53. UMD Symposium 2018 – Part 2

Knowledge into Practice: Agile

[music]

00:02 Susan Parente: Agile is about giving people what's valuable to them, because that's our job, is to figure out, out of what you're asking, what is it you really need?

00:14 John Johnson: You can have a project, that costs say $500 million, takes 10 years to build, and if you deliver it and no one uses it, it's completely useless. It's worth negative $500 million to that organization.

00:29 Bruce Gay: Design thinking is the creative and systematic approach to problem solving, by placing the user at the center of the experience.

00:37 Sean Eddings: What if instead of focusing on keeping people busy all the time, we focused on delivering work to the customer as quickly as possible? Right? What if that was our motivation?

00:45 Jason Dunn: The most important thing about fixing, is not taking in to the process, the assumption that you have to save the project.

00:54 Jeff Dalton: And remember, Agile is a philosophical cultural model, it's not a software development model. So, what we're saying really, when we're transforming to Agile, most people will say, "We're going to get the team to adopt some of the techniques, but we're not really going to change who we are." And Agile, is about changing who you are.

01:16 Kendall Lott: Welcome back PMs to this, our second in a three part series from the 2018 Project Management Symposium, hosted by the University of Maryland's Project Management Center for Excellence. The theme of the symposium was turning knowledge into practice. In Part One, Episode 51, we culled highlights from the symposium's leadership track. For Part Two, our theme is Agile. The presenters look at Agile through a variety of different lenses and I think you'll find some interesting ideas to consider here. You'll learn more about design thinking, risk in Agile, maximizing throughput, rejecting the hierarchy and so much more.

01:50 Announcer: From the studios of Final Milestone Productions overlooking the White House in downtown Washington DC, This is PM Point of View, the podcast that looks at project management from all the angles. Here's your host, of Kendall Lott.

02:05 Kendall Lott: When you think of Agile what comes to mind? Speed? Small batches and multiple iterations? Creativity? Flexibility? Yeah, that’s all here. You will also hear about Teams –

teamwork and team health. Visuals: they strengthen the agile approach. The more visuals the better (says the audio podcast host). You will also find a couple of interesting topics such as using Agile to enhance the Procurement process; and how to deal with a failing project. And it should no longer be a newsflash to PM Point of View listeners, but a cornerstone of these lessons: “Value.” Defining it up front and using it as a guiding principle through development.

02:32 John Lightner: What I came away with, was that Design Thinking isn't just a luxury for project management, but it's a critical part of meeting cost and schedule.

02:40 KL: Why do you say that?

02:41 JL: Well, because Design Thinking is actually problem-solving that makes the end product a little bit more compatible with the process.

02:49 KL: That was John Lightner one of the many enthusiastic attendees at the symposium. He's talking about Bruce Gay's presentation on Design Thinking and Project Management – a tool box for framing and reframing the problem, when you start with *not* knowing the final result. Listen in to hear how that works

03:03 Bruce Gay: How many times have your projects started out and had a delivery, and then all of a sudden, the user adoption was less than you expected? I know I've had cases where our end users were not totally happy with the product that we shipped to them, and that was mainly because we were not solving the right problem for what these individuals needed. So, Design Thinking helps us do the right thing and then all the rest of our project management tools help us do the thing right.

[music]

03:40 JL: There's an organization called the Design Management Institute, that is a little bit of a lobbying and a membership organization. They've gone out and looked at how can we show monetarily and financially how companies can benefit from having Design Thinking as part of their organizations and what they did was go out and look at organizations across the US and picked these 15 or so and some of them are pretty well-known. There's Apple, there's Disney, there's Steelcase, Starbucks, and they've looked over the past decade or so, and saw how were their returns versus the standard 400/500? In this case in mid-2015, it was a 200% above what the standard is, and these are all organizations that have design that is being supported at the C-Suite, design that's in Management and then design that is used as innovation and a tool for change.

[music]

04:46 BG: So, we as project management professionals we have initiation, there's some execution, there's monitoring and controlling, and there's a closing of a project. Typically, we know at the beginning it's predefined, what we're trying to build or execute against. Design on the other hand, looks like this: It's pretty squiggly, and so there's quite a bit of chaos. There's a lot of things going on here at the beginning, and I can tell you that when I was a project manager about 10 years ago, this was hard for me to understand and manage, because it wasn't easy to monitor and control the staff that was doing this work. It was not exactly easy to understand the thought process, or why we were doing things, why we were doing further exploration, why were the teams doing prototyping and throwing things away in order for them to get to a converged problem space?

05:36 BG: One thing that's important about design and incorporating design on your projects, is there is multiple framing and reframing of the problem space. So quite often, these are projects that you start out, not necessarily knowing what your final result's going to be, but you need to spend time to be able to research and prototype for it.

[music]

06:00 BG: Design Thinking and design isn't really a process, it's more of a toolbox that we as project managers and team members can go in and pull from these. So one of the tool boxes is user research. This is sitting down with your end users, being able to pull out of them both their articulated and their unarticulated needs for the product or the process that you're building for them. Idea generation. This is another important tool box that you'll find from design thinking. This is beyond what we have for normal brain storming. There's things such as brain writing, there's a lot of different methods that you can use to be able to pull together and generate ideas that can be tossed against the wall and basically turn into prototypes, at some point. Visualization is pretty important to Design Thinking.

