Getting Stuff Done

Stan Graves

NESC Webcast

April 14, 2021

QuantiTech

A Few Sources of My Studies in "Lean"

Toyota Production System (TPS). Alcoa Business System. Work of guru Steven Spear

- Doesn't translate into engineering, procurement, finance, and other business processes
- Terms like TAKT time, Pull, Kaizen, Judoka, Heijunka, Standard Work, Make-to-Use, etc. actually turn administrative people off. They don't want to learn a new language

Goodrich Aerospace almost went out of business in the mid-1990's. Their highest revenue source was the vending machines in the employee break rooms

- They reinvented themselves through a series of initiatives: *Lean in the Factory, Lean in the Office, and Lean Product Development*. Completely different company in 4 years.
- I've come to believe that a company has to be going bankrupt before it can undertake such sweeping transformations. It's easier to start over in the building across the street..

Lean Product Development: Kelly Johnson's Skunk Works

Adaptive Project Management: Siemens Software Development

Rapid Learning Cycles: more on that later

Scrum: I'll quote from Jeff Sutherland's work a lot in the following pages.

- I like Scrum because it applies directly to Project Management by using many of the same principles as the TPS, but without all the Japanese words….

"Cadence is a rhythm that helps us transform unpredictable events into predictable delivery of useable products."

I'll show several examples of how cadence helps us *get stuff done*.

Happiness Equals Success = Getting Things Done*

Happiness = success

- Happiness metric is predictive. *Happy employees create more value*.

What makes people happy? **Easy!!** ***Getting things done.***

Getting *things done the right way* has the following attributes:

1. **Autonomy**: ability to control your own destiny.
2. **Mastery**: feeling that you are getting better at something.
3. **Purpose**: feeling that you are serving something bigger than yourself.

Attributes of the greatest teams you know: Sports, Military, Innovative Products

1. Transcendence. A purpose that is greater than the individual.
2. Autonomy. Freedom to make decision on how to take action and be respected as masters of their craft. Having the ability to improvise.
3. Cross functional. The team has all the skills needed to achieve the mission.
4. Process Focused. The team asks, "What can we change about how we work?" What is our biggest sticking point?" How can we improve?

*"*SCRUM. The Art of Doing Twice the Work in Half the Time*". Jeff Sutherland and J. J. Sutherland. 2014.

Wrapped Up in Our Own Kudzu

I went to MSFC several years ago to talk to leadership about rapid product development and agile project management ideas.

If you are from the south, you will understand the following analogy.

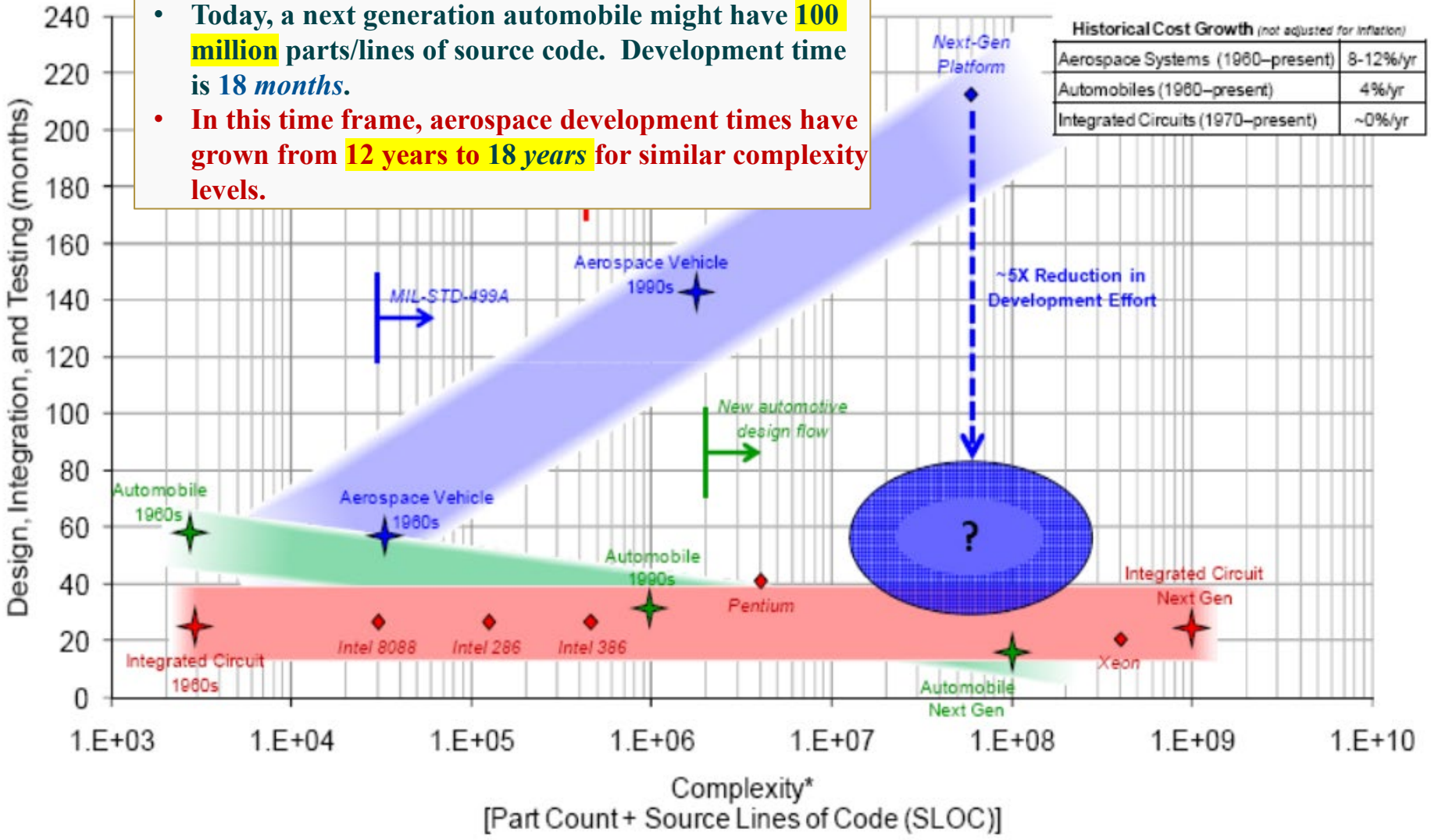
My friend Chris Singer, then Science and Engineering director, said "***We are so wrapped up in our own Kudzu, we can't get anything done***!"

