ENXUGANDO A MÁQUINA

ISMAEL MELO & PAULO CAROLI

Enxugando a Máquina

Lean MVP & Pontos de Função

Ismael Melo e Paulo Caroli

Esse livro está à venda em
http://leanpub.com/enxugando-a-maquina

Essa versão foi publicada em 2017-08-24

Conteúdo

Agradecimentos

Aos familiares, amigos e colegas de trabalho, e todos aqueles que de alguma forma contribuíram com *insights*, dicas e/ou experiências sobre o tema abordado, o nosso agradecimento. Esta realização não teria sido possível sem o apoio de vocês.

Prefácio

O uso de metodologias ágeis com pontos de função é um assunto recorrente e alvo de muitas discussões entre agilistas, especialistas em métricas de *software* e gestores de TI, principalmente nos últimos anos.

Com o intuito de esclarecermos que a convivência entre os métodos é viável, sendo possível inclusive potencializar suas virtudes, nós decidimos publicar este livro. E sinceramente esperamos que ele sirva como um "caminho das pedras" para uma abordagem de serviços enxutos e mensuráveis.

Nossa proposição foi construída com base em princípios de colaboração e valorização das pessoas, tão presentes na filosofia ágil e enxuta, mas sem prejuízo à capacidade de monitoramento e impessoalidade nas contratações oriundas dos conceitos tradicionais de governança de TI. Desta forma, o controle e a burocracia são vislumbrados apenas em um nível considerado suficiente e necessário, com foco na eliminação de desperdícios através da garantia de visibilidade e transparência dos custos envolvidos no planejamento, execução e gestão do ciclo de desenvolvimento de *software*.

Nós pretendemos que esta publicação desperte o interesse das comunidades de usuários dos temas abordados e que ela realmente seja útil às organizações que usam e/ou contratam serviços de desenvolvimento de *software* ágil e enxuto, de qualquer natureza ou porte.

Bom, agora você poderá começar a leitura e preparar-se para acompanhar e/ou contratar seus projetos enxutos com segurança, transparência e eficácia. E isso com o menor impacto possível na essência da metodologia ágil!

Introdução

Neste livro, você será apresentado a um novo modelo de referência para o desenvolvimento e contratação de serviços de *software*, utilizando uma proposta de trabalho mais ágil, colaborativa, enxuta e mensurável. Esperamos que você aprecie este estudo e também que consiga efetivamente utilizá-lo no seu dia-a-dia.

Primeiramente, para alcançarmos nosso objetivo de auxiliá-lo a fazer mais e melhor com menos, nós revisaremos aguns conceitos e técnicas relacionadas ao *Design Thinking*, *Lean Thinking* e principalmente ao *Lean Startup*, métodos hoje muito utilizados na indústria de manufatura tradicional, mas que também têm se destacado ao orientar o desenvolvimento de projetos de *software*, facilitando a descoberta, construção e entrega de produtos e serviços de alto valor para o negócio e menor desperdício na execução do processo produtivo.

Na nossa visão (e é isso que pretendemos expor), estamos retratando um cenário em que a utilização de métricas de *software*, em especial pontos de função, pode suportar muito bem o monitoramento do processo do desenvolvimento, tornando a operação enxuta mais transparente, eficaz e eficiente:

- Transparente, devido a maior tangibilidade, previsibilidade e comparabilidade das contratações e/ou demandas realizadas;
- Eficaz, devido à obtenção de resultados de forma condizente com a expectativa dos seus usuários, e também consistente durante a evolução do produto sendo desenvolvido; e
- Eficiente, pela capacidade de monitoramento e acompanhamento dos resultados com vistas ao melhor uso dos recursos, ao tratar as fontes de desperdício.

Salientamos ainda, que esta obra considera o conhecimento adquirido e documentado no livro "Direto ao Ponto", escrito pelo

Paulo Caroli, vinculando-o a uma série de conceitos relacionados à contratação e gestão do desenvolvimento de produtos de *software*. Assim, temos um estudo bastante rico e abrangente, que de alguma forma tem se mostrado muito proveitoso às organizações as quais temos apoiado. Esperamos que ele seja útil na resolução de alguns de seus problemas e que você também possa nos fornecer *feedback* tão logo seja possível!

Quem deve ler este livro?

O conteúdo deste livro foi escrito para contextualizar, facilitar e nivelar o entendimento sobre o uso de metodologias ágeis e enxutas por gestores e executivos de TI, consultores e especialistas em desenvolvimento de *software*, principalmente aqueles que trabalham com isso e/ou entendem o contexto abordado, mesmo que superficialmente.

Esperamos que a sua experiência de leitura seja agradável e habilite-o como um agente de mudança cultural na sua organização, com vistas a auxiliá-la a entregar mais, melhor e de forma antecipada, otimizando a visão de progresso e custos a partir de direcionadores básicos de eficiência.

O que ele aborda?

Esta obra pode ser considerada um estudo prático-teórico, estruturado em 7 capítulos e cuja finalidade é possibilitar o entendimento necessário aos seus leitores para o desenvolvimento e/ou contratação de projetos de *software* cujo desenvolvimento seja iterativo e incremental em uma perspectiva ágil e enxuta.

Já no Capítulo 1 você será contextualizado sobre a filosofia enxuta, assim como entenderá alguns conceitos e métodos relacionados à cocriação. E no Capítulo 2, Paulo Caroli lhe apresentará uma visão enxuta de desenvolvimento de *software* utilizando a abordagem de MVP.

No Capítulo 3 você verá os métodos de trabalho mais conhecidos e utilizados para desenvolver *software*. Eles foram ilustrados com o propósito de nivelar o conhecimento dos utilizadores deste guia.

O Capítulo 4 apresentará opções de gráficos e outras representações visuais típicas de acompanhamento para gestão de projetos ágeis. E nos Capítulos 5 e 6 você será contextualizado por Ismael Melo sobre as principais métricas de tamanho de *software* e estimativas derivadas, assim como conseguirá visualizar o funcionamento do modelo de contratações adotado pelo Governo Brasileiro, que usa este e outros artíficios como um meio para prover transparência de custos e qualidade.

O capítulo 7, por sua vez, apresentará um *framework* simplificado para acompanhamento e/ou contratação de projetos ágeis com o suporte de um modelo de estimativa e medição em pontos de função, propiciando assim uma perspectiva diferenciada para utilização de uma metodologia iterativo-incremental com a abordagem de MVP.

Como aproveitar bem o seu conteúdo?

Para um melhor proveito do conteúdo deste livro, sugerimos que o leitor compreenda completamente os conceitos apresentados nos capítulos iniciais, o que também facilitará o entendimento da proposta que estamos apresentando.

Ao final da publicação há uma lista de "Termos e Referências" relacionados aos assuntos abordados, especialmente elaborada para aqueles que não possuem muita intimidade com eles. Use caso necessário.

1. Cocriação & Pensamento Enxuto

As práticas colaborativas e participativas para o desenvolvimento de novos negócios e produtos ampliam fronteiras sobre como pensamos nas escolhas das soluções mais inovadoras, proporcionando transformar algumas ideias inéditas em novos negócios (PRATES, 2011). Todavia, a cultura organizacional liga-se intimamente à maneira como as decisões são tomadas, fazendo com que isso seja um dos principais fatores viabilizadores da tomada de seus benefícios (BORIA, 2011, Pág. 43).

BORIA (2011) defende que as decisões ou são tomadas pelas posições mais altas em detrimento da opinião daqueles que efetivamente atuam na operação (devido à colaboração ou independência) ou são delegadas, para que todos possam contribuir com o direcionamento estratégico (pela hierarquia ou competências).

É com base nesta afirmação de BORIA (2011), representada pela antítese da Figura 1, que serão abordados a seguir os conceitos relacionados a algumas práticas de cocriação e aculturamento para o desenvolvimento de produtos enxutos, fortemente estruturados sob a perspectiva de métodos de trabalho mais colaborativos, criativos e participativos, em detrimento aos meios tradicionais e com os quais nos habituamos.

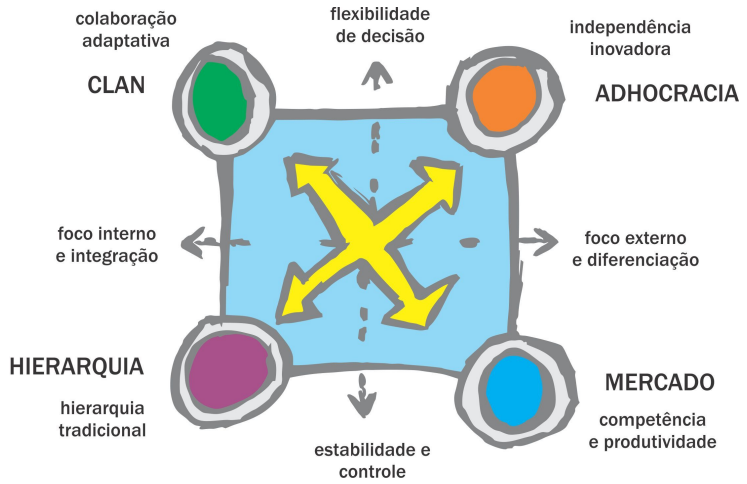


Figura 1 – Antítese e a Cultura Organizacional

Ao longo desta obra, de acordo com o mapa ilustrado acima, trabalharemos com a premissa de uma abordagem que fomenta a flexibilidade de decisão. Ainda, enquanto que na perspectiva interna, valoriza-se a colaboração adaptativa, externamente dá-se valor à independência inovadora, o que por sua vez viabiliza a inovação e diferenciação dos produtos e serviços providos pela organização.

Na proposta deste livro, contudo, cabe destacar que entendemos como viável uma perspectiva mista de gestão, para prover estabilidade e controle apenas no nível necessário, principalmente para análises de tendências, custos e/ou gestão de produtividade, ou ainda para acompanhamento dos projetos e produtos de *software*, assuntos que serão tratados nos próximos capítulos.

1.1 *Design Thinking*

O "pensamento de designer" ou *Design Thinking* (em inglês), é um processo colaborativo de pensamento crítico e criativo que permite organizar informações e ideias, para tomar decisões, apri-

morar situações e adquirir conhecimento (VIANNA, 2012). Sendo assim, pode-se dizer que ele é um método que trabalha essencialmente a cocriação e utiliza algumas práticas valiosas de descoberta. Desta forma, e até mesmo por possibilitar a coleta de resultados imediatos a partir de ações simples, acaba por auxiliar também no aculturamento da empresa.

Como filosofia, o *Design Thinking* tem também o objetivo de promover o bem-estar na vida das pessoas, com base no comportamento do consumidor de um produto, serviço ou informação, considerando suas experiências (PRATES, 2015). A sua utilização abrange diferentes ramos da indústria, podendo ser considerada também uma ferramenta estratégica que incentiva a criação conjunta a partir de uma visão por processos (Figura 2). E nesta perspectiva, o processo de *Design Thinking* é definido, conforme as seguintes fases (VIANNA, 2012):

- Imersão (entendimento) - envolve a contextualização do projeto;
- Ideação (criação) - tem o intuito de gerar ideias inovadoras para o tema de projeto;
- Prototipação (teste) - tem como função validar hipóteses a partir da formulação de questões, criação de protótipos, testes, avaliações e conclusão;
- Desenvolvimento (aplicação) - o próprio uso do conhecimento adquirido através do processo criativo.

O *Design Thinking* é um método bastante eficaz e presente nas organizações que buscam realmente entender, avaliar e implementar o que os seus clientes e/ou *stakeholders* pensam sobre seus produtos e serviços. Veja a seguir, um resumo sobre como cada fase funciona.

1.1.1 (1) Imersão

A fase de imersão é realizada a partir de três etapas (sendo a última opcional):

- "Imersão Preliminar" - etapa ou subdivisão da fase de Imersão que visa o entendimento inicial do problema em foco e caso necessário, o seu reenquadramento, podendo ser executada também através de uma "Pesquisa Exploratória" ou "Pesquisa *Desk*";
- "Imersão em Profundidade" - etapa onde são identificadas as necessidades dos *stakeholders*, assim como as prováveis oportunidades que possam emergir do entendimento de suas experiências, utilizando técnicas tais como "Entrevistas", "Cadernos de Sensibilização", "Sessões Generativas", técnica "Um Dia na Vida" e técnica "Sombra";
- "Análise e Síntese" - consolidação visual do conteúdo, identificando padrões de contexto e estabelecendo ferramentas de apoio, tais como: "Cartões de *Insights*", "Diagrama de Afinidades", "Mapa Conceitual", definição de critérios norteadores, identificação de "Personas", "Mapa de Empatia", "Jornada do Usuário" e "*Blueprint*".

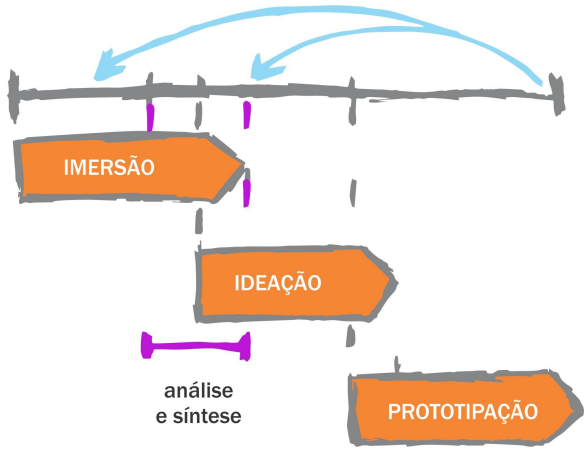


Figura 2 – *Design Thinking* como um Processo

Não por acaso, você perceberá nos próximos capítulos que utilizaremos diversos dos conceitos utilizados nesta fase, principal-

mente ao aplicar o método "DiretoAoPonto" no ciclo de desenvolvimento de *software*. Desta forma, não estamos apenas pensando no aculturamento ou nos aspectos da cocriação, mas numa proposta consistente construída sobre um único paradigma.

1.1.2 (2) Ideação

A ideação objetiva gerar ideias inovadoras utilizando-se de ferramentas de síntese, aplicadas às informações oriundas da etapa de "Análise e Síntese". Ela estimula a criatividade e gera soluções conforme um contexto trabalhado.

