Notes for Improving Scrum clinic - Part 4 - Ver 40 (Work-in-Progress)

Joe Little — April 27, 2026
Webinar — **LeanAgileTraining.com**

© Joseph Little 2026

Intro



Contents - 4

- Intro - slide 2
- Team Too Small - slide 4
- Harder cases - slide 8
- Case: Not Software Development - slide 10
- Basic Problems - slide 14
- Case: The Team is “meh” - slide 23
- Case: Adding Complexity - slide 30
- Case: Scaling - slide 39
- Problem: Understand why? - slide 52
- Case: PO Sucks - slide 58
- Case : TBD - slide 64
- Questions - Discussion - slide 65



Team Too Small



Imagine you have 4 small teams

- One with 4 people
- One with 2 people
- One with 5 people
- One with 3 people
- Assume: within these 14 people, we can find 2 ScrumMasters, 2 POs.



What are problems

- Some or many people are not committed and focused
- Some are easily distracted
- Minimal redundancy of knowledge if a person leaves
- Minimal ability to focus on one product and build a LOT in that product
- Fewer people who can help well.



What are advantages

- Two teams of 7.
- Each person is 100% in one Team
- Good ratios within Team
- Less distraction
- More redundancy of knowledge if a person leaves
- Now: ability to add skilled people MORE and build a LOT in one product
- More people who can help a team member will.



Harder Cases



Many possible situations

- Here are a few:
  - Product is not software development
  - Multiple teams (scaling)
  - Heavy waterfall mindset
  - Crazy number of Distractions
  - Missing key members of Team
  - Many Others
-



Case: Not Software Development



Simple Version

- You can define the product
- You can imagine incremental pieces of the product
- There is a visible way to verify and validate each pieces as it is built
-



Hard - why?

- Lots of these kinds of products are being built with Scrum
- BUT: Commonly each team feels “isolated”: “no one else is building our specific product with Scrum”
  - Or, so it feels. Reality is probably different
  - Few people to ask for help (but you can find some)
-



First Job: Define the product

- You must define the product. It's ok if you get it wrong at first
- You must start to imagine incremental pieces of the product. ie, Stories that can be built in a Sprint (2 weeks)
- Imagine how to verify and validate that each piece is good enough, according to a DOD
- Experiment.
  - You probably will not get it right at first, but still you will get more transparency
  - Expect to revise as you learn



Basic Problems



Silos and skill sets

- Almost every Team starting scrum has this problem
- Scrum is asking people (and skills) to come out of the “silos” and join a team
- With software development, agile-scrum has lots of learning and ideas about this.
- With a non-SW product, you have to “re-learn” all of this specific to your product
- **Key**: We must build and verify/validate a piece (a story) inside the same sprint
- So, all the related skills must (eventually) be inside the Team.



Team can be helped by Chickens

- This is not discussed enough.
- Every team always gets help from “chickens" (part-timers outside the team)
- Many types:
  - A person
  - A groups
  - A Scrum Team
  - A vendor outside the company
  - Etc



What can chickens do?

- Almost anything:
  - Coach
  - Do work alone
  - Do work with our Team
  - Build a specific “deliverable”
  - Give us a wrapped “widget” that we can insert in our product (Ex: a battery)
  - Other…



So, chickens overcome silos

- The Team can use chickens to overcome the problems arising from the old silos.
- SM has to teach the Team and the organization all the little ways to make this “chickens” solution work
  - Case by case
  - Person by person
  - Soon: Many common patterns of using chickens



Knowledge sharing vs hoarding

- Share "everything"
  - OK, within some limits
- Helpful in so many ways



Collaboration

- A real art, and very useful.
- Many ways. Let's discuss.
- Do it more
- Learn to do it better



How will we decide

- Many methods
- Recommend:
  - Everyone talks (timebox)
  - Best expert in that domain decides
  - Pretty fast
  - Decisions will be wrong
  - Re-visit later
- Best way to learn fast. Other advantages.



