Implementing Product Development Flow: The Key to Managing Large Scale Agile Development

Will Hayes – SEI

Software Solutions Conference 2015

November 16–18, 2015

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0002729

Agenda

Common Perceptions about Agile

Cadence and Synchronization

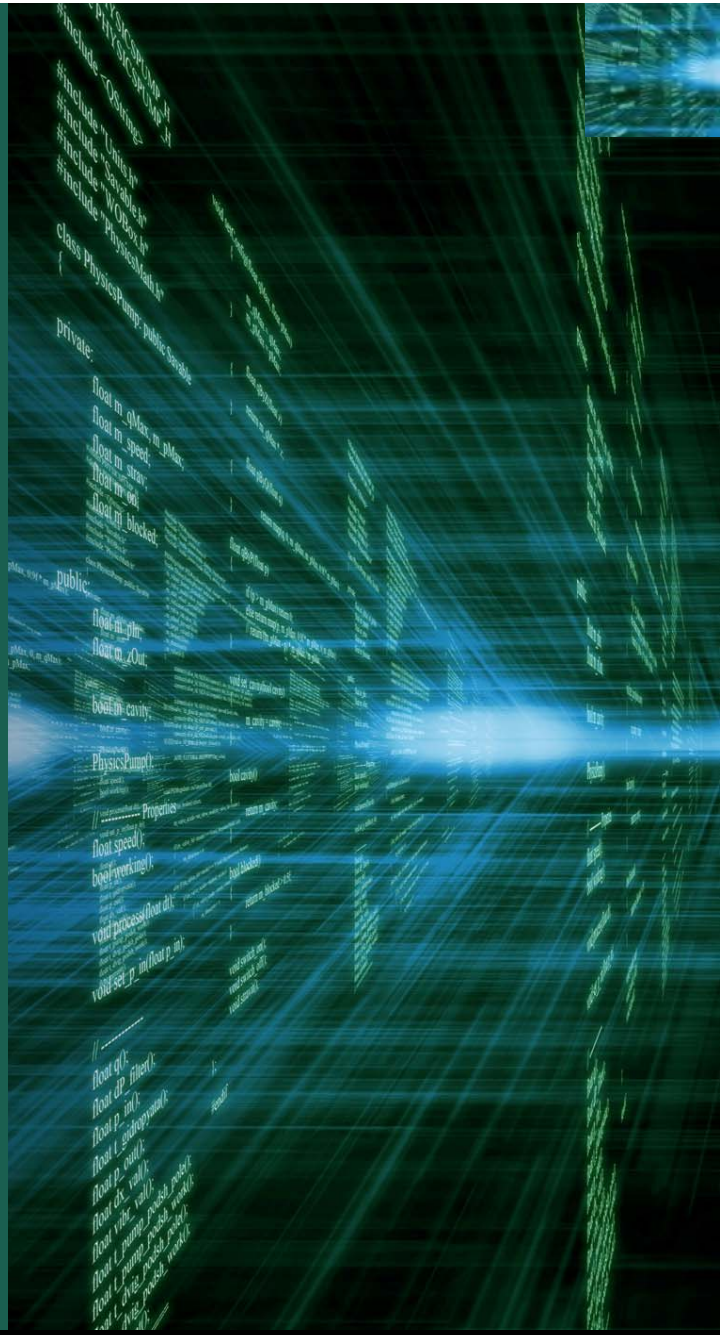
Unhealthy Focus on Utilization

Cost of Delay

Conclusion

Agile

Common Perceptions



The So-Called "Traditional Approach"

It's not the heavy weight of documentation that gets you…

It's the long wait for course-correcting feedback that can kill your program…

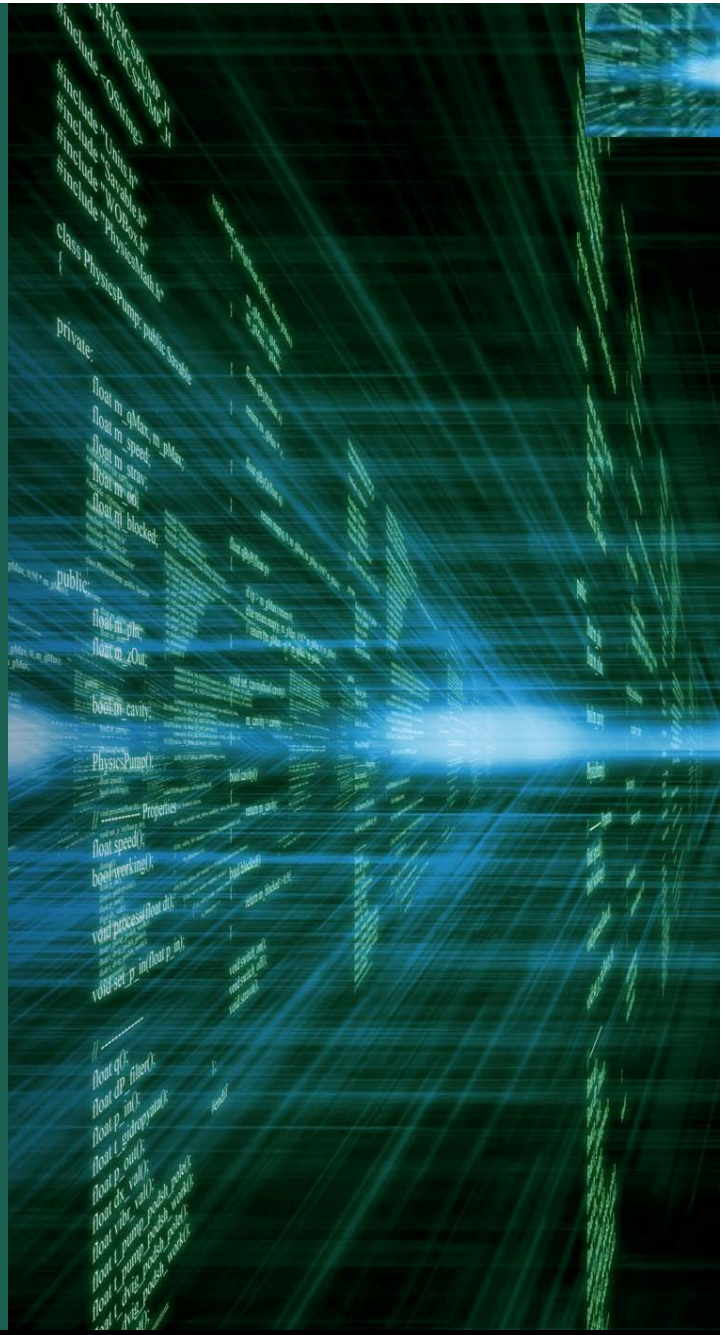
According to Mark Twain…

It ain't what you don't know that gets you in trouble. It's what you know for sure that just ain't so.

