Past the Tipping Point:

Understanding Firefighting in New Product Development

Nelson P. Repenning

Assistant Professor

Operations Management/System Dynamics Group
Alfred P. Sloan School of Management, MIT
30 Wadsworth St., Cambridge, 02142
voice:617/258-6889 fax:617/258-7579
e-mail:<nelsonr@mit.edu>

Design, Implementation, and Execution

- **Improved practice** requires both a more effective set of tools and processes and organizations that use them effectively.
- While much current research focuses on the ***design*** of new tools and processes, organizations also struggle with both the ***implementation*** and ***execution*** of existing best practices.
- This often occurs despite general agreement that the prescribed methods are better than those in use:
  - **"*The [new process] is a good one. Some day I'd like to work on a project that actually uses it".***

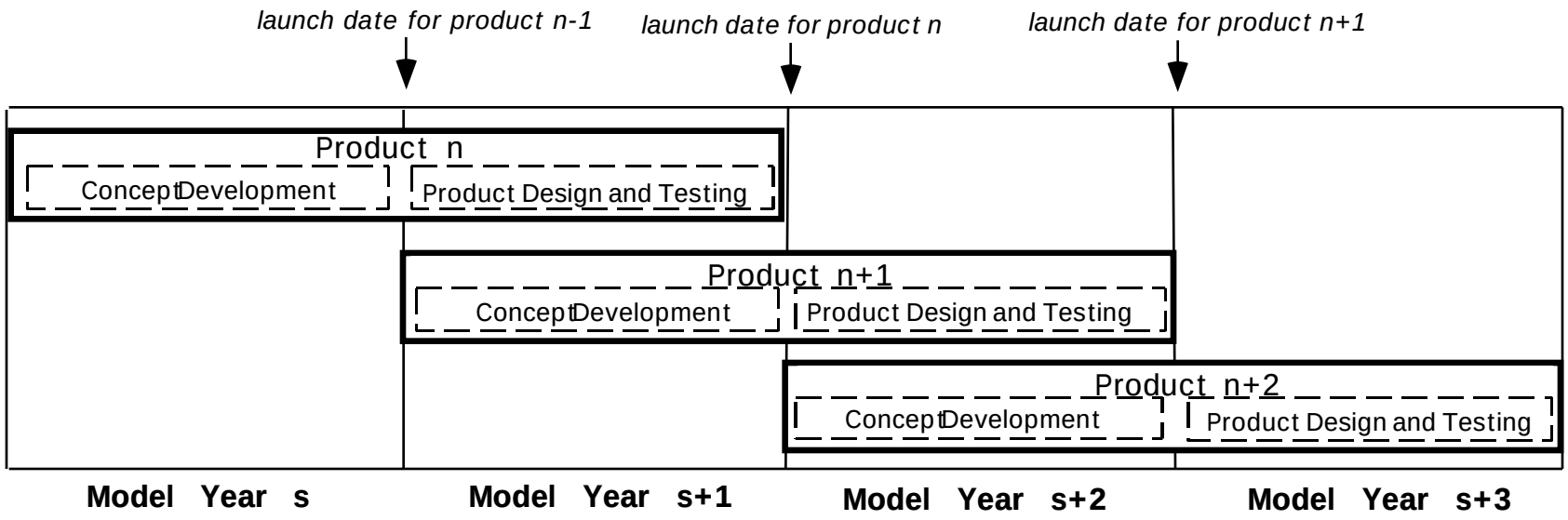
An Example: Fire Fighting

- The allocation of resources to unanticipated problems late in a product's development cycle, or ***fire fighting***, is a widespread phenomenon that can significantly degrade performance.
  - ***"…the completion date [the date at which the project is ready to launch] is getting later and later each year. We are starving the ensuing model years to make the one we are on. We never have time to do a model year right, so we have lots of rework and so on."***

    --PD engineer at Mighty Motors
  - ***...if you look at our resource allocation on traditional projects, we always start late and don't put people on the projects soon enough...then we load as many people on as it takes...the resource allocation peaks when we launch the project.***

    -- manager at Alpha Automotive
- If everybody understands fire fighting is bad, why does it persist?

