



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

Will Hayes – SEI

Software Solutions Conference 2015
November 16–18, 2015



Software Engineering Institute

Carnegie Mellon University

© 2015 Carnegie Mellon University

Distribution Statement A: Approved for Public Release; Distribution is Unlimited

Copyright 2015 Carnegie Mellon University

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.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM-0002729



Agenda

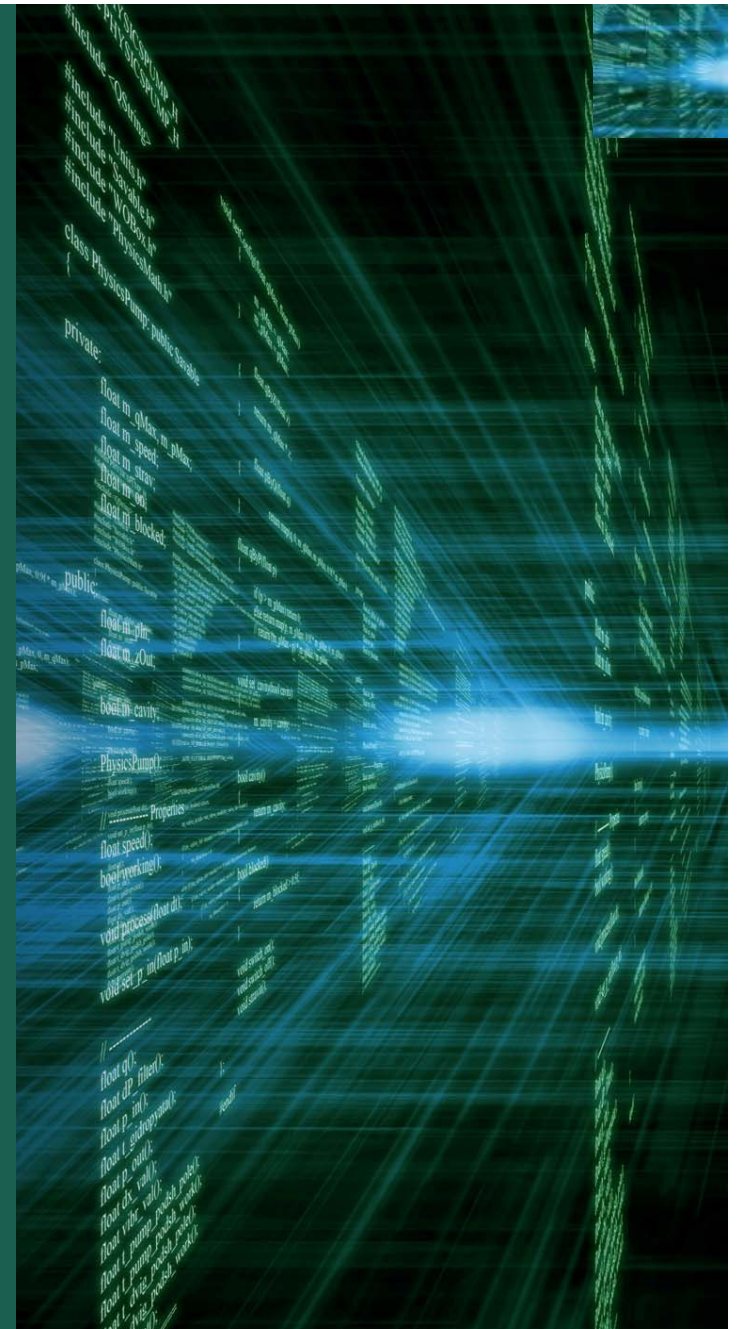


Common Perceptions about Agile
Cadence and Synchronization
Unhealthy Focus on Utilization
Cost of Delay

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...

