**"ESQUEÇA MÉTODOS ÁGEIS"**

Fabio Akita Co-fundador da Codeminer 42, empresa de desenvolvimento de software relata em seu vídeo, o lamentável estado atual do mundo do "Ágil". O recado é muito simples: não existe bala de prata. Agilidade não são processos, metodologias, nem a miríade de numerologia e astrologia que assola este mercado. É hora de recuperar um pouco da essência original do mundo da Agilidade.

Eu acompanhei o nascimento, a evolução e a decadência desde quase quando o movimento surgiu. Lembrem que eu desenvolvia software desde os anos 90 e a idéia das práticas ágeis foi uma mudança muito bem vinda pra mim, por isso é particularmente frustrante ver o que isso se tornou hoje.

Eu nunca tive muita vontade de participar de eventos ou conferências sobre Ágil, sempre achei perda de tempo. Eu tenho minha própria empresa de desenvolvimento de software, mas eu não “vendo” agilidade. O foco sempre usar as práticas de desenvolvimento de software como algo básico, e não vender implementação de metodologias exotéricas feng shui de projetos. Eu pessoalmente não gosto de quase tudo que consultorias “ágeis” vendem hoje em dia, e detesto todo tipo de jargão moderno que tenta empacotar a idéia de agilidade em pacotes atraentes para empresas que não sabem o que estão fazendo. Lean Kanban, SAFe ou qualquer bobagem assim.

O episódio de hoje não é explicar Agilidade em detalhes ou fazer um checklist enorme de tudo relacionado ao assunto, mas explorar porque eu acho que os valores originais foram esquecidos e as práticas foram distorcidas e acabaram nos levando de volta ao ponto anterior, à época das tediosas metodologias monumentais dos anos 90. A camiseta que estou usando combina com o tom que vou usar pelo resto do episódio, então já fica aviso pros mais frágeis que o tom de hoje é Rated R.

(...)

Como sempre, vamos começar do começo. Todo mundo faz referências ao Manifesto, mas quantos de vocês sabem quem são os tais signatários, os autores originais do manifesto? Martin Fowler todo mundo conhece por outros livros como Refactoring e seus artigos muito bons publicados no seu site pessoal. Alistair Cockburn tinha sua metodologia Crystal Clear. O Jim Highsmith tinha sua metodologia Adaptive Software Development. Kent Beck e Brian Marick todo mundo associa com Test Driven Development. E no caso de Kent Beck, junto com Ward Cunningham e Ron Jeffries publicaram uma das coisas mais importantes pré mundo Ágil: as práticas de desenvolvimento de software de XP ou Extreme Programming. Se tem um único conjunto de métodos e práticas de desenvolvimento de software que você deveria estudar, é XP.

Ward Cunningham também criou a idéia original de Wiki e tem um dos wikis antigos mais interessantes: o C2, um dos lugares que eu sempre gostei de ler faz anos ou décadas. Dave Thomas e Andy Hunt a maioria deve conhecer tanto pelo livro The Pragmatic Programmer, que até hoje é uma das leituras que eu diria que é muito recomendada pra todo programador, mas também pela editora online PragProg que publicou muitos livros técnicos importantes.

Outro signatário foi Steve Mellor que alguns veteranos podem conhecer por causa do Shlaer-Mellor do fim dos anos 80, uma das muitas metodologias de análise e design de sistemas orientados a objetos. Eu não sei nada sobre Arie van Bennekum, James Greening ou Jon Kern, mas ainda temos Robert Martin, mais conhecido como Uncle Bob que adora falar de Clean Code até hoje. Mike Beedle, Ken Schwabber e Jeff Sutterland, que fundaram o famigerado Scrum, que pra muitos ainda é sinônimo de Agilidade.