06:56 BG: Being able to take your ideas and concepts out of your head and put it on paper is important. Verbal communication is not as precise as something that we can visually see. And then prototyping is the last tool box that teams can use. Obviously, it's always good to try out things before you either throw money at the engineering of it, so, in some of my use cases here, we're going to have paper prototypes, we're going to have prototypes that are on white boards. I've seen prototypes too where they've done mock-ups of iPhones using cardboard and paper, so there's a lot of things we can do.

[music]

07:37 BG: Design reduces overall development costs. Tom Gilb back in the '80s wrote in his Principles of Software Engineering Management that if you find defects in the development phase, they're 10 times as expensive to fix than during design. And then after launch, you're going to find them approximately 100 times more expensive, so the key thing here, we spend as project managers a lot of time thinking about planning. There's also a need for us to put a lot of time into design, exploring the problem space, exploring what our users really need, before we spend a lot of money at the development stage, and this could be both for hardware, software, process, services.

08:20 BG: This can be used across the board. Being able to incorporate your users earlier in the process will help mitigate the risk and failure, and you and your team's missing requirements. I know when I first got into this, I'm thinking wow, usability tests, having extra staff that are user experienced is going to be expensive. Spend your money on smaller usability tests and repeat those multiple times. We typically try to get anywhere from five to nine different users through that process. And it's going to help you catch most your usability issues that your product is going to find before it's launched.

[music]

09:03 BG: Have the actual users of your projects involved and tightly coupled with your development teams. I'm involved in a joint partnership with GE Healthcare, and we're redesigning this radiology workstation and what we did was we did in fact embed design personnel into our projects. We're up to eight. They spent a lot of time exploring what is the job of a radiologist? How do you do your work, where are your pain points? They mapped out their clinician experiences and then we also included the designers in the scrum teams. We also did a lot of co-design sessions. This is where we get multi-disciplinary teams together, and that was very useful. This is actually something that we can, as project managers, put on our teams pretty easily. This is where you get stakeholders, your internal teams, experts maybe from the industry together and they will help work through technical or programmatic problems as you build.

[music]

10:14 KL: Sean Eddings speaks to through-put and speed in his presentation on The Traffic Fallacy The agile connection? He sees the problem as flow efficiency, not activity maximization—follow this logic....

10:21 Sean Eddings: A little while ago I was stuck in traffic and I thought to myself, "Geez, why is this still a thing, right? In 2018. So I googled 'where is the worst traffic in the world?' And I came across an interesting story. In 2004 Houston had a major traffic problem. It was taking far too long to go from A to B. The Katy Freeway, this stretch of 30 miles was the second most congested road in America. So measured in hours, they calculated that drivers sit about 25 million hours a year in traffic, so they recognized this was just a totally undesirable state of affairs. Too many cars, not enough freeway. So imagine this was something someone reasonable said. I could get behind this- “more lanes, more room for more cars”, intuitively, I guess that would mean shorter travel times. So therefore, “let's build more freeway”. And that's what they did.

11:10 SE: They spent $2.8 billion to widen to 26 lanes and this problem has become worse. Travel time has increased 55% from 41 minutes to 64 minutes. The mileage, 28 miles, has remained the same. So, how could this be? Kind of flies in the face of common sense? They realized that it wasn't the same amount of cars now, right? As the amount of road increased, the number of cars increased too.

[music]

11:48 SE: So, I took out my agile and Lean principles thinking and I realized hey, despite the capacity increases, there are too many vehicles in progress. That means continued high road utilization, high throughput times, and idling vehicles. So, when the freeway is full, right? You need slack or space on the road to handle the inevitable variations in vehicles. Vehicles are different sizes. You've got your 18 wheelers, and you've got your motorcycles. You've got people who drive recklessly, you've got people who drive conservatively, different skill levels, beginners, people who've been on the road for 40 years. Too much variation. You need slack in the system. So, instead of increasing the amount of road, since we can't control the variation in vehicles, perhaps the city, the state, could experiment with a vehicle in progress limit. Reduce the number of vehicles on the road at any given time.

[music]

12:45 SE: In software development, our freeways, you may have a backlog or essentially a project that you're working on. And you're looking at it and you're like, "Oh wow. We're behind schedule. Why is nothing getting done? Well, we should just add more people. Get more people on this project. More people, move faster, etcetera etcetera. Oh, hey, look. There are two members here who have greater than one thing assigned to them that's marked as in progress.". Everyone being incredibly busy, right? There was always so much work to do, but value, right? The things that we were being hired to do was taking an incredibly long time to get to the customer, which is our ultimate measure of success. People had too much on their plate. Other people, not enough. Tasks were sitting around in unknown various states waiting for people to become free to work on them. One individual had something like 80 tasks assigned to him at one time, and it's like how could that possibly be true?

[music]

13:39 SE: We experimented with, kind of, flipping our thinking. And what if instead of focusing on keeping people busy all the time, we focused on delivering work to the customer as quickly as possible, right? What if that was our motivation. So, perhaps a more fun way of saying this, around the office, is what if our goal as a team was to take our ideas and get them into action. Put them into the real world, as quickly as possible. When you're thinking as a team, how do we want to do? We have a customer need, how do we get to the customer as quickly as possible, and work at a sustainable pace, not work weekends? So we experimented with three things. A lot of this has its origins in Lean Thinking, but if anyone here has looked into Kanban systems, a lot of this comes from that too, which is based largely on Lean Thinking. But one of the things that we started experimenting with, was visualizing our own process. We limited our work-in-progress and we worked together to improve as a team. So, none of these experiments made people work any faster, asked them to code faster, work the weekend.