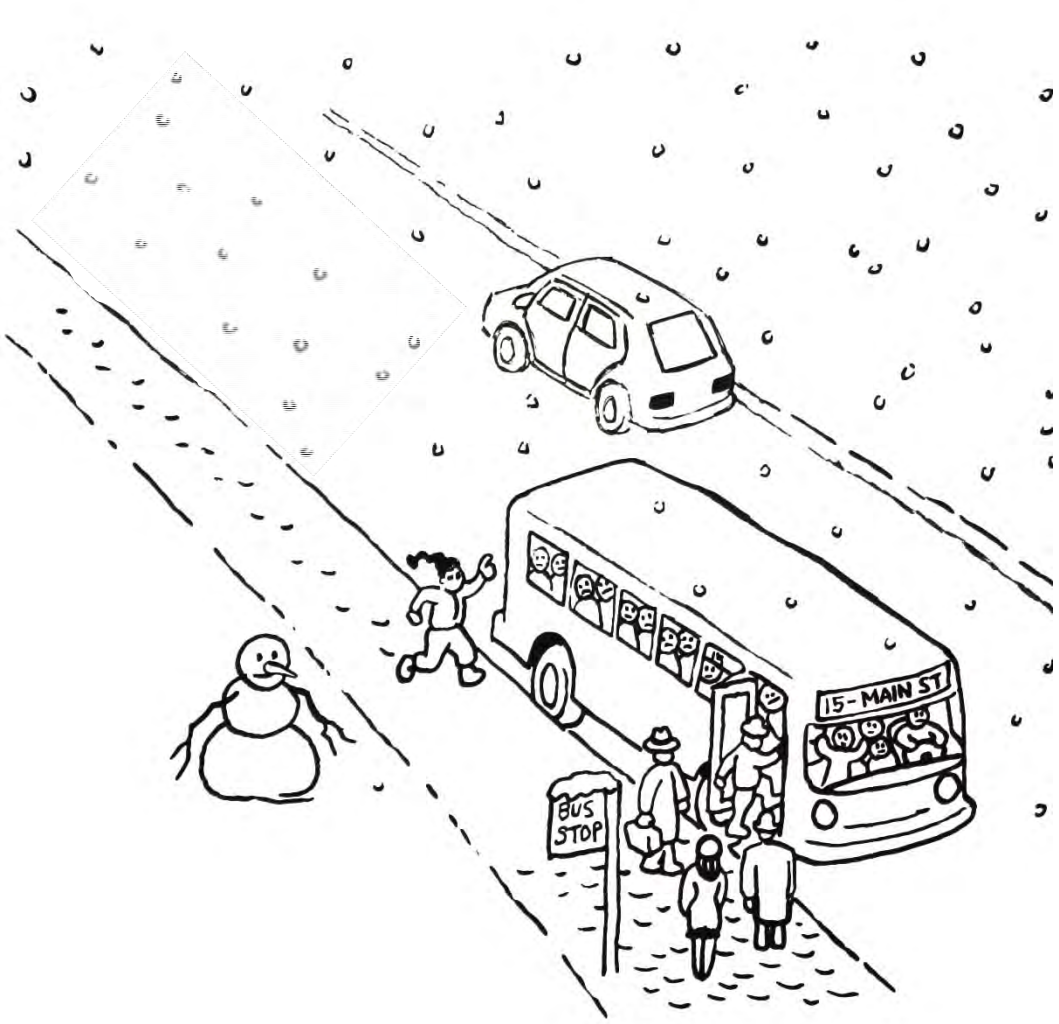
Samuel Langhorne Clemens

Product Development Flow

Cadence & Synchronization



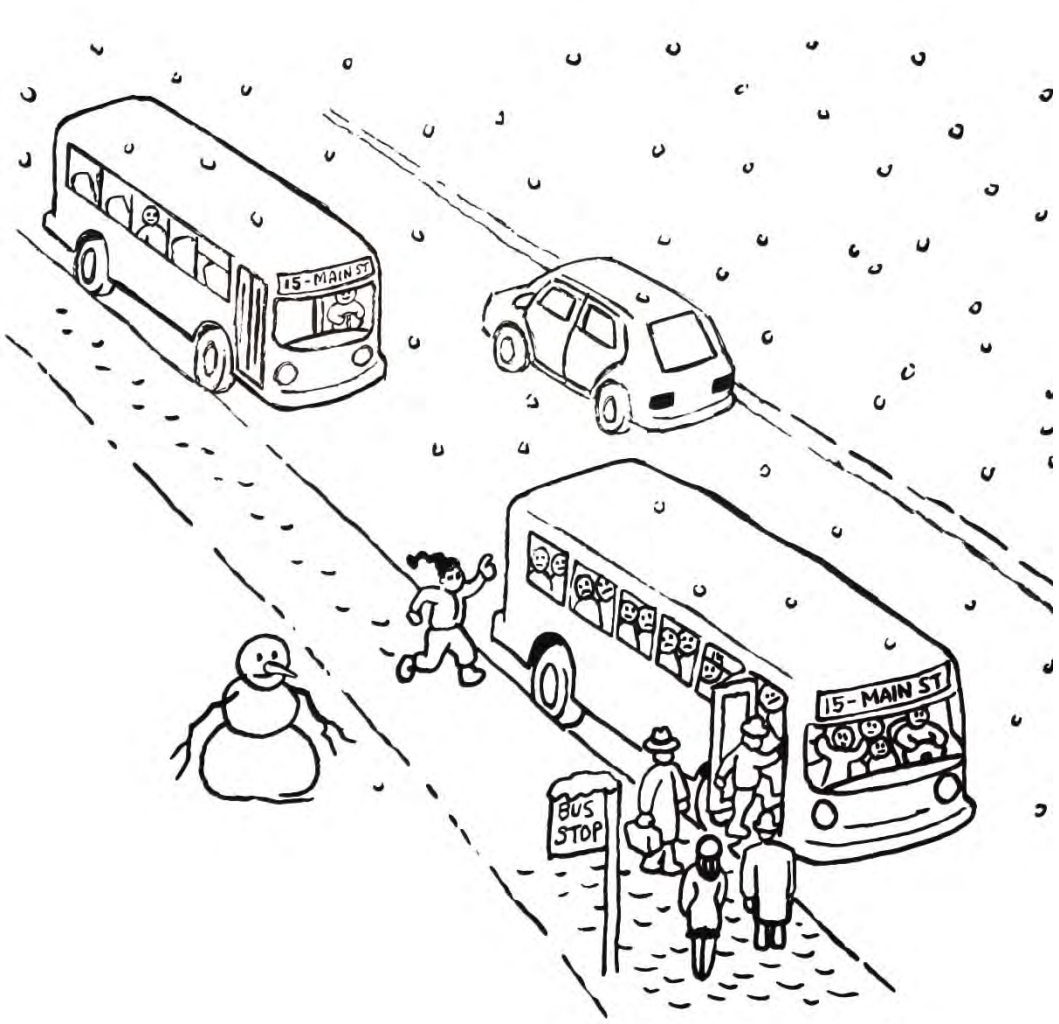
Cadence Enhances Predictability



A Late Bus:

- Makes people scramble to get aboard
