1

BV Engineering is...

- the values, principles and practices that enable us to deliver more and more Business Value [from a given team] as we improve.
- a learning and incremental improvement approach to giving customers more of what they really want, looking at the whole process, end-to-end.
- a framework for getting better.

2

Attributions

- Some people who directly or indirectly contributed: Peter Drucker, Takeuchi & Nonaka, Jim York, Chris Matts, Kent McDonald, Womack & Jones, Jeff Sutherland, Kent Beck, Mary & Tom Poppendieck, Taiichi Ohno, Ken Schwaber, Dave Muldoon, some friends at "a large financial institution in Virginia", and many others.

Joe Little, CST & MBA

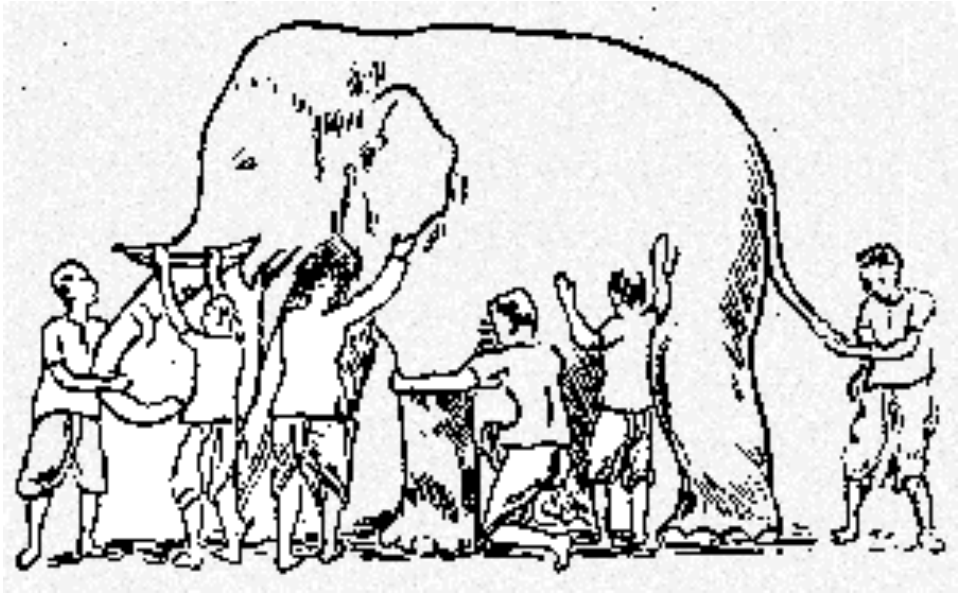
- Agile Coach & Trainer
- 20+ years in senior level consulting to well-known firms in New York, London and Charlotte
- Focus on delivery of Business Value
- CST, CSP, CSM
- Was Senior Manager in Big 6 consulting
- Head of Kitty Hawk Consulting, Inc. since 1991
- Head of LeanAgileTraining.com
- Started trying to do [Agile] before reading The Mythical Man-Month

  - http://agileconsortium.blogspot.com
  - jhlittle@kittyhawkconsulting.com

A Start

"You've got to be very careful if you don't know where you're going, because you might not get there." Yogi Berra

"Some people, if they don't already know it, you can't explain it to them." Yogi Berra



6 Blind men and an elephant

My main stance

"I'm mad as hell and I'm not going to take it anymore." (movie quote)

"The biggest thing to fix is how we do BV Engineering."

One quick analysis (to follow)

- This is a thought-experiment. For one thing, it proves to me the high importance of a decent Product Owner.
- This is an exercise you should do with your team. See what you learn.

What are the numbers for your team?

- Cost per year
- NPV delivered per year
- Derive: "The multiple"

Let's do the math...

- Assume team costs $1,000,000 per year
- Assume normal multiple is 3x (ie, delivers $3,000,000 in BV)
- Assume the "real work" itself does NOT get any faster

Could a better Product Owner make a difference?