[music]

14:42 SE: So, the first step of visualizing our process was pretty easy, right? We've got out of the electronic tool. We built the physical board with columns representing work states, which brought a high level of visibility to the team, so the whole team could see what was in progress, what was incoming, who was working on what. Were things stuck, too little, where could people help each other? Being the key point there. And if there are any bottlenecks. If there was a bottleneck, the team could then in that moment discuss it, anything we could do about it. And it made that hidden work, "Oh, I've got 20 things assigned to me that are marked as in progress, but really, 11 of those are waiting on something else, right? So, we could talk about how do I want to represent the things that are waiting on someone else and why are they waiting on someone else.

15:20 SE: So, really, what this means is we, kind of, had a place to encourage these really productive valuable conversations the team could have. So, the fundamental idea here was that the team could, in the morning, kind of, take stock of where they were in the context of the whole system that they were a part of, and determine what the next best course of action was. So WIP limits, or Work-In-Progress limits. You're going to limit our work-in-progress here, because how many things, fundamentally believe, can you work on at one time, right? I can work on one thing at a time. I may have 10 things going on around me, but really, I can work on one thing at a time. So, it's important that this board represents what's really going on.

15:57 SE: So, if the team can, kind of, observe the board in action, and decide where are things getting stuck? Should we dedicate more time to one section today? We've got a lot of things together in QA. QA seems to be bottling up. Maybe we just need to get all these things out of QA, move them into ready-for-production, or perhaps just production. And there's no hard set limit on WIP limits, work-in-progress limits, right? There are a couple of rules of thumb here, where a lower WIP limit is generally better, right? A pretty high WIP limit can leave work idle, a low WIP limit means people may be hanging around looking for something to do for too long. So you want slack, but not too much slack.

[music]

16:38 SE: So, we have these cards which represent an individual flow unit. So this is the thing itself, that sits on the board and the items that move through the cue are these flow units, and they are essentially just the task, right? And you could represent this in a number of different ways. At the time, when we were rolling this out across the teams, we kept them, kind of, short and sweet. Perhaps on little sticky notes. The teams have, kind of, matured a little bit now, so they're more comfortable maintaining this thinking in electronic tool. You take your list of work that you need to do. You want to make sure that it's prioritized based on business value: One, two, three. That's your backlog in a very simple sense. Teams work from the top of the backlog to the bottom of the backlog.

17:19 SE: That's the order in which they approach the project, and they self-organize to figure out they've got their work, and they pull it through the system, as they themselves decide what makes sense to work on as a team and who the best person is. And this has its roots in something called flow efficiency. So, traditionally, the structure was the team member has a lot of things assigned to them. So, if you put a camera on the team member's shoulder, they were always really busy. But if you look at what was the most important thing, which is the flow unit, it was idling. There were a lot of times it wasn't doing anything. Whereas, if you flip it a little bit and you focus on flow efficiency, you may have multiple people, kind of, working on one thing, as a team, moving things through.

18:00 SE: And the camera on the team members shoulder may be idle and that's okay, because if you look at the flow unit itself, the task, the thing, the idea that we're getting through to production, that was always busy. There's this quote from a book called This is Lean, that I highly recommend for folks who're interested in exploring this further. “Flow efficiency is the sum of value-added time in relation to throughput time.” And it's an operational strategy, right? So, flow efficiency is about reducing your non value-adding activities.

[music]

18:36 KL: Agile feels risky to many people, and so we ask, “Is risk management even relevant or possible in agile development?” Susan Parente’s response is an emphatic “yes” in her presentation: “Agile Quantitative Risk Analysis.”

18:41 Susan Parente: Whenever you hear the word agile, the next word you should be thinking about is value. Providing value to customers, that's like a fundamental thing. And decreasing project risk, especially around scope. That does not mean that agile projects have no scope risk, but they have a highly reduced amount of scope risk, because the commitments are smaller and over shorter periods of time. So you might want to consider using agile when your project has one of these elements: A high level of uncertainty, that would be a high level of scope risk; Complexity, doing something that involves multiple systems and integrations of different systems; Something innovative, you're doing something nobody's ever done before, and; The customer wants something really soon. If in a high level of urgency with agile, cause you deliver in increments, you can actually provide them value upfront very quickly, and they can get something from that. So it may not be the full solution but maybe they get a piece of it.

[music]

19:42 SP: In traditional project management, we define the scope upfront, we're clear on what the scope is, and then we estimate the time or a schedule and the cost of our budget based on that scope. We flip that around with agile or adaptive project processes and what we do is we say we're going to fix the time, so we're going to time box our work, and we're going to fix the costs. How are costs fixed? We have five to nine members and we have them, you know, working for this amount of time, so we know how much it's going to cost and we know how long it's going to take, what we don't know is what we're going to get. So now this is a big problem for every federal contract, right? I've never met a federal client that is willing to put out time and money to have something done but not be sure at the end if that's going to be what they want.

20:30 SP: If you ask a pure Agilist person, "How do you deal with this?", they'll say, "We can commit to somebody in a certain time frame and a certain cost that we will give them what is most valuable to them. We cannot commit to what that would be." Now, there are some things you can do with that. You can have some caveats. You can have some milestones.You can have some high-level requirements. For example, if I was building a vehicle and I want to get from here to downtown DC, that's a requirement that in the end I need a vehicle that gets me from point A to point B. Now, is it going to be covered in the rain, is it going to run on gas, is it going to be electric?