A Model

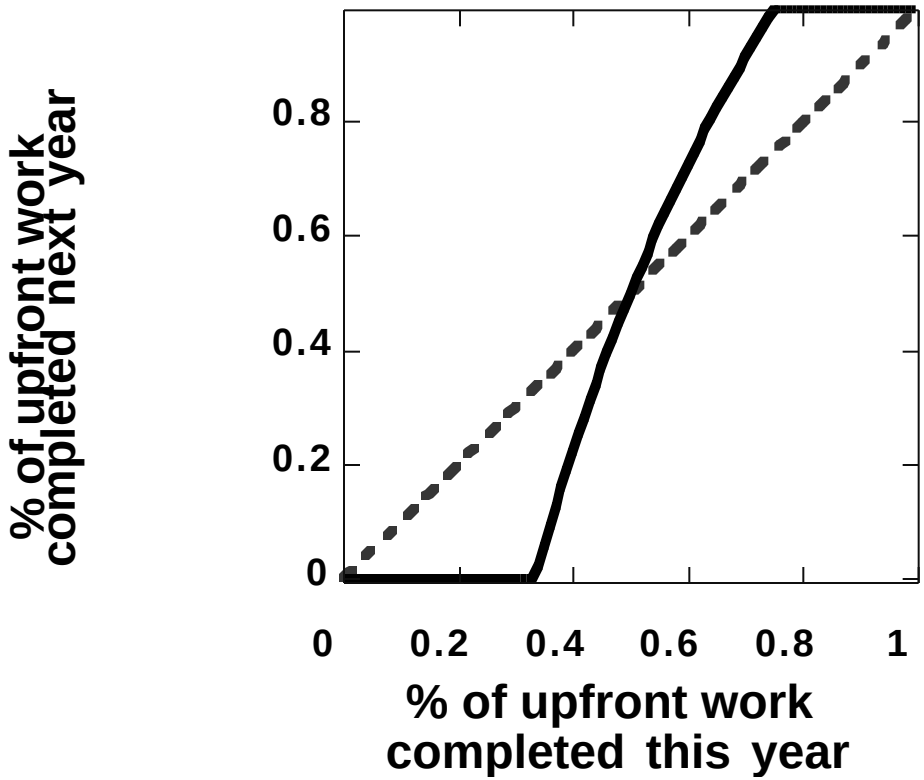


- Two kinds of development work
  1. **concept development work -** take place two years prior to launch
  2. **detailed design work-** takes place one year prior to launch
- Launch date is fixed (this is relaxed in subsequent work).