Our complex business systems, rules, and processes create backlogs, queues, work stoppages, and lack of individual accountability. "I'm waiting".

Kudzu vines grow over healthy trees and kills them by blocking sunlight. *Eradication often requires killing the vine and the tree, and starting over.*

Historical schedule trends with complexity

- In the 1990s, a car had a million part/lines of source code. Development time for a new car was 3 years.
- Today, a next generation automobile might have 100 million parts/lines of source code. Development time is 18 *months*.
- In this time frame, aerospace development times have grown from 12 years to 18 *years* for similar complexity levels.



Multitasking Makes You Stupid*

Humans can only keep *4 things* in short term memory at one time

Do one thing at a time. Finish it.

- 1 project = 100% of time available = 0% loss to context switching
- 3 projects = 20% of time available per project = 40% waste to context switching
- 5 projects = 5% of time available per project = 75% loss to context switching

Half done isn't done at all.

Do it right the first time. And fix it now.

- **Fixing it later can take 20 times longer.**

Working too hard makes more work. It creates errors and more work.

- Take vacations. Rest. Get more done.
- Too much work makes people stupid.

No heroics. Heroic effort should be viewed as a failure of planning or of the process.

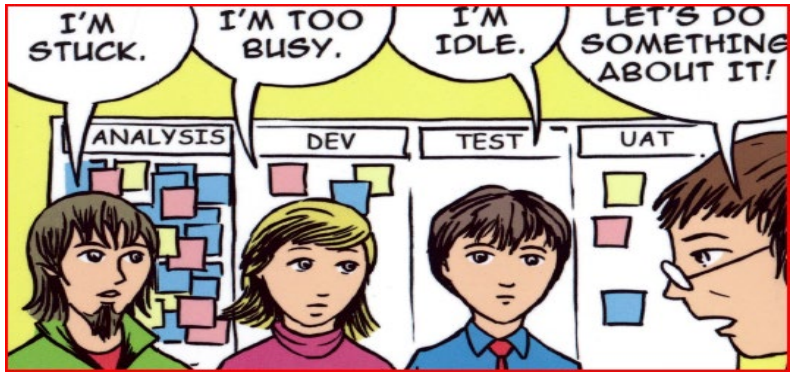
Stupid policies create stupid forms and stupid meetings and stupid approvals… If your work seems like a Dilbert cartoon, fix it.

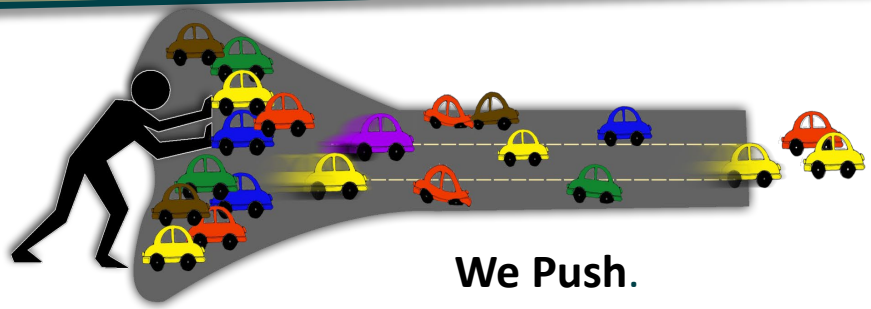
*"*SCRUM. The Art of Doing Twice the Work in Half the Time*".
Jeff Sutherland and J. J. Sutherland. 2014.