- We make the stories 20% better
- We use Pareto's "85-33" rule to get more done in less time
- We identify more high value epics
- We motivate the team, so that they are more productive
- We assure that we actually hit the mark, rather than just say that we did

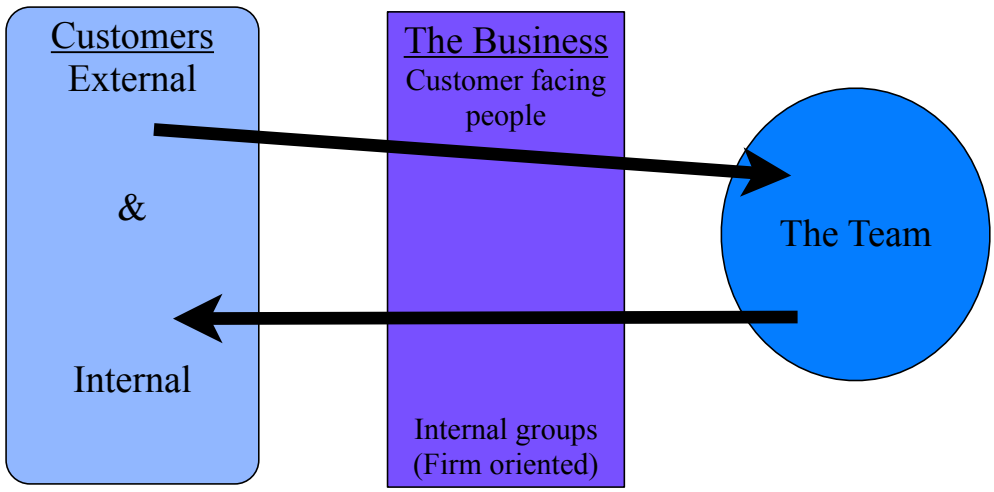
- What's that worth? 3X more BV?

One version....

	Year 1	Year 2	Year 3
Cost of Team	$1,000,000	$1,000,000	$1,000,000
Orig Value Delivered per Year	$3,000,000	$3,000,000	$3,000,000
NPV	$7,460,556		
ID Better Stories (+20%)	$3,600,000		
Deliver Top 33% (85% of BV)	$3,060,000		
Deliver Top 33% again	$3,060,000		
Deliver Top 33% again	$3,060,000		
TOTAL FIRST YEAR	$9,180,000	$9,180,000	$9,180,000
Better NPV	$22,829,301		
Better/Original	3.1		

What the Product Owner does

BV Engineering



What are we addressing? - 1

- What are the requirements?
- What does the customer want?
- Who are the customers?
- What is their problem?
- Do we have the right people involved?
- How do we get feedback that we understand?
- How do we decide which project to do?
- How often do we release?
- What's the MMFS?
- How do decide when to release?

What are we addressing - 2

- What is BV?
- By checking the BV Model against reality, we are proving whether the process or theories or the Model is correct.

What are we addressing? - 3

- How long should 'final testing' take?
- How do we sell and install the product?
- What other things do we add around the product?
- Did we hit the target?
- Did we make money?
- How do we support the product?
- What have we learned?
- What goes in Version 2?
- The road to the next thing?

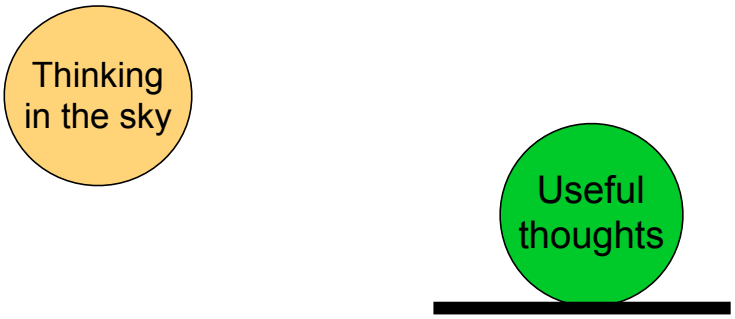
Some problems

- **We set up the telephone game**
- **Customers are not consistent**
- **The needs of the customers and of the firm are sometimes in contradiction (or at least somewhat antagonistic)**
- **It is difficult to accurately measure success**