21:13 SP: There's no commitment to that, but I can commit to that value add, to that business result that you want. Let's say you make an assumption that at the end of the project, because stuff happens, life happens, things are going to change, even external requirements may change, at the end of that project you're going to get 80% of the scope completed. If you knew that, you would be better off to go with this value driven agile approach, because you will get the 80% of highest priority. Whereas in a plan-driven you're going to get 80%, but you don't know what 80% you'll get. You may not get your top priorities. And if you've ever been to the National Auto Show, you can look at one of those cars spinning around. There isn't necessarily an engine in there. Okay? So that's like 80% of a car, but if you need to get from point A to point B that is not the car you wanted, okay? The one without the cup holder, and no paint on it, and maybe doesn't even have back seats. That will get you from point A to point B.

[music]

22:22 SP: So this brings us to agile risk management, which is why you're here. Okay, so yes, it's inherent in inspection and adaption, which is part of agile projects. We're always inspecting and adapting. We want to focus on maximizing value, we can think of that as opportunity or positive risks, in addition to minimizing threats or negative threats. We want to do regular and iterative inspection as part of risk management, we want to evaluate risks for each iteration and actually look at them for maybe the next two or three iterations. You know, there's a reason at the end of the iteration we review how we did right before we plan how we're going to do it. It's inherent again, to manage risk as part of agile projects, and we want to capture the lessons learned.

[music]

23:19 SP: How quickly can we respond to change? That's going to affect how we can deliver value to our customers. So some things that we can do, we can look at response efficiency, how much time, how much cost, how much resources does it take to respond to change. And we can actually quantitatively assess that. We can also look at what is the extensiveness of that response. So if there's 100 changes and we respond to 80% of them, then that's our efficiency with that. You want to communicate with your customer by escalating risks and work with them to identify concerns and reduce risks.

24:00 SP: The other thing about risk management for Agile is, it's really built into Agile practices. For example, in a daily stand-up meeting, we look at what has been done, what are we planning to do, and what are the barriers or impediments. The barriers or impediments is another way of talking about risk. The other way we can implement it is through planning, and we look at the backlog, our product backlog... We look at it prioritized by value, but also consider that, they call it a risk adjusted backlog, we can do a subjective assessment where we look at the features and see which features have the highest risk.

24:39 SP: Now why is that important? Well, if a feature has a high risk, then we might want to address it sooner in the project versus later. Or if it has a high risk because of an unknown factor that we believe we'll have more information about later, then maybe we'll address it later in the project. So this risk adjusted backlog... It doesn't have to be a whole crazy analysis. You can just say, "What is the level of uncertainty with this feature?" So that's a way to kind of get started with implementing risk management for Agile. And that’s something you probably want to put in place before you do more quantitative risk analysis measures.

[music]

26:00 KL: From analyses of design, flow and risk we turn to the soft skills. Jeff Dalton talks about Agility for Leaders – how leaders need to re-fashion their leadership techniques in an Agile environment, an environment defined by self-directed teams.

26:17 Jeff Dalton: My wife and I decided to move to the Florida Keys, and we bought a house there on September 9th, 2017. A little friend named Irma came to visit. What happened after Irma was really fascinating. The roads were all closed, FEMA wasn't there. There was no water, there was no fuel, there were no telephones. There were just the 10,000 people that stayed on the higher Florida Keys, so about 10% of people stayed. And what they did is, is they self-organized around Facebook groups and they found each other and they all gathered together in schools and then they went from neighborhood to neighborhood looking for survivors. This notion of self-organization, how we can pull together and make stuff happen without traditional leadership, what we call, "command and control" or "low-trust leadership" in our business, is amazing and was really witnessed in Irma. I experienced it personally. I wasn't there, but I connected with one of these guys on the Facebook group, and he went to my house and actually took video of everything and was showing me what was left, or what wasn't left in this particular case. And to ensure that my neighbors were safe and all these kinds of things.

[music]

27:32 JD: So worldwide, people have lost faith in bureaucracy at an astounding rate. Here is some data from Gallup. People have lost faith in organized religion. 65% in 1980, down to 41%. The financial system, going from 60% to 27%. We've lost faith in the media. We've lost tremendous faith in business. Congress is the worst one, from 42% to 9%, but what that means is that self-organization and Agile, by definition, because that's what Agile is about, is a result of people gradually losing faith in command and control leadership. It isn't a cause of it. You hear a lot of Agilists talk about how they invented something. They reacted to something.

[music]

28:38 JD: So the question is, is how can you make that big? How do you scale that? How do you lead that? Those of us that are leaders, and I know we have a lot in the audience, went through a process to become leaders that doesn't involve self-organization. The self-organization that we have to experience with Agile participants is much different than the skills that we have to have to lead organizations that are Agile. Skills required to become a leader in an Agile organization are antithetical to the skills of learning to become a leader in a typical organization. And part of the reason we're struggling is, is a plethora of models that are out there in the industry. So if you're using a PMBOK, ITIL, CMMI, ISO, COBIT, Strong, SAFe. Any of these models, you know what I'm talking about. The word capability is kind of thrown around in our industry a lot and capability is actually broken into three main categories. So, what I call the why-ability model, these are policies and Agile values.