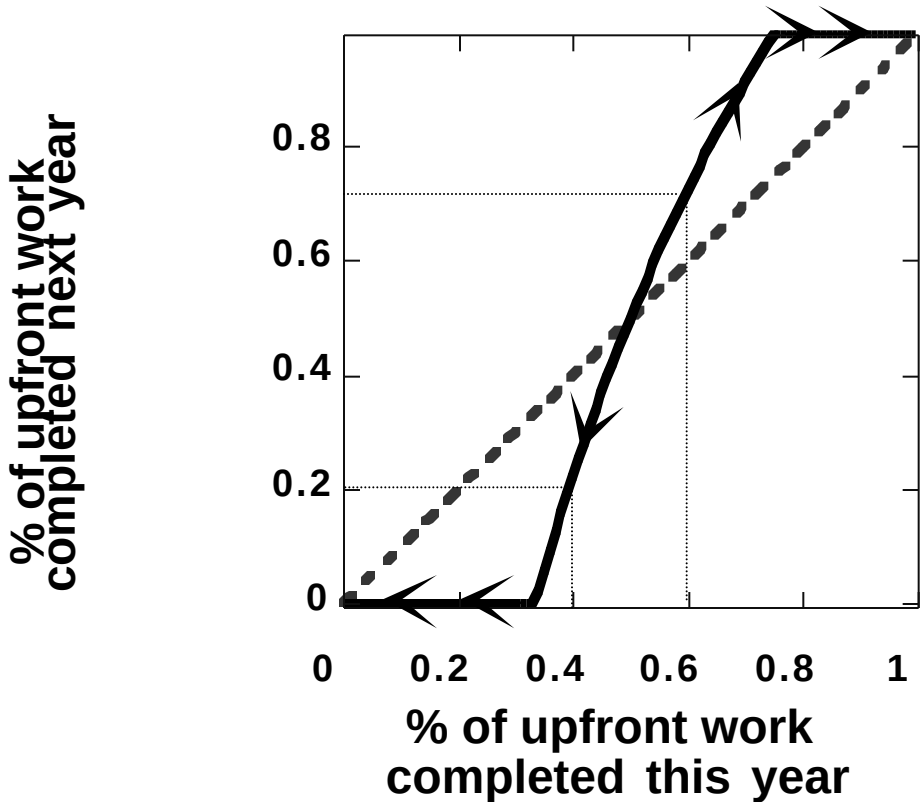
Biases towards downstream work

- People overweight salient and tangible features of their environment
  - **Projects in the design phase are more salient and tangible than projects in the concept development phase**
- People are risk averse and ‘ambiguity’ averse
  - **investments in design activities have a more certain outcome than investments in concept development**
- People are biased toward activities that produce immediate returns
  - **The benefits of concept development are realized with a longer delay than the benefits of design work**

We can summarize the dynamics of the system with a phase plot.



We can summarize the dynamics of the system with a phase plot.

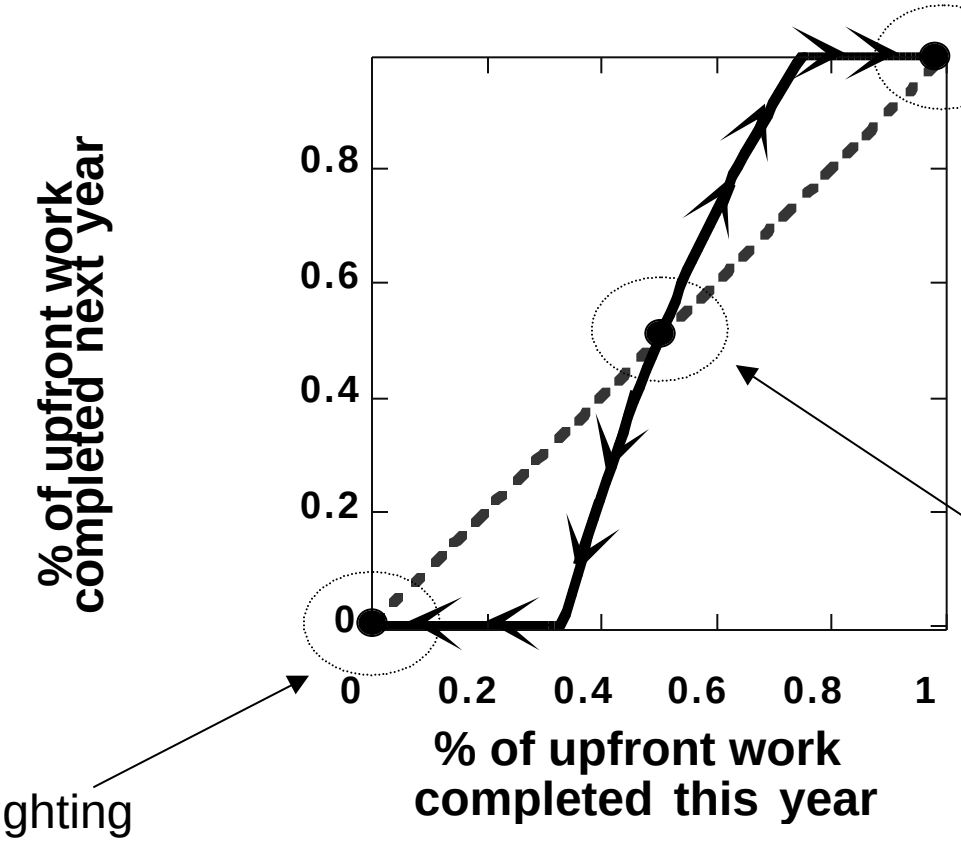


- So, if we accomplish 60% this year
- We will accomplish about 75% next year
- creating a *virtuous* cycle of increased attention to up-front work

- But, if we accomplish only 40% this year
- then we only accomplish about 20% next year
- creating a *vicious* cycle of decreased attention to up-front work

The plot tells us how much concept development work we will accomplish next year, given what we accomplished this year.