Value in deciding quickly

- Almost never have complete information
- Value in trying something and learning by doing
- Most decisions are not that hard to reverse



Case: The Team is “meh”



Symptoms - 1

- The Team is fully remote
- The Team is quiet mostly
- The Daily Scrum lasts too long, no one cares
- Lots of small problems, no one seems to care
- No real attempts to get better
- Do not seem to talk to each other enough
- Not clear if they are motivated



Symptoms - 2

- Productivity is low compared to prior Team (pre-Covid)
- Engagement is low
- The Team over-commits on Sprint every time
- Managers feel “at a loss”, “don’t know what to do”



NOT (?)

- Lack of talent (at least not key)
- Dependencies on others
- Don’t want to be here
- Lazy
- Managers are terrible (The Office) or mean-ish (Dilbert)



Root cause(s)? - 1

- Many are posited by managers and agile coach:
  - Being remote (for them)
  - Do not yet think of themselves as a Team
  - No sense of improving
  - Not “safe”
  - Introverted, and have not gotten to know each other well — hence, low “talking”
  - Big cultural change for these people (on average)



Root cause(s)? - 2

- Naturally work in isolation
- Think they are supposed to work in isolation
- Over-promising for Sprint, so no “wins” yet
- True at task level
- Insufficient explanation of the why we do the Scrum things
- Very different understandings of what Scrum is
- When they hear that some things can’t be fixed, they feel nothing can be fixed. So, hermit mode.



Root cause(s)? - 3

- Too much interrupt work
- Something outside the Team
- Team is following the “tacit” company culture (from whom? how does this happen?)
- Some big impediment that they are not talking about



Case: Adding Complexity



KISS

- Always good advice: Keep It Stupid Simple
- One Team with our kind of work? Complex enough.
  - Goal: Optimize the one full Team
- But: We feel pulled or we feel life forces us to deal with more complex situations. Often. (Make it a LOT less often.)
- “Things should be as simple as possible, but not simpler.” Einstein



Adding More Teams

- A lot more to manage
- We start to have a prioritized project portfolio
- Each Team gets 6 months of work at a time
- Each Team / PO gets to decide: Release 2 is Prod A or Prod B



Key Idea: First Things First

- True at prioritized project portfolio level
- True at product backlog level
- True at task level

- Problem 1: Which is the most important?
- Problem 2: Priorities could change



A Team gets “interrupt work”

- Someone asks the Team to add a PBI or story to the Sprint Backlog
- Why?
- Very disruptive, demoralizing
- So, all work must go to PO first: Mostly “no”, but some “even exchange”



Key: Work one thing at a time

- Minimize distraction. Increase focus
- Improve morale
- Focus on stoppages of flow. Fix impediments now.
- Helps deliver the most important thing first



Scaling

- Scaling has many definitions, almost each person a different one…
- Let’s focus on two:
  - Organizing a larger group with real Scrum Teams and others
  - Having 3 Scrum Teams work closely together



Larger Group

- 54 People
  - 1 Boss
  - 6 Scrum Teams (x7) = 42
  - DBAs - 3
  - Archs - 3
  - Misc - 3
  - Managers - 2
- See picture (Paper app)



Must have “individual contributors”

- People not in Scrum Teams
- Add to Scrum Teams with needed skills



Case: Scaling



Scaling

- Scaling has many definitions, almost each person a different one…
- Now: 3 Teams working closely together on 1 Product
-



Scaling is hard

- Too many people
- Too complex
-



3 Teams Working Together

- It’s big, it’s a mess, we need it now.
- Ans: 3 Teams work together
- Key: Share “everything” with all 21 people
- But: That is impossible.
- So, this scaling is…
  - Well: Very hard
  - Often not a good idea
  - But might work, help



Alternatives

- Don’t with new Teams. Cruel to try.
- Alt1: Just One Good Team
- Alt2: Divide and conquer
- Alt3: Dream Team. Much simpler
- Alt 3b: “Best of” Team. Simpler, maybe best you can do
- Alt 4: Reg Team PLUS
  - Really good SM
  - Really good Manager



Don’t do this with beginning teams!