29:40 JD: So if you're familiar with the Agile manifesto, you're familiar with the concept of a why-ability model. Why are we doing this? This is the reason we're doing this. That's often coupled of with what I call a What-ability model. Models that tell us what to do. PMBOK is a what-ability model. CMMI is a what-ability model. Strunk is a what-ability model. What these models don't do for us and for leaders and for a large organization, is what I call the how-ability part. The behavior that we hope people to exhibit in a large scale organization. So if we're going to, say we're going to do self-organizing in Agile, we have to standardize behavior. So there has to be behavioral guidelines, what they call "guardrails" in the Agile world, and we have to have a way of leading people who are behaving in a self-organizing way. Very different than having a Microsoft Project work plan and a set of tasks and tracking everybody as we go.

30:45 JD: So we often see this where, leaders of Agile organizations are running their teams using what-ability models. Like PMBOK. But they're expecting their teams to behave in an Agile way. High trust, high collaboration, transparency, self-organization, self-subscription. These things don't go well together with PMBOK, so this is a culture divide. So we've got a lot of young developers coming into the system right now. A lot of the younger workers are really excited about self-organization, and here we are running an organization saying, "No, you must run this project this way."

[music]

31:28 JD: So, this is just an interesting dichotomy. Think about this: We want to run large-scale, self-organizations, so we need really strong leadership. It doesn't seem to make sense at first, right? Well, here's the thing: We want to have strong leadership, just not the kind of leadership we think. So a strong leader in the traditional sense is setting goals and objectives and tasking people, they are setting budgets, they're tracking people, they're holding people accountable, they're making sure that all of the work gets done. They're statusing their senior management and they're making sure projects come in on time and on budget. A leader in a self-organizing setting is motivating people to do that without leadership.

[music]

32:17 JD: So a good Agile leader focuses on providing. Equiping people with the right infrastructure, with the right physical plan, with the right tools. They focus on teaming. Building healthy teams that are self-organizing and transparent and collaborative, but they affirm and confirm that all of these things are happening. So as it turns out, it's all good to have self-organization, but people need to be checked sometimes. People need to be tasked with looking at how this is going. They need to help people envision solutions, just not build solutions.

[music]

33:00 JD: And the first thing we do when we go work with a company, is we work with their leaders to transform them. These leaders have spent decades, sometimes 30, 40 years, becoming who they are. And when there's 10 levels of leadership, we need to start at the highest level of the company and work through 15 levels before we even get down to the team, or else they’re going to crash and burn the minute we leave. Leaders need to start with who they are, and who they are is clearly defined by the values that they project. We start by teaching them about how to have values that are enabling. And the Agile manifesto, if you're familiar with that, has this all defined, right?

33:42 JD: If you're familiar with collaboration and transparency, fail fast. All of those things are part of enabling teams and projecting values that provide envisioning. And we've developed this set of nine core values that every Agile leader learns about early on: Openness, focus, commitment, respect, visibility, sense of humor, courage and fail fast. And as an Agile leader, we need to teach them how to project those values. So, right now we have a scenario where most leaders are mouthing the words, but not living the values.

[music]

34:23 JD: So this stuff isn't magic. They have to build an infrastructure around it. And one of the things that we have to do is demonstrate values traceability. So, values traceability is an infrastructure concept that we came up with that says, "If I have this set of the nine core Agile values, then every single framework, method, and tool that we use has to support and be traced directly to that value." So if you take an Agile framework like Scrum, it fits perfectly well of course, the nine Scrum values...

34:55 JD: And then the techniques that go along with Scrum, like daily stand-ups and sprint demos and Planning Poker and all of those kinds of good things, fit perfectly well and there's bi-directional traceability there, right? Then we move to the providing part and this is really interesting because leaders don't know what to provide to Agile teams to be successful. Face-to-face collaboration and visual information management are critical to Agile's success. So we need big signs and we need digital signs and we need things that are obvious and that are in front of us that we can see. You can't implement it without it. So these kind of cultural changes are important, and leadership has to learn that, they have to provide a whole different kind of infrastructure, than what a typical organization defines.

35:42 JD: The other thing is that leaders need to live and breathe this infrastructure themselves. So we have to ban private offices and we have to ban them being on a different floor than the people that work for them. So the CIOs have Scrum and Kanban boards in their offices or in their open space. They do have some private space to have private meetings, of course, but they live and breathe Agile, they use sprint planning, and they do delivery, and they have Kanban boards, and they have cards. So in other words, they're projecting the values that they're asking people to protect.

[music]

36:19 JD: Teaming is everything on an Agile team. There's a fantastic book... If you haven't read Brian Robertson's book, Holacracy. It's all about self-organizing teams and how rigorous and disciplined they are. And it's really fascinating. We tend to think of self-organization as low rigor and low process. It isn't, it's high rigor and high process. And teaming and self-governing is really difficult. Self-organizing requires accountability. Well, how does a Self-organizing team hold itself accountable? Only one way. Through an infrastructure of peer accountability, right? Peer pressure, right? But we have to do it within the context of a framework. We just can't tell each other what to do. So, behavior needs to get overlaid above some more vertical traceability of agile values down to the agile techniques.

[music]

37:19 Jason Dunn: Stopping a project is a very hard decision. So I'm really going to talk about three things: being prepared, being courageous, and making timely decisions.