São algumas das ferramentas utilizadas para fomentar a "Ideação": o "*Brainstorm*", o "*Workshop* de Cocriação", o "Cardápio de Ideias" e a "Matriz de Posicionamento".

Não entraremos no mérito do que é e de como aplicar as técnicas supracitadas, sendo que para isso existem outras publicações já bastante exploradas e conceituadas, de fácil referência na internet ou bibliotecas.

1.1.3 (3) Prototipação

A prototipação tem como principal função validar um conjunto de hipóteses a partir da formulação de questões, criação de protótipos, testes, avaliações e conclusão, podendo ocorrer concomitantemente com as fases de "Imersão e Ideação". E isso proporciona a seleção e refinamento de ideias, tangibilizando-as, avaliando-as e validando-as interativamente junto a uma amostra do público. Assim, pode-se antecipar a visibilidade sobre eventuais gargalos e problemas, o que tende a contribuir para a redução de riscos e otimização de custos relacionados.

Na aplicação tradicional do *Design Thinking*, a prototipação normalmente utiliza-se de recursos rudimentares, tais como "Protótipos de Papel", "Modelos de Volume", "Encenação" ou "*Storyboards*", "Protótipos de Serviços", entre outros.

Você perceberá também, que de certa forma, esta fase é análoga

à aplicação do *Lean Startup* e a estratégia de MVPs, conceitos que serão abordados a seguir.

1.1.4 Desenvolvimento

O desenvolvimento não é uma etapa formal do método, mas o momento dedicado ao uso do conhecimento adquirido por meio do processo criativo para transformar as soluções identificadas em negócios, estratégias e/ou oportunidades concretas. Ou seja, é o ponto em que o processo de *Design Thinking* foi encerrado já possibilitando desenvolver o serviço, produto ou ação em questão, e portanto, viabilizando assim a coleta dos resultados esperados.

1.2 *Lean Thinking*

A indústria automobilística foi o berço da filosofia *Lean*, como uma alternativa proposta por Kiichiro Toyoda e posteriormente implementada por Taiichi Ohno (ambos funcionários da empresa Toyota, no Japão pós guerra), ao modelo de produção em massa das montadoras norte-americanas, cuja estratégia fundamentava-se sobre um modelo de ganhos de escala. O grande diferencial desta nova abordagem era que todas as peças deveriam chegar à linha de montagem somente no momento da sua utilização (*Just-in-Time*), ou seja, a sua produção ocorreria pouco antes de serem efetivamente necessárias.

Difundido mais tarde para outros setores da economia, o *Lean* também ficou conhecido como *Lean Manufacturing*, ou ainda "Sistema Toyota de Produção" (o título do livro lançado por Taiichi Ohno). Ele remete à obtenção dos materiais corretos, no local correto, na quantidade correta e com o mínimo desperdício, para manter a flexibilidade e a adaptabilidade do sistema produtivo. Note assim, que o foco está na otimização do processo e não propriamente na capacidade total de produção.

Neste contexto, podem ser destacados como pontos chave do *Lean Manufacturing*:

- Qualidade total imediata – atuação na detecção e eliminação de defeitos, com a solução dos problemas na sua origem;
- Minimização do desperdício – eliminação de tarefas que não acrescentam valor para o negócio e/ou levam à insatisfação do cliente e demais *stakeholders* (isto é, aquelas que possuem relação com recursos usados indevidamente e que contribuem para o aumento de custos e tempo);
- Melhoria contínua – busca contínua por redução de custos, melhoria da qualidade, aumento da produtividade e compartilhamento da informação;
- Processos *pull* – os produtos devem ser retirados pelo cliente final e não empurrados para o fim da cadeia de produção;
- Flexibilidade – produção rápida de diferentes lotes e com uma grande variedade de produtos, sem comprometimento da eficiência esperada do processo;
- Relação de longo prazo com os fornecedores – tomam-se acordos para compartilhar o risco, custos e a informação.

Atualmente em voga, o termo "Pensamento Enxuto", em inglês "*Lean Thinking*", é derivado do conceito "*Lean Manufacturing*", sendo estes muitas vezes citados como sinônimos. Ele foi usado pela primeira vez por James Womack e Daniel Jones, na obra de mesmo nome (WOMACK, 2004), e desde então é utilizado para referir-se à filosofia de liderança e gestão que tem como objetivo a eliminação do desperdício e a criação de valor.

As características centrais do *Lean Thinking* são elencadas abaixo:

- Organização em equipes, envolvendo pessoas flexíveis, com múltipla formação, com elevada autonomia e responsabilidade nas suas áreas de trabalho;
- Estruturas de resolução de problemas ao nível das áreas de trabalho, em sintonia com uma cultura de melhoria contínua;
- Operações *Lean*, o que leva os problemas a revelarem-se, e a serem posteriormente corrigidos;

- Políticas de recursos humanos voltadas aos valores e a capacidade de comprometimento, encorajando sentimentos de pertença, partilha e de dignidade;
- Relações de grande proximidade com fornecedores;
- Equipes de desenvolvimento multifuncionais;
- Grande proximidade com o cliente.

Dentre os princípios do "pensamento enxuto" podem ser destacados (POPPENDIECK; POPPENDIECK, 2011), portanto:

- Eliminação de desperdícios;
- Respeito às pessoas;
- Busca pela qualidade;
- Busca pela simplicidade;
- Aperfeiçoamento do todo; e
- Entregas rápidas.

1.2.1 *Lean Startup*

Conceito derivado da filosofia *Lean* e bastante utilizado na indústria tradicional, assim como atualmente também na indústria de *software*, o *Lean Startup*, ou "*Startup* Enxuta" em português, é uma abordagem que proporciona a idealização de produtos e serviços com foco na geração de aprendizado sobre mercado, produto e cliente, a partir da validação de hipóteses.

No *Lean Startup*, as hipóteses são formadas por crenças construídas com base na experiência e observações dos envolvidos, e que devem ser validadas para serem transformadas em fatos (SABBAGH, 2013).

O método promove um novo jeito de pensar e de construir produtos inovadores para negócios sustentáveis, a partir da identificação e eliminação sistemática de desperdícios e de uma combinação de ideias relacionadas a *marketing*, tecnologia e gestão (RIES, 2012). Características as quais, residem em cinco princípios básicos:

- Empreendedores estão em toda parte – aplicação independente de especificidades ou estereótipos, e especialmente voltada para cenários de extrema incerteza;
- Empreendedorismo é administração – gestão focada na extrema incerteza, como uma instituição a ser gerida;
- Aprendizagem validada – busca pela validação científica dos experimentos, podendo testar cada elemento do modelo de negócios;
- Construir-medir-aprender – foco na transformação de ideias em produtos, medindo como os consumidores reagem e decidindo sobre mudar ou perseverar;
- Contabilidade para inovação – busca por melhores resultados e atribuição de responsabilidade aos empreendedores, atentando para a medição de progresso, definição de marcos e priorização de trabalho.

Para a construção de produtos de *software*, o *Lean Startup*, segundo SABBAGH (2013), mostra-se muito útil, pois considera um laço de aprendizagem com ênfase em etapas de aprendizado, construção e medição (Figura 3), permitindo assim validar hipóteses, potencializadas através do conceito de produto mínimo viável (MVP, do inglês: *Minimum Viable Product*).

Um MVP é a versão mais simples de um produto que pode ser lançada com uma quantidade mínima de esforço e tempo de desenvolvimento, em um processo de aprendizagem contínuo e que possibilita o teste de hipóteses fundamentais ao negócio (RIES, 2011, págs. 77 e 93).

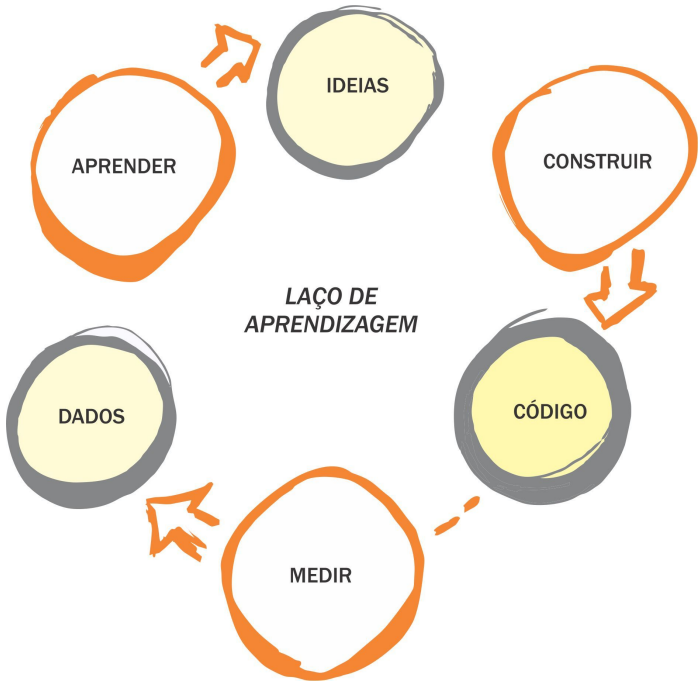


Figura 3 – Laço de Aprendizagem no *Lean Startup*

1.2.2 *Lean Kanban*

Kanban é um método formulado por David J. Anderson para gestão do fluxo de trabalho de um processo incremental e evolutivo. Influenciado pelo modelo Toyota (*Just-In-Time*), o método possibilita visualizar o fluxo de trabalho e, a partir disso, atuar no processo para não sobrecarregar os membros da equipe.

Através de uma abordagem de gestão visual perante a cadeia de valor, o processo é exposto aos membros da equipe, desde sua etapa inicial até a entrega do trabalho. E tipicamente a cadeia de valor é representada em quadros brancos com *post-its* ou ferramentas *online*.

O processo e os itens de trabalho, bem como os trabalhadores,

são visualmente representados nos quadros comumente chamados de *kanban boards*, ou *kanban* (isso mesmo, o nome do método e o nome do quadro se confundem). Assim, em linhas gerais, usando *Kanban* fica mais fácil para a equipe decidir "o que", "por quem", "quando" e "quanto" produzir.

No desenvolvimento de *software*, normalmente uma pequena tarefa leva de algumas horas a alguns dias para ser concluída. Além disso, você não consegue visualizar facilmente quantos requisitos estão atualmente em análise ou quantos estão sendo codificados ou testados. De fato, não conseguimos "ver" o item de trabalho relacionado ao *software*, e como este se move ao longo das etapas de todo o processo até que esteja pronto. Aqui é onde tudo começa: *Kanban* torna esses itens em construção visíveis e você consegue visualizar claramente os pontos de gargalo!

1.2.2.1 *Workflow* Visível

A ideia prinncipal do kanban é colocar o fluxo de trabalho na frente de todos. Por exemplo, em um quadro branco, ou na própria parede. Veja isso na fotografia do *kanban* abaixo, mantido por uma equipe de desenvolvimento de *software*:

Figura 4 – Um exemplo de workflow visível

Como a parede é uma superfície bidimensional, o *kanban* é apresentado em um formato tabular, onde as etapas de trabalho são títulos de colunas, e os items de trabalho, as fotos das pessoas, e outras marcas relacionadas ao trabalho preenchem o espaço na parede. Estes cartões podem ser organizados em uma linha horizontal ou não. Tudo depende da equipe e de como ela representa e organiza o seu trabalho na parede.

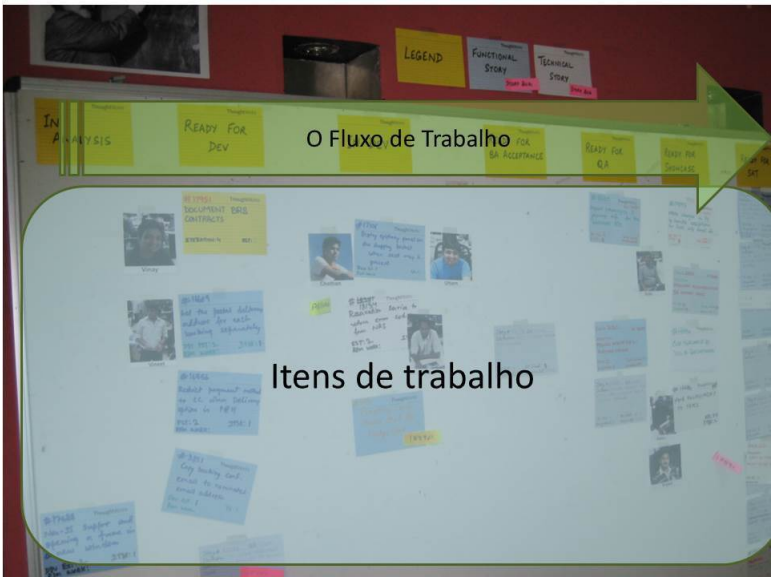


Figura 5 – Etapas e itens de trabalho

1.2.2.2 *Work In Progress* (WIP)

Limitar o trabalho em andamento, ou *WIP* (de *work-in-progress*, em Inglês), implica que o *kanban* segue um sistema puxado. Logo, o trabalho em cada etapa do processo é limitado, de forma que um novo item somente seja "puxado" para a próxima etapa se existir capacidade disponível dentro do limite *WIP* de tal etapa.

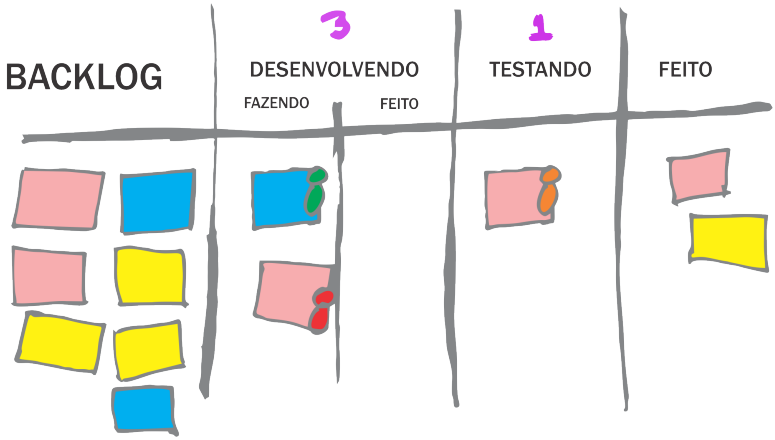


Figura 6 – Limitando o WIP

As restrições de *WIP* identificam gargalos e áreas problemáticas no processo e auxiliam o time a tomar decisões para resolvê-los.