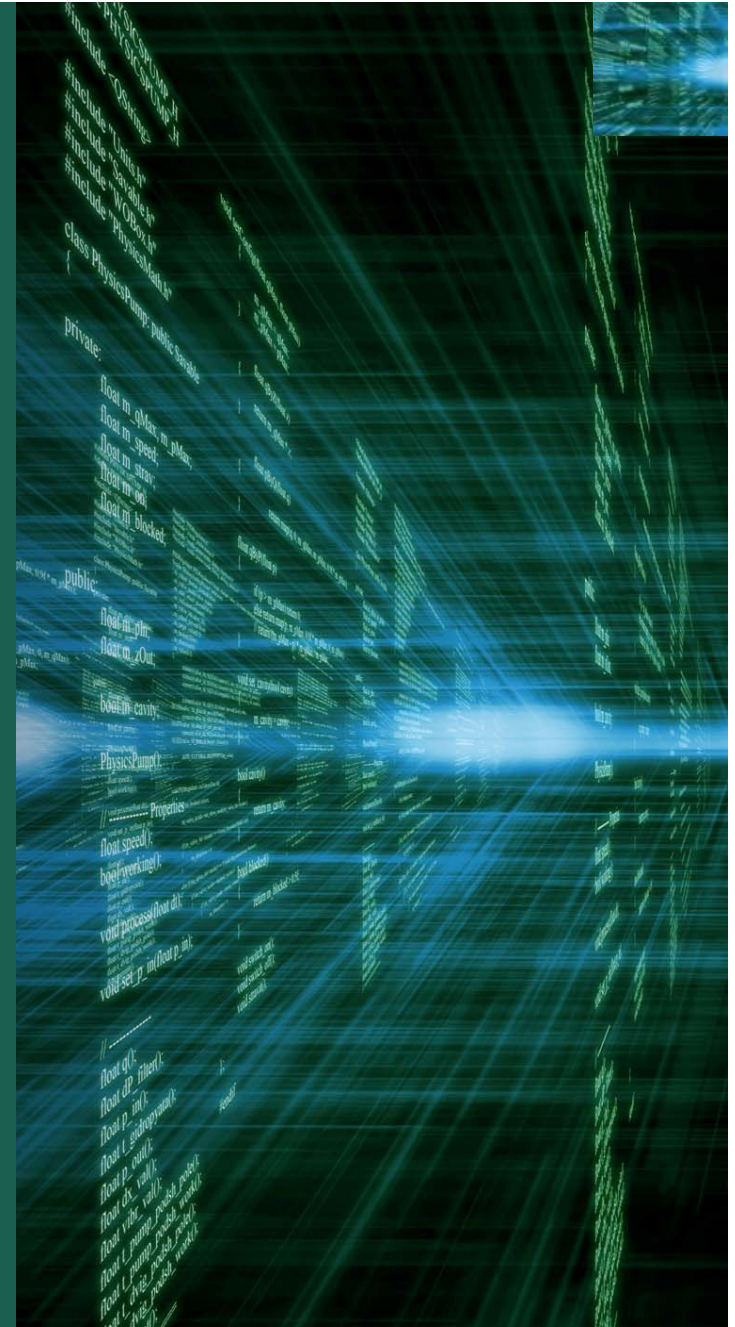


It ain't what you
don't know that
gets you in trouble.
It's what you know
for sure that just
ain't so.