[music]

37:31 KL: That's Jason Dunn on deciding when to save a project and when to shut it down. It's not something that we as PMs like to discuss a lot. But he has some excellent ideas regarding the warning signs, the triggers, and the procedures.

37:46 JD: We'll talk about the warning signals and signs, and I'm going to then walk you through three paths for this position for troubled projects. I think the important thing though, is let's set up why we need to talk about this. And it's certainly an issue at most organizations, but this is what we know, based on data that was collected around 2015. Is once you get to that final funding date, less than 1% of the projects were killed. Once that train starts down the track, it's very hard to stop it, but this is also what we know. If you look at projects that delivered from a business standpoint, 80 plus percent of their business benefits. Then only about 40% of the projects meet that expectation. So, if we look at then, what was estimated, we spent in 2017 on projects $6.3 trillion. Let's just say, 10% of those projects we shouldn't have done.

38:47 JD: That's $630 billion dollars. This is probably even more important. So we did a study, and really this study was focused on team health. So we looked at team health, and so we asked about 30 questions of teams at the time. One of the questions that we asked was, "Do you think this project... " and this was... We asked them right as they were going for funding for the project, we said, "Do you think this project has an adequate budget to meet its needs?" "Do you think this project has a realistic schedule?" "And do you think this project has a fairly reasonable and clear set of project objectives?" So we took those three questions and what we found out is that if the project team said, "No" to two of them. It was highly predictive of whether that project was going to be a cost or schedule success or business success.

39:41 JD: So this is the thing, is that project teams have an uncanny but a very predictable way of killing projects. We asked people, "Is this a good budget?" "No." "Are you going to change what you're asking for?" "No." "Is this a good schedule." "No." "Are you going to ask for more time?" "No, this is what we were told we have." All right. So this – a lot of what we're talking about here... Two things culture and courageousness. It's tough to kill projects.

[music]

40:17 JD: When I talk about making sure that we're set up to deal with problem projects that might come up, there's really four things I'm going to talk about. The first is understanding what your trigger points are going to be. You're going to watch for signs, you're going to investigate, okay? If you see those signs come up, right? So maybe your trigger points are hit and then you're going to have to make a decision. All right, so that's kind of the process flow around dealing with troubled projects.

40:47 JD: The most important thing is that you don't wait until there's a fire to figure out where the emergency exits are. With projects you don't wait 'til the project is going down in flames to figure out, "Oh, I think maybe we should do something about it." Lots of organizations they don't have any of these types of things in place. So you’ve got to start thinking about it very early on. You want to customize it to the project, but generally it's going to be a common framework. So you don't have to do it new every time. Come up with something that's based on metrics. I'll give you some examples of that as we go through, but come up with something specific. The last piece is once they're triggered, be prepared to know exactly what you want to do.

[music]

41:38 JD: Let's talk about some warning signs. Does everybody know the difference between a lagging indicator and a leading indicator? If you don't, lagging is, it's a measure or metric of something that's already happened like cost growth, our costs are going over our estimates. Well, that's already happened. It's too late. Leading are, "Oh, this looks like something that could cause our costs to grow," right? So, very early on in conceptual phase, when you're setting your estimates, then you're going to allow a wider latitude around an issue before you get the trigger point. When you're later, if you're during execution, you're developing, you're programming, then the moment you start to get over 10%, you might want to start to take a look at some things. Beause by the time you get into programming and certainly into testing, you're going to have really, at least half or more of your monies been spent. You have a really good sense how much that project should cost.

[music]

42:47 JD: Schedule, same thing, if it's very early on I'm not going to worry. If you come to me in a 16 month project in the first two months and you say, "You know, I said it was 16, but now it's going to be 18." I'm not too worried about that. But if you're pretty late in execution, then as your date starts to jump around, that tells me something. That tells me something's going on in your project and you may or may not have a good handle on what that is. So I'm going to ask some questions and we might reach a trigger point with that. Scope, we can't finalize the scope during the planning phase so we go into the execution phase with optional scopes still on the books, significant changes to scope, dropping significant scope.

43:37 JD: We do projects because they bring certain benefits. Those benefits hopefully, are tied to scope. The moment you start you shedding scope, you start to shed benefits. If you're starting to drop significant scope, that is an early warning bell to me. Then, if you got changes to your technical solution, major changes to your contractor, when you're starting to have major changes to your contractors or vendors, the ship's already on fire.

[music]

44:09 JD: The leading indicators are team health, quality of requirements. Those are the types of leading indicators. So if we persist, let's say we have something called Phase One where we do our requirements. If Phase One goes from two months, into three months, into five months because we're struggling to nail down the requirements, that is an immediate warning bell for the PMO. So we're going to start to really dig into that project. We do health assessments every month on all of our, what we call consequential projects. That for us, could potentially be a trigger point depending on how long it goes on. Engagement. If there are people who are going, "You know, we're just going to skip that piece of it because we've got to get to this other stuff," the moment that you start skipping key pieces of it is another warning bell.

45:05 JD: If you've got poor engagement in the business, issues of trust between business and IT or the business and contractors, those are all warning bells. This was the other one that continues to bedevil us a little bit, is we get in, we're developing an IT system and all of a sudden somebody says, "This process is crappy. It's not good." What is the worst thing you can do in life as an IT manager, is you can slap an IT system on top of a bad process where you immortalize that bad process forever. The moment that I hear, "Oh, we need to go back and look at some of the business processes we got time out", then you need to stop the development piece and you need to figure out what it is that you need because you don't probably understand the problem that you think that you're solving. Those are all some of our warning signs that we use.