Limitar o *WIP* é o grande diferencial do método *Kanban*. Tal artimanha é o divisor de águas entre *task boards*, ou quadros visuais – como eram conhecidos antes da influência de David Anderson com a divulgação do método *Kanban* e os quadros *kanban*.

1.2.2.3 Otimização do Fluxo

Segundo David Anderson, o ponto principal de implantar um *Kanban* é criar uma mudança positiva. E antes de criar essa mudança o time tem que saber o que mudar, o que pode ser feito olhando como os itens de trabalho estão fluindo através do processo e analisando as áreas problemáticas em que o trabalho engargala. Desta forma, poderá realizar mudanças no processo de trabalho para resolver tais problemas.

E assim sucessivamente: identificando problemas, e agindo para resolvê-los, tendo por base a visualização e limites *WIP* do *kanban*, melhorando o trabalho e o processo, na busca contínua de maior eficiência.

O *kanban* pode ser visto como uma ferramenta de comunicação para rapidamente adquirir e possibilitar a troca de conhecimentos

entre os membros de uma equipe. O objetivo principal é dar a todos os membros uma visão compartilhada do processo e do estado atual do trabalho. Assim, temos representações visuais para as fases de trabalho, para as pessoas, e para o próprio trabalho.

O método *kanban* vai ainda além disto: a partir de uma simples abordagem ele permite a visualização do *workflow* e dos limites de *WIP*. A equipe então trabalha em um sistema puxado, que auxilia o time a identificar gargalos e a atuar sobre áreas problemáticas do processo. Sendo assim, concomitantemente o *kanban* fornece à equipe: uma ferramenta, uma prática e um processo de melhoria contínua.

2. Produto Mínimo Viável (MVP)

Com origem no *Lean Startup*, o "Produto Mínimo Viável" ou "MVP", do inglês *Minimum Viable Product*, é a versão mais simples de um produto que pode ser disponibilizada para a validação de um pequeno conjunto de hipóteses sobre o negócio. Basicamente, isso significa que você não quer desperdiçar tempo, dinheiro e esforço construindo um produto que não vai atender as expectativas. Assim, para mitigar estes riscos, é razoável que você possa querer entender e validar as hipóteses sobre o negócio antes de tentar encontrar uma solução definitiva.

O MVP ajuda nessa validação e aprendizado da forma mais rápida possível. E diferentemente dos produtos criados da forma tradicional, normalmente com um período longo de criação de protótipo, análise e elaboração, o objetivo do MVP é somente a validação de um primeiro passo ou visão (um produto mínimo, portanto), notadamente bem menos elaborado do que a sua versão final.

Um MVP foca no produto mínimo de modo que ele se mostre viável para verificar a corretude de um direcionamento ou conjunto de proposições. E em se tratando de desenvolvimento de *software*, isso remete a um conjunto inicial de funcionalidades consideradas necessárias no curto prazo e que depois serão submetidas à validação, com fins de aprendizagem sobre o negócio.

Um exemplo de MVP no mundo real é o da *startup* EasyTaxi. Os seus sócios decidiram que, ao invés de passarem meses desenvolvendo uma solução para intermediação digital de corridas de Taxi, criariam apenas uma página *web* simplificada cuja função seria enviar o endereço de origem informado pelo usuário aos seus próprios e-mails. Depois, eles mesmos ligavam para as cooperativas e completavam o processo. Com este primeiro MVP, puderam

constatar que havia interesse no serviço, validando uma hipótese fundamental do negócio, com um uso mínimo de tempo e recursos, reduzindo também os riscos associados.

2.1 Origem

É importante ressaltar que a ideia de MVP está originalmente vinculada àquelas que foram popularizadas pelo estilo Toyota de manufatura enxuta [1] [2] [3]. Steve Blank, um empreendedor do Vale do Silício, criou uma metodologia [4] definida com base no desenvolvimento do cliente. Este foi o início do movimento *Lean Startup*, o qual teve seu ápice com Eric Ries e o lançamento do seu livro [5], com o mesmo nome do movimento.

Quando Eric Ries popularizou o MVP com a publicação do seu livro *Lean StartUp*, o termo já estava em uso vários anos antes do surgimento do movimento que dá nome à publicação, especialmente entre as *startups* e seus respectivos empreendedores e investidores do Vale do Silício. Explica-se: a expressão *minimum viable product* apareceu pela primeira vez em 2000 em um artigo de Willin Junk [6], de título "O equilíbrio dinâmico entre Custo, Cronograma, Recursos e Qualidade em Projetos de Desenvolvimento de *Software*", em português.

[1] Womack, James P.; Daniel T. Jones, and Daniel Roos. (1990) *The Machine That Changed the World.*

[2] Ohno, Taiichi. (1988) *Toyota Production System. Productivity Press.*

[3] Womack, James P.; Daniel T. Jones. (2003) *Lean Thinking. Free Press.*

[4] Blank, Steve G. (2013) *The four steps to the epiphany: successful strategies for products that win.*

[5] Ries, Eric. (2011) *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Publishing.*

[6] Willian, S. Junk. (2000) *The Dynamic Balance Between Cost, Schedule, Features, and Quality in Software Development Projects, Computer Science Dept.*, University of Idaho, SEPM-001.

2.2 Evolução

MVP não significa que o produto não irá evoluir e incrementar suas funcionalidades. Muito pelo contrário, a ideia por trás de um MVP é ter um incremento validado e guiado pelos resultados iniciais, com a correção ou a confirmação do curso orientando os incrementos seguintes. Incrementos estes que são também um MVP: novos produtos mínimos adicionados aos produtos mínimos já validados. E estes, por sua vez, novamente constituindo produtos mínimos, os quais são submetidos a novas verificações sobre o direcionamento do produto, agora já mais elaborado e talvez um número maior de usuários (permitindo validar novas hipóteses, ainda mais estruturadas).

Observa-se que é muito importante compreender que o MVP promove uma criação evolutiva. Logo, a arquitetura, bem como o ferramental de construção do produto, devem idealmente também possuírem esta característica de evolução gradual e contínua.

Em 2010, Jez Humble e David Farley publicaram o livro *Continuous Delivery* [7]. Nele, elaboraram sobre um processo de entrega rápido e de baixo custo, permitindo a criação incremental de produtos de *software*. Isso foi batizado *Continuous Delivery* (em português, "Entrega Contínua"), e abrange a disciplina de desenvolvimento de *software* que promove entregas mais rápidas e com maior frequência.

Apesar do livro *Continuous Delivery* entrar em detalhes sobre produtos de *software* e o fluxo de trabalho utilizado para sua criação, a essência da ideia de "Entrega Contínua" é a mesma que Eric Ries recomenda para o *Lean StartUp*: ciclos rápidos para validação das hipóteses.

Ciclos rápidos e frequentes, aliás, permitindo tempos de liberação muito curtos e com baixo custo de experimentação o que traduz-se em fatores críticos de sucesso principalmente para projetos de *software* de inovação ou diferenciação. No livro "DiretoAoPonto"

[7] Jez Humble and David Farley. (2010) *Continuous Delivery*, Addison-Wesley.

[8] Paulo Caroli descreve uma receita com atividades de análise e planejamento efetivo baseado em MVP.

2.3 Incrementos

O produto é construído de forma incremental, com MVPs recém-criados sendo adicionados ao produto consolidado já existente. A entrega contínua e incremental proporciona o aumento da percepcão de valor do produto ao longo do tempo, enquanto que o processo de criação de produto tradicional não fornece qualquer valor até o final, quando todo o produto está pronto.

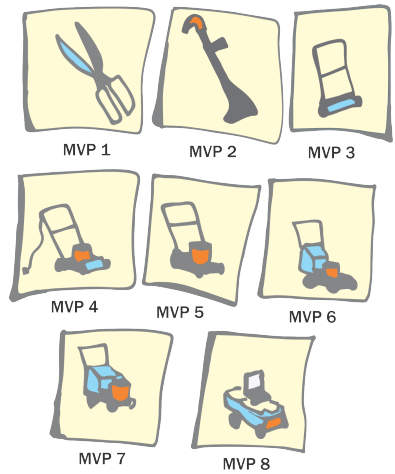


Figura 7 – MVPs para "cortar grama"

A Figura 7 mostra como trabalhar com MVP oferece pequenas validações ao longo do tempo, enquanto o estilo de criação do produto "mais tradicional" só fornece a validação do todo (novamente, ao final da demanda de desenvovimento).

Inicialmente um tesourão forma o primeiro MVP... mas há realmente grama para cortar? Alguém vai utilizar tal aparato? A

[8] Paulo Caroli. (2014) Direto Ao Ponto, criando produtos de forma enxuta, *LeanPub*.

validação dessas hipóteses impulsiona a evolução do produto para o próximo MVP.

Talvez um aparato mais cômodo, com um cabo… e que tal adicionar rodas? E assim por diante até que o produto evolua de MVP a MVP, sempre incrementando e validando tais incrementos.

O MVP promove, portanto, uma abordagem incremental, em que apenas uma pequena parte de um requisito ou ideia mais abrangente são tratadas por vez. Cada um desses incrementos é projetado, criado e preparado para ser adicionado ao produto, agregando mais funcionalidades ao mesmo (que tendem a ser "as funcionalidades certas" a serem construídas).

Em essência, uma ideia de produto de software é sequenciada em uma série de hipóteses menores, mais simples e, logo, mais fáceis de entender, criar e contabilizar. E cabe lembrar que felizmente *software* não é manufatura: logo, no mundo de *software*, um cortador de grama pode ser criado adicionando-se rodas e cabo a um tesourão.

3. Métodos de Desenvolvimento: nivelamento

Para efeito deste livro, podemos considerar três abordagens genéricas que podem ser utilizadas para desenvolvimento de *software*: modelo tradicional, modelo ágil e modelo enxuto. Embora existam outras formas de desenvolver *software*, assim como derivações e híbridos destes métodos de trabalho, não trataremos todas as referências disponíveis, mas as mais significativas para o desenvolvimento do tema de estudo.

Embora este capítulo trate um assunto provavelmente conhecido por grande parte dos leitores, é importante para o entendimento geral da proposta do livro, realizar um nivelamento de conhecimentos, principalmente porque é necessário enfatizar tópicos, princípios e valores intrinsecamente ligados a cada abordagem com vistas a nos mantermos atentos à razão de suas escolhas.

3.1 Tradicional

Os métodos de desenvolvimento de *software* no modelo tradicional englobam paradigmas derivados da abordagem conhecida como *waterfall* (Figura 8), ou "cascata" em português, assim como os seus conceitos relacionados (SABBAGH, 2013).

Inicialmente descrito por ROYCE (1970), o modelo tradicional segue uma linha determinística para o desenvolvimento de produtos e é tipicamente sequencial, sendo composto por um conjunto de atividades fortemente dependentes (PRESSMAN 2006), o que podemos comparar com algo similar às linhas de produção das fábricas de manufatura. Desta forma, observamos que há pouca adaptabilidade e teoricamente baixa propensão a mudanças.

Trabalhando no modelo tradicional, é esperado que os papéis, artefatos e responsabilidades dos profissionais envolvidos sejam bastante claros e as etapas seguidas bem definidas, durante todo o processo de desenvolvimento de *software* (SABBAGH, 2013, pág. 19).

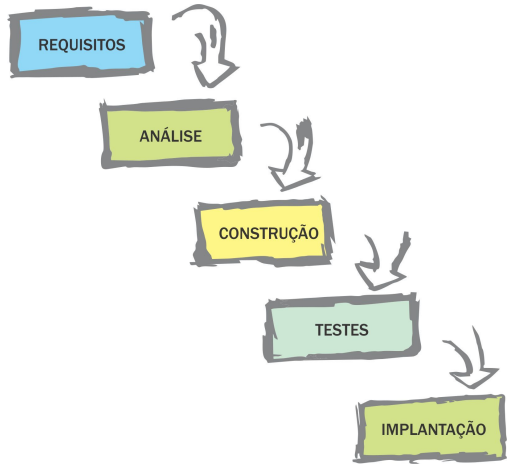


Figura 8 – Etapas do Desenvolvimento *Waterfall*

É importante destacar ainda, que o próprio ROYCE (1970) criticava a aplicação do método "cascata" para atividades de desenvolvimento de *software*, afirmando que seu uso era arriscado neste cenário, uma visão que é compartilhada por SABBAGH (2013). Ele considera que os métodos tradicionais são fortemente prescritivos e que de forma geral são caracterizados pelo foco em planos detalhados que são definidos no início do projeto, preocupando-se com o custo, escopo e um cronograma detalhado, com micro gerenciamento e poder centralizado, em uma estrutura de processos complicados, prevendo extensa documentação.

POPPENDIECK (2011, pág. 45) também defende que o *software*, por sua própria natureza deveria ser projetado para se adaptar às mudanças durante todo o seu ciclo de vida, e que por isso o processo de desenvolvimento não deveria ser determinístico como

nos métodos tradicionais, mas empírico.

3.2 Ágil

Devido a crescente complexidade e necessidade de inovação na concepção de determinados tipos de produtos, no ano de 2001 surgiu um movimento na comunidade de desenvolvimento de *software* o qual culminou com a publicação de importantes críticas às práticas de desenvolvimento e gerenciamento de projetos *software*, até então predominantemente executadas no que hoje chamamos "modelo tradicional".

Esta publicação foi realizada por um grupo de desenvolvedores e engenheiros de *software*, com a justificativa de que o modelo cascata já não mais seria condizente com a realidade das empresas que necessitavam deste tipo de serviço. Ao apanhado foi dado o nome "Manifesto Ágil" (BECK et al, 2001).