Yielding a system with two execution modes (equilibria) separated by a tipping point

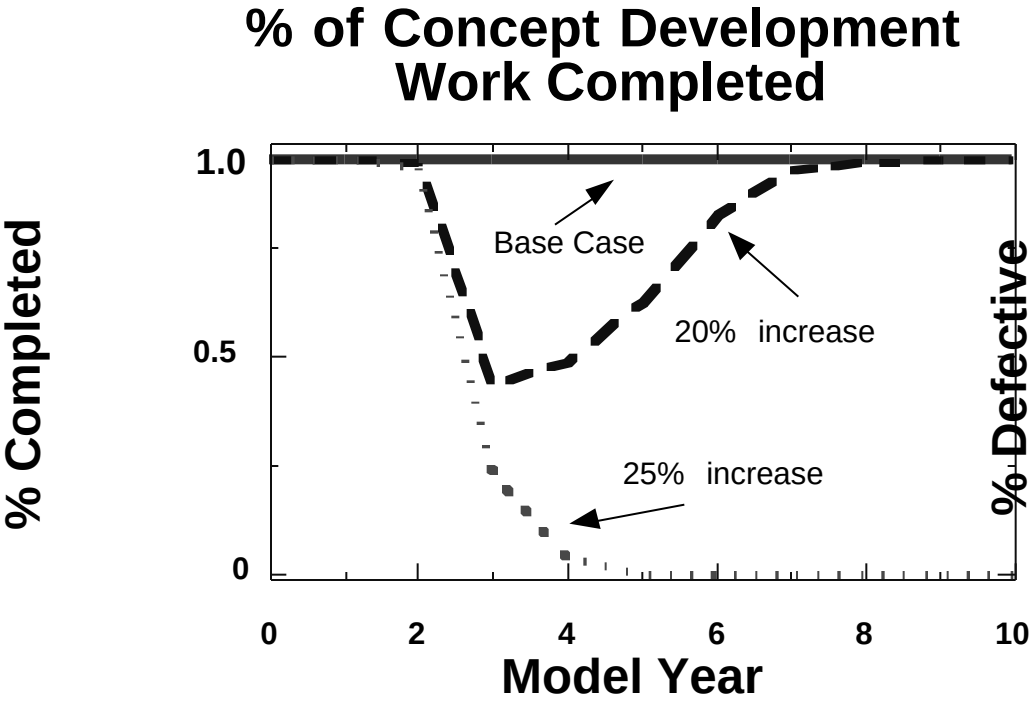


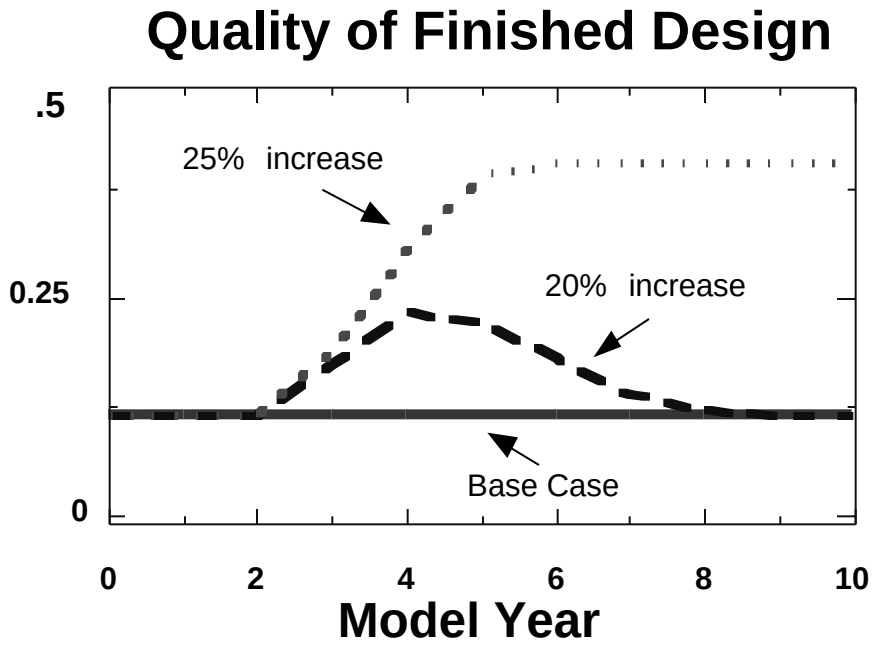
Desired Execution Mode:
100% of up-front tasks
are completed