Em 2010 eu tive a oportunidade de trocar idéia com o Bob Martin. Ele é muito eloquente e um excelente orador, é muito fácil ser convencido só pelo seu carisma. A mitologia do Manifesto Ágil é outra história que já se transformou numa lenda maior do que realmente foi. Todo mundo já ouviu a versão desses dezessete cavaleiros da agilidade, que se reuniram em Snowbird, Utah, numa távola redonda e de lá saíram as escrituras sagradas que definem a Igreja Dogmática da Agilidade!! Ohhhhhh Vamos ouvir dele como as coisas aconteceram.

(... trecho da história do agile …)

Provavelmente existem vídeos de outros agilistas mas esta palestra do Dave Thomas na conferência GOTO em 2015 é uma das melhores que já achei. Vamos ouvir o trecho dele explicando como surgiram com a estrutura do texto do manifesto em si:

(... trecho da história por Dave Thomas …)

Nessa palestra que vou deixar o link nas descrições, o Dave Thomas levanta um ponto importante. O Manifesto se chama “Manifesto do Desenvolvimento Ágil de Software”. Não é “Manifesto Ágil”. Falando dessa forma parece que o manifesto em si de alguma forma tem a “qualidade” de ser ágil, como ele diz, parece que o manifesto vai sair por aí dançando. Ágil não é um nome, é um adjetivo. Ou você é ágil ou não é. Não existe isso de “Você faz ágil?” ou “Estamos fazendo Ágil da forma correta?” Mas quem precisa vender alguma coisa, não tem como vender um adjetivo, e daí as consultorias, cursos, conferências, passaram a usar o termo como um nome, quase como se fosse possível dizer “quero levar meio quilo de Ágil, por favor, … pra viagem”

Eu não vou repetir o que o Dave Thomas explica nessa palestra, por isso recomendo que vocês assistam depois. Ele também escreveu um blog post um ano antes em 2014 na mesma época em que eu escrevi um artigo no meu blog: Ágil está morto, que vou deixar linkado nas descrições abaixo também. Muitos de nós já declaramos a “morte” do Ágil mas ele continua existindo nessa forma irreconhecível como um zumbi em The Walking Dead.

O que todo agilista de verdade vai concordar é que o termo foi prostituído. A beleza do Manifesto é sua simplicidade. Todos os profissionais de prática dessa área, com anos de experiência nas costas, consegue ler esse manifesto e se identificar. Ele descreve quatro simples valores que definem o que os agilistas chamam de “ser Ágil”. Seria algo similar às sete virtudes do Bushido, o caminho do Samurai. Ou os 10 mandamentos da Bíblia. O problema disso é que também é muito simples de distorcer a interpretação pra qualquer coisa que você quiser.

É como eu dizer “os 5 valores de ser saudável”. Ser saudável não é uma coisa que você compra numa prateleira. Ou um programa com 10 passos que você implementa em um mês e automaticamente você é saudável. Não adianta nada fazer almoços veganos e tomar multi vitamínicos e continuar comendo pizza na janta e enchendo a cara no fim de semana. Ser saudável é uma mudança completa em todos os seus hábitos e rotinas, todos os dias, e não alguma coisa que você liga e desliga quando quiser. A mesma coisa com honra. Não existe ser honrado de vez em quando. Mesma coisa com ser bom, não adianta roubar às terças e quintas e ser bom nas segundas e quartas. Ser ágil é a mesma coisa: é uma estilo de vida. Nunca um procedimento ou metodologia. Ou você é um profissional de desenvolvimento ágil de software ou não é.

Dado tempo suficiente, qualquer interpretação repetidas muitas vezes vai parecer mais verdadeira que a intenção original. Tome Scrum, por exemplo, por um período de tempo, talvez até hoje, muitos chegaram a associar sua implementação como o equivalente a se tornar Ágil. Scrum na verdade foi inventado nos anos 80, o guia sequer menciona o termo “ágil”. Mas era o texto mais simples e palatável para não-programadores.