A proposta ágil definida no "Manifesto Ágil" foi elaborada com base em quatro direcionadores principais:

- Indivíduos e interações mais que processos e ferramentas;
- *Software* em funcionamento mais que documentação abrangente;
- Colaboração com o cliente mais que negociação de contratos;
- Responder a mudanças mais que seguir um plano.

Contudo, além destes direcionadores, o Manifesto Ágil conta ainda com 12 princípios, os quais são relacionados abaixo:

- Priorizar a satisfação do cliente através da entrega contínua e adiantada, com valor agregado;
- Permitir mudanças nos requisitos (mesmo que tardiamente), visando vantagem competitiva ao cliente;
- Entregar frequentemente *software* funcionando, preferindo a menor escala de tempo possível;

- Unir as pessoas do negócio e os desenvolvedores diariamente no trabalho despendido durante todo o projeto;
- Construir projetos em torno de indivíduos motivados, com ambiente, suporte e confiança necessária para executar o trabalho;
- Buscar o relacionamento direto entre as pessoas;
- Primar pelo *software* funcionando como medida primária de progresso;
- Manter um ritmo constante, indefinidamente e de forma sustentável;
- Aumentar a agilidade através de contínua atenção, excelência técnica e bom *design*;
- Buscar a simplicidade, maximizando a quantidade de trabalho não realizado;
- Conceber equipes auto organizáveis, emergindo melhores arquiteturas, requisitos e *designs*;
- Possibilitar que a equipe reflita sobre sua eficácia, em intervalos regulares, refinando e ajustando seu comportamento conforme necessário.

Ao reforçar a importância da abordagem ágil e derivando seus princípios, HIGHSMITH (2004) sugeriu a aplicação de técnicas de gerenciamento ágil em projetos que tratam de novos produtos, enfatizando que as empresas precisam desenvolver uma cultura que promova a adaptabilidade para absorver mudanças, com direcionadores bem definidos, o que encoraja a auto-organização, de forma perfeitamente combinada à capacidade de autogestão, colaboração e interação dentre os membros de um time.

O desenvolvimento de *software* ágil é tipicamente iterativo e incremental, sendo os requisitos de *software* separados em pequenas funcionalidades e planejados em iterações. Assim, o *software* é construído de forma incremental, com funcionalidades recém-desenvolvidas sendo adicionadas ao produto já existente. A figura abaixo descreve a natureza do desenvolvimento ágil de *software*.

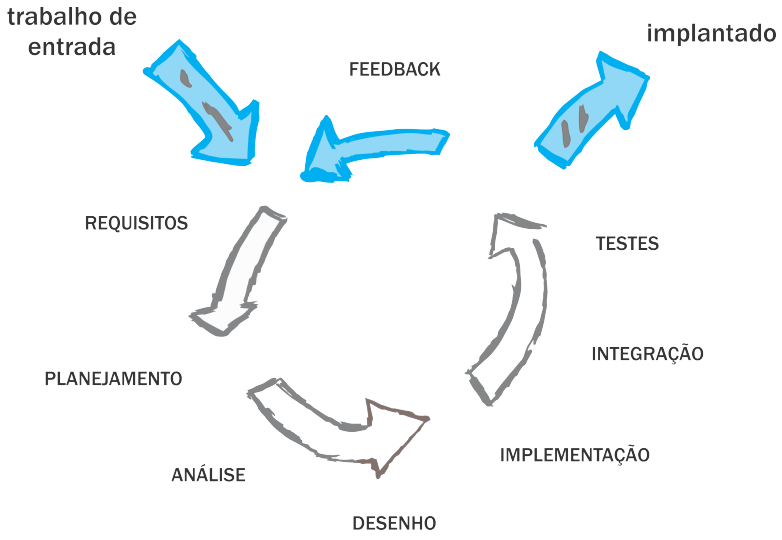


Figura 9 – O ciclo de trabalho

No modelo de desenvolvimento de *software* ágil há pequenos ciclos de desenvolvimento, que normalmente duram de uma a quatro semanas. E como você pode ver na figura acima, há várias atividades intrínsecas a esta forma de trabalho. Essas atividades diferem, a depender do estilo e da maturidade da metodologia ágil dentro da organização.

Destaca-se também, que a entrega incremental de funcionalidades proporciona um aumento do valor do produto ao longo do tempo, enquanto que o processo de desenvolvimento de *software* tradicional adiciona valor apenas na entrega do produto final idealizado. A Figura 10 ilustra a entrega incremental típica de métodos ágeis por meio de iterações consecutivas (ela demonstra três iterações consecutivas).

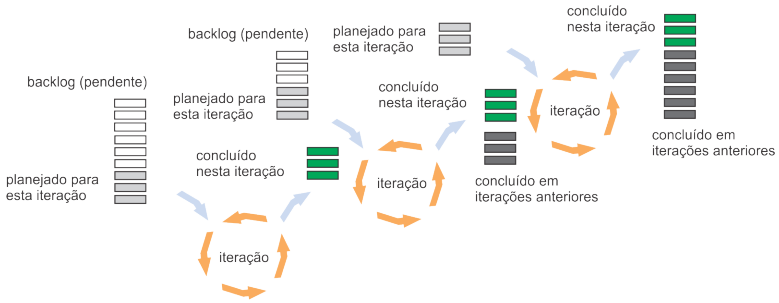


Figura 10 – Entregando gradualmente

Os requisitos do produto são organizados em um *backlog*, ou seja, uma lista dos pequenos pedaços de requisitos. Em seguida, deste *backlog*, uma certa quantidade de trabalho é selecionada para uma iteração, formando uma porção menor do *backlog*. *Scrum* chama esses *backlogs* respectivamente de *backlog* do produto e *backlog* da iteração (*Sprint*). Subsequentemente, à medida que o trabalho progride, este é continuamente integrado com o trabalho concluído anteriormente. Por fim, o trabalho realizado dentro da iteração, ou é contabilizado ou espera pelo complemento de iterações futuras para ser contabilizado.

O gráfico de funcionalidade em função do tempo (abaixo) fornece uma representação visual eficaz para comparar métodos ágeis e estilos de desenvolvimento no modelo tradicional. Neste gráfico, o eixo X representa o tempo decorrido, e o eixo Y, a funcionalidade entregue. A seta azul, por sua vez, demarca um momento na linha do tempo.

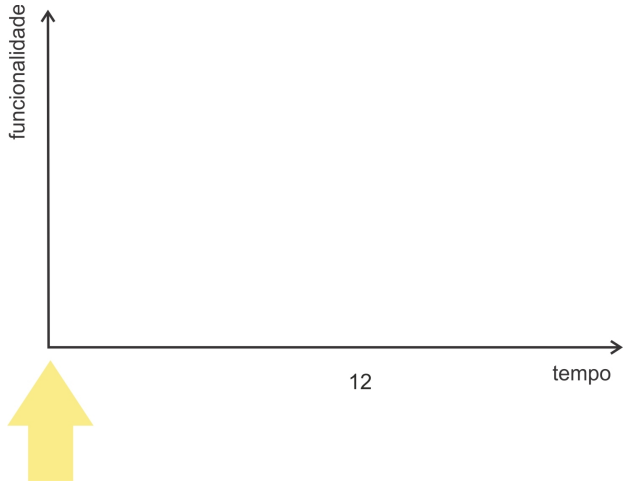


Gráfico 1 – Funcionalidade em função do tempo

Note que o gráfico acima representa um projeto no início e nenhuma funcionalidade foi concluída ainda, independente se no modelo desenvolvimento ágil ou tradicional.

As figuras a seguir mostram o estilo ágil comparado com equipes de desenvolvimento de *software* tradicionais:

Sequência tradicional

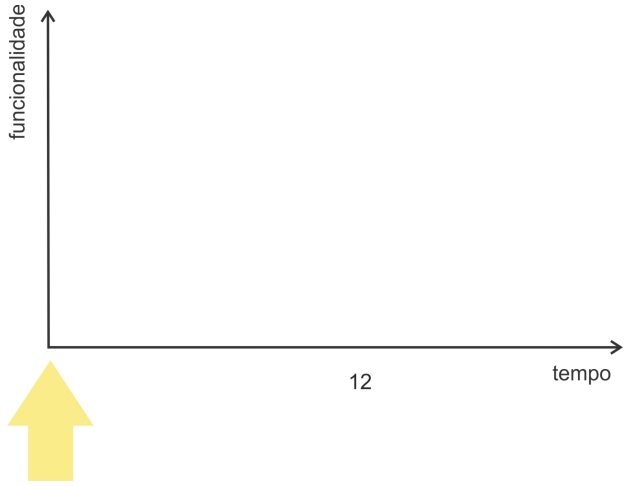


Gráfico 2 – Desenvolvimento tradicional 1

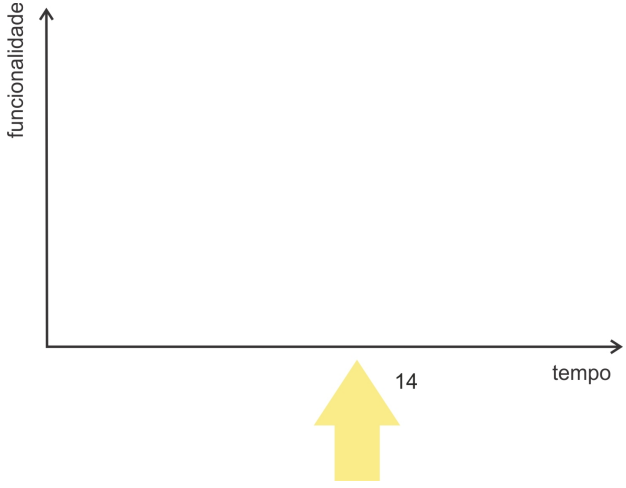


Gráfico 3 – Desenvolvimento tradicional 2

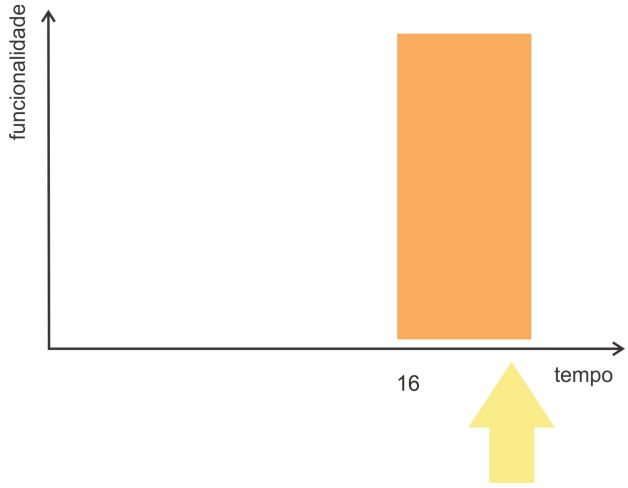


Gráfico 4 – Desenvolvimento tradicional 3

Sequência ágil

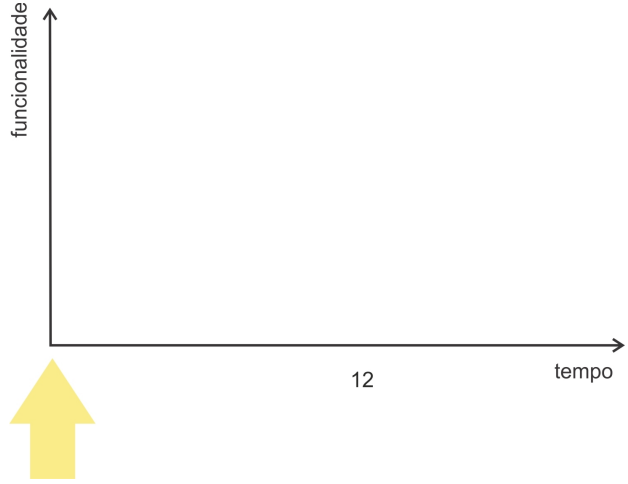


Gráfico 5 – Desenvolvimento ágil 1

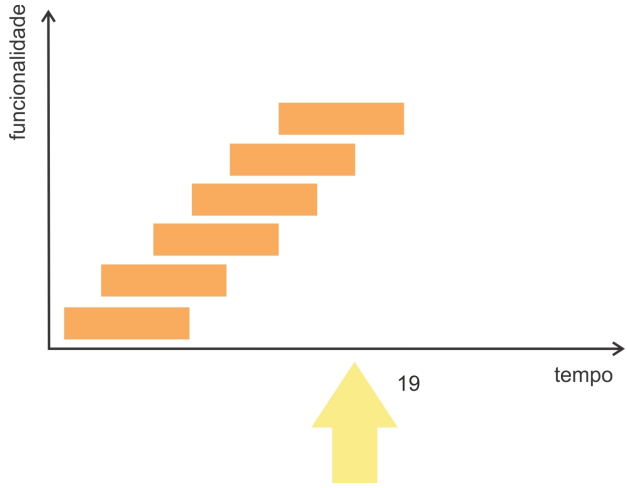


Gráfico 6 – Desenvolvimento ágil 2

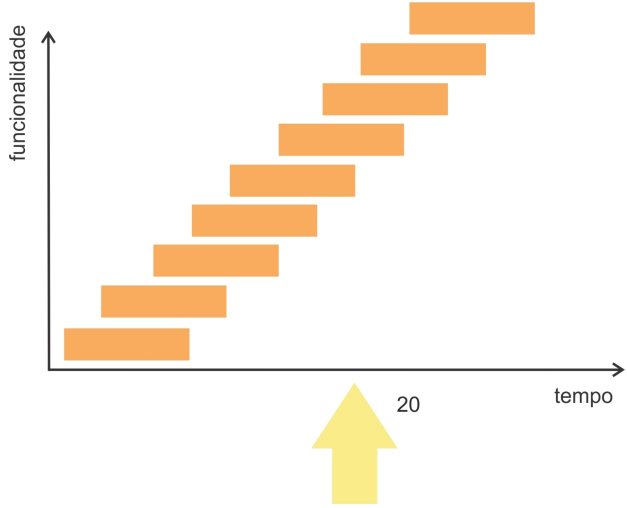


Gráfico 7 – Desenvolvimento ágil 3

Esta sequência demonstra como entregas ágeis incrementam funcionalidade ao longo do tempo, enquanto o estilo de desenvolvimento de *software" mais tradicional entrega o todo somente no

final.

