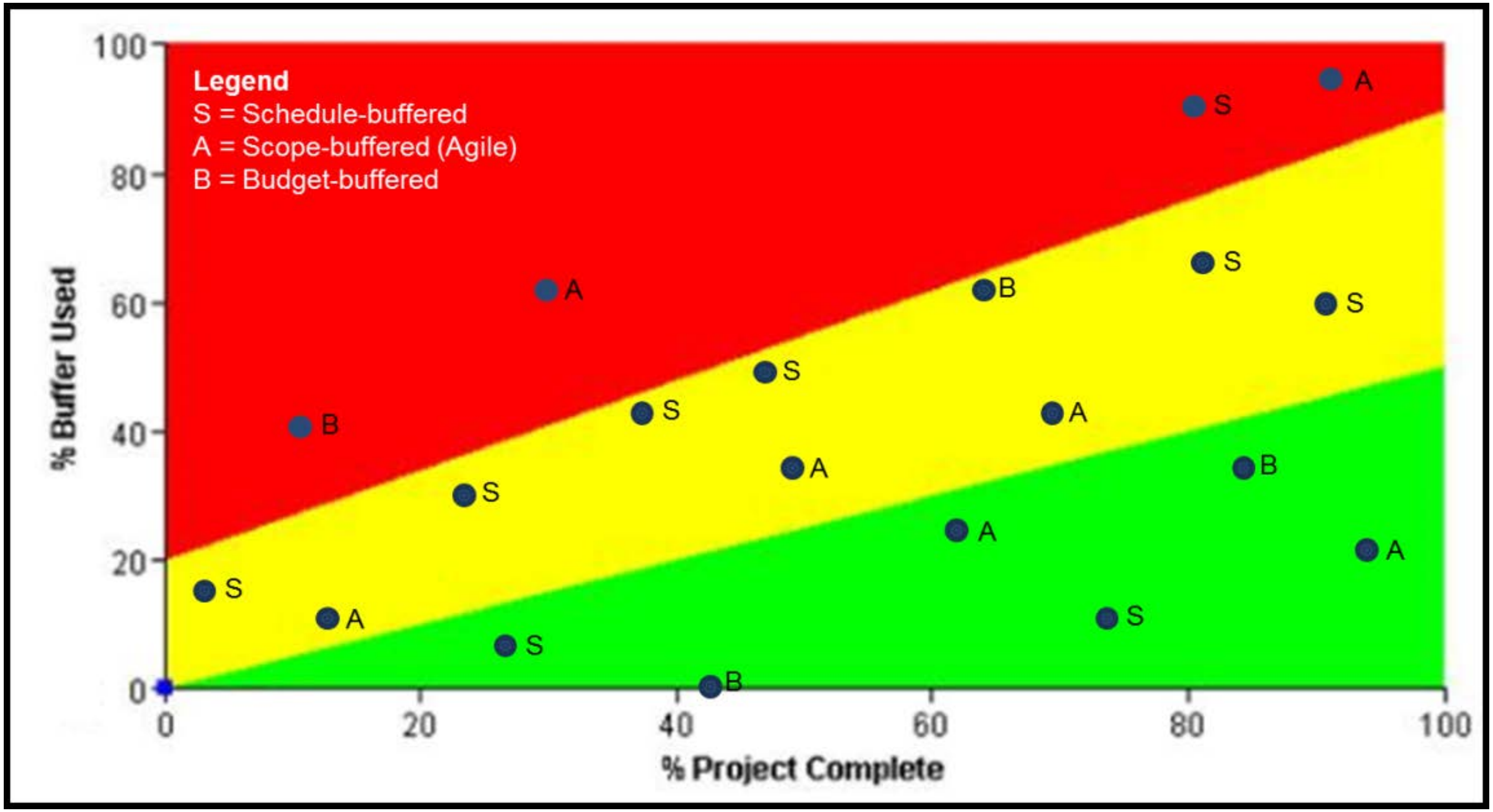
Time-Based Buffering for Budget-Buffered Projects