Hoje em dia muitos adotam o tal de Kanban da mesma forma, que não se parece em nada com o Kanban original inventado na Toyota. As premissas e os fins são completamente diferentes, mas o nome é bem marketável, por isso é usado. Distorção dos objetivos originais funciona assim: muitos poderiam associar pagar uma contribuição pra Igreja como o equivalente a ter virtudes pra ir pro céu. Implementar as cerimônias do Scrum ou as técnicas de Kanban não torna ninguém automaticamente Ágil, da mesma forma como contribuir pra uma instituição não o torna Bom. E falando em Bom, o Bob Martin coloca muito bem porque o termo Ágil foi distorcido até se tornar irreconhecível:

(... trecho do Bom de Bob Martin …)

Se eu quiser ser mais extremo, não existe isso de metodologia Ágil. Existem métodos e práticas adotadas por profissionais ágeis. Não existe “implementar Ágil”. Existe “ser Ágil”. Toda consultoria que quer diferenciar seus produtos, e já percebeu que o valor da Agilidade vem caindo, está vendendo alguma variação com a etiqueta de Lean ou Kanban, ou simplesmente adotando práticas de outros corpos de conhecimento como PMI ou CMMI e re-etiquetando como parte de Ágil, ou então tentando inflacionar etiquetando alguma coisa como “Ágil Escalável” pra grandes organizações. Tudo isso é um monte de besteiras. Nas palavras de um signatário do Manifesto “Ágil”, o que significa ser ágil?

(... trecho dave thomas agil …)

Como Dave Thomas explica na palestra e como eu também venho explicando nos últimos quase 10 anos, agilidade é simplesmente andar um passo de cada vez, avaliar o resultado a cada passo, ajustar o caminho se o resultado não estiver de acordo com o esperado e repetir. Só isso, é exatamente o que devemos fazer pra atingir qualquer objetivo que não sabemos como atingir. Se quiser uma definição mais formal, é exatamente o que está na origem do Lean de verdade da engenharia de produção: o ciclo PDCA de Deming, que em resumo é planejar, executar, checar e agir. Todo processo cíclico deriva da mesma idéia. O DMAIC de Six Sigma é exatamente a mesma coisa: definir, medir, analisar, melhorar e controlar.

Todas as técnicas que se nascem desse ciclo de melhoria contínua, como um Kanban, fazem sentido para resolver os problemas encontrados especificamente na engenharia de produção da Toyota décadas atrás. Sob as premissas daquela época, sob as restrições daquele contexto, que são completamente diferentes de desenvolvimento de software do século XXI.

E falando em engenharia de produção. Parem, pelamordejesus, de querer associar técnicas de engenharia de fábricas em software. Vamos entender de uma vez por todas: software se chama soft porque não é hard. Hardware, equipamentos, coisas palpáveis, obedecem as leis da física. Tem peso. Sofrem a força da gravidade. Existem limites e restrições muito específicas sobre o que é possível ou não fazer com materiais físicos.

Todos os campos da engenharia foram sendo refinados pra lidar com isso. Fabricar carros, fabricar navios, construir prédios, produzir bens de consumo. É nesse tipo de ambiente que faz sentido falar em Kanban, Lean, Six Sigma. PQP Monte Carlo em estimativa de software não faz absolutamente nenhum sentido. De novo: só porque os números computam, não quer dizer que você tem qualquer resultado útil. É a mesma coisa com numerologia, só porque os números contam uma história que te agrada, não torna isso uma ciência. Isso não uma prova, não é sequer evidência de coisa nenhuma. É apenas coincidência e o resultado de torturar números pra contar a história que você quer ouvir.