- Evil
- Stupid
- Mean!
-



Say NO!

- The first question is: should you say no?
  - Let’s discuss why and why not
- The next question: can you say no?
  - At least make an argument
  - And start collecting data about whether it works
-



Other, often better, solutions

- Get the “Dream Team”
- Get a “best-of” Team, and help them
- Other



3 Teams (scaled)

- How:
- 3 full Teams
  - 7 people each, good PO, good SM, good Team
  - Mastered basics of Scrum
  - Ready to try scaling



3 Teams - 2

- Scrum of Scrums
- Add Scrum of Scrums Master
  - Key role, needs experience, hard job
- Add Chief Product Owner
  - Key role, needs experience, hard job
- Manage chickens well
  - More, much harder
- Run meetings well!



If you must use 3 teams

- Don’t do all of SAFe
- Get 3 good, experienced teams that have “mastered” one-team Scrum
- Get Experience:
  - Get a SofSM
  - Get a CPO
- Help this cluster-mess.
- IT IS HARD!



Key Issues

- How to use Business Stakeholder (BSHs) effectively
  - CPO is on-point
- How to use “chickens” effectively
  - SofSM is on point
-



K.I.S.S.

- Keep It as Simple as you can!!
- Smart people always make things too complex.
- Smart people tend to believe that they are important and that success depends on them being a hero.
- Keep focus on the individual Team.



Problem: Understand Why



We must understand why

- Knowledge Workers cannot do a “process” because you say so
- They must understand why
- And they forget
- “Hi, I’m Joe. I’m a recovering waterfall.” Slowly waterfall ideas will start to creep in.
- “Culture eats strategy for breakfast.” That is, your culture is taking them back to waterfall.



How do we explain?

- We describe the “process” and show how the ideas, values, principles support and explain the “practices”.
- So:
  - Ideas
  - Values
  - Principles
- And it turns out there are many ways



The main sets

- Agile Manifesto
- Agile Principles (12)
- Scrum Values (5)
- 3 Pillars of Empiricism



What else?

- Common sense
- Many quotes
- Lean Thinking (lean ideas)
- Theory of Constraints
- XP Ideas (per Kent Beck + others)
- Other



Can you explain these well?

- Do you remember them all?
- Can you explain them well, in each context?
- Can each member of your team explain them well?
- Can all the important managers explain them well?
- How do you know them still understand “why” well? How would you know that had started to forget?



Case: PO Sucks



Most Product Owners…

- Typically good at their former job
- Typically have some good skills sets for this situation
- Typically do not start out knowing Scrum
- Typically are not allocated enough to this Team
- Typically do not understand the job they are getting into



Often the biggest impediment

- Is that the PO is not good enough yet
- The PO is big and important
  - A weak PO can really drag down the Team (although it might not be clear that the PO is the cause)
-