Some more problems

- **La donne e mobile**
  - The customer is always changing his mind & who the customers are is always changing
- **Stuff is happening out there**
  - Everything in the environment, both for the customers and for us, is changing
- **Wow, this technology stuff is always changing**
  - A brilliant product today is yesterday's news tomorrow
- **"I know it when I see it"**
  - The customers can't tell you what they want
- **"What we've got here is a failure to communicate"**
  - It is impossible to accurately convey what you want

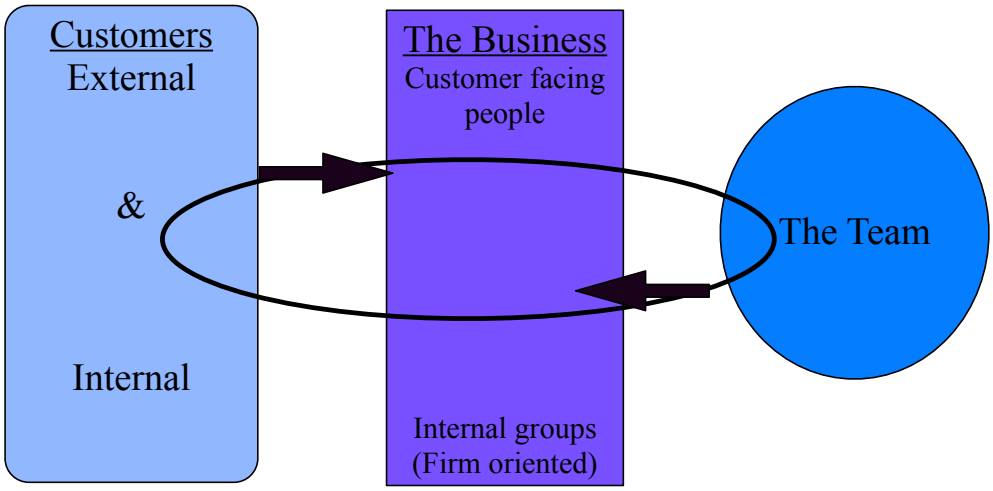
The other big problem



In theory there is no difference between theory & practice. In practice, there is. (Yogi Berra)

To know and not to do is not to know. (A martial arts master)

Is it better this way?



Some possible axioms

1. A "technical success" is no success at all
2. You win by learning faster than the next firm
3. You win with small "scientific" experiments; frequent and fast
4. The numbers never get precise, but that does not mean 'use no numbers'
5. Numbers can be useful, but that does not mean 'human judgment is no longer needed'
6. There is no one best approach to BV engineering

Start Exercise #1

Real Situation

- It must be a real situation….at least to the "Product Owner" of the group.
- All others help PO describe his/her situation.

Exercise 1:

- As a team, you will have 5 mins.
- Pick a PO and a specific situation (project).
- Define BV in words for all those who must use it. In the context of a specific situation.
- Define the basics of your BV Model. "If x and y and z, then over 3 years we will make $3 million from this software." x, y, z are more variables (assumptions) in an equation.

Conditions of Satisfaction

- Your BV definition must compelling to key participants
- The linkage from the product (to be built) and your BV (to be realized) must be reasonably obvious or explained.
- Hint: Most teams deliver against multiple definitions of BV (eg, reduced risk and higher NPV).

What is BV?

- Opinion: Defined by people
- Which means: There are many many opinions. Which means what?
- Involves customers, stakeholders of the firm, even workers
- No one right answer for all situations

What is Business Value?

- ROI/NPV
- Reduced risk
- Higher customer satisfaction
- Higher revenues
- Lower costs
- Proxy: More eyeballs
- Proxy: More usage
- Proxy: More units sold
- Other: Some lean, 6Sigma or other metric
- Special: Movement toward some org goal

Hints

- Decide the PO quickly.
- Get a real situation. Minimal abstract discussion; mostly concrete specific discussion.
- Argue some (that's where you learn).
- When you come to a fork in the road, take it. (ie, the PO has to decide when to "decide & move on")
- The SM is responsible for getting everyone involved, and some less involved
- The SM is responsible for "team mojo"

Debrief

- In a sentence, not repeating what someone else said, what was the (next) biggest thing you learned?
- Biggest = most useful ??