We need to create flow in our work processes

We multitask

Capacity Utilization Varies





We Push.

Tasks are stopped – those who could help us get our tasks going are busy doing their own tasks.

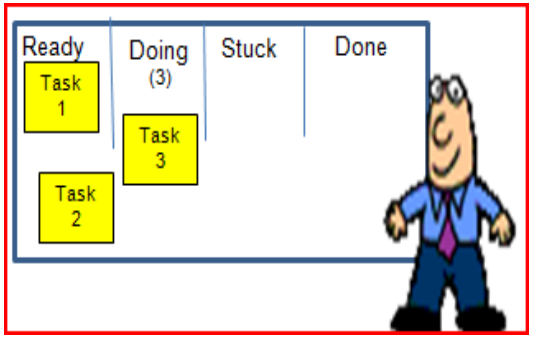
Ideal: Tasks are visual including stops. The need for help is visible.

Visual Work Boards

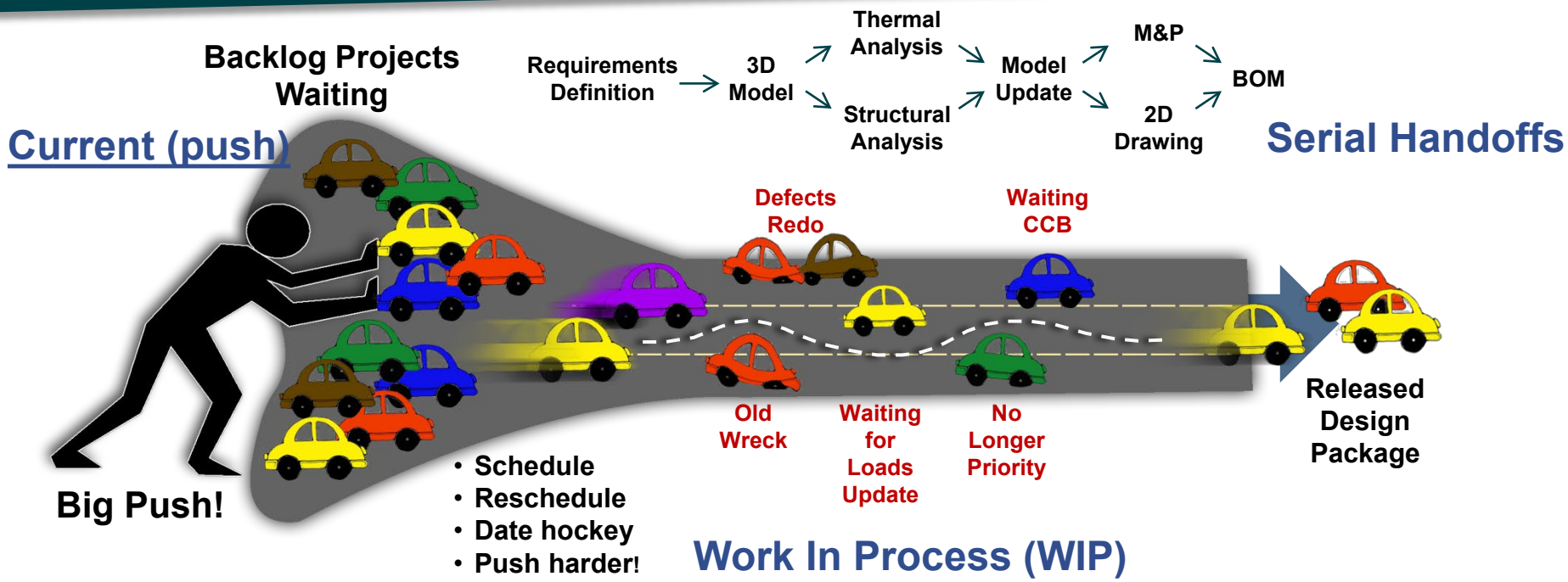
Visual Work Boards at the Team Level

- **Daily standups. 10 minutes.**
  - Roadblocks? *Where are we stuck?*
  - Who is working to get the task moving?
- **One month of tasks placed in Backlog queue.**
- **"To Do": 2 weeks worth of tasks locked in until completed** (one week or one month for other teams)
- **Project leadership determines backlog priority. What's next.**
- **Project management is "agile" because**
  - Most important work is always next up.
  - Unimportant work is eliminated from the task list

Similar visual work boards at the individual designer or analyst desk. Supervisor helps problem solve and get tasks moving.