Serão apresentados a seguir, alguns habilitadores para o desenvolvimento de *software* ágil, os quais serão abordados para nivelamento em relação aos métodos e conceitos utilizados na nossa proposição para a "máquina enxuta", tais como: XP, *Scrum* e *DevOps*. É uma leitura rápida e útil para contextualização.

3.2.1 *eXtreme Programming (XP)*

A metodologia ágil de desenvolvimento chamada *Extreme Programming* (XP) foi criada por Kent Beck nos anos 1990 e cobre diversos aspectos técnicos do desenvolvimento de *software*, tais como codificação, *design* e testes (GOMES, 2013). FRANCO (2007) complementa esta visão, afirmando que o *XP* surgiu como uma tentativa de solução para os problemas causados pelos longos e morosos ciclos de desenvolvimento dos modelos de desenvolvimento tradicionais e que por este motivo consolidou-se no mercado.

O *XP* é considerado uma proposta de desenvolvimento de *software* simples, que visa entregar o que o cliente solicitou, no prazo acordado com ele. Ele define uma equipe única de trabalho, e considera que os gerentes, clientes e desenvolvedores devem também formar uma equipe, dedicada e destinada a entregar *software* com mais qualidade (AUDY e PRIKLANDNICKI, 2007).

Segundo HIGHSMITH (2002), o *XP* foca em interações de cliente e desenvolvedor que devem ocorrer em um ambiente apropriado para este fim: times dividindo o mesmo espaço físico, compostos por dez ou menos desenvolvedores e com um cliente no local, dedicado integralmente ao projeto. Como em outras metodologias ágeis, o desenvolvimento com *XP* ocorre com base em iterações curtas (de três semanas ou menos).

3.2.2 *Scrum*: um *framework* ágil

HIGHSMITH (2004) define o Gerenciamento de Projetos Ágeis, em inglês *Agile Project Management* (*APM*), como um conjunto de

valores, princípios e práticas que auxiliam a equipe de projeto a entregar produtos e/ou serviços de valor em um ambiente complexo, instável e desafiador – onde os valores e princípios referem-se aos conceitos e as práticas voltadas para o "como" realizá-lo. O *APM* caracteriza-se por ser simples, flexível e iterativo, com grande capacidade de adaptação às práticas de gerenciamento existentes e passível de aplicação em ambientes dinâmicos cujas especificações são regidas pela inovação, em complemento às teorias ditas "tradicionais" (AMARAL, 2011, págs. 2, 3 e 8).

Outro autor, AUGUSTINE (2005), afirma que o gerenciamento ágil de projetos deve ser constituído pela energização, capacitação e habilitação da equipe envolvida no projeto, para prover entregas mais rápidas e com maior valor para o negócio através da integração dos clientes num processo de aprendizagem contínua, com adaptabilidade a mudanças e de acordo com suas necessidades e ambiente.

Neste contexto, o *Scrum* é considerado um *framework* ágil para a construção e gestão de projetos complexos, inicialmente proposto para desenvolvimento de *software* mas que tem sido aplicado para diferentes contextos de projetos complexos e trabalhos inovadores. Ele é especialmente adequado para projetos de *software* onde requisitos mudam rapidamente ou são altamente emergentes. Ele progride através de uma série de iterações chamadas *Sprints*.

Uma *Sprint* promove uma cadência semanal, de uma a quatro semanas, a depender da preferência do time. Diariamente o time realiza uma reunião para verificar o andamento das tarefas de trabalho, sendo esta reunião chamada *Daily Sprint*. Nela, basicamente todos os membros do time ficam de pé (para que a reunião não demore demais), e todos devem responder a três perguntas, as quais auxiliam o time a se auto-organizar, buscando o alinhamento diário em relação ao trabalho realizado e ainda a ser realizado. As três perguntas são: "o que fiz ontem?", "o que vou fazer hoje?" e "o que está impedindo o progresso do meu trabalho?".

A metodologia sugere que cada *Sprint* comece com uma breve reunião de planejamento e termine com uma reunião de revisão do trabalho realizado. Estes são os princípios do gerenciamento

de projetos *Scrum*: ciclos curtos e cadenciados com reuniões de alinhamento, com acompanhamento da evolução do trabalho e do time.

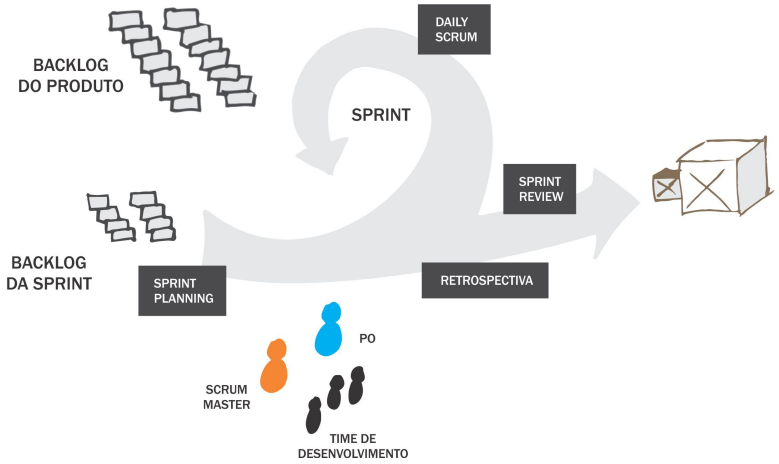


Figura 11 – O *framework* Scrum

Além das reuniões de (1) *planning* e (4) *review*, o *Scrum* orienta mais duas reuniões, que ocorrem a cada *Sprint*: (2) "Retrospectiva", a reunião que promove o momento *kaizen*, onde o time busca a melhoria contínua do processo, entrega e interação entre as pessoas; e (3) "*Grooming*", a reunião onde o *backlog* do produto é revisitado para aprimorar o entendimento sobre os próximos requisitos, ou seja, os candidatos ao próximo *Sprint*.

No mundo ágil *Scrum*, evitamos descrições completas e detalhadas sobre como tudo deverá ser feito na *Sprint*, sendo que muito do que é feito é deixado para o time de desenvolvimento decidir. Isso porque o time vai saber a melhor forma de resolver o problema em questão. E é por isso que a reunião de planejamento do *Sprint* é descrita em termos de metas e resultado desejado. o resultado desejado é um compromisso com um conjunto de funcionalidades a serem desenvolvidas no próximo *Sprint*. Buscando assim o equilíbrio entre autonomia, flexibilidade e comprometimento do time. E cabe ressaltar, que este compromisso é revisitado ao final da *Sprint*,

na reunião de *review*.

Segundo SABBAGH (2013), o Scrum atualmente é a forma mais comum de se trabalhar em projetos de *software*, tendo sido adotado por organizações de diversos tamanho e tipos, sejam multinacionais, *start-ups*, empresas privadas ou públicas. Ele ressalta ainda, que ao utilizar *Scrum*, existem termos que frequentemente serão citados e que falam por si sobre a natureza do método, tais como: facilitação e trabalho em equipe, auto-organização, metas de negócio, motivação e relacionamento com os clientes – e ainda, que alguns dos principais benefícios do método são os que seguem (SABBAGH, 2013):

- Entregas frequentes de retorno ao investimento dos clientes – partes do produto funcionando podem ser entregues desde cedo e frequentemente, através de ciclos curtos de desenvolvimento, o que reduz o prazo de introdução do produto no mercado;
- Redução dos riscos do projeto – há *feedback* constante dos clientes e demais partes interessadas, estabelecendo uma melhor relação entre o que foi solicitado e aquilo que efetivamente foi recebido como produto, embora também seja importante a presença do time multidisciplinar, para redução de dependências;
- Maior qualidade do produto gerado – o próprio time de desenvolvimento é encarregado de garantir a qualidade do produto entregue e de validá-lo junto ao requisitante (como isso ocorre com maior frequência, diminui-se o risco de problemas acumularem-se as entregas);
- Mudanças utilizadas como vantagem competitiva – na verdade são vistas como oportunidades e não como empecilhos, como no método tradicional, muitas vezes, são até mesmo uma consequência do ciclo de aprendizado;
- Visibilidade do progresso do projeto: o conjunto de práticas e artefatos do Scrum visa garantir a visibilidade e transparência

do progresso do projeto para seus participantes e envolvidos, além de prover uma sensação de progresso perante os clientes, que recebe frequentemente partes do produto funcionando;

- Redução do desperdício – o desperdício é reduzido e evitado na busca pela simplicidade, já que o time que usa *Scrum* tende a produzir e utilizar apenas o que é necessário e suficiente (produzindo apenas o que os usuários irão utilizar, planejando apenas com o nível de detalhe possível e utilizando apenas os artefatos necessários e suficientes);
- Aumento de produtividade – diversos fatores potencializam a produtividade de times que utilizam Scrum, dentre eles: o trabalho da equipe e a autonomia do time na realização desse trabalho, a existência de facilitação e de remoção de impedimentos, a melhoria contínua dos processos de trabalho, além da busca incessante por um ritmo sustentável de trabalho e maior motivação do time.

3.2.2.1 Papéis

Scrum fomenta uma equipe multifuncional e auto-organizada e a eficiência do time depende da capacidade dos membros para trabalharem em conjunto fazendo o melhor uso das suas habilidades individuais em um contexto de trabalho multifuncional. Na prática, isso significa que o time *Scrum* é auto-organizado, mesmo sem um líder de equipe que decide quem vai fazer qual tarefa e como. Desta forma, as tarefas e problemas do time são levantados por todos e as questões relacionadas também são decididas pelo grupo.

Os times *Scrum* são apoiados por dois papéis específicos e um generalista: *Scrum Master*, *Product Owner* (PO) e Time de Desenvolvimento:

- O *Scrum Master* deve ser alguém experiente com o *framework* e capaz de ajudar o time a usar o *Scrum* para alcançar seus objetivos de alto nível. Os melhores profissionais *Scrum Master* são aquelas pessoas que sentem mais satisfação de

facilitar o sucesso dos outros do que seus próprios. Alguém com este papel deve se sentir confortável e seguro com a metodologia a ponto de dar todo controle em relação ao produto para o *Product Owner* (*PO*), e todo controle do desenvolvimento à sua equipe.

- O *PO* representa o negócio (os clientes ou usuários), e orienta a equipe para a construção do produto certo. Ele deve liderar o esforço de desenvolvimento, através de esclarecimentos e priorizações sobre o trabalho a ser executado.
- O Time de Desenvolvimento é um grupo de desenvolvedores com variadas formações e capacidades. Ele possui todo o conhecimento necesssário para que o produto possa ser desenvolvido.

Tipicamente, o *PO* trabalha com o *Product Backlog*, a lista mestre dos requisitos do produto a ser criado. É sua função, priorizá-lo com base no valor do negócio e no alinhamento realizado entre as partes interessadas (tanto internas quanto externas à equipe *Scrum*). Assim, o *PO* deve estar disponível para a equipe, para orientar o time a cada momento ou indagação.

Destaca-se que a combinação de autoridade e disponibilidade do *PO* torna-o peça chave do *framework*, valorizando a auto-organização e a autonomia; portanto, ele deve respeitar o direcionamento e a capacidade da equipe para criar o seu próprio plano de ação.

O "Time de *Scrum*" – formado pelo *Scrum Master*, o *PO* e o Time de Desenvolvimento – participa ativamente de todas reuniões com um alto nível de autonomia, transparência e comprometimento.

Devido a busca pela melhoria continua, idealmente, o Time de *Scrum* performa em níveis elevados de rendimento. É importante destacar que muito disso é alcançado pelo entrosamento do time, do alinhamento cadenciado via *Sprints*, e da clareza de cada papel e reunião.

Conforme SABBAGH (2013), com a divisão de papéis, na prática, o Time de *Scrum* compartilha a responsabilidade sobre o escopo, custo, tempo, qualidade, risco e gestão do trabalho de

desenvolvimento do produto, ou seja, os seus integrantes trabalham de forma colaborativa e são igualmente comprometidos com a satisfação dos clientes do projeto (SABBAGH, 2013, págs. 107 e 108), vide quadro abaixo.

Ponto-chave a ser gerenciado	Product Owner	Time de Desenvolvimento	ScrumMaster
Retorno sobre o investimento	X		
Necessidades/objetivos de negócios	X		
Clientes e demais partes interessadas	X		
Visão do Produto	X		
Releases	X		
Tarefas de desenvolvimento do produto		X	
Qualidade interna do produto		X	
Qualidade externa do produto	X	X	
Estimativas ou previsões		X	
Processos (funcionamento do Scrum)			X
Impedimentos no trabalho			X
Relacionamento e motivação do Time		X	X
Riscos	X	X	X
Comunicação	X	X	X

Quadro 1 – Responsabilidades do Time *Scrum*

3.2.3 *DevOps* e *Continuous Delivery*

Completamente aderente ao processo de desenvolvimento ágil e enxuto, SATO (2013) define *DevOps* como um movimento cultural e profissional que está tentando quebrar barreiras para prover a entrega de *software* em produção. A utilização das suas práticas têm se tornado cada vez mais comum, pois possibilitam entregar *software* em produção com maior frequência e com maior estabilidade e robustez – realidade já de muitas empresas, como Google, Amazon, Netflix, Flickr, GitHub e Facebook (SATO, 2013).

Conforme SATO (2013), embora óbvio, apenas nos últimos anos as empresas vem realmente percebendo que a própria tecnologia pode ser usada a seu favor e que o atraso em um *deploy* para

produção significa muitas vezes retardar sua habilidade de competir e se adaptar a mudanças no mercado; sendo hoje comum as organizações realizarem por vezes dezenas ou até centenas de *deploys* por dia – uma abordagem conhecida como "Entrega Contínua" e que faz parte de uma estratégia de *DevOps* moderna e competitiva.

3.3 Enxuto

Podemos considerar ainda um terceiro modelo de trabalho: o enxuto. E embora olhemos com bons olhos para as metodologias ágeis, nossos principais direcionadores acabam por ser os enxutos, devido a sua maior proximidade com os princípios da eficácia.