Tipping Point:
Determines shift from
virtuous to vicious

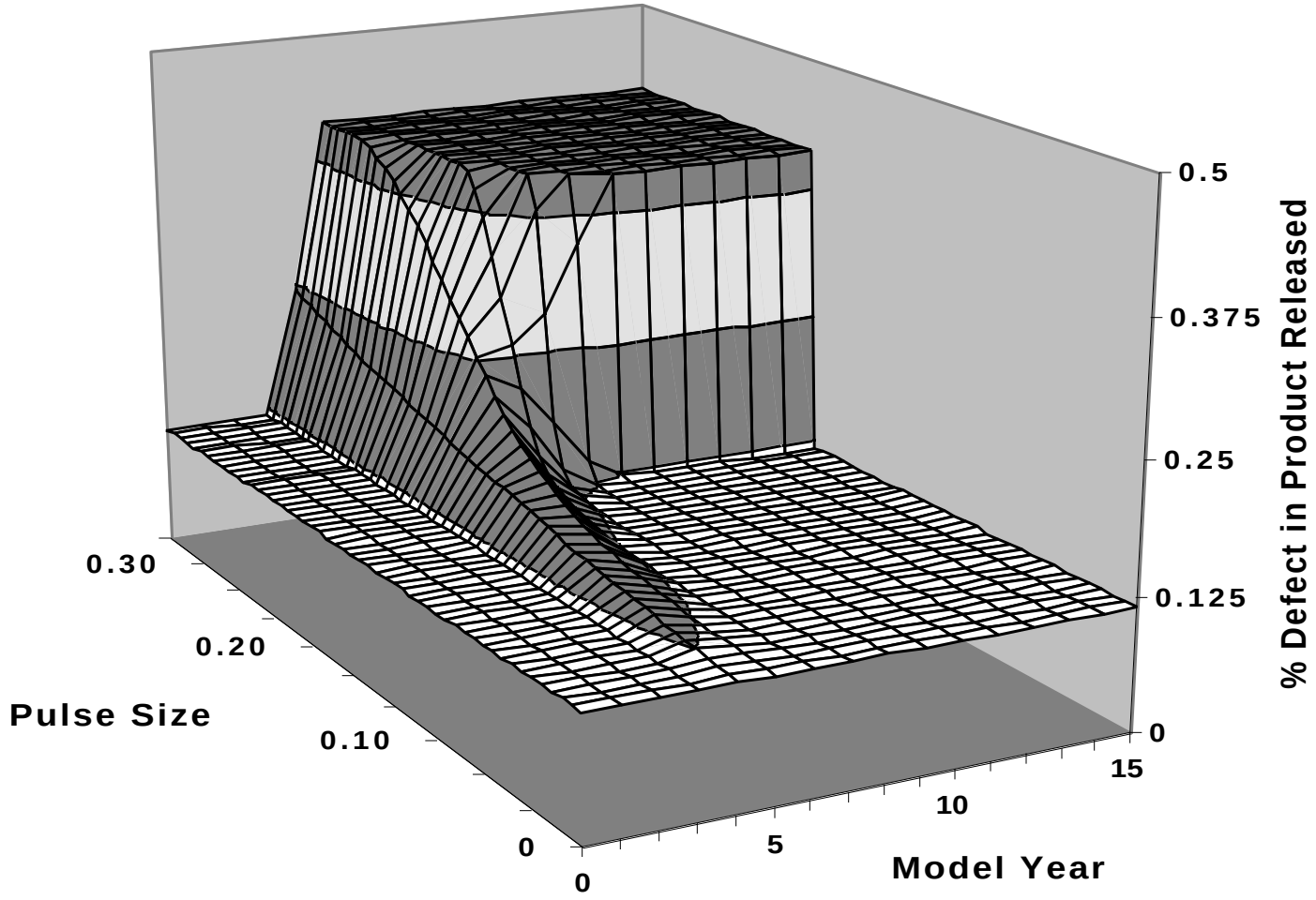
'Fire Fighting
Execution Mode:
0% of up-front tasks
are completed