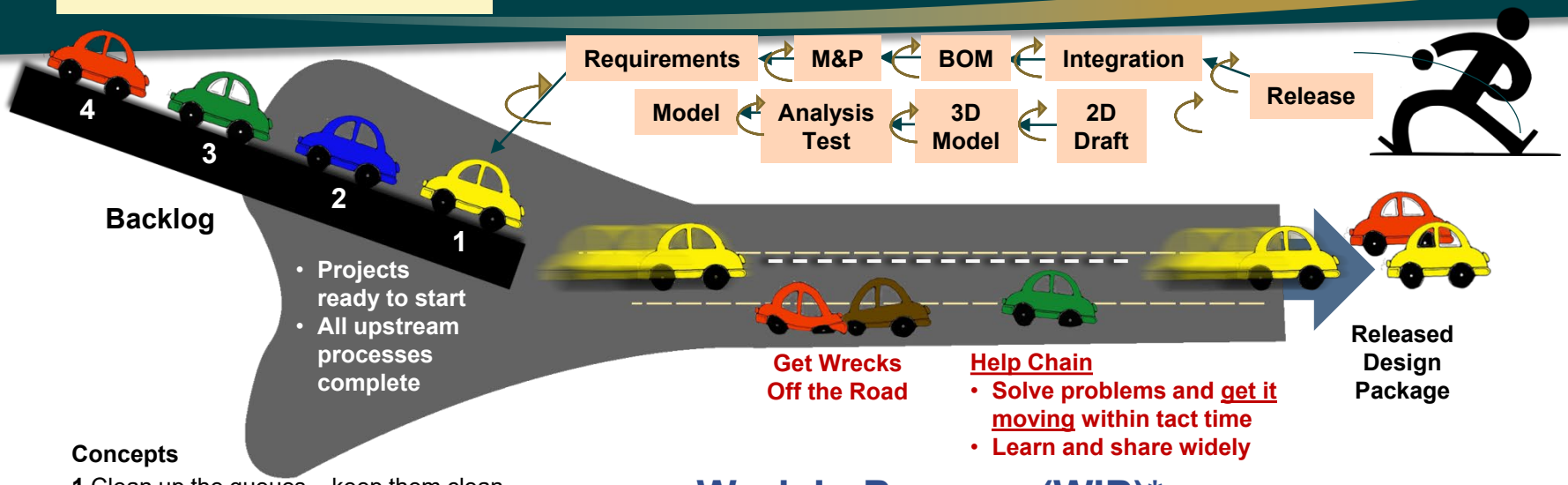
Traditional Design Process



Program Priority
FIFO Slide

New Design Process

New: Pull Flow



Concepts

1 Clean up the queues – keep them clean
- Get rid of obsolete junk

2 Pay attention to projects that are stopped
- Why?..."It's not due yet!"
- Because, paying attention to things that are stopped reveals the waste in the system and makes problems visible

3 Shorter cycle times eliminates waste, reduces cost, increases quality, reduces risk, and increases motivation

4 Pull allows engineers to spend their time on creative tasks and less time reporting status

5 Allows for rapid learning and rapid product development cycles

Work In Process (WIP)*

*Pull enabled by limiting WIP

A day in the Life

The design creation and release process takes about 20 steps. Tribal knowledge: nothing comes out of drafting. It's a black hole. Or, it's held up waiting customer approval.

When we put together a visual map of the Work-in-Process, it revealed two bottlenecks. Configuration Management approval, and Data Management approval at document release.

Old Process

Configuration Management Specialist picks up design package to review. It stinks. Send an email to the project engineer. Set it aside. Pick up the next package. Ditto. She could only approve about half the packages. The rest went into limbo until crisis time. Very painful since 98% of the work is already done.

New Process:

40 project and integration engineers trained and certified in CMII. Packages sent for CM approval were correct. Reject rates fell to 1%

CM Review moved to the beginning of the project, before the design package was sent to drafting. Change description and justification.

Final review verifies that only the authorized changes are made. *Re: CMII processes*.

Old Process

Data Management Specialist review prior to document release assures (1) all approvals are in place, and (2) only the authorized changes are included in the change package.

Reality was that once the design change was initiated, the designer would make additional (needed) changes. Result: package rejected and sent back to start over (painful). Specialist inundated with phone calls from program managers to release the document.

New Process

We convinced the designer to release the document with a "technical lien" that would prevent the part's manufacture, and start a new change. Manufacturing and Quality could start their work, and the next change would be ready shortly.

Traditional Project Planning Doesn't Work*

Traditional project planning approaches (Gantt charts and the waterfall method) don't work.