Software, por definição, é uma abstração sem forma física e sem limites físicos. A mesma coisa vale pra literatura, música, pintura, qualquer coisa no campo criativo e das idéias. Software não tem peso, não tem cor, não tem cheiro, não obedece nenhuma lei da física a não ser o tempo de execução determinado pelo clock das CPUs ou GPUs. O principal: dois programadores diferentes tentando representar em código a mesma especificação da mesma tarefa vão chegar em códigos diferentes, gastando tempos muito diferentes. Diferente da engenharia do mundo real: duas construtoras diferentes, seguindo a mesma especificação, vão construir as mesmas casas dentro de custos e tempos muito similares, com qualidades similares, dadas as mesmas condições.

Mas no campo do soft, código, palavras, idéias, as diferenças são gritantes entre os melhores e os medíocres. Estamos falando em ordens de grandeza. Às vezes um medíocre pode precisar de dezenas de linhas e dezenas de horas pra representar uma tarefa em código, mas os melhores podem precisar de uma linha e alguns minutos. Não existe paralelo entre as técnicas de engenharia de produção e engenharia de software além de aspectos superficiais. Quanta gente já tentou gerenciar produtividade de escrever livros usando Lean Writing? Quantas vezes você já viu alguém controlando composição de música usando Kanban Songs? Soa ridículo? Lean Software soa tão ridículo quanto.

A parte bem conceitual de um Lean faz algum sentido: evitar desperdícios. Errar rápido. Mudar de direção quando faz sentido. Mas é só. As técnicas e métricas em si, não fazem sentido. Mas todos os que defendem isso tem números pra “provar” que as técnicas funcionam. Bobagem. A grande vantagem de usar números para explicar coisas pra pessoas não treinadas em matemática é que é muito fácil torturar números pra dizer qualquer coisa que você quiser. Qualquer um que lida com números sabe como é fácil mentir com números. Nós mesmos nos enganamos, muitas vezes sem querer. Mas sempre tente entender as condições sob as quais esses números foram colhidos, com que precisão? Com que procedimentos? Quão repetíveis são esses testes? Como são configurados os grupos de controle?

Repetindo: ter algo que parecem ser evidências não é nem de longe perto de ter provas. Não subestimem o método científico, não é assim que ciência funciona. Diferente de uma engenharia civil onde podemos teorizar, calcular, e provar fisicamente o funcionamento, o mesmo não pode ser dito de absolutamente nenhuma metodologia ágil. Ah, mas no Google funciona! Ah, mas os squads do Spotify funcionam! Ah mas na empresa X ou Y funciona. Bullshit. Sob quais condições? Quais premissas? A resposta mais óbvia é: dado 100 tech startups que surgiram e morreram, uma sobreviveu chamada Spotify, que por acaso tem elementos que parecem uma metodologia. Mas não foi por causa dela.

Vamos estabelecer o seguinte: contratar uma consultoria que vende implementação ou coaching de metodologias ágeis costuma ser perda de tempo na maioria das vezes. Mas por que parece que dá resultados? Se partirmos do princípio que quem você contratou de fato sabe como fazer software. Seria como contratar um personal trainer. Só funciona se você vai realmente continuar a disciplina mesmo quando ele não estiver presente. O problema é que o treinamento não é só de uma pessoa, normalmente é uma equipe inteira, e o ideal seria a empresa inteira. Depois que o coach ou consultoria for embora, o normal é a maioria das pessoas voltar a fazer as mesmas coisas erradas que faziam antes. E por isso todo mundo vende metodologias e processos de controle, e cerimônias, e algum jeito de medir se todo mundo está seguindo esses procedimentos.

Releia o Manifesto: “Indivíduos e Iterações em vez de processos e ferramentas”. É o primeiro valor do Manifesto. E qual é a primeira coisa que você pensa em fazer? Contratar uma consultoria externa com indivíduos que não fazem parte do seu time, para implementar processos e ferramentas e mecanismos de controle. É basicamente a pessoa que entra em dieta toda segunda feira, perde peso por algumas semanas, posta no LinkedIn “olha só como minha dieta funciona” mas no mês seguinte ganhou mais peso do que tinha perdido, e ninguém mais ouve falar dele.