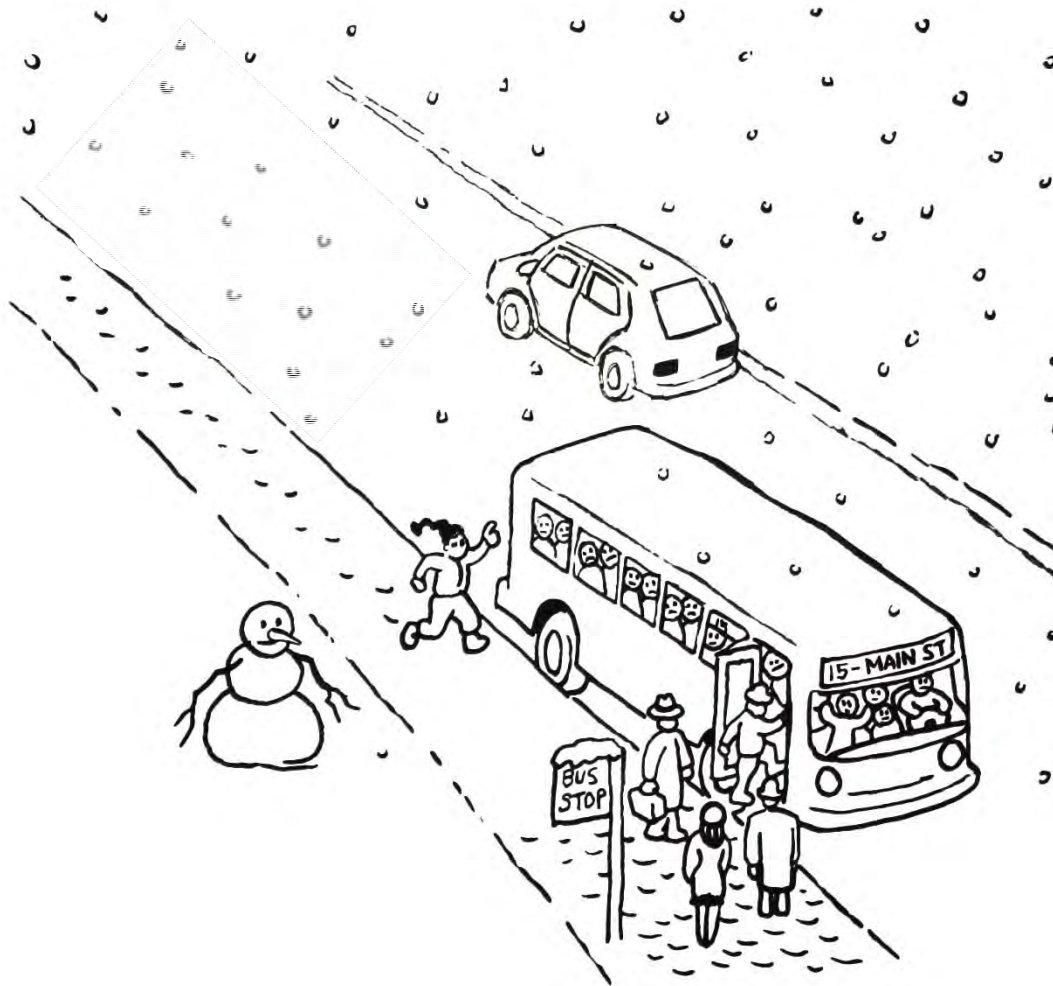
Samuel Langhorne Clemens



Product Development Flow Cadence & Synchronization



Cadence Enhances Predictability

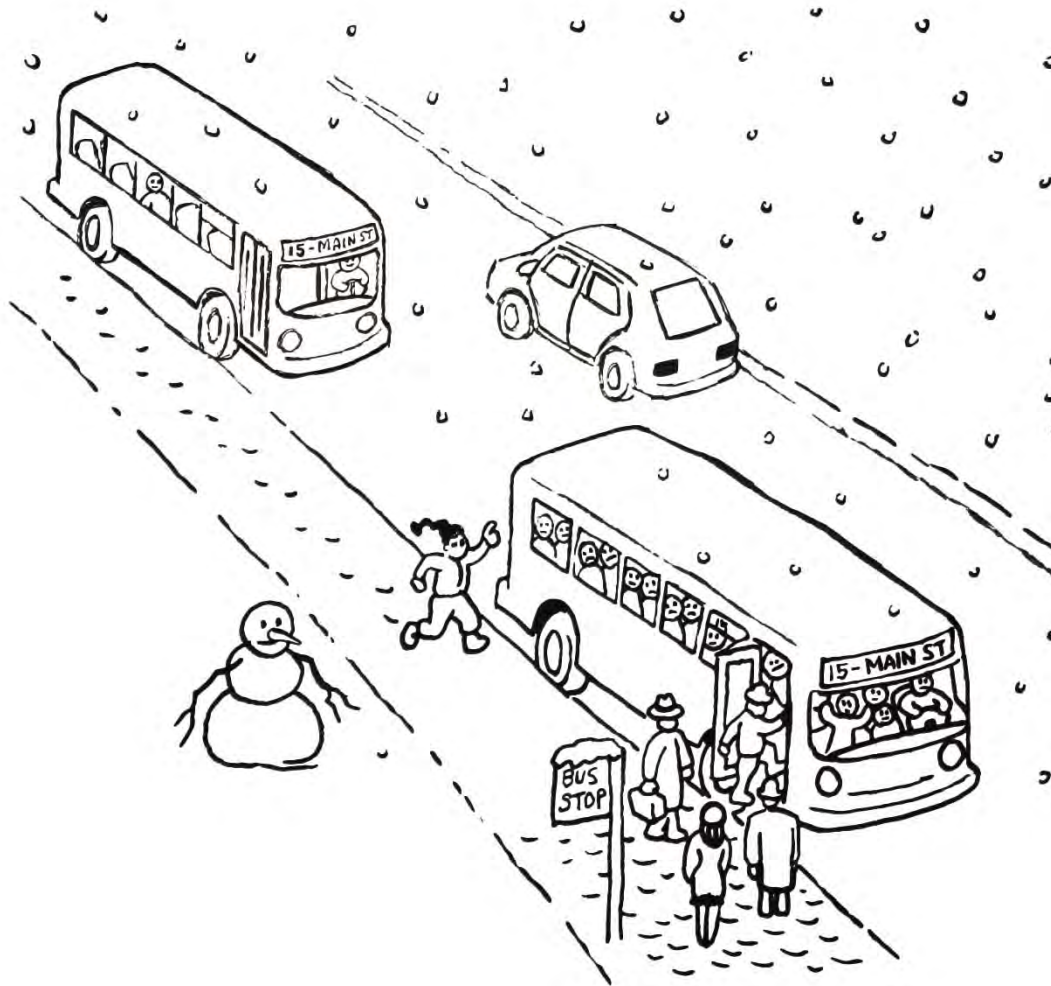