- *This is especially true for teams involved in the creative work of crafting something new.*

Traditional approaches create vast number of documents, graphs, and charts to *achieve the illusion* of predictability and control.

- Tremendous effort goes into providing status, return-to-green plans, variance analysis…
- Teams are incentivized to make the plan look as if it is working. Basically, to lie to us.
- Attempts to control change leaves problems unaddressed and stifles innovation

Every project involves discovery of problems whose solutions *require bursts of inspiration*

Traditional methods lead to frustrated customers not getting what they want. Projects are delayed, over budget, and all to often abject failure.

And these approaches lead to frustrated employees because *80 % their precious time* creates *stuff*, but not real value. *80% of what they do every day is complete waste*.

*"*SCRUM. The Art of Doing Twice the Work in Half the Time*".
Jeff Sutherland and J. J. Sutherland. 2014.

Size Matters*

Brooks' Law: *"adding more people to a late development project makes it later."*

Teams of 7 – 9 people is ideal.

- *A team of 20 or more people will produce less value than a team of 5 people.*

Everyone on the Project team needs to know what everyone else is doing.

- Complete transparency: progress made, challenges, future tasks, team goals

Impact of group size on number of communication channels. Connections = n(n-1)/2

Size of Team	Number of Connections
5	10
6	15
8	28
10	45

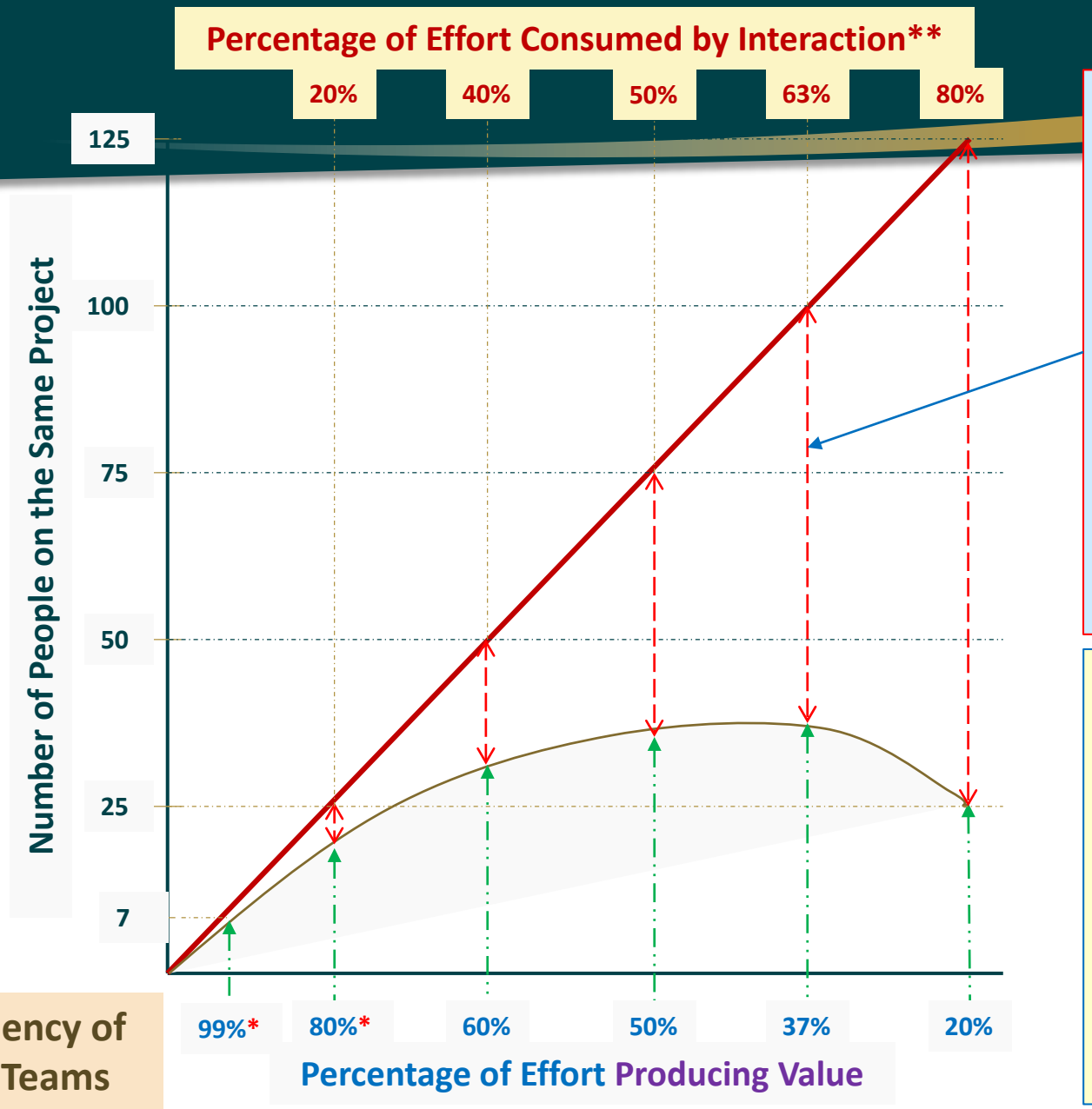
When the team gets too big, the ability of everyone to communicate with everyone else gets muddled