Two opposite approaches

- Proctor & Gamble: A 'traditional' but highly disciplined approach. "What does the customer want?" "How do we advertise our product?"
- Google: A new (or old?) approach. Let's try something, and see if they like it. If so, then we'll build on it.

Proctor & Gamble

- Full marketing program
- Focus groups, customer interviews, observation, customer segmentation (& lots of other tools)
- Financial (& other numeric) forecasts
- Multiple experiments, high rigor
- Advertising

Google

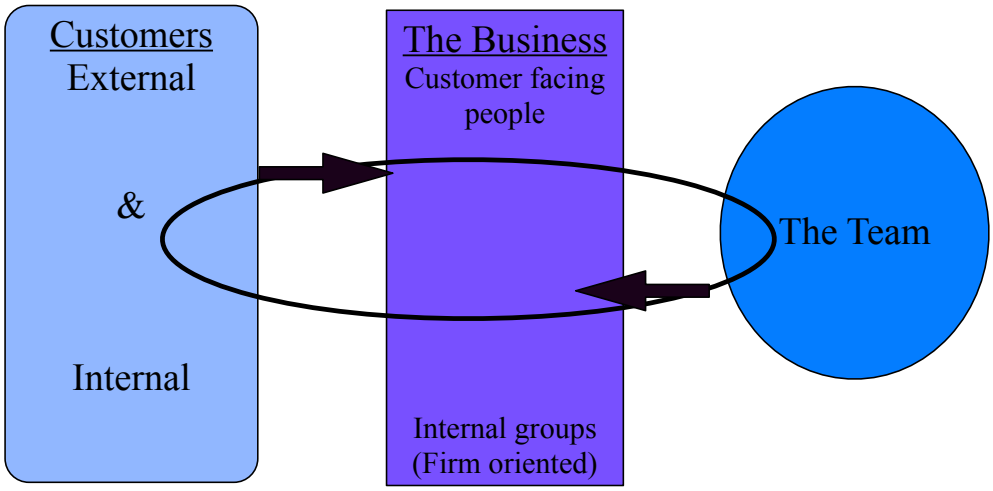
- Let employees create what they want
- Get a prototype out there "in the real world"
- See who bites
- Develop product incrementally based on customer input
- Monetize later (after we have a real product that a bunch of people really want)
- Get more "at bats"

Hallmarks of real BV Engineering!

1. The process is visible and articulated & improved
2. Failures in BV communication are identified and corrected frequently, quickly
3. There is a theory, and a concerted attempt to prove out the theory
4. There is appropriate dynamism and change
5. Business & Technology are partners
6. Success is forecast and also measured after the fact
7. Human judgment is involved (it's not just the numbers)

The BV process is visible and articulated

- Do you understand your's, end-to-end?



IMPORTANT

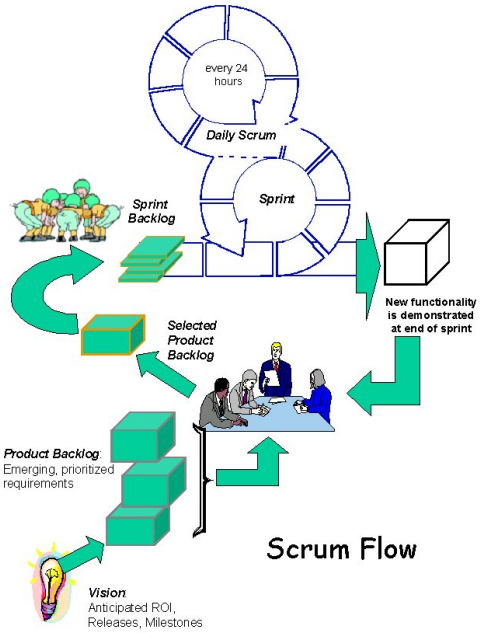
- End to end to end to end

The process is always being improved

- Is your process always being improved?
- Does everyone know that?
- What is the approach to improvement?