Na interpretação de POPPENDIECK (2011, pág. 44), as práticas enxutas de produção e de gerenciamento da cadeia de suprimentos não são facilmente "traduzidas" para o desenvolvimento de *software*, pois tanto o *software* como o desenvolvimento são bastante disformes em relação ao contexto de operações e logísticas. E por este motivo, são necessárias algumas adaptações.

POPPENDIECK (2011) também consolida uma ampla variedade de teorias e experiências "traduzindo" o *Lean* da indústria de manufatura (*Lean Manufacturing*) para viabilizar sua aplicação no contexto de desenvolvimento de *software*. Nesta perspectiva, conseguimos visualizar também que os problemas fundamentais de produção são também aqueles que podem ocorrer no desenvolvimento de *software*, tais como:

- Lidar com incerteza e mudança;
- Melhorar continuamente os processos; e
- Entregar valor aos clientes.

A autora defende também, que a conversão das teorias enxutas para o desenvolvimento de *software* impacta em sete princípios e sete desperdícios, sendo demonstrados abaixo os "Sete Princípios":

- Eliminar o desperdício;

- Integrar qualidade;
- Criar conhecimento;
- Adiar comprometimentos;
- Entregar rápido;
- Respeitar as pessoas;
- Otimizar o todo.

A perspectiva dos "Sete Desperdícios" no desenvolvimento de *software*, por sua vez, identifica os seguintes pontos a serem atentados e trabalhados (POPPENDIECK, 2011, pág. 93):

- Trabalho inacabado – o estoque do desenvolvimento de *software* é o trabalho inacabado;
- Funcionalidades extras – considerado o pior item desta lista, é a ação de adicionar funcionalidades que não são realmente necessárias ao cliente e são adicionadas no escopo de desenvolvimento;
- Reaprendizagem – desperdício de conhecimento devido a ignorar a participação, contribuição e engajamento das pessoas;
- Transferência de controle – evitar realizar transferência do controle do trabalho;
- Troca de tarefas – evitar trocar tarefas;
- Atrasos - impactam na (in)satisfação e pode prejudicar relações de interdependência com outros times;
- Defeitos - também impactam na (in)satisfação do cliente e ocasionam retrabalho.

POPPENDIECK (2011, págs. 82, 83, 87 e 89, 91) define ainda uma série de questões que devem ser levadas em conta ao desenvolver *software*, tais como as citadas abaixo:

- Projetos e produtos são coisas diferentes, visto que projetos possuem início, meio e fim e produtos possuem início, mas não necessariamente um fim;

- Evitar a complexidade, pois esta é uma das principais aliadas da dívida técnica e do desperdício – cujo custo associado é exponencial;
- Optar por times perenes, embora exista a tendência de constituir uma nova equipe de desenvolvimento a cada novo projeto;
- Enxergar-se como uma empresa de *software*, tendo o negócio como seu cliente;
- Cada funcionalidade desenvolvida deveria passar por uma prova de valor, justificando o porquê da sua existência para enxergar o real valor para o negócio;
- Buscar dividir o *software* em conjuntos mínimos de funcionalidades úteis, para que o negócio possa utilizar o *software* mais depressa;
- Não automatizar a complexidade: lembre que primeiro o processo candidato à automação deve ser esclarecido e simplificado.

3.3.1 Uma receita "DiretoAoPonto"

O *Minimum Viable Product* (MVP), já citado anteriormente, como vimos, é a versão mais simples de um produto que pode ser disponibilizada para o negócio. O seu foco reside no produto mínimo e na validação de pedaços menores, obviamente bem menos elaborados do que uma versão final, mas já uteis para verificar se um direcionamento está no caminho correto (CAROLI, 2015).

A Figura abaixo mostra como o MVP oferece validações pequenas ao longo do tempo, diferentemente da forma tradicional de criação de produtos, que provavelmente só o validaria funcionando na sua versão final. No exemplo, o trator cortador de grama (MVP 8) representa o produto final.

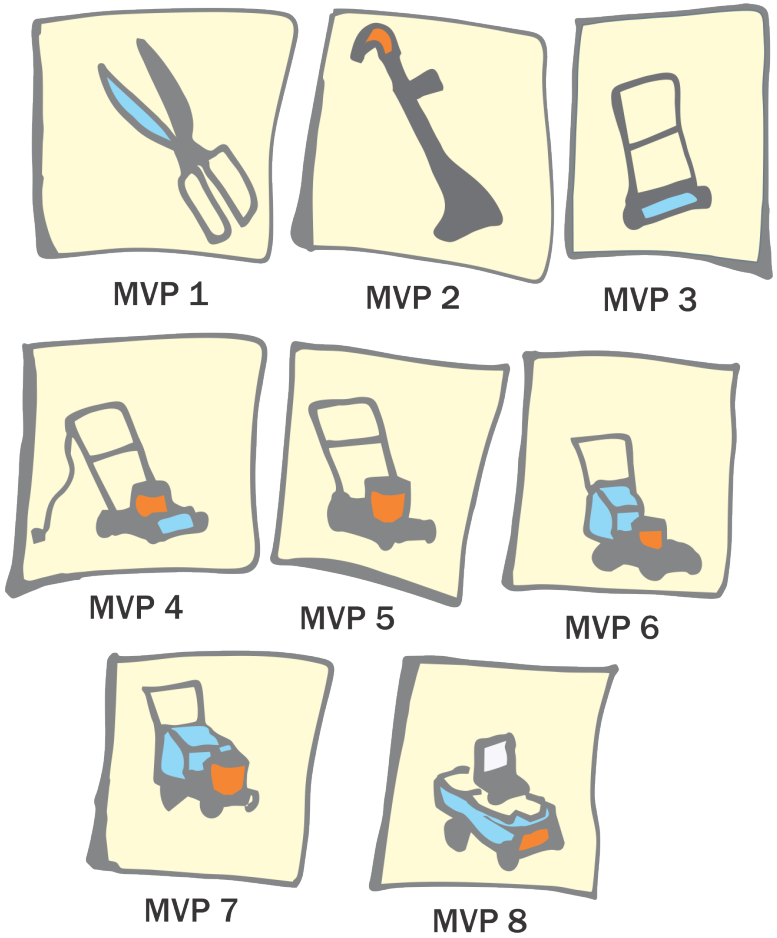


Figura 12 – MVPs para cortar grama

Considerando um planejamento por MVPs, as hipóteses de negócio são sequenciadas em uma série de outras hipóteses menores, mais simples e, logo, mais fáceis de entender. Elas também são elaboradas de forma mais rápida e com um produto disponibilizado mais cedo, o que viabiliza o seu uso pelo usuário final (CAROLI, 2015).

A "Concepção Enxuta" (*Lean Inception*, em inglês), ou sim-

plesmente "Concepção", como chamaremos a partir deste ponto, é definida como um *workshop* presencial e colaborativo realizado para alinhar as necessidades de negócio e o entendimento sobre os MVPs (CAROLI, 2015). Assim, o sucesso de uma Concepção está diretamente atrelado à capacidade do grupo envolvido em colaborar e participar ativamente das dinâmicas.

Em uma única semana de trabalho colaborativo, os envolvidos na Concepção serão capazes de compreender os objetivos do produto e os seus principais usuários. Nesta cerimônia será identificado um escopo funcional de alto nível e que deve ser suficiente para definir uma estratégia de lançamento incremental dos MVPs.

O "DiretoAoPonto" é uma "receita" (como o autor prefere chamar) que possibilita o entendimento e planejamento da entrega incremental de MVPs, organizando suas ideias e recursos em torno de um modelo que busca entender a principal finalidade do produto, considerando as jornadas dos usuários para realizar as entregas incrementais de produtos viáveis.

A receita "DiretoAoPonto" é definida por uma sequência de atividades cuja aplicação mostra-se rápida e eficaz (analogamente à receita de um bolo), conforme abaixo:

- Descreva a visão do produto;
- Priorize os objetivos do produto;
- Descreva os principais usuários, seus perfis e as suas necessidades;
- Entenda as principais funcionalidades;
- Compreenda os níveis de incerteza, tais como o esforço e valor de negócio por funcionalidade;
- Descreva as jornadas mais importantes dos usuários;
- Crie um plano de entrega incremental do produto, impulsionado pelo conceito de MVP (Sequenciador de *Features* e Canvas MVP);
- Estime o esforço por amostragem;
- Calcule os custos e especifique datas e cronograma de entrega.

O Sequenciador de *Features* é o principal artefato definido durante a Concepção. Ele consolidada o planejamento de *releases* de um produto sendo trabalhado (MVPs), os quais configuram porções de código que incrementam e/ou aumentam o produto, facilitando sua compreensão pelos envolvidos em um projeto de *software* desenvolvido colaborativamente. Ele contém os recursos e a proposição de valor a ser entregue em cada porção parcial de *software* potencialmente entregável, tal como a sua ordem de liberação. Estas porções são consideradas incrementos, que constituem conjuntos de hipóteses validadas e orientadas pelos resultados iniciais, estimulando a sua correção ou a confirmação do curso que será utilizado para guiar os próximos incrementos. E estes são novos produtos mínimos adicionados aos produtos mínimos já validados; mais uma vez, produtos mínimos, entretanto viáveis para fazer novas verificações sobre o direcionamento.

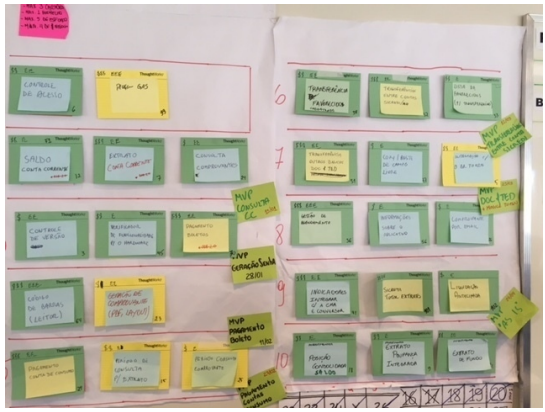


Figura 13 – Exemplo: sequenciador de *features*

Depois de definir o Sequenciador de *Features*, o time cria os canvas MVP para os primeiros MVPs identificados no sequenciador. No Canvas MVP, o time detalha cada MVP, de forma a esclarecer as seguintes perguntas:

1. **Visão do MVP** – Qual é a visão para este MVP?

2. **Métricas para validar as hipóteses** do negócio – Como podemos medir os resultados deste MVP?
3. **Resultado esperado** – Que aprendizado ou resultado estamos buscando neste MVP?
4. **Funcionalidades** – O que vamos construir neste MVP? Que ações serão simplificadas ou melhoradas neste MVP?
5. **Personas & Plataformas** – Para quem é este MVP? Em que plataforma estará disponível?
6. **Jornadas** – Quais jornadas são atendidas ou melhoradas com este MVP?
7. **Custo & Cronograma** – Qual é o custo e a data prevista para a entrega deste MVP?

Figura 14 – Exemplo: Canvas MVP1 e Canvas MVP 2 ao lado do sequenciador de features

Mais informações sobre esta receita podem ser obtidas na obra

"DiretoAoPonto", que apresenta um passo-a-passo de todas atividades que precisam ser realizadas durante o *workshop* da Concepção.

4. Representações Visuais

4.1 Fluxo Cumulativo

O Diagrama de Fluxo Cumulativo (ou em Inglês *Cumulative Flow Diagram*, CFD) é uma valiosa ferramenta de gerenciamento para:

- 1) rastrear e prever a realização de itens do trabalho;
- 2) indicar a necessidade de agir sobre o fluxo e o processo de melhoria.

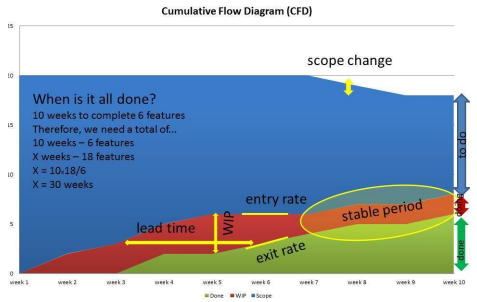


Gráfico 8 – Parâmetros de fluxo em um Diagrama de Fluxo Cumulativo

O CFD fornece uma representação gráfica do andamento do trabalho no sistema, tornando visíveis os gargalos e eventuais instabilidades. É uma ferramenta simples, porém muito eficaz para demonstrar o trabalho em andamento (WIP – *Work in Progress*, em inglês), taxa de entrada, taxa de saída, tempo de atravessamento, taxa de transferência, tempo decorrido, trabalho completo, trabalho restante e escopo total.

Veja abaixo uma sequência que mostra CFDs nas primeiras nove semanas de um projeto.

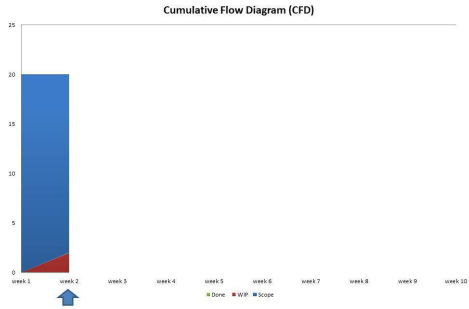


Gráfico 9 – CFD na Semana 2

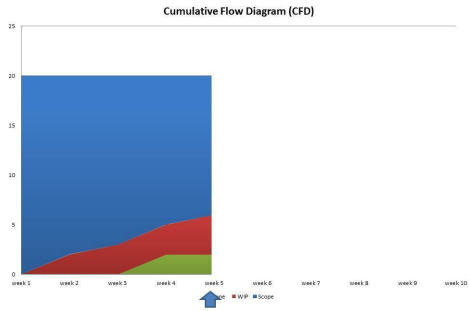


Gráfico 10 – CFD na Semana 5

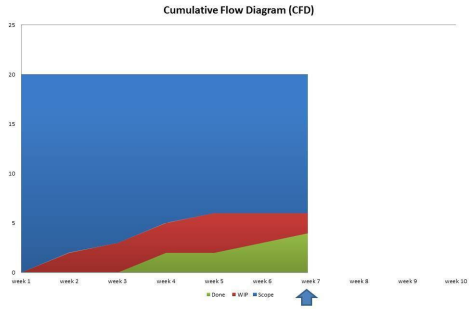


Gráfico 11 – CFD na Semana 7

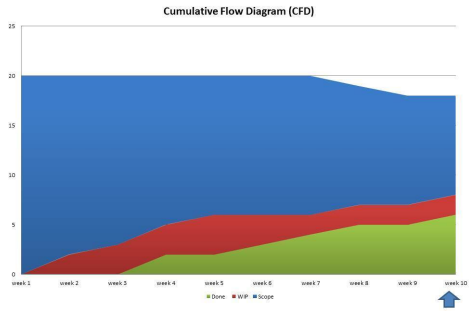


Gráfico 12 – CFD na Semana 10

4.1.1 Interpretando

No gráfico abaixo, você pode visualizar um outro CFD com muitas de suas propriedades. Cada um deles será explicado nas próximas seções.