- Too many cross-currents
- Socially and functionally breaks into sub-teams that work at cross-purposes
- Cross-functionality gets lost
- Meetings take hours instead of minutes

*"*SCRUM. The Art of Doing Twice the Work in Half the Time*". Jeff Sutherland and J. J. Sutherland. 2014.

QuantiTech

Inefficiency of Large Teams



**Interaction
- Planning.
- Prep for the meetings
- Meetings. Statusing
- Talking about the work.
- Not making decisions
- Behind schedule
- Over budget
- Variance Analysis
- Return to green plans.
- Errors. Re-do. Late.
- Requirements creep
- Doesn't work
- Multitasking
- Managing Complexity

*Agile Project Management
- Producing something tangible every two weeks.
- Learn a little. Take another step forward.
- Focus on the process
- Identify process problems
- Learn. Adjust. Adapt.
- Solve Problems
- Decide

Rapid Learning Cycles – 30 day Sprints for Engineering

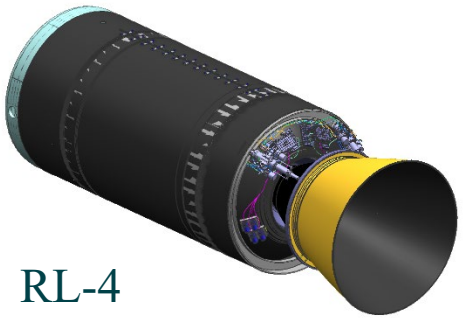
Process

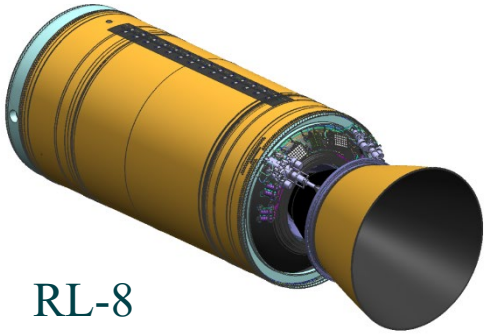
- Model released every 30 days
- A usable, testable product delivered on a set cadence
- One source of knowledge
- Everything known at that time is available to everyone

Results

- Makes creation of new knowledge predictable
- Forces schedule discipline
- Limits changes in design or added scope
- Self integrating. Don't need an army of engineers to integrate nozzle, case, igniter, propellant and TVC systems
- Manufacturing, Quality, Tooling have the most current data
- Customer also has latest model for vehicle integration.
- On Stratolauncher, early design had the wing attached in a different place than the customer
  - Recovery took 30 days instead of 6 months, and was found early


RL-1


RL-4


RL-8

Stratolauncher: Launch Vehicle Development

SLS story:

- I told the SLS Chief Engineer that with a Skunk Works approach and Rapid Learning Cycles, he could get the job done with 25 equivalent people.
- He couldn't fathom how that was remotely possible. *In fact, he believed he needed even more people*.

Stratolauncher story:

- 25 engineering EPs developing two new stages (TVC, case, propellant, nozzles)
  - About 15 full time people, plus pieces of another 60 specialists
- "Normal" NASA, Navy, or Air Force development program would have 200 EP.
- I asked the Chief Engineer, "*What would you do with another 175 full time EP*?" Stumped. He couldn't fathom what he would do with that many people.
- Engineering VP said, "*I know*. Keep them the hell away from the other 25 people."

Scrum: Basic Principles*

Scrum: Fewer people in less time can deliver more stuff with higher quality at lower cost.

- Scrum embraces uncertainty and creativity. It places structure around the learning process, enabling teams to assess both *what* they've created and *how* they created it.
- Scrum gives team tools to self-organize and rapidly improve both the speed and quality of work.

Basic principles

1. Prioritize the plan by identifying the top value added features. Do these first.
   - *80 percent of the value is in 20 percent of the features.*
2. *Every two weeks, create something of value that works*.
3. Teams discuss not what they did, but how. How can we work together better? What was getting in the way in the last one? What is slowing us down?
4. Inspect and adapt. Learn. Figure out ways to do it better.
5. Fail fast. Fix early. Working in short cycles allows early user feedback and the team can immediately eliminate what is obviously wasteful effort.

*"*SCRUM. The Art of Doing Twice the Work in Half the Time*". Jeff Sutherland and J. J. Sutherland. 2014.