- Small example: Which stakeholders are involved? Do we have the right ones? Are we making the most use of them? Are we overweighted in compliance, legal, regulatory input? How good is our process of engaging them, and getting the most with the least effort? Are we creating knowledge just-in-time?

Where do you check for communication failures?



And are there other points or methods?

A theory, that is being proved out

- Is the theory stated as such, or is it assumed to be right?
- How it is being proved out?
- What happens when (not if) it is (somewhat) wrong?

Dynamism and change

- The appropriate amount of dynamism and change will vary by situation.
- In general, my experience is that we are adapting too slowly.

Business & Technology are partners

- In fact, there is minimal distinction. Anyone can help a partner learn. Can question. Can propose
  - Remember: There is no technical success
- The Technologists often know more about the customers than you'd think
- Should we talk about the failure modes here?
- Everyone on the Team understands what real success would be

Success is measured

- 1 to 3 key "end" metrics. Identified. Forecast.
- Then the real results are obtained.
  - Perhaps not perfectly, but reasonably
- And learned from. (Was the product wrong? Was the theory wrong?)
- And communicated back to the Team

Human judgment

- Yes, stuff often happens that makes one question whether the "scientific" experiment was fair
- Yes, one can still have a hunch that the product will succeed later (if not now)

- So, metrics do not absolve managers from tough human judgment about the actuals and other information they get back

The unbearable lightness of metrics

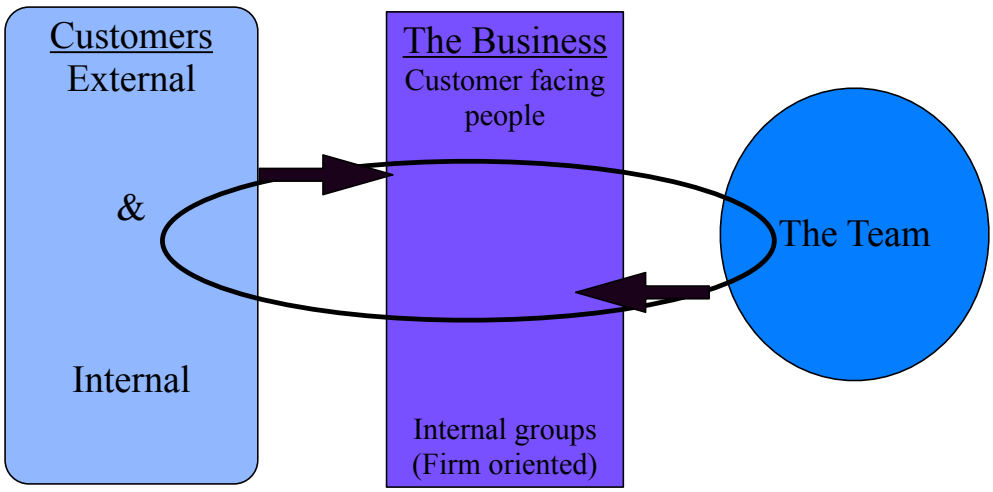
- We use metrics (about the past) to take forward-looking action
- Metrics help us see how bad we were at predicting the future
- Metrics help us learn (perhaps first, by helping us see how much we don't know)

Multiple Steps are important

- Some firms focus too much on one or two steps (eg, initial focus group, user story creation, the PO review of completed stories, the product launch)
- It is not one play; it is the culmination of plays that wins the game

- Examples: Understand the customer better and spend more time to assure that the Team understands the customer's problem better and better

It is not one play...