Response to Unplanned Increases in Workload





Response to Unplanned Increases in Workload



Implications

- A **transient** increase in workload can cause a **permanent** decline in process capability
- Thus, managers face a trade-off between steady state performance and robustness to self-reinforcing fire-fighting dynamics
- Aggregate resource planning is very important, but very hard to do
  - **Plan must be updated for each unplanned increase in resource allocation**

Why does this phenomenon persist?

- Overloading appears to still be a persistent problem in industry
- Incentives remain focused on downstream activities
  - ***"...around here the only thing they shoot you for is missing product launch, everything else is negotiable."***
    --engineer at Alpha Automotive
- Wouldn't people learn to manage the system better over time?

Learning in Dynamic Environments

- When assessing causal relationships people are influenced by the following:
  - **Cues to causality (Einhorn and Hogarth 1986)**
    - Temporal order
    - Covariation
    - Contiguity in time and space
  - **Misperceptions of Feedback (Sterman 1994):** People typically:
    - Have an open-loop view of causality
    - Ignore feedback
    - Fail to acknowledge delays between actions and outcomes
    - Do not account for non-linearity

So, what do managers and engineers learn in this system?

- Focusing attention on the product nearest its launch causes people to spend more time on product design and less time on concept development
- This has two effects
  1. **Initially, the performance of the specific *project* improves...**
     - this happens immediately, it is easy to observe, and the impact is very certain
  2. **but, over time, performance of the *system* declines**
     - this happens only with a delay, it is hard to observe, and the impact is very uncertain

> **The system 'teaches' managers that downstream work is more valuable (even if its not!)**

Norms and Incentives co-evolve with the state of the system

- ***Occasionally there is a superstar of an engineer or a manager that can take one of these late changes and run through the gauntlet of all the possible ways that it could screw up and make it a success. And then we make a hero out of that person. And everybody else who wants to be a hero says "Oh, that is what is valued around here." It is not valued to do the routine work months in advance and do the testing and eliminate all the problems before they become problems. What is valued is being able to make a change in the last minute and ramrod it through.***

--senior managers at Mighty Motors

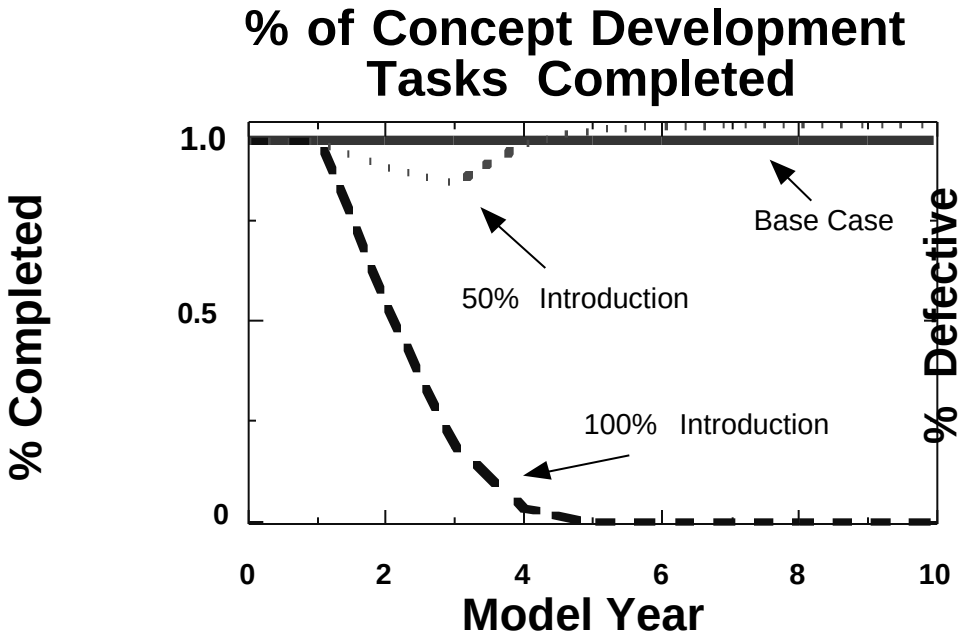
But, the situation is worse...