Kelly Johnson's Skunk Works

XF-104 Mach 2 Fighter
1st flight – 5 Mar 1954

U2 1st flight – 1 August 1955

1st flight - 30 Apr 1962

SR-71 Prototype – 22 Months

F117 Prototype – 1st Flight Dec 1977

Kelly Johnson's Skunk Works 14 Rules

1. One Strong, Knowledgeable Leader
2. Minimal Program Office Size
3. The number of people having *any connection* with the project must be restricted in an *almost vicious manner*. Use a small number of good people (10% to 25% compared to the so-called normal systems).
4. A very simple drawing and drawing release system with great flexibility for making changes must be provided.
5. Record Important Work
6. Manage Your Program And Never Surprise The Customer
7. Manage Your Subcontractors
8. Know What Your Team Is Providing
9. Lead Your Test Program
10. Minimize Specifications
11. Fund The Program Right The First Time, for the life of the program, with discipline to absolutely meet budget
12. Establish Trust With Your Customer
13. *Access by outsiders to the project and its personnel must be strictly controlled*.
14. Reward Performance: by pay, not by the number of personnel supervised

Increasing Program Technical Complexity Has Resulted In a Cascade Of Effects Counter To Kelly's Skunk Works Vision

- *Larger Teams, And . . .*
- *More Time, Which Encourages . . .*
- *Specialization, Complicated By . . .*
- *Increasing Complexity Of Tools, Which Requires . . .*
- *Further Specialization, Aided By . . .*
- *Cubeland, Reinforced By . . .*
- *Email (A Powerful Conflict Avoidance Mechanism)*

All Of Which Encourage Dis-Integration

Adaptive Project Management: Siemens

- Work on the most important things first.
- Minimize Work in Process (WIP). Don't multi-task.
- Define the work in time-bound chunks. Typically 2 weeks. Maybe 30 days.
- When you start a scrum cycle, finish it without interruption.
- Done means that they have created a product that is usable to the customer.
- Done is done. 95% isn't done at all.
- ***Before they write one line of code*, they define what is to be accomplished in terms of meeting customer requirements, and *they define a test*.**
- I repeat. They define a test that will validate that the code works as intended -- before they start writing the code. **Hypothesis. It should do *this*. It should not do *this*…**
- **Build in quality. Never pass along a defect.**
- Review frequently what went well, what didn't, and use this input to improve the next cycle.

- **Perform frequent integrations across 50 scrum teams to assure that sum total of all the piece parts works as a system.**
- Don't over-subscribe the people. 6 or so hours of focused creative effort is planned.
- The process creates stability and predictability to our knowledge creation process. Previous: "*When will you be done?*" "*I don't know. I have to invent a solution. You have to let the creative process do its thing*." New: "In 14 days."
- All available resources are focused on creating the most value for their customers as quickly as possible.
  - It is quality focused. Defects cost 10 times as much to fix as to prevent.
  - It is schedule focused, but only in the sense that it focuses on delivery of the most important things first.
  - **It is adaptive. The top level plan is reviewed every three months. As the business changes or the customer priorities change, the task list changes.**

Receiving Inspection Story

Before

Multi-program complaint: Nothing is coming out of Receiving Inspection.

I spent 4 hours just watching one of the inspectors.

A day in her life:

- Open computer systems: MRP, Quality, Eng.
- Pick up an item to receive. About half done:
  - Stuck. Some part of the data pack is not compliant with the purchase order.
  - Call Quality Engineer. No answer. Leave a message. Call Procurement Specialist. No answer. Leave a message.
- Get phone call from irate program manager. Urgent priority needed on another item. Drop whatever you are working on.
- Sigh. Set package aside. Pick up next package.
- About half the time, the item could not be received.
- As I looked around the room, it was cluttered with half-received materials, unneeded tools, etc.

She went home every day feeling like a loser

After

QE and Buyer co-located

Hundreds of items in limbo cleared up. (Some were 3 years old)

Got rid of unneeded tools and gauges (5S)

Established a single person to work with various programs and manufacturing to establish a single priority list.

- 90 day forecast, 30 day plan
- 7 day deliverables
- Rolling cart with one day's deliverables for each inspector.
- Supervisor could add an urgent item to the cart (and subtract one)
- All the inspector had to do was pick the next item off the cart. No panics for her.
- And, if there was a problem, the co-located QE and Buyer would fix it immediately

All the inspector had to do was inspect.

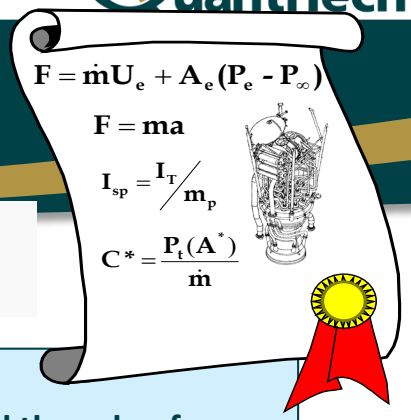
- No threatening phone calls

She went home every day feeling like a winner.