- They don't know when the next one will get here

Cadence Enhances Predictability

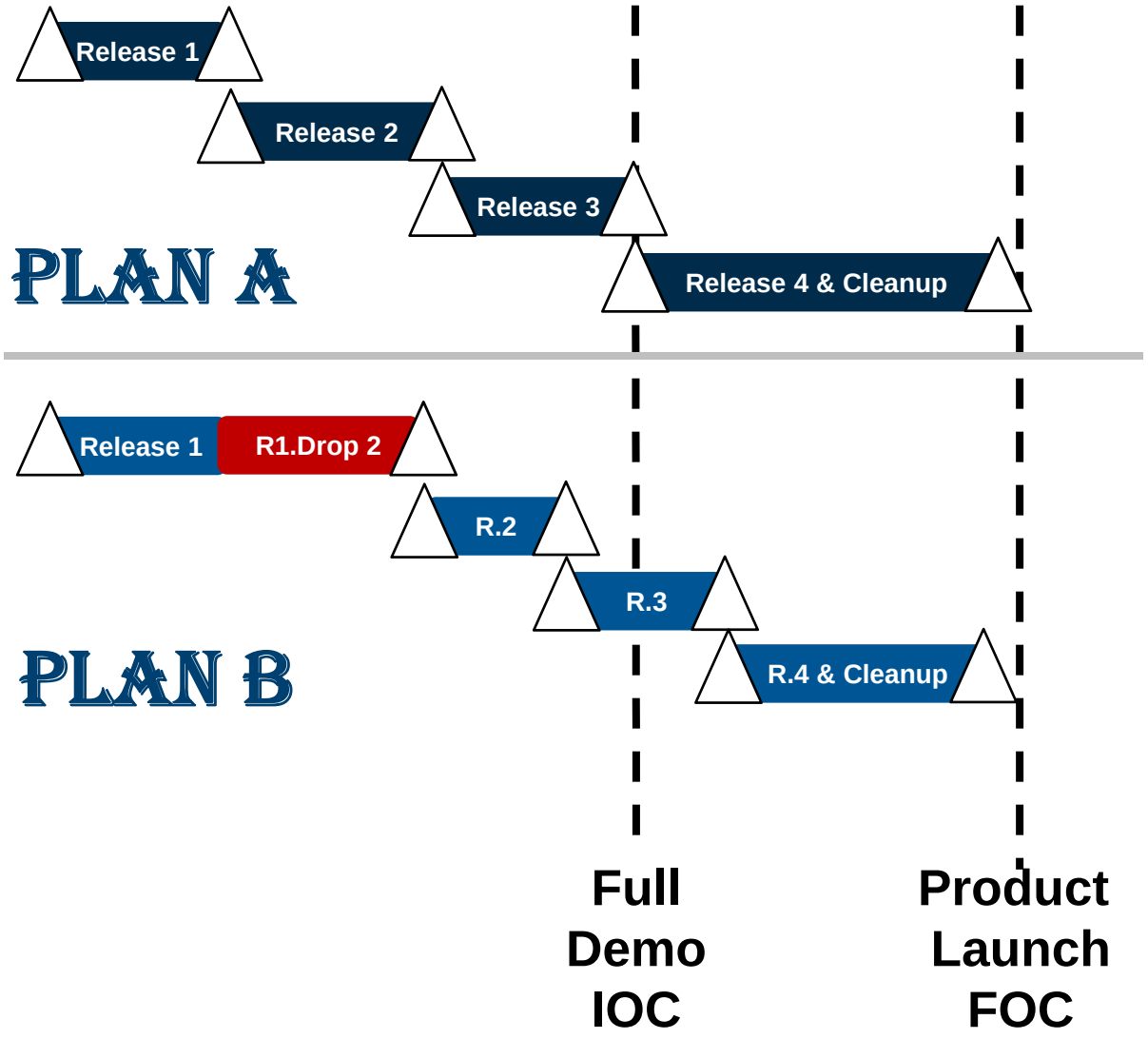


A Late Bus:

- Makes people scramble to get aboard
- They don't know when the next one will get here