- Key question: How much schedule will $X of budget buffer buy us?
  - For labor-intensive projects, this is conceptually straightforward
  - Identify a schedule-accelerating mix of skills from additional staff resources available to our project, and calculate
    - The “burn rate” of these additional staff resources
    - The time allocation required of each of these additional staff resources to execute assigned tasks
    - The amount of project schedule acceleration expected to be achieved once these additional staff resources’ assigned tasks are complete

If Only It Were That Easy…

- Admittedly, this can often be much more complex, especially when any of the following exist:
  - Buy vs. build options (i.e., different ways to blend labor and materials)
  - Uncertainty over the availability of the right mix of staff resources, for the specific task durations estimated
  - Uncertainty over the cost of additional staff resources
  - Uncertainty over the quality of additional staff resources
  - …and more
- Additionally, the amount of schedule acceleration expected varies according to which tasks are targeted for acceleration, and on when in the project we decide to consume budget buffer
- Even with these complicating factors, however, this is a manageable problem for an experienced PM armed with the right PM tools

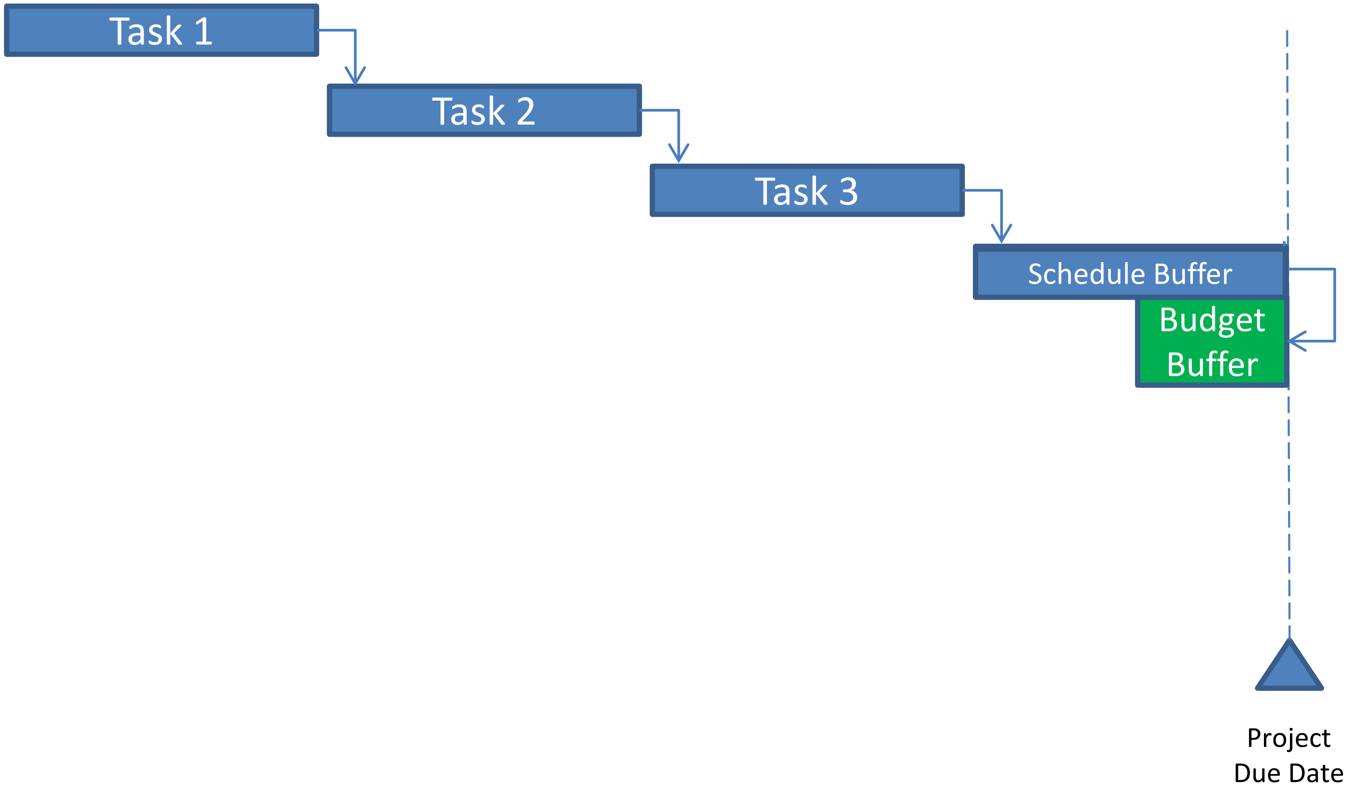
Portfolio Buffer Balancing with Flexible Buffer Types