Elements of BV Engineering 1

1. PO Team
2. Product Backlog
3. PB prioritized by BV
4. Priority Poker
5. Story Points (proxy for cost, for cost-benefit analysis)
6. Minimum Marketable Feature Set
7. Reprioritize before each Sprint
8. Increase velocity (remove impediments)

Elements of BV Engineering 2

1. Making the stories smaller
2. Value Stream mapping
3. Kano Analysis
4. Voice of the Customer
5. Having the team live with the customers
6. Pareto chart (eg, of causes of customer problems)
7. Process charts or high level use cases
8. Other Lean, Six Sigma, or TQM tools

Elements of BV Engineering 3

1. Understanding the importance of minimizing technical debt
2. Agile portfolio management
3. What quality means to the customer and why it is 'free'
4. Just-in-time knowledge creation
5. Modifying the BV model frequently (& the values in the model)
6. Removing impediments
7. Comparing our BV Engineering to theirs
   - We're different; what does it mean?

Elements of BV Engineering 4

1. Identifying better sources for good user stories (eg, observation, "living with", experts, user interaction, "prototypes", etc)
2. Identifying good user stories
3. Fleshing out good user stories with an Agile specification
4. Improving the monetization of User Stories (or themes)
5. Improving the conversations around the user stories
6. Getting better feedback faster

Some metrics I like

1. NPV (net present value)
2. ROI (return on investment)
3. Faster end-to-end cycle time
4. Increased sales
5. Increased market share
6. More eyeballs (on a webpage)
7. Improved eyeball demographics
8. Reduced costs

More metrics I like

1. Reduced risk (although I prefer if this is made more concrete by being monetized…see underwriting)
2. Net promoter score
3. Any specific metric showing higher customer satisfaction
4. Others??

Lies, damn lies & statistics

- It is not having numbers…
- It is making good use of numbers (that are reasonably accurate)

About the following "theories"

- A theory is a stated or unstated way of looking at the world. Ideas that affect how we act.
- The following 5 pages is a list of theories I see people use.
- Not complete; just there to stimulate your thinking.
- Some I agree with strongly; some I disagree with strongly. (Some are stated in a sarcastic way.)
- The point is to enable you to discover your firm's underlying (implicit or explicit) theories.

Theories (examples) - 1

- The customer won't change her mind in [6 months].
- The customer knows what he wants.
- The customer can explain clearly what she wants.
- The customer only knows it when he sees it.
- The customer does not want software, just a solution to her problem.
- The Sales guys are the best ones to explain what the customer wants.

Theories (examples) - 2

- No one in our firm could possibly learn about BV by using metrics.
- Numbers are too hard to collect, so it is better to ignore any potential benefit from them.
- It's really good to use documentation to convey "requirements", since we get to lose all the Tacit knowledge.
- The telephone game is useful in conveying requirements.
- It is too risky to ask the customers for feedback on unreleased products.

Theories (examples) - 3

- What the customer wants and what the shareholders want are always aligned.
- All projects are equally valuable.
- We already put projects in priority order, so all PBIs within an approved project are "required".
- It would be wrong to tell the Team the expected NPV of the effort; they might ....
- Getting feedback on how bad the upfront NPV estimates are could never help us learn ...[x]

Theories (examples) - 4

- There is only one kind of user: "the user".
- There is no distinction between the user and the buyer.
- Cust 1 and Cust 2 always want exactly the same thing.
- We should optimize delivery 'end-to-end'.
- "End-to-end" starts when we get the business requirements document, and ends when we hand-off the SW to the final test group.
- The functionality needed by customer set 1 is never in conflict with the needs of customer set 2.

Theories (examples) - 5

- A bad [X] from team [Y] is not my problem. However bad, it could not have been better. I should just do what I am told. Then things will turn out for the best in this best of all possible worlds. [Apologies to Voltaire.]
- There are no cost-benefit trade-offs in our work. And anyway, IT's costs, always fully understood up-front, have nothing to do with delivering business value.
- Knowledge creation and knowledge decay have nothing to do with business value.

Theories (examples) - 6

- The customers don't want anything quickly, so we should take as long as necessary to build it efficiently.
- IT is just a cost center, so projects should only be cancelled if the team is bad.
- The manager can always accurately describe the requirements known by all the people that report to him.
- We should never let the coder talk to the end user.
- Testers don't need to know "business value".