A Late Bus:

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



Cadence Enhances Predictability



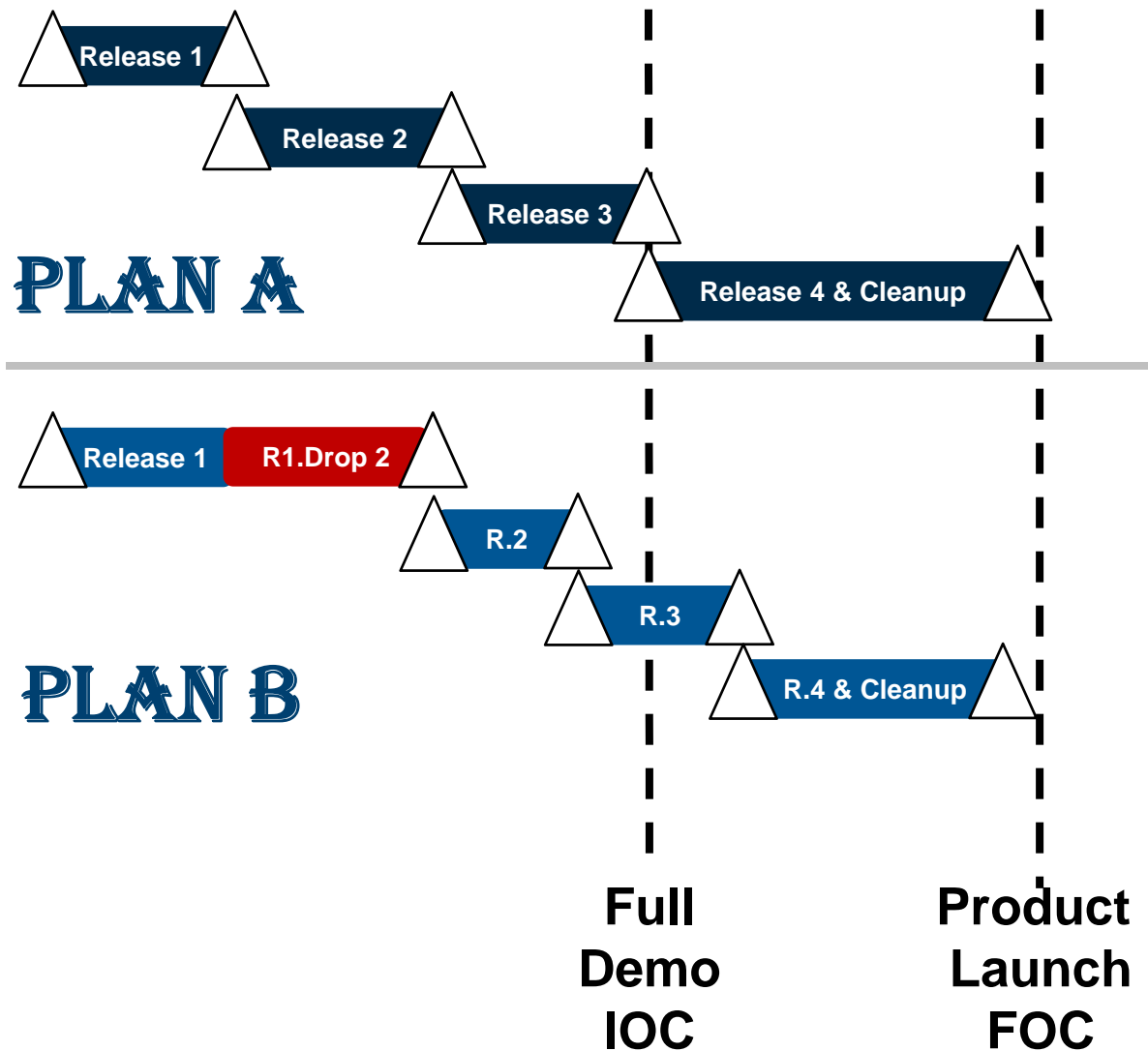
A Late Bus:

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

Then the next bus comes along empty



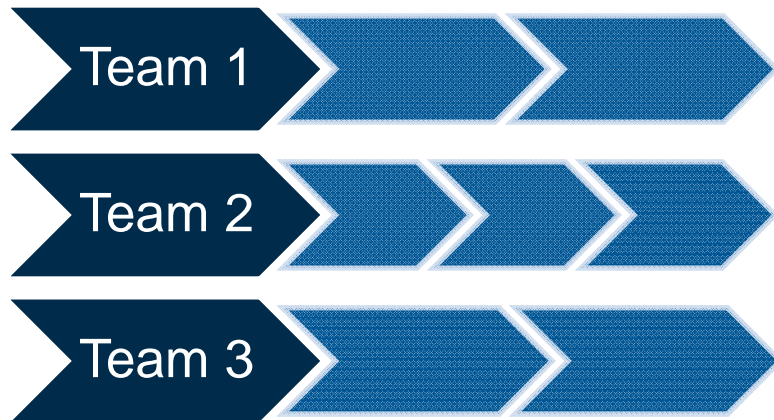
Late Releases Become “Feature Magnets”



- 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

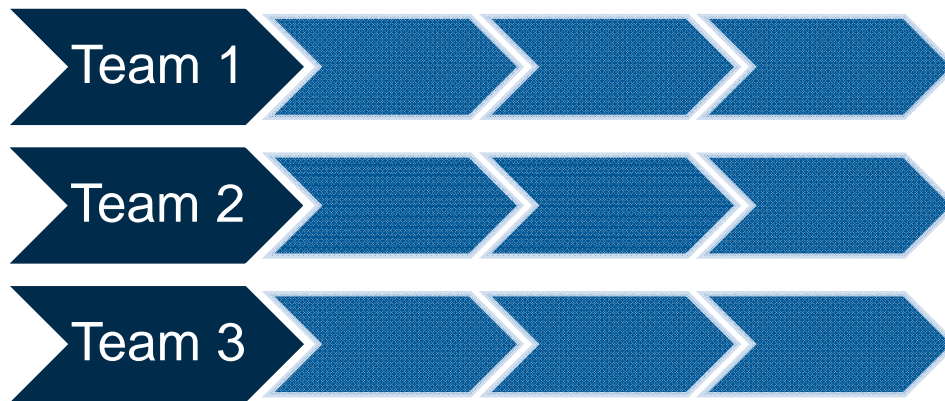
Non-synchronized schedules can lead to counter-productive dynamics...