A principal razão de porque as pessoas e as empresas fazem isso, é o mesmo motivo de sempre: medo e insegurança. Ninguém gosta de estar errado. Um medroso ou medíocre sempre vai terceirizar a decisão pra algum expert ou coach ou mentor ou consultoria, dessa forma ninguém pode dizer que ele estava errado já que ele fez o que todo mundo está dizendo pra fazer: “implementar Ágil”. Hoje em dia você tem essa atitude esnobe de “como assim você não faz ágil?”

Pras consultorias ou coaches, a situação é muito fácil. É que nem um personal trainer que precisa trabalhar pouco, e colocar um espelho convexo pra parecer que está dando resultados. E, principalmente, não ficar muito tempo. Implementar, colocar alguma numerologia que cospe algum número interessante pra parecer que existe controle, e ir embora. Assim você tem o benefício de ter um “caso de sucesso”, já que tem post numa rede social deve ser verdade. E se depois de implementado os resultados começarem a desaparecer - como obviamente vão - eles sempre podem dizer “olha só, vocês não continuaram o processo que implementamos, por isso tá dando errado, mas tudo bem, só contratar a gente de novo que ajustamos”. E bem vindo ao mercado de ganhar dinheiro em troca de pouco ou nenhum resultado. É o melhor tipo de indústria que tem. Você pega qualquer coisa que parece um resultado e faz posts no Linkedin como se fosse salvador da pátria e esconde embaixo do tapete tudo que não mostrou resultado.

Por que as pessoas querem uma boa definição formal do que é uma especificação de software, seja na forma de user stories ou qualquer coisa assim? Porque a maioria não tem saco pra escrever, e quem vai programar não tem saco pra ler. Então perde-se mais tempo discutindo o formato e o protocolo do que em realmente discutindo o que precisa ser feito. Já que ninguém senta e chega num consenso do que é uma boa produtividade, vamos discutir lead time, vamos discutir story points, ou inutilidades como WIP, não importa, qualquer numerologia arbitrária em vez de realmente entender quais são os fatores que estão afetando a produtividade naquele contexto em particular, como a indecisão dos donos da empresa ou o acúmulo de más contratações que já deveriam ter sido demitidos.

Ou então, já que programador odeia ter que estimar, vamos associar agilidade com não-estimar, o estúpido movimento de hashtag #noEstimates. Essa acho que tentou pegar tração mas não foi muito pra frente. Eu diria que se um programador gosta de no-estimates pros projetos devia tentar vender isso junto com no-estimates pra ser pago também, acho justo. E é verdade, eu já discuti bastante porque eu acho que estimar é uma tarefa ingrata mas necessária, e não ajuda nada que os clientes tem uma grande dificuldade de entender como as coisas são feitas e quais são as expectativas reais. Mas pelo menos todo não-programador sabe que se perguntar a um programador quanto tempo alguma coisa leva pra alguma ficar pronto, e ele responder, 200 horas, você sabe que em 200 horas seja lá o que foi pedido, vai estar pela metade. É quase uma lei universal isso.

E é exatamente o tipo de problema que a idéia de agilidade original quer tentar resolver. A idéia de parar de brincar de “vamos fingir que você não tá me enganando e vamos fingir que eu tô acreditando”. Agilidade não tem nada a ver com implementar uma ferramenta, ou um processo, ou uma forma mensurável e numérica de controle. Agilidade é fazer o que precisa ser feito para atingir os objetivos concordados, sem meias palavras, com as expectativas alinhadas.

O programador promete o que tem segurança que consegue entregar e a discussão é se isso é o suficiente e se vale a pena pro cliente, e o cliente entender que se não cabe tudo, ele precisa decidir se metade é suficiente. Ou então ninguém nem começa e o projeto deve ser abandonado. E sim, não começar algo inseguro demais é ser ágil, é evitar desperdícios. Eu tô cansado de recusar projetos porque eu acho que tá tão mal definido que vai me dar mais dor de cabeça do que vale a pena. Prefiro ser claro e dizer: volta pra casa, o dia que você souber o que quer, daí me procura.