Theories (examples) - 7

- Business value never changes over the months we deliver the project
- We could not possibly help the customer firm prioritize the "requirements" that the diverse people from that firm have given us
- The best possible feedback is when it goes into production
- Understanding "requirements" requires much more frequent feedback using working software. Directly from the best customer representatives we can find.

Theories (examples) - 8

- Interviews are the best way to elicit the requirements
- Use cases are the best way to articulate the requirements
- Showing the real users working software frequently is the best way to learn about requirements
- The customer fully understands their problem and it never changes during the project
- The appearance of alternate solutions has no impact on the customer requirements

Theories (examples) - 9

- Exogenous variables (war, weather, economic) are not important enough to pressure us into delivering faster.
- The key manager at the customer accurately reflects everything that 'the customer' wants.
- The customer says they want releases only once a year, and there is nothing we could possibly change to get them to accept releases more frequently.
- The cost of testing and the level of bugs in field trials has no influence on customer willingness to accept faster releases.

A note on sarcasm

- I have made every mistake, so the sarcasm is always partly about myself.
- Sarcasm is the acid that frees us from our box. Maybe a bit painful, but useful.
- Even in being painful, one also feels compassionate. "I must be cruel only to be kind."

Exercise 2.

- Map out one specific BV Engineering process.
- At your table, led by one PO.
- Timebox: 40 mins. (4 "days" with each: 2 mins of Daily Scrum and 8 mins of real work.)
- Required output (see later).

Situation

- **A real, specific situation for the Product Owner.**
- **Situation: mainly one product. (But could use other definitions.)**
- **Other team members act as consultants.**
- **Mapping only; not fixing. Current state, not future state.**

Required Output

- **A "map" (picture) of some sort**
  - Ideally end-to-end (whatever that means)
  - Show the process (at a high/medium level)
- **Describe the BV Model (better)**
- **Describe the underlying theories**
- **Describe the timeboxes and feedback loops (either in the picture or in writing)**

In describing the BV Process

- Do it for a specific situation: a group or one team.
- About 20-30 steps (not more, not a lot fewer)
- Can include "project portfolio" management

Should (indirectly) address...

- Do we have a PDCA cycle?
- Who is involved? Where?
- How long does it take? (Or, how many cycles to "get the whole thing done"? Or is that a meaningful concept?)

- It can be similar to a Value Stream Map, but is it then a PDCA cycle?

Timeboxes/Feedback loops

- Describe the timeboxes used in your BV Engineering
- Describe the feedback loops, and where new learning is used to get better.

Hints 1

- Think outside the Scrum "box" (or the box you have put Scrum in)
- Just describe, don't fix.
- PO rules; just enough info for the PO to understand.
- Both PO and SM have a role in keeping the team from getting stuck.
- Do "the best we can" in this timebox.
- If you don't know, guess for now. (And check later.)

Hints 2

- "Could we do a VS map?" Yes, and where is the PDCA cycle?
- It can include only one (Scrum) team or multiple teams.
- For here, KISS is probably a good idea. But you can use this basic framework as a start for complex situations. "Things should be as simple as possible, but not simpler."
- For here: one product is enough.
- You won't be able to keep yourself from fixing, just not too much.

Hints 3

- Do something in all 4 areas:
  - Map
  - BV Model
  - Theories
  - Timeboxes/feedback loops
- Don't get stuck too much in one area.

Debrief

- In a sentence, not repeating what someone else said, what was the (next) biggest thing you learned?
- You can "show" results.
- Biggest = most useful ??

Exercise 3. Improve the BV Model

The BV Model

- Improve the BV model and show (more) its underlying theories and assumptions
- f(x,y,z) = $T
- What is/are the function(s)? (addition, multiplication, etc, etc)
- How many variables? 3, 5, 8?
- What are the best assumptions about "best" values for the variables?

The End

For now....

Retrospective

- What do you remember?
- What will you act on tomorrow?
- What thing(s) will you do to improve your BV Engineering?

Contact Info

Joseph Little
Kitty Hawk Consulting, Inc.
LeanAgileTraining.com
jhlittle@kittyhawkcnsulting.com
704-376-8881