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



Synchronization Promotes Visibility



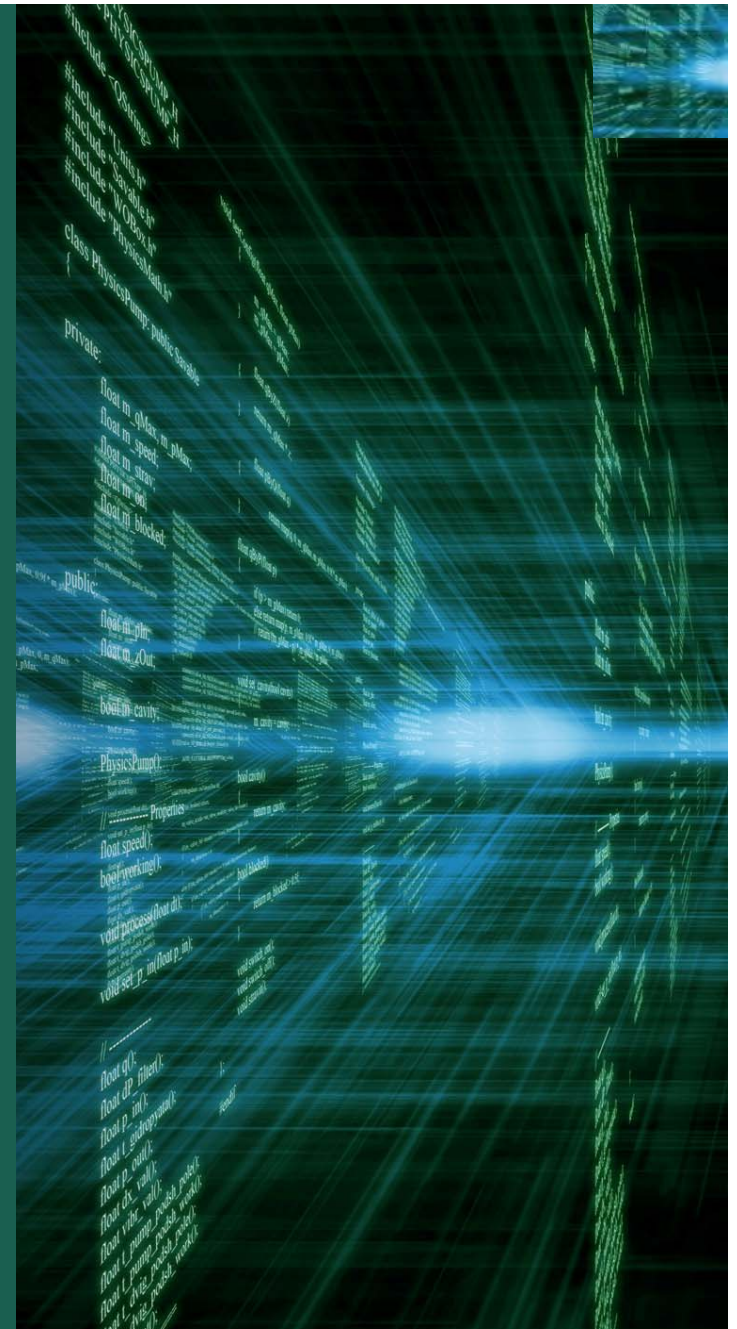
Added incentive to maintain cadence...

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:

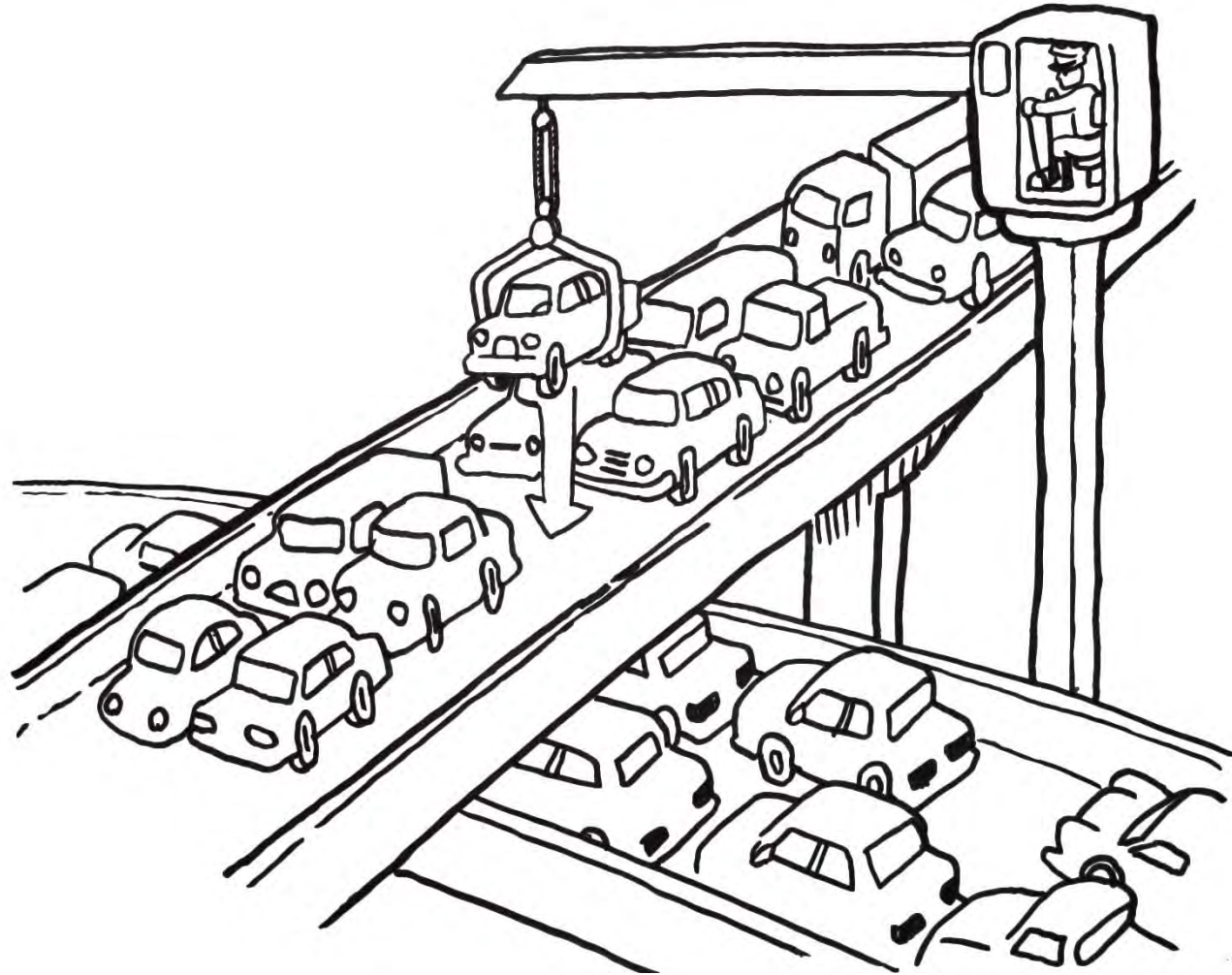
- Magnifies the impact of variation
- Maximizes task-switching overhead
- Assures slower overall progress