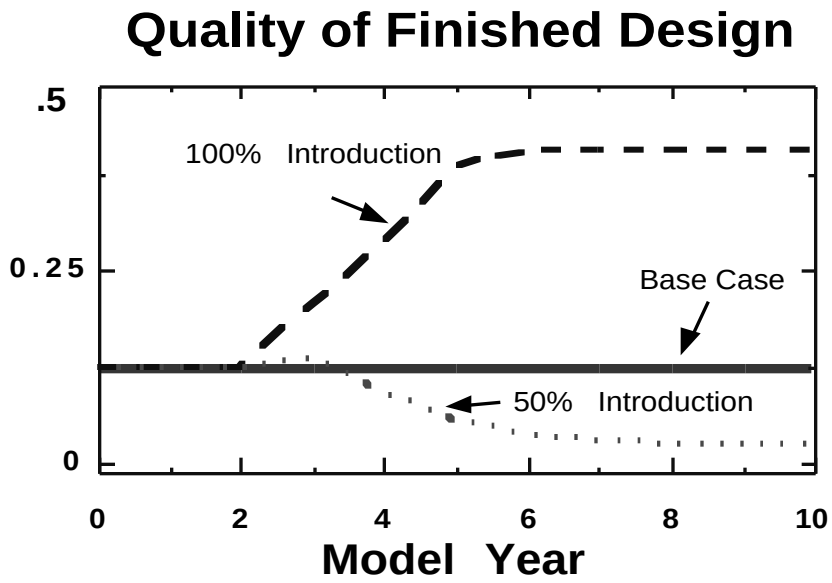
- As performance declines, managers must make some attribution of cause
  - The psychology literature suggests a bias towards blaming people rather than systems for low performance
- If managers (erroneously) attribute low performance to the attitudes of their staff, what do they do?
  - Increase workload directly by increasing number of projects
  - Increase workload indirectly by increasing frequency and granularity of measurement and reporting schemes.
- All of these actions increase the demands on engineers time, thus making the problem worse rather than better.