Then the next bus comes along empty

Late Releases Become “Feature Magnets”

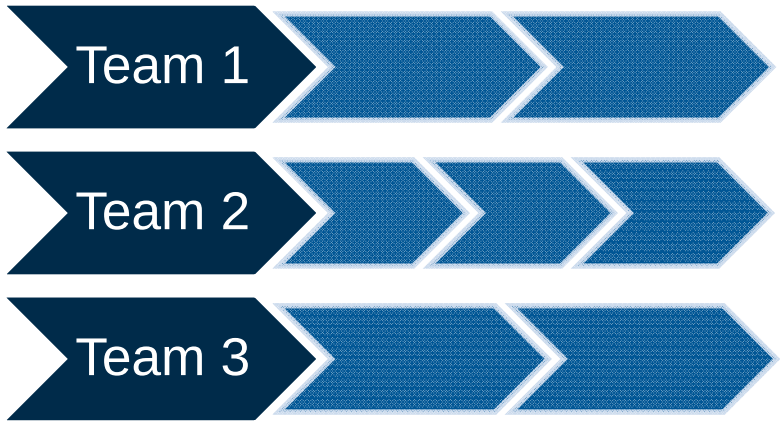


As things start to slip

- Influential people get ‘their priorities’ moved up, rather than deferred
- Pressure increases on early releases
- Functions slated for final release can’t be guaranteed…

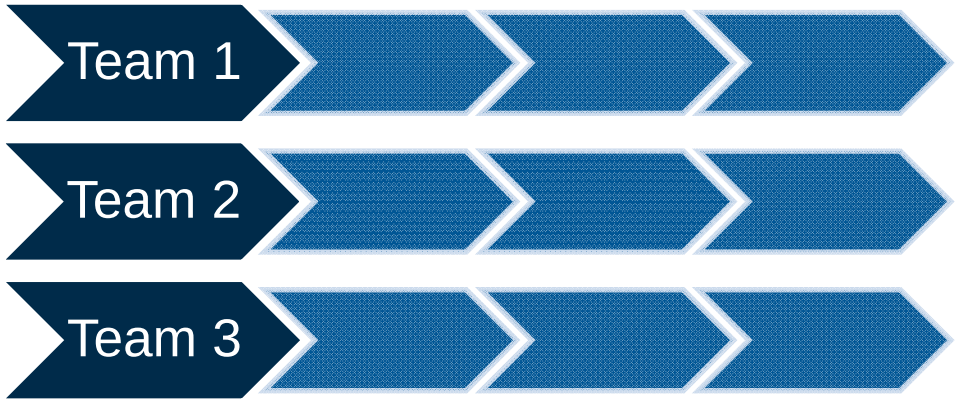
Importance of Synchronization

Non-synchronized schedules can lead to counter-productive dynamics…



One team's schedule slip can give other teams the schedule relief they didn't want to ask for…