[music]

46:03 JD: Now, if things go wrong, you really have three things that you can do. You can either reset your expectations. That's a situation where your project's pretty far down the road. It's still important. We still need it. We think we've got a really good handle around what the problem is and so we're just going to finish it up. We're going to basically not reset our base lines necessarily, because we want to learn from these projects, but we're going to reset our thinking about how much it should cost and how long it should take.

46:36 JD: This is the one that we tend to jump to pretty quickly and then we're going to fix the project. We're going to jump in, be PMO heroes. We're going to save the project. We're going to put a team on it. It's all going to be good. Everybody's going to be happy at the end. If it's a mandate, then we're probably going to try to figure out some way to get that done, even if the first way we thought of was not a good way. Or the situation where your benefits, if it's not a mandated project, as an example, the project you had to do, aren't severely compromised. You're still going to get benefits out of it, so you probably might want to save that project. And then, of course, the final one, the one that we don't do very well, is we're going to cancel the project.

[music]

47:25 JD: The number one reason why people don't want to cancel their projects is because it's related to their job. The most important thing you can do when you're going to cancel a project is to make sure that someone has a job that they can go to.

[music]

47:40 JD: Close out the project, of course lessons learned. When you do your lessons learned, you're going to talk about the process. What did we miss in the process? How can we strengthen the process to get that right? This is one that that people have a little bit of a hard time with. When you cancel a project at the right time, pop some champagne.

[laughter]

48:02 JD: That's a good thing. Reward people. Make people feel good because they've done the right thing.

[music]

48:17 John Johnson: The emphasis in terms of a goal is speed, speed being the most important aspect of a project or business.

[music]

48:24 KL: That's John Johnson, whose presentation is on Earned Capabilities Management or ECM. His co-presenter, Christopher Harris, discusses FIRE, which stands for fast, inexpensive, restrained and elegant. Developed by Dan Ward, it's a framework for rapid, frugal innovation.

48:43 JJ: My name's John Johnson. And this is Chris Harris, and we're here with talk to you about Earned Capability's Management, and how when we combine it with the FIRE principles, it offers a framework for rapid acquisitions. Projects exist to benefit the organization, and so we should stop measuring our products based upon the development of the product itself, but actually, on the benefit it gives the organization. And the way that we can do that is we define what we're delivering as a capability, and the capability is the ability to perform work of a certain quality, capacity and efficiency. If we can say that we are delivering a new capability into the organization, then we can define how that's changing the organization, and what the return on investment is, and if the return on the investment is positive, that's a good investment. And if it's more positive than another project, it's a better investment. So what does that look like in terms of a calculation?

49:38 JJ: It's the idea that, if we can do a simple process model for the organization, for example, if you are delivering benefits in terms of insurance, if you are saving lives in disaster response, if you are making profit, throughput is the goal. So, set the objective, baseline the organization performance of how well you are delivering those benefits now, and then measure, after you deliver the capability, the change. So if you increase throughput, that's positive, if you decrease cost that's positive and then you divide it by the total cost of the capability. Put all these dollars in annual terms, so you can compare one against the other, and you're good to go.

[music]

50:27 JJ: So if we work this way then we can appreciate two very important things: Timing impacts and market impacts. If it is costing you right now $160,000 a day in order to build a plant, and that plant is going to be able to realize $350,000 in revenue as soon as it's done, I can double my daily cost to speed up that project, and if I could halve the time, I'm winning. Right? In almost all cases, it makes sense to speed up a project if you actually look at the returns. That's the market returns. The timing impacts are a little bit more interesting, in that it deals with the technical uncertainty.

51:05 JJ: In the year 2000, if I was developing an economy car, I'd have a really good market for that. Fast forward 17 years later, I've got Amazon to deliver my stuff for me for free, I've got Skype to allow me to be there on the birthday without having to be there, and I've got ride sharing services where the trip used to cost 30 bucks, now it costs 10 bucks. If I was making more than one trip a day I should buy a car back in 2000, today, I'd have to be going more than just to and from work on average. This is the timing impact of the capability of having the ability to drive yourself, no longer needed.

[music]

51:45 JJ: So how do we know what to do? What's our job? Identify that, so important, that's why defining the problem is so important. That's why Agile comes in and says, "Let's define this problem, let's test our understanding of that problem immediately and then let's find what the real problem is and let's solve that." It's because of the fact that finding this and doing it is the most important part of your project. So what we want to do, is we want to at any point along our development of a project, we want to examine as often as we can, and we want to integrate up to the highest level that we can and then test at the system level and say, "are we actually fixing that four percent that one thing?

52:21 JJ: Are we actually hitting the one thing and deriving value?" That's how we win in our projects. We define what we want to do and the major capabilities to get there, then we wanted to actually deliver on this release, so we break it down, we break down the stories and tasks. What does this mean about our architecture? It needs to be vertical, and it's totally worth it for all of the buffering, for all of the extra development, and if we have to build extra tests, it's totally worth it too. Being faster at testing, means that we have to do less design.

52:52 JJ: Also being better at testing and being able to test all the way up to the highest level of the system, means we can actually identify the one thing. So the Rolling Wave plan is how we get there, you plan farther out at the highest level of your capabilities, you break it down, and then at the lowest level, you're closing, closing, closing all the time. Integrating and closing until you can test and then you test to find out where you are. So with that, that's the ECM part of the speech, I'm going to hand it over to Chris.