Example:
Interviews from Alpha Automotive

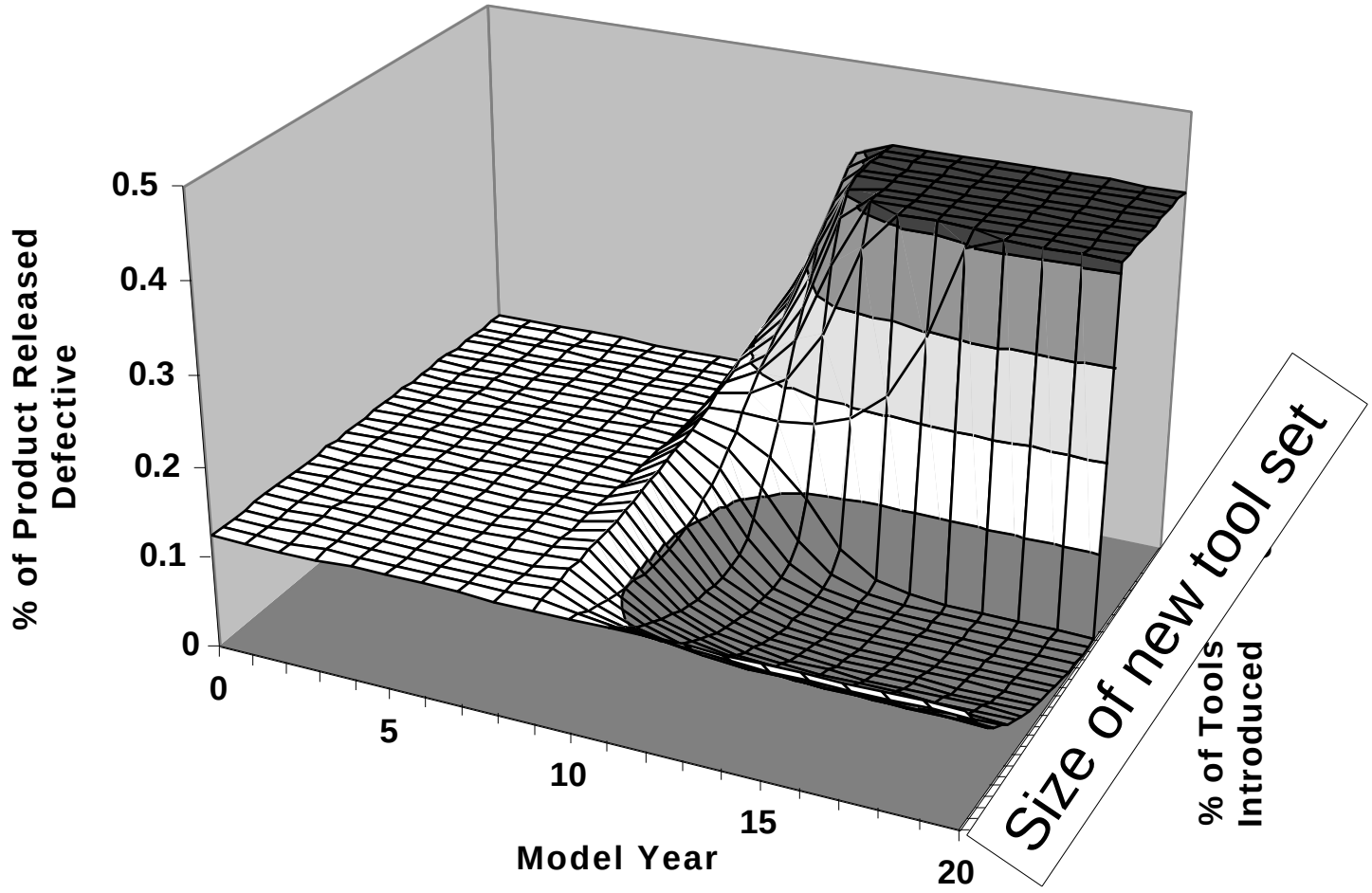
- **What managers said...**
  - ***engineers by trade, definition, and training, want to forever tweak things...It's a wild west culture.***
  - ***A lot of the engineers felt that it was no value add and that they should have spent all their time doing engineering and not filling out project worksheets. It's brushed off as bureaucratic.***
- **What engineers said...**
  - ***People had to do their normal work as well as keep track of the work plan. There just weren't enough hours in they day, and the work wasn't going to wait.***
  - ***...under this system...the new workload was all increase.... In some cases your workload could have doubled.***
  - ***How do we catch up? We stayed late. Most of the team was working from 7:00 a.m. to 8:00 p.m. and on weekends. A lot of people worked right through the Christmas vacation.***

What happens when you introduce new tools and processes?





What happens when you introduce new tools and processes?



Introducing too many tools at once can make performance worse rather than better

Implications

- **Aggregate resource planning is key to maintaining the integrity of the development process**
- **Cancel projects early and often.**
  - Do not assume that problems identified during concept development will be “worked out.”
- **Plan for “worse before better” when implementing new tools and processes**
- **Reward project managers for following the chosen process, not for “saving” troubled projects**