How to get Started

1. **Form dedicated product teams for all deliverable products**
   - People become invested in a workable deliverable product promoting engagement, innovation, and agility
   - Move away from "by the drink" to promote effective interaction and innovation
   - Combats multitasking "stall" by driving work to completion as quickly as possible
2. **Co-locate teams: promotes team/product loyalty**
   - Open collaborative spaces increase the frequency of interaction by design
   - Meetings are held impromptu, real-time, rather than delayed scheduling
   - Human face-to-face interaction facilitates the multiple aspects of communication
   - Humanistic factors are a motivation source ("We're in it together", "I'll do it for you")
   - Facilitates making information and status visible and physical (on walls, etc), increasing interaction with tools ("Planning" and "Standard Work")
3. **Promote servant-leaders**
   - Leveraging highly interactive, independently thinking teams means empowering them
   - Ensure they have the resources, information, skills, and tools to do their best
   - Set strategy and establishing planning, then work to review roadblocks and inhibitors

Scrum Process Results



Small (6) team of scientific programmers. They create software solutions to complex engineering equations in fluid dynamics, heat transfer, and structures

Before Scrum

- Engineers have an infinite appetite. Hundreds of requests in queue. Budget for 6 people.
- Each scientific programmer worked alone. They typically picked projects that were most urgent or most interesting.
  - Difficult projects went to the bottom of the list. Too hard -- takes too long.
  - Constant interruptions for urgent projects
  - Once a project was started, the engineer would ask for 'just one more feature'.
- The analysts went home feeling like losers because the backlog of requests just kept growing, and the urgent requests never went away.
- It was impossible for 'customers' to get a status on their project requests. When will you start? When will you get it done?

I don't know.

After Scrum

- Integration engineer performed the role of "Scrum Master". Worked with the engineers to develop a prioritized backlog. Deleted dozens of obsolete requests.
- Each month, a Scrum was started with well defined scope that could be completed in 30 days, and with individual deliveries due to the team each Friday.
- The 6 analysts worked on difficult projects as a team, breaking the work into weekly chunks.

Results

- The number of projects completed doubled the first year, and doubled again the next.
- Working as a team drove individual accountability to deliver as promised, and eliminated scope creep.
- The analysts enjoyed working as a team
  - More creative solutions. Getting stuff done.
- Engineers: *When will you get my project done*?

"Its in queue for the May Scrum in 2 months."

Discipline: We've lost personal accountability

As our work rules and processes became more complex, and computer-related systems more cumbersome:

We developed a culture wherein it was *okay to have a good excuse*, rather than actually getting something done.

"I'm waiting."

"I need…"

"My boss gave me another assignment."

We *lost the discipline* to do whatever it takes to meet our commitments. Getting the job done didn't seem important.

"You are a big meanie."

Leaders have been taught to be kinder, listen with empathy, and not hold people accountable.

Our Chief Engineer for ASRM was a tough disciplinarian. Not always liked, especially by the slackers, but extremely well respected.

ASRM Nozzle Development Story

I was the lead engineer for development of the ASRM nozzle design

The Chief Engineer for the project had the following rules and process

- We will plan all the tasks that will take us to PDR, CDR, DCR, etc.
- **We will have a detailed plan for the work to be completed each week.**
- If we get this week's work done, all the scheduled milestones will fall in place.
- **Nobody goes home until their work is done that week**
- If you went to the Chief early with a problem, he would get you help or relief..
- If you simply didn't get the task done, but had a good excuse, **he would hand you your head.**
- If you did it twice, you were *off the team*.

Further Study

What is Scrum, and Why is it important?

"*SCRUM. The Art of Doing Twice the Work in Half the Time*." Jeff Sutherland and J. J. Sutherland. 2014.

How to implement and execute Scrum?

"*Essential Scrum. A Practical Guide to the Most Popular Agile Process*." Kenneth S. Rubin. 2012.

This isn't new

"*The New New Product Development Game. Stop running the relay race and take up rugby"*: Hirotaka Takeuchi and Ikujiro Nonaka. Harvard Business Review. January – February 1986.

Why limit work in process, and why you should not schedule more than 6 productive hours each day.

"*Six Myths of Product Development*." Stefan Thomke and Donald Reinertsen. Harvard Business Review. May 2012.

Final Thought

A disciplined application of the *lean* methodologies will create *stability and predictability to our knowledge creation process*.

Stability reduces chaos and creates a calm environment that results in best quality, superior safety, and highest employee satisfaction.