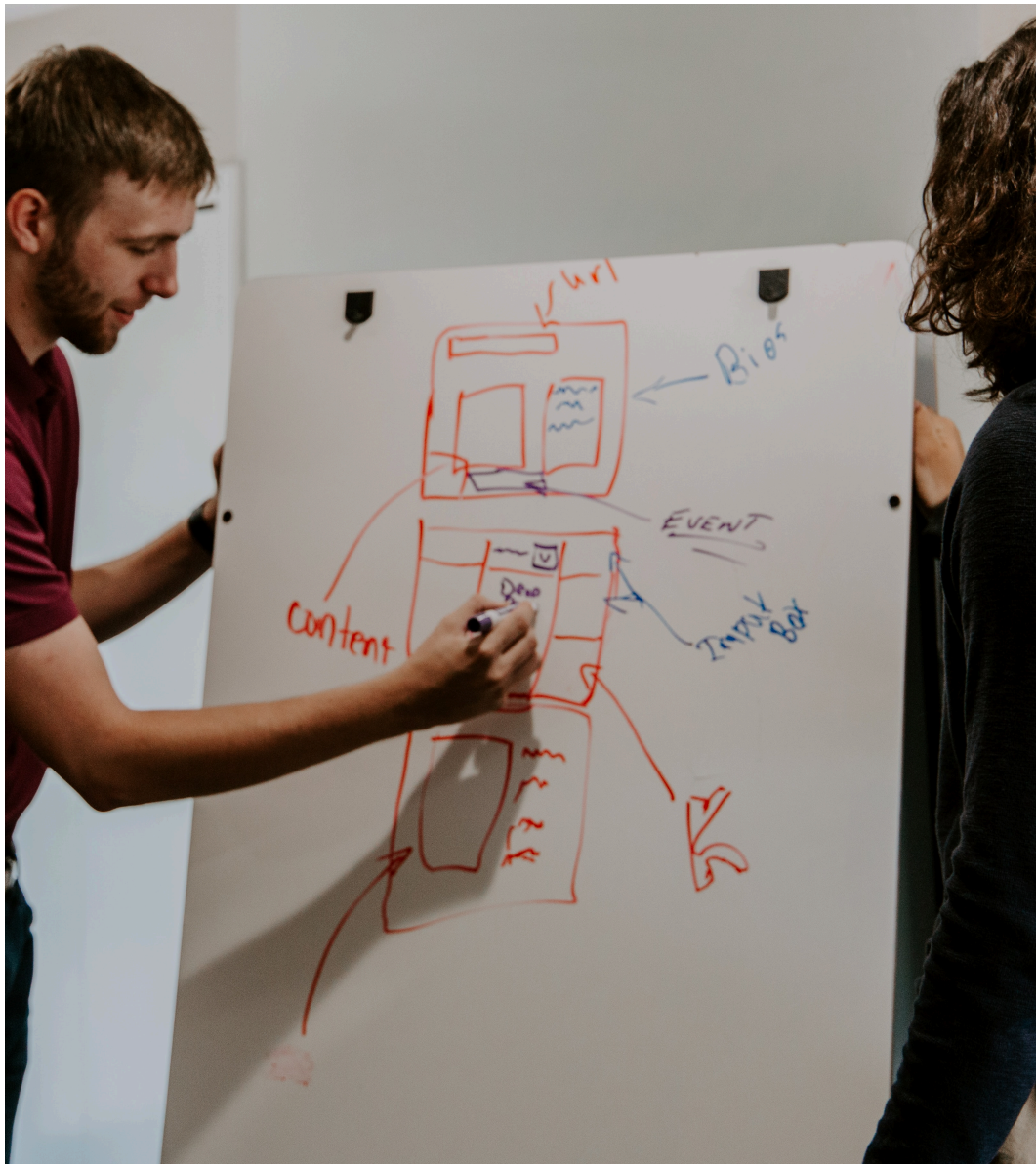
Common Action Items

- Get the PO more allocated
- Get the PO on only one Team (of 7)
- Get the PO trained (eg, CSPO)
- Get the PO coaching
- Find “Tom Brady”; have them talk. A lot.
-



Common Weak Areas - 1

- Lack of Knowledge
  - Make list of 18 top domains
  - Check depth in each one
- Not inspiring
- Not decisive (under uncertainty)
- No strong enough to say “no”
- Not a minimum viable product
-



Common Weak Areas - 1

- Can’t get vision accepted by “the culture”
- Does not understand DOR and DOR process
- No understanding of 80-20 Rule
- Inability to execute on 80-20 idea



Case: TBD



Discussion, Questions



Discussion, Questions

- Don’t be shy!
- Turn off the mute
- OR: Type them in the Chat



Other webinars

- Two Types:
  - About courses and workshops (30 mins)
  - About agile questions or issues (60 mins)

- Where:
  - LeanAgileTraining.com
  - MeetUp



More Info about Us

- Courses & Workshops: **leanagiletraining.com/lean-agile-and-scrum-courses**
- Joe Little
- jhlittle@leanagiletraining.com
- (704) 376-8881
- **leanagiletraining.com**