Eu falei disso no episódio sobre Priorização que você pode assistir linkado acima. A muleta das metodologias fica mais preocupada em ver o formato em que determinada requisição deve ser feita, se no formato de stories, cadastrado em qual ferramenta, como marcar quem fica responsável e bla bla bla em vez de se perguntar se a tal requisição sequer deveria existir, pra começo de conversa.

Quantas vezes alguém já chegou em mim pra um projeto pequeno e fala coisa do tipo, “ah, e deve ser fácil, mas eu queria ver se dava pra colocar um mensageiro no meu MVP, tipo o Messenger do Facebook, sabe?” e eu digo, “ah, claro, dá pra fazer sim, você vai pagar o mesmo que o Facebook pagou pra fazer o deles?” E esses dias alguém me pergunta “ah, vocês conseguem fazer um Spotify?” ah vá, nem vou perder tempo.

Hoje em dia em grandes empresas, com montes de equipes ou “squads”, gasta-se todo o tempo enchendo os backlogs ou kanbans de coisas pra fazer e depois comemoram porque o “throughput” desse sprint foi maior do que o do sprint anterior, ou porque conseguiram diminuir o “lead time”, ou porque o tamanho dos “lotes” estabilizou, Quanto bullshit.

O que deveria importar é: o software planejado que tinha mais valor foi entregue ou não? Qual foi o retorno do investimento? Qual é o custo total de propriedade do software atual? Essas coisas ninguém responde. Por isso a coisa mais importante que é não desperdiçar, ninguém sabe se tá sendo cumprido ou não. Na maior parte do tempo, numa empresa com sei lá, umas 10 equipes desenvolvendo software, eu poderia olhar por 1 dia e decidir cortar umas 2 equipes inteiras, ou mais, e não impactaria o negócio como um todo. Muitas vezes os que deveriam tomar as decisões têm essa noção, mas poucos tem o culhão de cortar, e assim vai queimando mais recursos do que deveria produzindo software que não traz valor pro negócio enquanto outras partes que poderiam estar sendo priorizadas sofrem ficando pra trás.

Toda a idéia de alguém nível C numa empresa, um CEO, um CTO, um COO, é tomar as decisões difíceis que funcionários não podem tomar. Não tem como saber com 100% de certeza qual decisão é a melhor. Mas a pior coisa é não tomar nenhuma. Sempre parece mais fácil se esconder atrás de alguma muleta, tipo um Scaled Agile Framework. Daí dizer pros investidores como isso vai melhorar as métricas X ou Y da empresa, mas vai levar um ou dois anos pra implementar. E literalmente coçar o saco se escondendo atrás disso em vez de estar fazendo o que deveria: tomando uma decisão de verdade.

Na vida real, tudo é um risco. Em particular gestão de projetos é em boa parte gestão de riscos. Gerenciar riscos não significa evitar todos os riscos. Pense como contratar um seguro. Não existe zero riscos a zero reais. Quanto mais você quiser de cobertura, mais caro vai ser o seguro e mais caro a franquia se precisar usar. Pior do que tentar economizar não fazendo seguro é achar que é possível ter cobertura completa. Se você tentar garantir 100% de certeza mais você vai errar. É melhor assumir uma perda pequena: a realidade é que você não tem como acertar 100%, então pare de tentar.