[music]

53:23 Chris Harris: Okay now this is the part of the presentation where we transition the concepts of Agile into the government contracting world through the prism of FIRE. And we'll use these principles because we want to implement them via the marginal contracting, as set forth in FAR Part 39. And, at its essence we know that FIRE is about keeping projects simple in terms of budget and scope which leads to speed and quality and greater overall value to customers, in terms of costs and the capability that is actually delivered to you.

[music]

54:04 CH: This was the NASA lead, faster, better, cheaper initiative as it was called, that actually used these FIRE principles in order to get things done, and it led to some very interesting results. 16 missions, 10 of which were very successful, the five trips to Mars, the one trip to the moon, four Earth-orbiting satellites and one asteroid rendezvous. So all of these missions cost less than one Saturn Cassini mission, which means that the government got 10 wins for one.

[music]

54:40 CH: So how does this FIRE principle tie back to Agile? In simplest terms, they both use small batches to get things done. So what FIRE does, is it helps us learn faster so that the knowledge gap declines much more quickly and then I can actually manage and deliver more quickly. That is the idea. So now that the project manager who's responsible for controlling the day-to-day activities and then delivering the project to the government, they're saying, "Well, how can I make sure that this goes over without a hitch?" Because we can plan all day, but if we can't get it on contract and execute it, it's not going anywhere fast.

[music]

55:28 CH: So what we are focused on, is making high level statements where people can understand the context of what they're doing. For example, if you want a search capability, don't bind yourself by saying “a search feature that allows two levels of complexity” and they'll go down and they'll say all this stuff. Because when you get in there and you actually discover what the customer needs, then chances are you may need to go into the right direction. Because when you're developing for the government class, and of course, everybody knows this, they bring in 20 people, but the number of stakeholders that you actually need to talk to is probably more like 40 or 50. So when you get in there, you will actually find out what you need to do to support everyone. So we try to keep it safe by making a high level statement in terms of business value that you're trying to deliver, and that is the ability to locate and share documents efficiently and effectively.

56:21 CH: Now that you have established these capabilities at a high level, then you logically group them into modules. And then each one of those could be a plan on the contract and have the capability in it, that the agency says that they want. Now that you've developed your modules, we're just winding our way down to more and more detail. You've got your must-haves, your should-haves, your could-haves. From the most important down to those that are of lesser importance. And of course, you've got to start with the most important stuff first. And now that you have that, you are now ready to do something controversial in many circles, but I think it's the way that the government should move to stop wasting money, and that is what's called a "firm-fixed-price contract" in the development environment. Because you have at least defined the shell of what you need to do.

57:16 CH: Now the way that you go into your firm-fixed-time contract, is that you fund the must-haves, but you also fund many of the should, or all of the should-haves, in order to create a scope buffer. What does that do for everybody? Well, on the contractor's side it provides less risk. On the government's side, I get a lower price because I actually don't overprice the must-haves, because I'm sure you're going to do some changing in there.

57:45 CH: Okay. So, how do we winnow our way down to making sure that we get something of quality? So that is called, listing the assumptions that you will need in order to actually scope this correctly and ensure that it's delivered in accordance with the quality and terms and conditions that you need. You know what that causes you to do? As a contractor you go, "Well, all right. Since you went from this level of security to the next level of security, now the complexity increased. There are so many integration points, this, that, and the other thing. And now I know how many user stories I need." And then of course, that translates into bodies, and yada, yada. And there you go. So, this is providing context and it’s also providing the definition of done.

[music]

58:35 KL: Speed and throughput are key. Don't be afraid to close down a project when the warning signs are clear and to be truly effective, Agile needs to be adopted as a culture. And did you notice how many presenters spoke to “value”? Again, it is the key principle in decision-making on projects, and producing outcomes that matter. Those are just a few of the takeaways from the Agile track at the University of Maryland's 2018 Project Management Symposium. If you want to learn more about the presenters or check out the slides that go with their presentations, go to www.PMSymposium.umd.edu/PM2018. Stay tuned for the third of our series from the Symposium on Construction, coming up in the Fall. To learn more about the University of Maryland's Project Management Center for Excellence, go to www.pm.umd.edu. Special thanks to this group of presenters, Jeff Dalton, Jason Dunn, Sean Eddings, Bruce Gay, John Johnson, Christopher Harris, and Susan Parente.

59:39 Announcer: Our theme music was composed by Molly Flannery, used with permission. Additional original music by Gary Fieldman, Rich Greenblatt, Lionel Lyles, and Hiroaki Honshuku. Post-production performed at M Powered Strategies.

[music]

59:53 KL: PMPs who have listened to this complete podcast may submit a PDU claim, one PDU, in the talent triangle, Technical, with the Project Management's Institute’s CCR system. Use provider code 4634, and the title "PMPOV0053 Knowledge into Practice Agile". Visit our Facebook page PM Point of View to comment and to listen to more episodes. There you will also find links to the transcripts of all of our one-hour productions. You can also leave a comment on the projectmanagement.com portal, evaluate us on iTunes, and of course, you may contact me directly on LinkedIn. I'm your host Kendall Lott. Until next time, keep it in scope and get it done.

[music]

1:00:37 Announcer: This has been Final Milestone production, sponsored by M Powered Strategies.