Change is inevitable, plan to learn

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

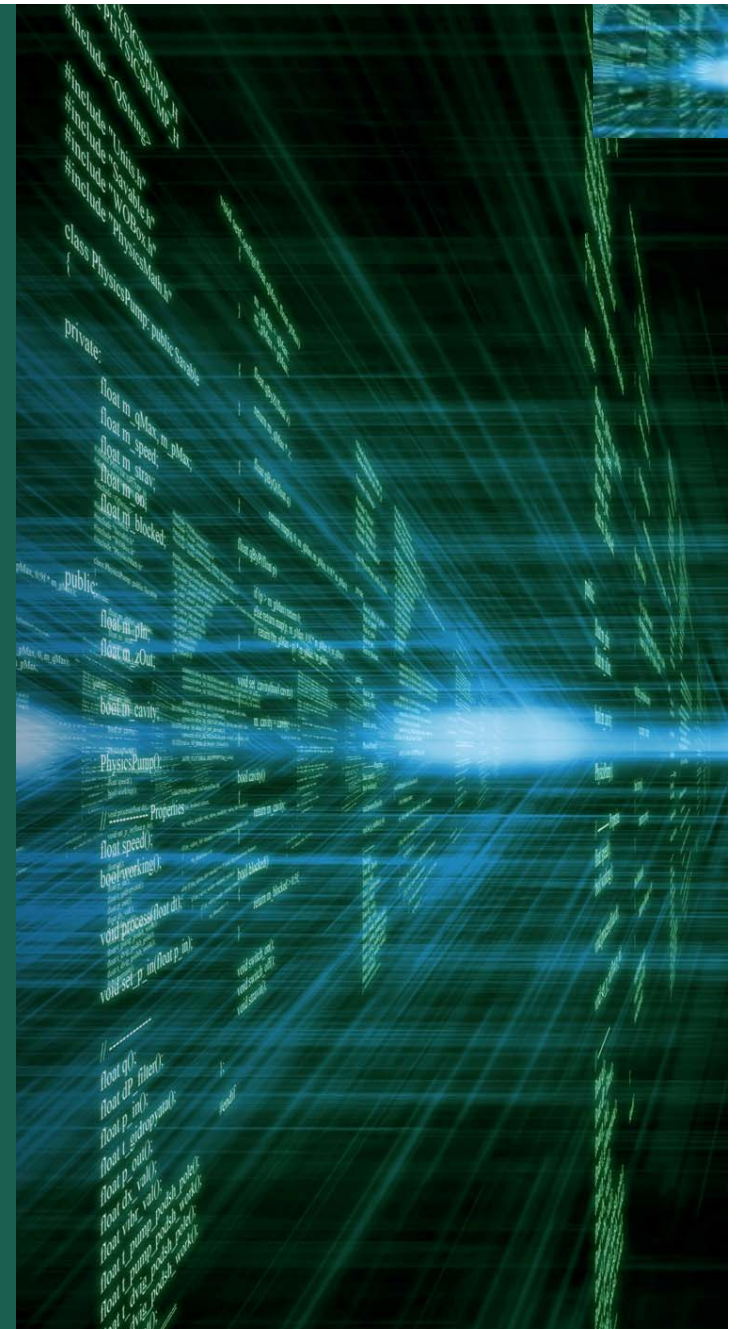


Maximum Utilization is Counterproductive



Product Development Flow

Cost of Delay



Spend Your Time Wisely



Look for the 'sweet spot' between

- *Analysis Paralysis*
- *Extinction by Instinct*

Reduce integration risk

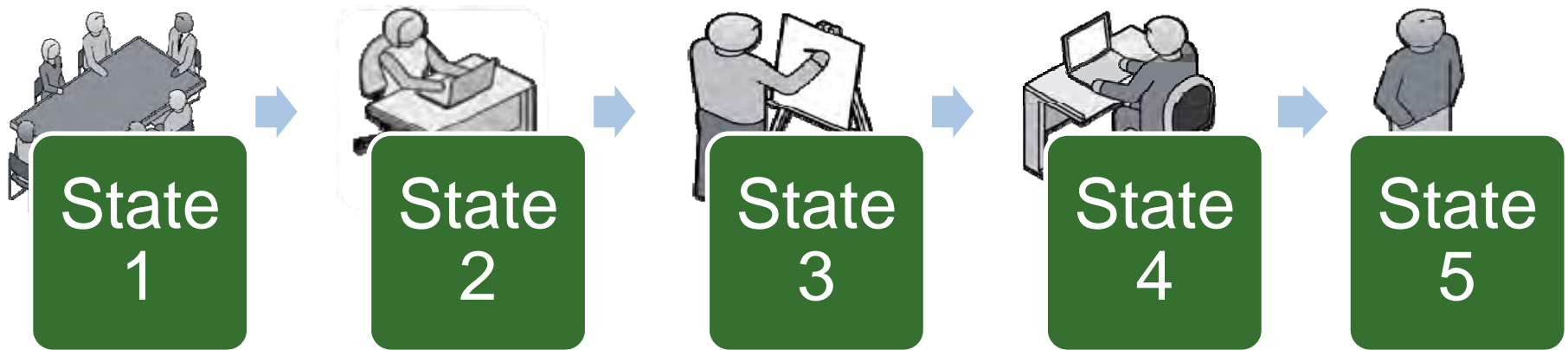
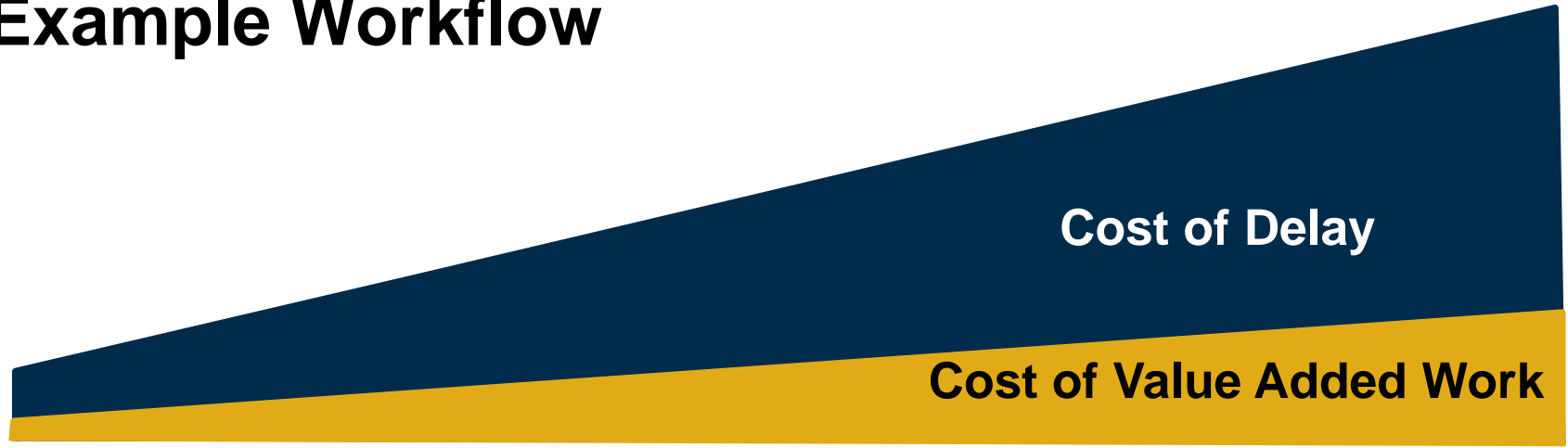
- Invest in architecture to set the stage for later work
- Validate with each iteration

Plan for learning

- Time the critical design choices with availability of information



Example Workflow

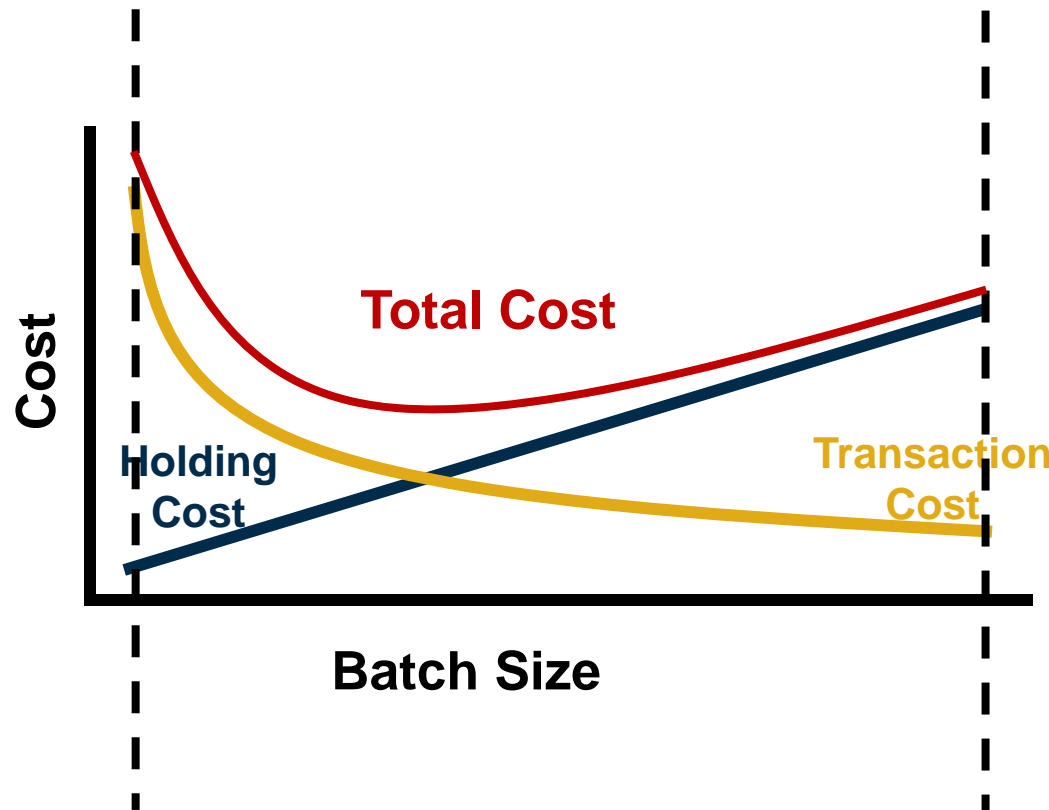


Build queues where knowledge can accumulate
Stage items in batches if they belong together