Se for pra micro gerenciar um projeto, adicionando o máximo de ferramentas e procedimentos de controle, é melhor nem começar. Num pensamento assim seria como eu dizer, “cara, você quer ter certeza que seu carro não vai bater? É muito simples, pra que complicar? Simplesmente não tire seu carro da garagem, nunca, assim você tem 100% de certeza”. Ou então quem quer ter 100% de segurança de que seus computadores não vão ser invadidos por hackers? Simples, nunca ligue na internet, fique offline o tempo todo. Mas obviamente nada disso é prático. É quando você é obrigado a assumir algum risco. Entenderam? Ter controle de verdade e “achar” que tem controle são duas completamente diferentes.

A partir do momento que você decide que precisa fazer mais do que simplesmente enfiar a cabeça num buraco e esperar o problema passar sozinho, significa que você precisa assumir o controle de verdade e assumir riscos. Bem vindo à vida de um adulto. Significa também que você precisa assumir o prejuízo de uma decisão errada. E novamente, sinto muito te dizer, você vai tomar decisões erradas, muitas, incluindo contratar funcionários ou fornecedores errados ou começar projetos que nem precisava. Em vez de apostar tudo que você tem numa opção só, é melhor fazer planejar um ponto de corte, testando por um curto período de tempo. E se não gostar, cortar o quanto antes. Simples assim. Nada disso é garantir que uma hora você vai chegar onde precisa, mas pelo menos vai ter mais chances de tentar.

Ser ágil é ser adulto. Você nunca vai transformar uma equipe de crianças mimizentas em adultos via metodologia. Todo moleque adora falar em querer ter mais autonomia, que é uma das coisas que parece que as metodologias ditas ágeis parecem fornecer, dado que a ênfase era ser mais leve e principalmente pela interpretação errada do primeiro valor do manifesto, que diz valorizar pessoas mais que processos. Não é isso que o valor quer dizer. Ela basicamente diz o óbvio: em vez de apostar num monte de micro controles pra gerenciar um projeto de software, partindo da premissa que todos os envolvidos são adultos, deixe os adultos discutirem (interações) e resolver o problema, como adultos.

Todo mundo quer ter autonomia. Mas só na hora que interessa. Autonomia é a junção de ter liberdade e assumir as consequências dessas liberdade. Não existe ter liberdade e não ter nenhum ônus, isso seria só libertinagem. Em inglês existe uma palavra pra isso: accountability. Não existe uma palavra equivalente em português, o mais próximo seria ser responsável mas é mais do que isso. Veja a definição da Wikipedia. Mas a idéia de ser ágil é ser accountable, não só ter mais liberdades. A diferença de alguém que faz alguma coisa por hobby e alguém que faz alguma coisa por profissão é que um profissional é accountable. Bem vindo ao mundo dos adultos.

Agora, se esconder atrás da metodologia dizendo bobagens como “puxa, não sei porque as coisas não estão andando como vocês querem, eu preenchi os tickets do Jira como vocês pediram, eu participei dos sprint plannings, e mesmo assim vocês não tão satisfeitos? Bom, não é minha culpa porque eu fiz tudo que pediram”. E isso é exatamente a mesma coisa que eu ouvia desde antes da era da Agilidade e é o oposto de accountability. Todo mundo que não quer trabalhar ou que vai seguir a lei do menor esforço, sempre vai se esconder atrás de todos os procedimentos que você der. Pois bem, tire todas essas muletas, e o que sobra é o trabalho em si. E agora você pode realmente medir as pessoas pelo mérito do trabalho individual de cada um.

Quando um manifesto é interpretado do jeito que todo mundo quiser, cada um pra justificar sua falta de profissionalismo, ele não tem nenhum valor. É como a Bíblia, se você vai ficar pegando pedaços pra ler ao pé da letra e outras vezes vai ignorar algumas partes ou interpretar diferente, pra que serve a Bíblia? Uma pessoa boa é boa com ou sem estrutura religiosa. Justificar uma atitude porque “está escrito na Bíblia” é o tipo de muleta sem sentido que não precisamos, especialmente no mundo profissional.