Synchronization Promotes Visibility

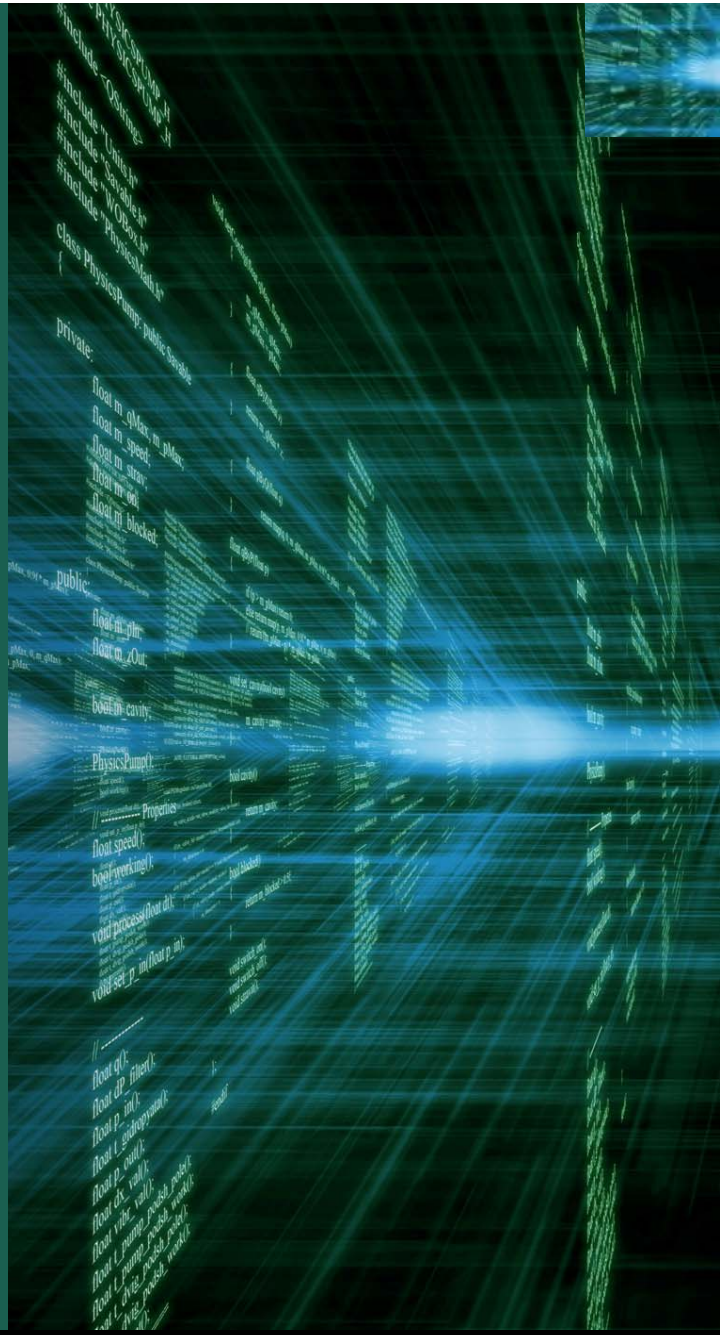


Added incentive to maintain cadence…

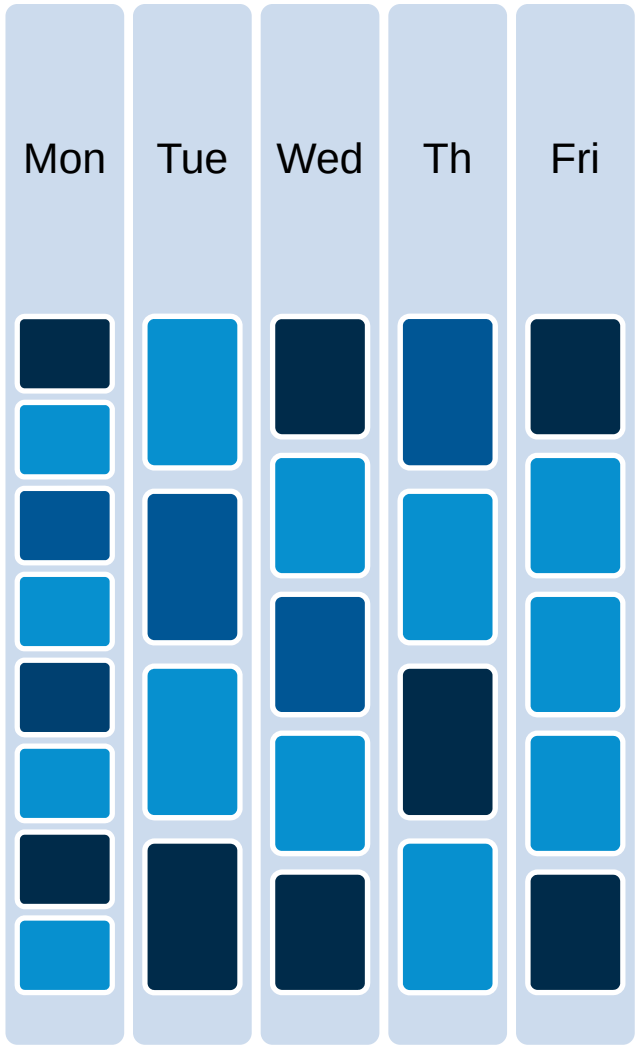
Frequent ‘synch-points’ offer more options for course-correction…

Product Development Flow

Unhealthy Focus on Utilization



Packing Scheduled Tasks is Prone to Risk



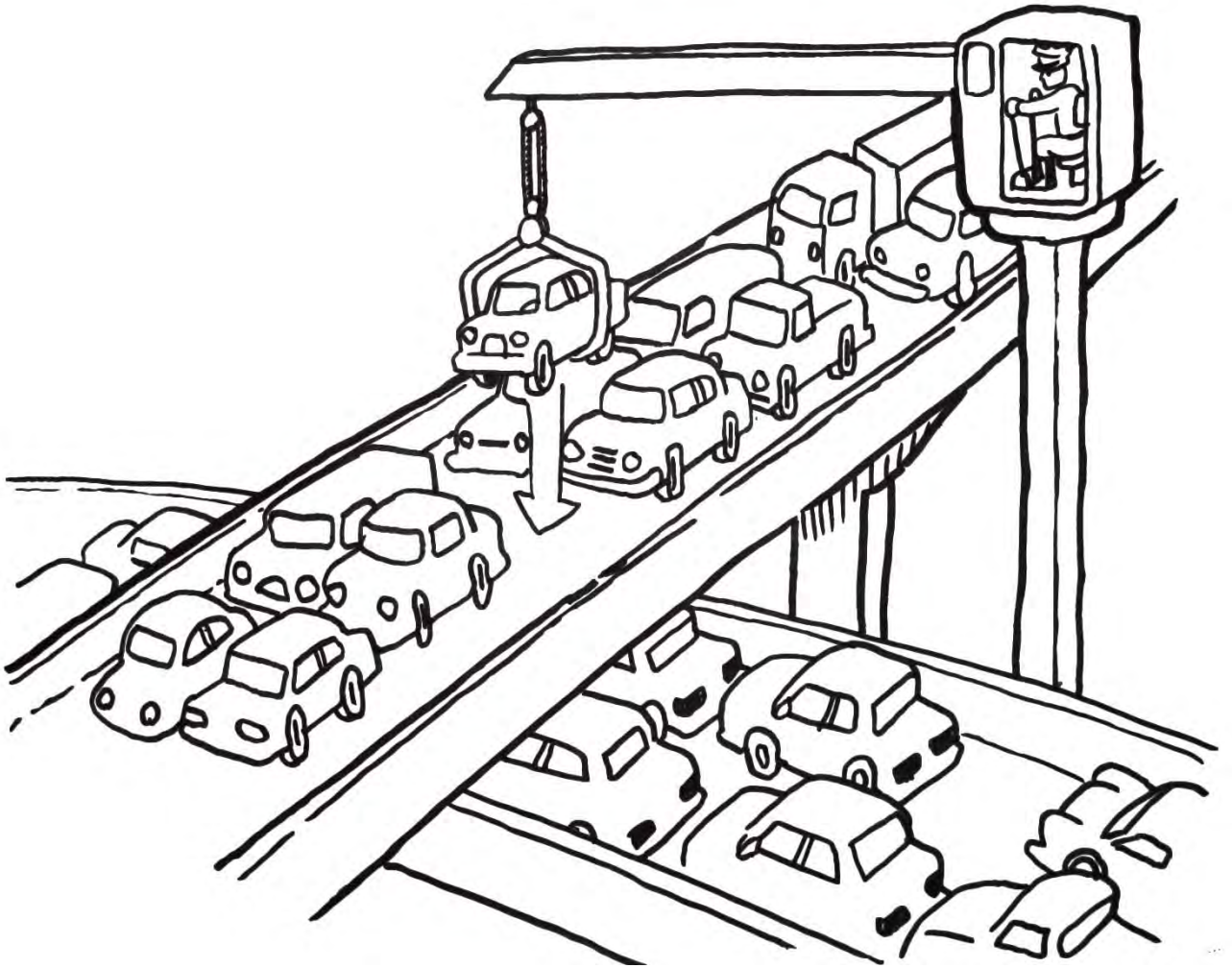
100% Utilization:

- Magnifies the impact of variation
- Maximizes task-switching overhead
- Assures slower overall progress

Change is inevitable, plan to learn

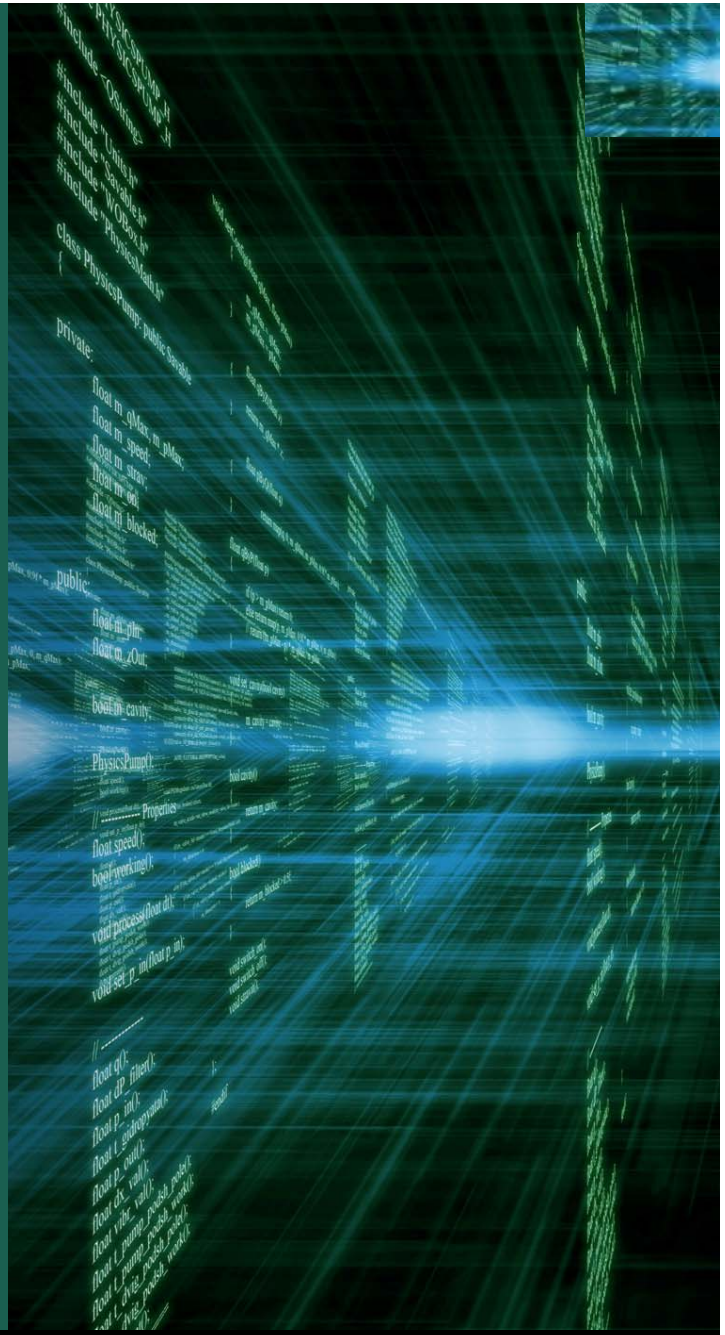
Multi-tasking is a myth we don't accurately comprehend