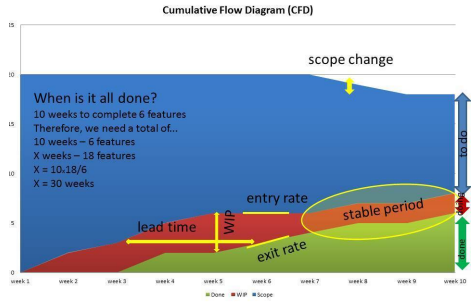


Gráfico 13 – Parâmetros de Fluxo em um Diagrama de Fluxo Cumulativo

Antes de tudo, é importante entender como o CFD é construído. O CFD apresentado é construído com base em uma tabela que é atualizada semanalmente. Abaixo, 3 demonstrada uma tabela com seu CFD correspondente:

time	Scope	WIP	Done
week 1	20	0	0
week 2	18	2	0
week 3	17	3	0
week 4	15	3	2
week 5	14	4	2
week 6	14	3	3
week 7	14	2	4
week 8	12	2	5
week 9	11	2	5
week 10	10	2	6

Tabela 1 – Modelo em Excel do CFD

Esta tabela descreve o fluxo de trabalho do sistema como: *escopo*

->WIP -> pronto; ou, em outras palavras, *a fazer -> fazendo -> feito*.

Muitas ferramentas constroem o CFD automaticamente para você, poupando o trabalho manual para a sua elaboração. Entretanto, para o seu aprendizado, é importante que você construa seu próprio CFD, simulando o que acontece em um projeto real.

Você pode fazer o download do modelo apresentado em http://www.caroli.org/cumulative-flow-diagram/.

> Itens de trabalho
>
> O número apresentado em cada célula da tabela representa a quantidade de itens de trabalho naquela etapa em uma semana específica. Neste contexto, é importante esclarecer que o item de trabalho é uma unidade que faz sentido para quem vai ler o CFD, como: funcionalidades, ponto de função, histórias, ponto de histórias, *bugs*, tarefas. No entanto, é essencial que não se misturem diferentes itens de trabalho em um mesmo CFD, ou seja, um CFD de funcionalidades deve conter somente funcionalidades, enquanto que um CFD de pontos de histórias deve conter somente pontos de histórias.

4.1.1.1 A fazer / fazendo / feito

Apesar de os CFDs serem comumente usados para fluxos de trabalho com muitas fases, é recomendado começar com um simples fluxo *a fazer -> fazendo -> feito* para entender completamente o seu funcionamento. Depois, quando você já tiver dominado o CFD e sentir a necessidade de mais dados, considere dividir a fase "fazendo" em um fluxo de trabalho mais detalhado.

A quantidade de trabalho em relação aos itens a fazer / fazendo / feito é descrita no CFD na imagem abaixo.

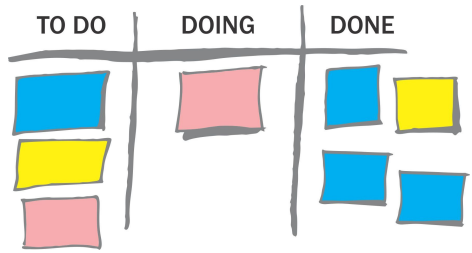


Figura 15 – Modelo *Kanban*: a fazer / fazendo / feito

4.1.1.2 Quando tudo estará pronto?

Quando todo o trabalho estará pronto? Esta é a pergunta mágica que todo mundo tenta responder. Porém, ao usar o CFD, há duas maneiras simples de respondê-la:

- 1) graficamente; ou
- 2) matematicamente.

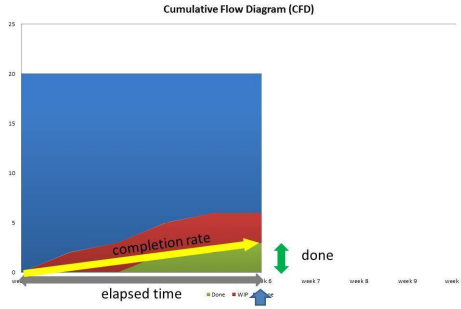


Gráfico 14 – Taxa de conclusão

Neste momento específico (representado pela seta azul na semana 6), você sabe a quantidade de trabalho que está pronta, e você sabe o tempo decorrido. Com esses dois parâmetros, você pode desenhar a linha da taxa de conclusão (a seta amarela no próximo gráfico) e você tem que responder a questão estendendo a linha da

taxa de conclusão até que ela alcance o total da linha do escopo de trabalho.

Apesar de poder fazer isso graficamente, há também a opção matemática: utilizando a regra de três.

A regra de três:

"*A regra de três, na matemática, é uma forma de se descobrir uma quantidade que tenha para outra conhecida a mesma relação que têm entre si entre outros dois valores numéricos conhecidos.*" (Wikipédia)

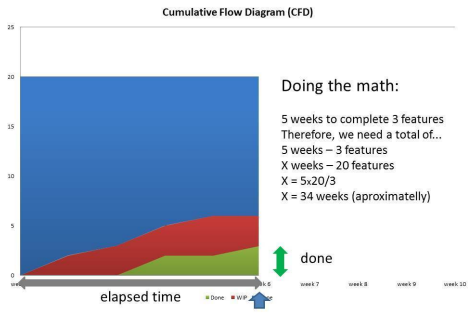


Gráfico 15 – Regra de três

4.1.1.3 Mudança no escopo

A linha do escopo é a linha horizontal, que computa o total de itens de trabalho. Essa linha é claramente definida e deve mudar apenas quando os itens de trabalho são adicionados ou removidos do escopo de trabalho.

O escopo total do CFD deve computar todos os itens de trabalho do sistema, sejam eles já feitos, em andamento, ou a fazer. Se um novo item de trabalho surge, ele deve ser adicionado no escopo de

trabalho total e a linha de trabalho total deve ser ajustada. Dessa forma, a nova linha torna fácil identificar quando um trabalho está sendo adicionado e também monitora quando itens de trabalho são removidos. Isso é ilustrado no gráfico abaixo:

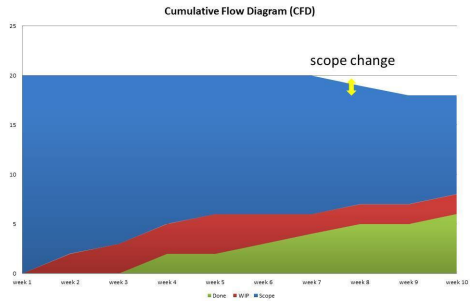


Gráfico 16 – O escopo foi reduzido

4.1.1.4 Trabalho em andamento

O trabalho em andamento (WIP – *Work in Progress*, em inglês), é o número de itens de trabalho atualmente em execução. Por exemplo, o cenário expresso na figura abaixo apresenta um *Kanban* com WIP de 1, onde: 3 itens de trabalho estão na fase "a fazer", 1 está sendo trabalhado (WIP) e 4 itens já estão feitos.

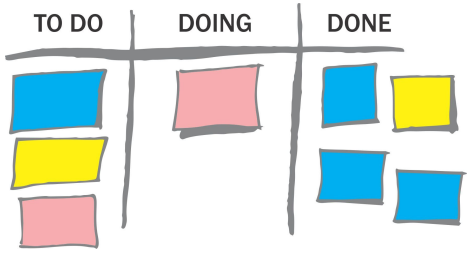


Figura 16 – Modelo kanban: a fazer / fazendo / feito

No CFD, o WIP é representado pela altura da linha vertical para a área WIP em um dado momento. O gráfico abaixo representa o

WIP na semana 5:

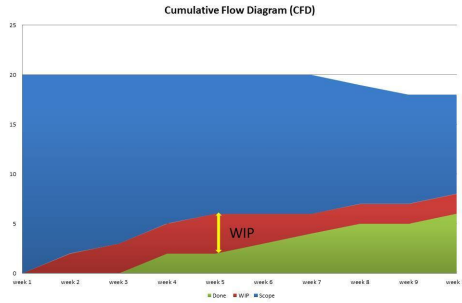


Gráfico 17 – WIP no CFD

4.1.1.5 Tempo de atravessamento

O tempo de atravessamento (*lead time*, em inglês) é o tempo decorrido entre o item de trabalho ser adicionado no sistema (fase "fazendo") e ele sair do sistema (trabalho completado; a fase "feito"). Para o *Kanban* "a fazer / fazendo / feito", apresentado anteriormente, o tempo de atravessamento é o tempo que leva para completar o item de trabalho (e apenas este, dado que o WIP é igual a 1) na fase "fazendo".

Como ocorre frequentemente, mais de um item estará em andamento na fase "fazendo". E por esta razão, normalmente a questão a ser respondida é: "em média, quanto tempo leva para concluir um item de trabalho?"

A resposta para essa pergunta é demonstrada no CFD, sendo que a imagem abaixo mostra uma linha horizontal, que representa o tempo de atravessamento. Mas então, quanto tempo leva para um item de trabalho passar pelo sistema? Ou ainda, fazendo a mesma pergunta com outras palavras: quanto tempo leva para um item de trabalho ir da fase "fazendo" para a fase "feito" (da área azul para a área verde no CFD)?

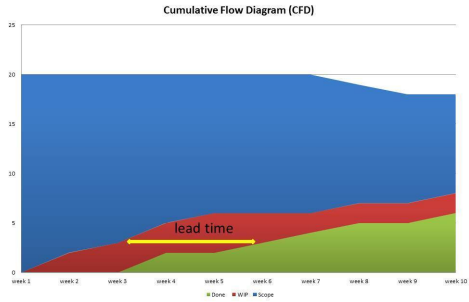


Gráfico 18 – Tempo de atravessamento no CFD

4.1.1.6 Taxa de transferência

A taxa de transferência é a vazão (quantidade por tempo) dos itens de trabalho passando no sistema. No CFD, a taxa de transferência é descrita pelo ângulo da linha dos itens de trabalho concluídos (a linha entre as áreas verde e vermelha). Abaixo, dois gráficos com duas marcas no CFD, comparando dois momentos distintos - note que o segundo gráfico exibe uma taxa de transferência bem mais baixa.

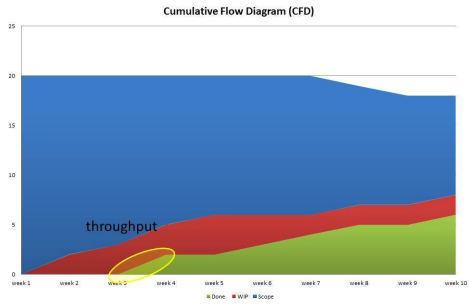


Gráfico 19 – Taxa de transferência no CFD no momento 1

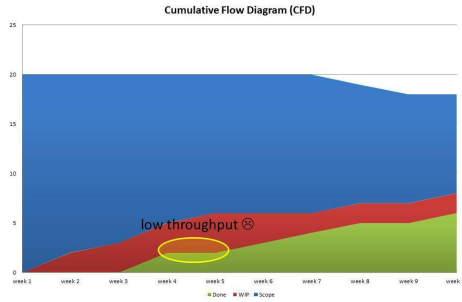


Gráfico 20 – Taxa de transferência no CFD no momento 2

4.1.1.7 Lei de *Little*

> Lei de Little: *O número médio de itens de trabalho em um sistema estável é igual à sua taxa de conclusão, multiplicado pelo tempo médio no sistema.* ∼ John Little, 1961

O texto acima é de *A proof for the Queuing Formula*, de John Little (1961); e é conhecido como a "Lei de Little".

John Little estudou e provou a relação entre o WIP e o tempo de atravessamento. Como ela nada mais é do que uma simples equação de primeiro grau, ao resolvê-la, você será capaz de encontrar o tempo médio para os itens de trabalho em seu sistema.

Um bar de *whisky* mostra um bom exemplo de sistema estável para ilustrar como você pode aplicar a Lei de Little para entender e monitorar o WIP, a taxa de transferência e o tempo de atravessamento.

Figura 17 – Bar de whisky

Digamos que eu e a minha esposa tenhamos um acordo: o lado direito do bar é meu, e o lado esquerdo é dela. Eu só bebo *whisky*, que está no meu lado do bar, onde cabem 12 garrafas.

Sempre que uma garrafa acaba, eu a removo do bar. Então abro uma nova, e a adiciono ao bar. Meu bar é um sistema estável: a taxa de entrada das garrafas é igual a taxa de saída delas.

O número de garrafas de *whisky* no meu bar é constante: 12 garrafas. Por ano, eu termino, em média, 6 garrafas de *whisky*. Então, qual é o tempo médio que uma garrafa de *whisky* permanece no meu bar?

4.1.1.7.1 Vamos aplicar a Lei de *Little*

WIP = T x L, ou seja, o número médio de itens de trabalho em um sistema estável (WIP) é igual à taxa média de conclusão (T) x tempo médio no sistema (L).

Usando os termos do bar de *whisky*: 12 garrafas (**WIP**, ou

número de garrafas de *whisky* no bar) = 6 garrafas/ano (**Taxa de transferência**, ou taxa média de conclusão) x **Tempo de atravessamento**, ou tempo médio de permanência no bar.

12 garrafas = 6 garrafas/ano x Tempo de atravessamento.