Initial Experience in a Relevant Context

- Client is a 700-person IT organization with ~50-100 projects being executed at any given moment
- Started with multiple buffer types on a single project
  - Major software rewrite with a highly compressed schedule, and fixed minimum scope
  - Schedule compressions added some risk and complexity, such that the 2:1 schedule buffer was perceived as potentially insufficient
  - Previous false starts, strong customer demand, a fixed deployment date, and high political visibility amplified the negative consequences of late delivery
  - Performed rough calculations for how much schedule acceleration could be achieved for a variety of scenarios—all focused on where schedule risk is perceived as highest—and arrived at an affordable budget buffer ample enough to address the biggest risk areas.
  - Given that the deployment date was fixed to align with fiscal cycles and customer preference, a key stakeholder asked, “What would we do if Murphy stays away and we finish the project early?”
  - We showed “Phase 2” scope that we could begin executing for inclusion in the initial deployment, effectively showing the schedule buffer interchangeably as a scope buffer.

Conceptual Representation



Initial Experience in a Relevant Context (cont'd)

- The client requested that the entire portfolio of ~50 projects being migrated to CCPM use buffer-type flexibility, as useful and appropriate, to enhance reliability.
- Software tool support for buffer-type flexibility is lagging, so workarounds are required as we work with the vendor community.
- We will need at least a year to obtain reliability outcomes across the portfolio
- Some key questions/challenges
  - Understanding in what circumstances a schedule overrun (i.e., exhausted schedule buffer) will necessarily consume budget/scope buffer
  - Understanding "the sweet spot" of when to use budget buffer

Applicability in Ship & Aircraft Design & Maintenance

- Known Unknowns in Maintenance & Repair
  - Scope uncertainty during planning
  - “Open, inspect, then decide what to do”
- Delays
- Exceeding Budget & Unspent Funds

Example: Keeping Ship Design Deliverables on Track

- A Defense Services Engineering, Technology and Consultancy Firm in Turkey
- Houses the largest Naval Design Team in Turkey
- STM is looking for methods to accelerate design work and achieve earlier deliveries for three Naval Surface Ship Design Contracts
  - Simultaneously delivering Technical & Production Data Packages to one local and one foreign Naval Shipyard and to Ministry of National Defense
  - Ship Construction flows depend on the arrival of design documentation; delays in deliverables delay the ship production
  - Number of Design Personnel is to be kept at a certain level

STM ENGINEERING TECHNOLOGY CONSULTANCY

Buffer-type Flexibility: Key Takeaways

- PMs pursue buffer-type flexibility intuitively—we need to take this intuition and lend it discipline in order to make it more effective
- In order for executives to optimize the reliability of their project portfolios, buffers must be visible, and represented in an “apples-to-apples” manner
- Showing all buffers as time-based is usually the most intuitive for executives, project managers, and team members
- Helping one project by using buffer from another is straightforward, though “cross-typing” may present some additional considerations
- Allows differing project-level methods—including Agile—as long as time-based buffering is adhered to

Q&A

Contact info:

- Michael Hannan, Fortezza Consulting, Mike@FortezzaConsulting.com
- Bahadir Inozu, Ph.D., NOVACES, binozu@novaces.com
- Hilbert Robinson, Delta Airlines, hilbert.robinson@delta.com