Maximum Utilization is Counterproductive



Product Development Flow

Cost of Delay



Spend Your Time Wisely

Look for the ‘sweet spot’ between

- *Analysis Paralysis*
- *Extinction by Instinct*

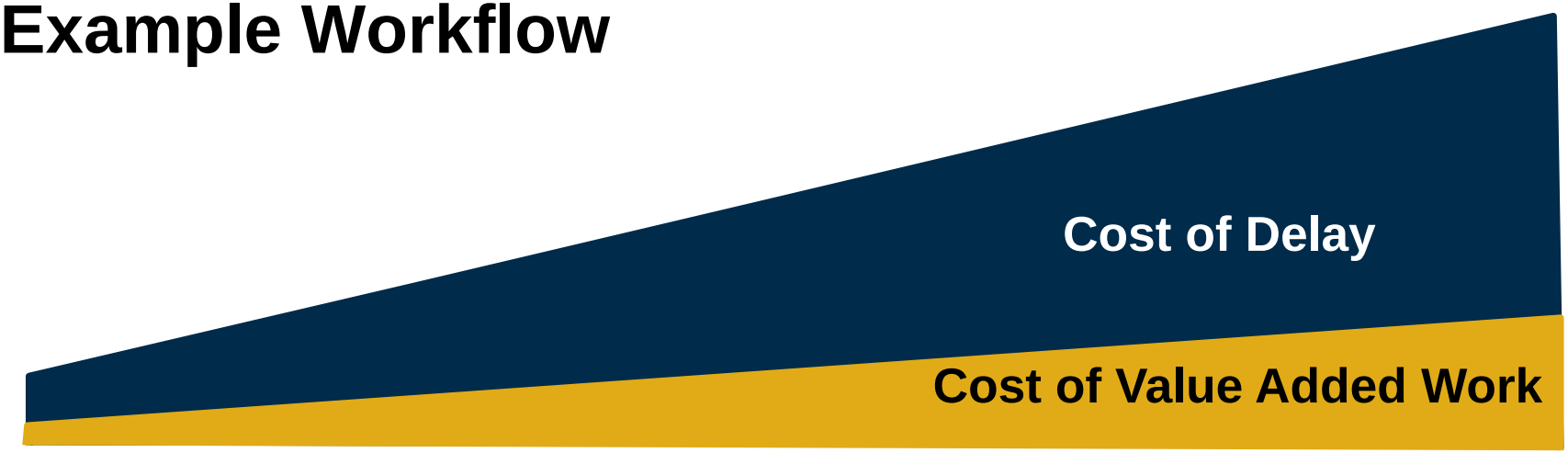
Reduce integration risk

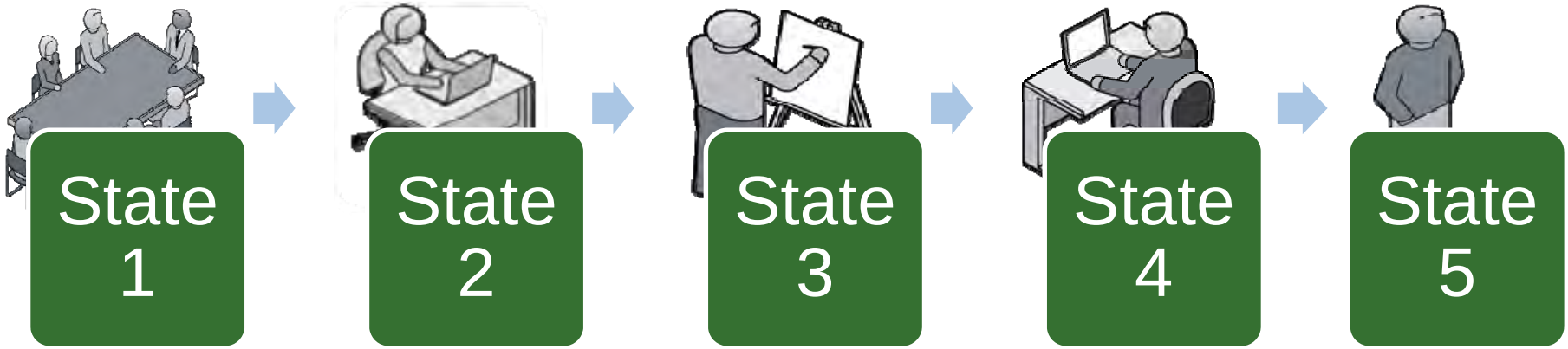
- Invest in architecture to set the stage for later work
- Validate with each iteration