Economies of Batch Size

Specify, build
test & ship a
SINGLE
line of code

Specify, then build,
then test & then ship
ALL lines of code

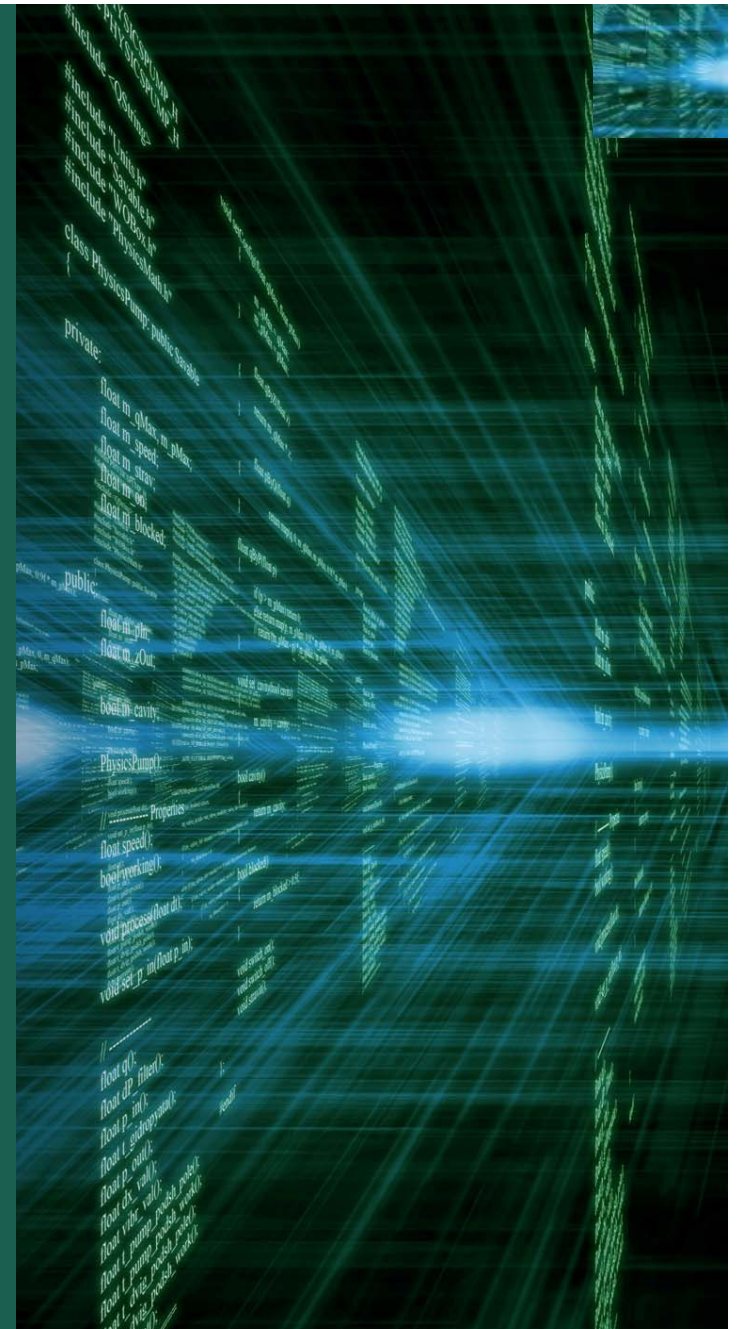


U-Curve optimization
problem as described in
*Principles of Product
Development Flow*, by
Don Reinertsen



Product Development Flow

Conclusion



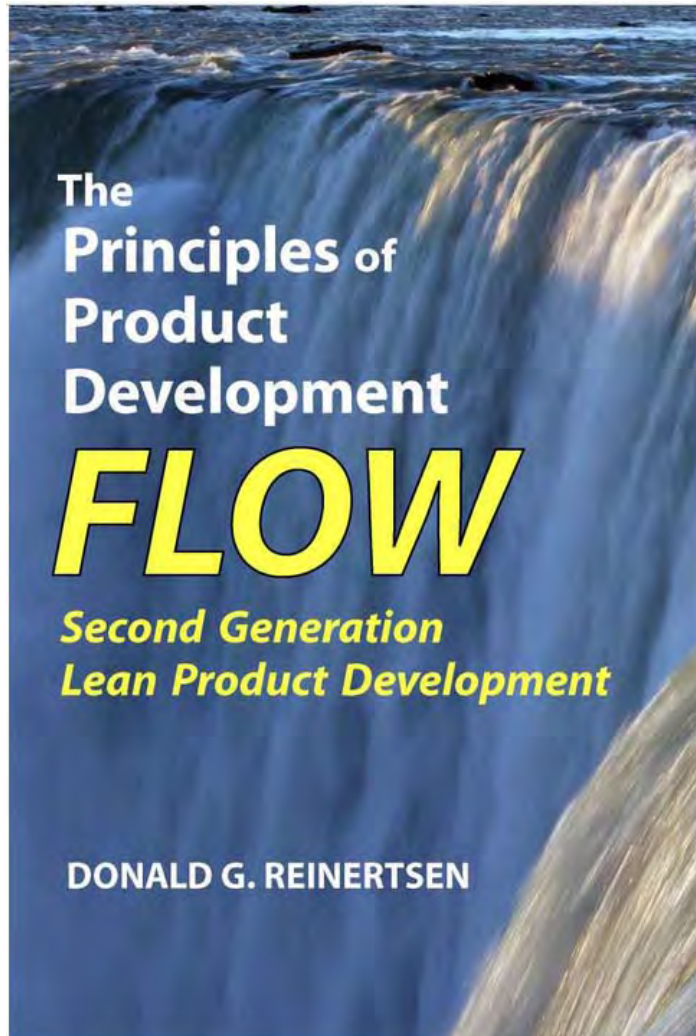
Priorities for Action

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