Você é dono ou gestor de uma empresa e quer uma equipe ágil? Pare de contratar crianças e tratar todo mundo como criança. Quando você quer adultos, você trata as pessoas como adultos. Em vez de fazer telefone sem fio, diga exatamente o que quer, e deixe os profissionais trabalharem. Os profissionais foram abaixo das expectativas? Mande embora. Agora, se você está no mercado de babá de programador, coloque metodologias ágeis.

Aos programadores, vocês querem ser profissionais? Sejam accountables. É o que eu comecei falando no episódio anterior. Ao usar as falhas óbvias nas metodologias pra esconder sua falta de vontade de trabalhar você está ocupando o lugar de outra pessoa que quer trabalhar e está desrespeitando sua profissão. Como eu disse no começo, se você é iniciante, as práticas que realmente interessam começam no Extreme Programming e não no Scrum. Test Driven Development, Small Releases, Continuous Integration, Simple Design, Collective Code Ownership (que é hoje em dia é basicamente Git) e práticas desse tipo é o que realmente se faz. Brincar de cartas pra estimar tarefas ou ficar usando chapéu em círculos pra dar vez pra se falar é o tipo de cerimônia ridícula que não faz nenhum sentido. O que é isso? Uma empresa de desenvolvimento de software ou uma creche? Daqui a pouco vamos estar todos abraçando árvores.

(... bob martin outros agilistas …)

Os reais objetivos e modo de pensar e agir de um agilista não é complicado. Não é pra ser complicado. Todos os corpos de conhecimentos em gestão de projetos, seja PMI, Lean, Scrum ou qualquer coisa não são necessariamente inválidos, não me entenda mal, são técnicas, meras ferramentas. Assim como uma caixa de ferramentas sozinha não constrói uma casa, você pode usar a combinação de ferramentas que quiser. Mas não acredite que existe um martelo especial que vai construir a casa melhor pra você automaticamente, isso não existe.

Em particular, você quer saber o que é agilidade? Leia os autores originais, os dezessete originais que o Bob Martin falou. Não precisa de mais do que isso. O foco não são as ferramentas, não são as cerimônias, nada disso. O objetivo não é atingir determinado número em algum gráfico arbitrário: é entregar valor. Se o que mais importa no seu projeto é atingir números, você precisa questionar seus valores.

Eu posso estender esse assunto ad eternum mas vou parar por aqui, já reclamei demais pra um episódio só. Mas a mensagem que precisa ficar clara: o estado atual do ecossistema de agilidade está poluído de prioridades erradas. Não é a toa que todo mundo se sinta tão confuso que precisa contratar um guru qualquer pra dizer o que fazer. Simplifique: jogue tudo fora, pare de ouvir os outros e simplesmente olhe pro seu problema de forma objetiva, sem achar que está fazendo errado por não estar usando algum novo jargão da moda.

No final das contas, claro que eu estou sendo hiperbólico pra salientar meu ponto. O dinheiro é seu e eu realmente acho que cada um deve fazer com seu próprio dinheiro o que bem entender, inclusive queimar ou jogar fora. E muita gente realmente gosta da presença de um guru ou coach, é que nem uma terapia, faz se sentir melhor e dormir melhor a noite. E ei, tudo isso é um motivo muito válido. Porém, se você acha que existe realmente alguma ciência por trás disso que você está comprando, pode parar. Se você tem uma doença séria e estão te vendendo homeopáticos, me desculpe, tirando o efeito placebo, você não vai se curar. Procure medicina de verdade. A era New Age já acabou, e ninguém vai sentir falta.

Vocês tem dúvidas específicas sobre esse tema? Talvez valha um episódio de repescagem pra responder perguntas específicas, então mandem suas dúvidas nos comentários abaixo pra eu saber. Se curtiram o vídeo mandem um joinha, assinem o canal e cliquem no sininho pra não perder os próximos episódios e compartilhem o video com seus amigos pra ajudar o canal. A gente se vê na próxima, até mais!