Plan for learning

- Time the critical design choices with availability of information

Example Workflow

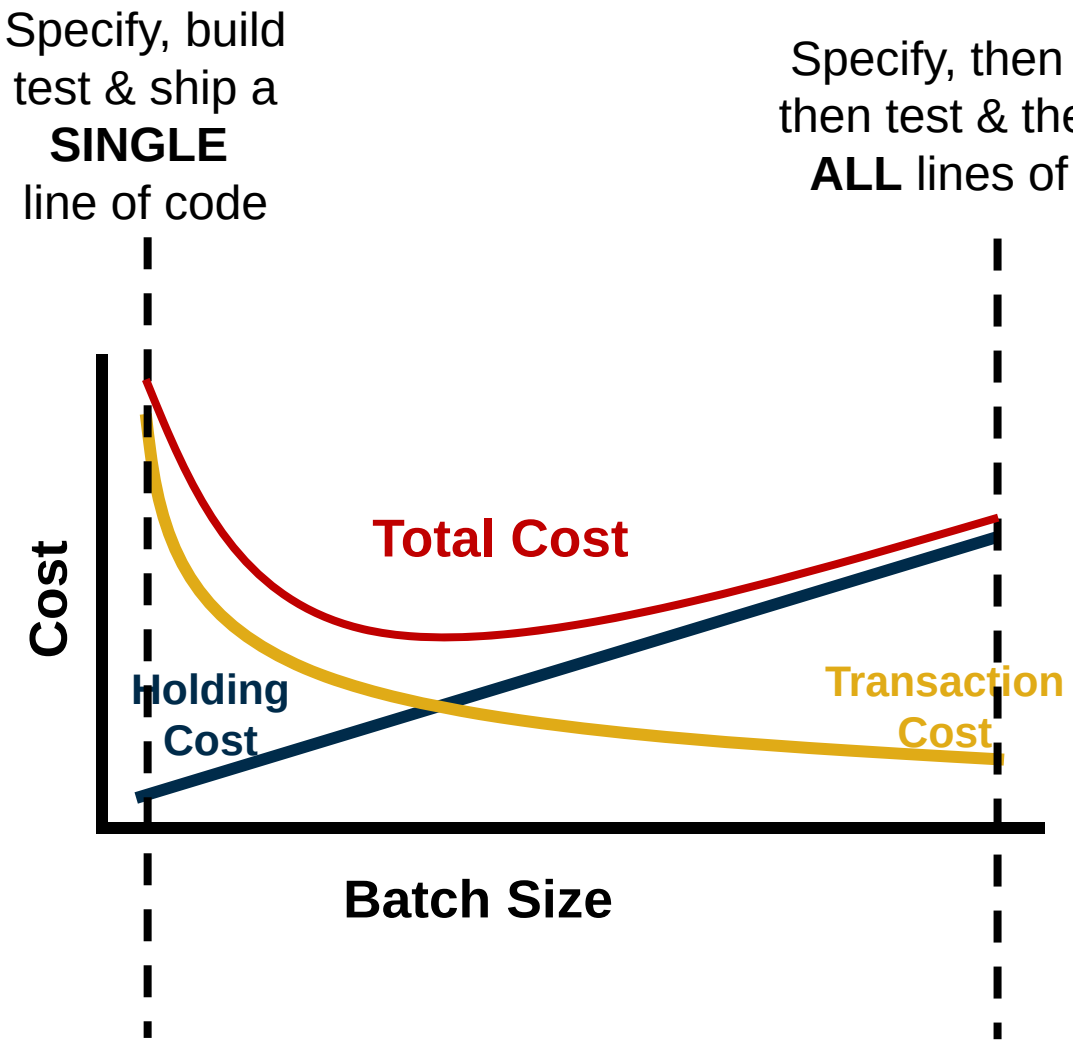




Build queues where knowledge can accumulate

Stage items in batches if they belong together

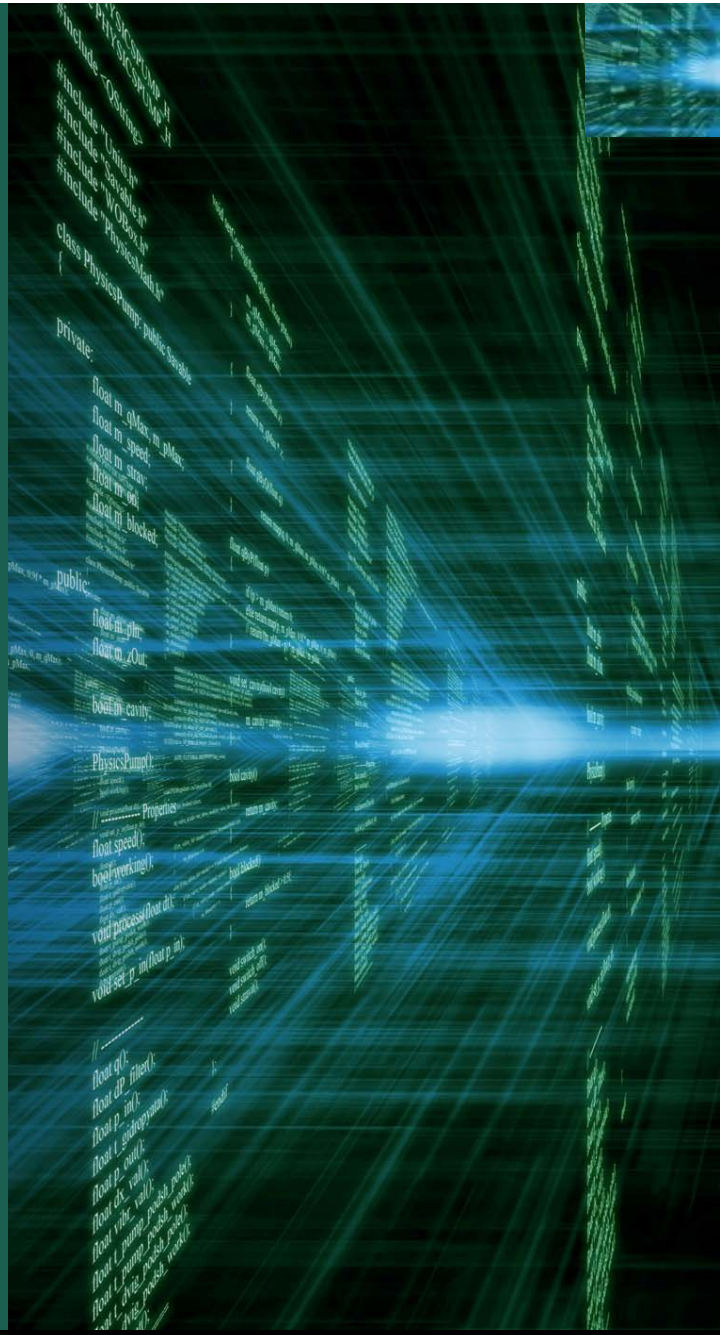
Economies of Batch Size



U-Curve optimization problem as described in *Principles of Product Development Flow*, by Don Reinertsen

Product Development Flow

Conclusion

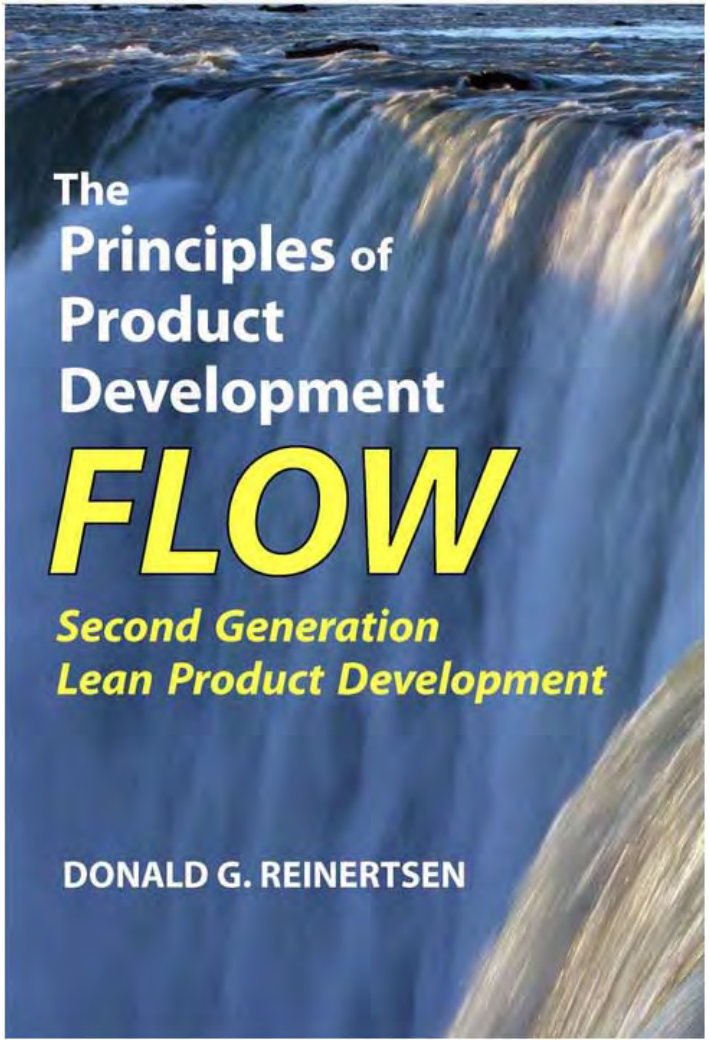


Priorities for Action

For Your Work Processes:

- Devise a regular cadence on which people can rely
- Synchronize often to reinforce cadence and visibility
- Resist the habitual focus on maximizing utilization
- Optimize at the system level – not at the unit level
- Characterize the cost of delay as an economic factor
- Balance holding costs and transaction costs

Credits



See also:

Managing the Design Factory: A Product Developer's Toolkit

Contact Information

Presenter

Will Hayes

Principal Engineer

Telephone: +1 412.268.6398

Email: wh@sei.cmu.edu