Dessa forma, temos que o tempo médio que uma garrafa fica no bar é de 2 anos.

É contraintuitivo! Você viu a fórmula e teve a resposta, mas você tinha em mente 2 meses. Dois meses, e não dois anos!

Essa é uma confusão comum. Quando você lê que uma média de 6 garrafas são terminadas por ano, provavelmente deve ter feito um simples cálculo: 6 garrafas por ano é igual a uma garrafa a cada dois meses. E seria 2 meses para uma dada garrafa, se no bar coubesse apenas uma garrafa.

Pense: o consumo de *whisky* não é apenas de uma garrafa; todas estão abertas e sendo consumidas. Então, se apenas um copo é servido, uma garrafa tem seu conteúdo diminuído, enquanto as outras 11 não se movem. E o bar tem 12 garrafas abertas.

4.1.1.7.2 Então, como funciona?

Ter menos garrafas no bar significa que cada garrafa terminará mais rápido. Como averiguamos, por ano, 6 garrafas são esvaziadas. Mas o bar comporta 12 garrafas. E isso certamente afeta o tempo médio de espera.

Por um momento, esqueça da Lei de *Little*. Vamos a outro episódio: *whisky* é bom para o coração, e por esta razão, eu bebo uma pequena quantidade todo dia.

Para essa comparação, considere que o meu hábito de beber é bem constante, então no último verão, eu fui para uma casa de praia, onde passei dois meses e levei algumas roupas, meu *laptop* e uma garrafa de *whisky*. Como o carro estava cheio, eu não ia levar todo o bar. Eu escolhi uma garrafa lacrada e a levei comigo.

Na mosca! Aquela garrafa estava vazia em exatos 2 meses. Então, o que aconteceu?

É simples, e John *Little* provou isso muito bem: o WIP, ou trabalho em andamento, na casa de praia era 1; apenas uma garrafa.

E a taxa de transferência era a mesma: de 6 garrafas por ano. Mais uma vez, caímos em uma conta simples: WIP = T x L.

Em outras palavras, o número médio de itens de trabalho em um sistema estável (WIP) é igual à taxa média de conclusão (T) x tempo médio no sistema (L).

Usando os números do bar da casa de praia: 1 garrafa (WIP, ou número de garrafas no bar) = 6 garrafas/ano (taxa de transferência, ou taxa média de conclusão) x **Tempo de atravessamento**, ou tempo médio no bar.

1 garrafa = 6 garrafas/ano x Tempo de atravessamento

Dessa forma, o tempo médio que uma garrafa fica no bar é de 2 meses (1/6 de um ano).

WIP = T x L; o tempo de atravessamento médio é diretamente proporcional ao WIP, e esta relação é a taxa de transferência. Como provado no exemplo do bar de *whisky*, menos WIP significa entrega mais rápida de itens de trabalho (garrafas de *whisky*). Menos significa mais rápido!

Ok, você deve estar cansado de matemática nesse momento... O CFD mostra a mesma coisa. O próximo gráfico mostra o CFD tanto com a representação do WIP como a do tempo de atravessamento, em dois diferentes momentos. Nela, você vai ver a questão principal da Lei de Little: WIP menor representa tempo de atravessamento mais curto.

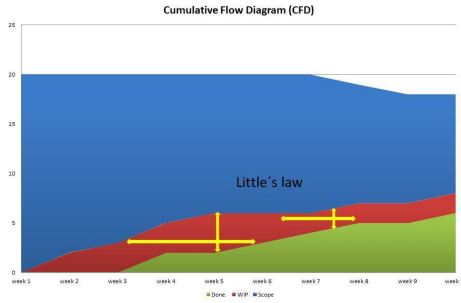


Gráfico 21 – Lei de Little no CFD

Note que o CFD apresentado foi obtido de um projeto real, provando o seguinte: uma vez que o WIP foi reduzido, o tempo de atravessamento médio também o foi.

4.1.2 O sistema é estável?

Esta frase descreve como as garrafas de *whisky* são removidas e adicionadas ao bar:

> *Sempre que uma garrafa termina, ela é removida do bar. Então, uma nova é aberta e colocada no bar.*

E nela, encontramos dois conceitos de sistema importantes: "Sistema Puxado" e "Sistema Estável". Veremos a seguir, mais informações sobre eles.

4.1.2.1 Sistema puxado

O Sistema Puxado descreve o movimento de itens de trabalho guiados pela demanda. No exemplo do bar, uma garrafa terminada abre uma vaga no bar. Assim, ela cria uma demanda para uma nova garrafa ser aberta e colocada no bar.

Essencialmente, o movimento dos itens de trabalho (garrafas de *whisky*) é guiado pela demanda vigente: uma garrafa é removida do bar, abrindo espaço para uma nova que será prontamente adicionada ao bar, ocupando o espaço vazio.

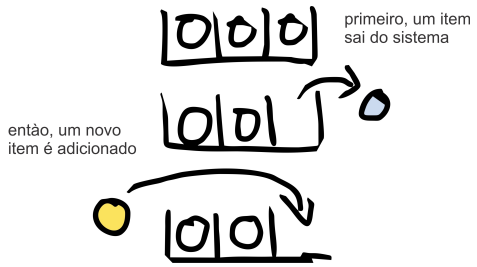


Figura 18 – Sistema Puxado

A manufatura *lean* descreve o "Sistema Puxado" em relação a um produto, que é puxado pelo sistema em vez de empurrado por ele. Um exemplo de um "Sistema Empurrado" seria adicionar garrafas ao bar sem que nenhuma garrafa tenho sido removida dele. Basicamente, novas garrafas seriam adicionadas sem qualquer demanda (espaço no bar) ser criada.

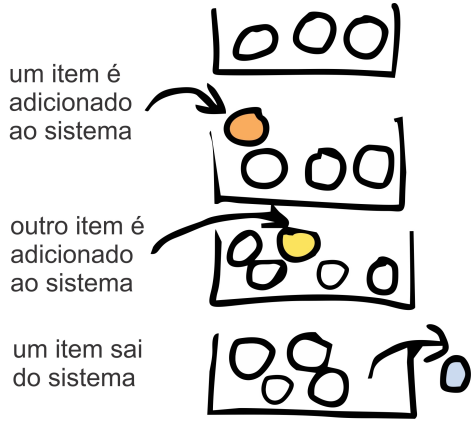


Figura 19 – Sistema Empurrado

Nos anos 80, a Ford Motors e a Toyota eram grandes exemplos de sistemas empurrados e sistemas puxados, respectivamente. Seguindo um sistema empurrado, a Ford produzia grandes quantidades de carros que ficavam nos pátios das fábricas e lojas esperando pelos clientes. Por outro lado, seguindo um Sistema Puxado, a Toyota focava na manufatura rápida de um carro customizado, assim que uma nova demanda era requerida por um cliente fazendo uma compra.

4.1.2.2 Um sistema estável

O bar de *whisky* é um sistema estável: a taxa de entrada de garrafas no bar é a taxa de saída das mesmas. Simples assim!

> *Um sistema estável é um sistema para o qual a taxa de entrada se iguala à taxa de saída* [1].

John *Little* definiu sistema estável em sua pesquisa, contribuindo também com trabalhos sobre medições e melhorias de processos.

4.1.2.3 Estabilidade do sistema no CFD

O CFD é seu melhor aliado para estabilizar o sistema. Nele, você pode claramente visualizar as taxas de entrada e saída. Os próximos três gráficos retratam:

- 1) a taxa de entrada, isto é, a taxa na qual os itens estão sendo adicionados ao sistema;
- 2) a taxa de saída, isto é, a taxa na qual os itens estão saindo do sistema; e
- 3) um sistema estável, em que a taxa de entrada se iguala à taxa de saída.

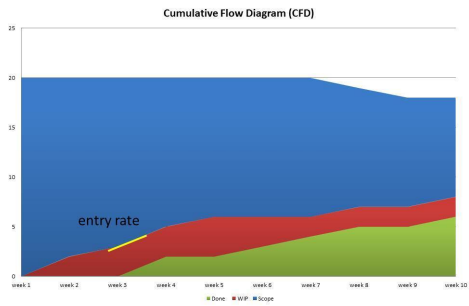


Gráfico 22 – Taxa de entrada no CFD

[1] *A Proof for the Queuing Formulaî, por Little J. D. C.* (1961)

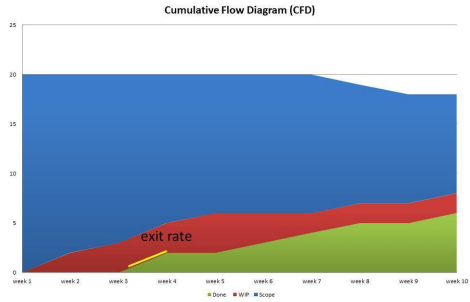


Gráfico 23 – Taxa de saída no CFD

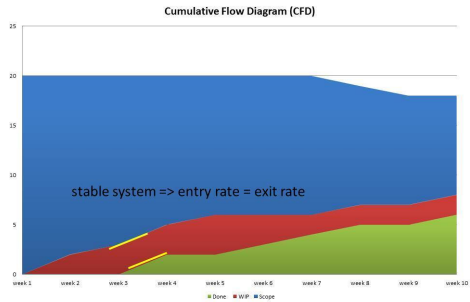


Gráfico 24 – Sistema estável no CFD

Enquanto as taxas de entrada e saída não forem iguais, o sistema estará instável. Os próximos dois gráficos mostram dois momentos em que o sistema está instável. No primeiro, a taxa de entrada está maior que a taxa de saída, fazendo com que o WIP aumente e o sistema fique em risco de sobrecarga.

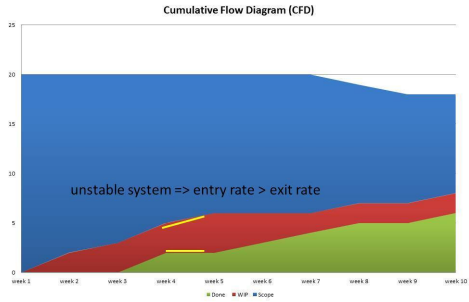


Gráfico 25 – Sistema instável: sobrecarga

No segundo gráfico, a taxa de saída é maior que a de entrada, fazendo com que o WIP diminua e o sistema fique sob risco de ficar sem itens de trabalho.

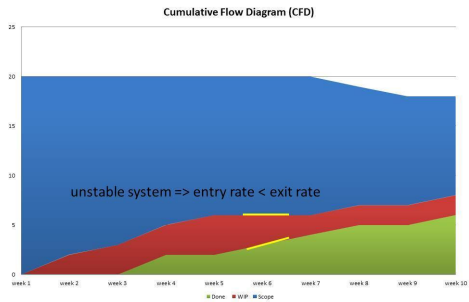


Gráfico 26 – Sistema instável: pouco trabalho

Nestes CFDs, você pode, mais uma vez perceber a Lei de Little reduzir o WIP, reduzindo também o tempo de atravessamento. Note que o modelo do CFD tem dados de um projeto real, e que períodos instáveis acontecem... mas o ponto principal é que o CFD é uma ferramenta para perceber e agir sobre a instabilidade. O próximos dois gráficos demonstrarão um projeto-modelo.

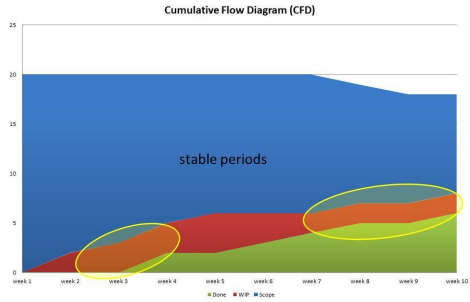


Gráfico 27 – Períodos estáveis no CFD

O projeto começou estável, depois passou por um momento de instabilidade (no qual o time agiu sobre ele), e então seguiu para outro período estável.

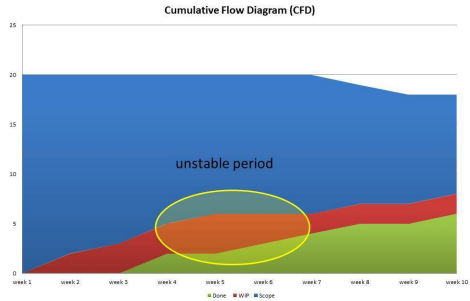


Gráfico 28 – Período instável no CFD

4.2 *Burn-up*

O *Burn-up* de *Features* do MVP contribui com o gerenciamento de tempo e escopo de um MVP. Ter o gráfico *burn-up* visível para todos constrói a confiança na gestão do tempo e no progresso das *features* do MVP. Desta forma, o *Burn-up* mostra-se como uma ferramenta essencial para dar visibilidade ao planejamento e para

realizar o acompanhamento da construção das *features* do MVP.

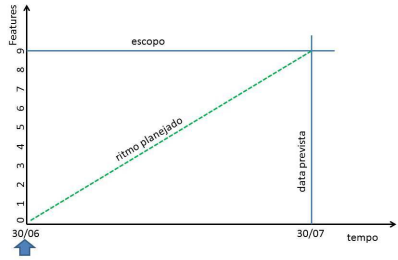


Gráfico 29 – *Burn-up* de *features* do MVP no dia 30/06

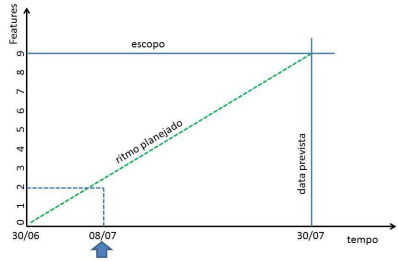


Gráfico 30 – *Burn-up* de *features* do MVP no dia 08/07

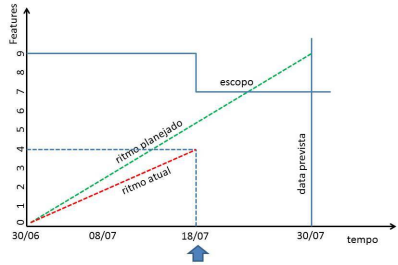


Gráfico 31 – *Burn-up* de *features* do MVP no dia 18/07

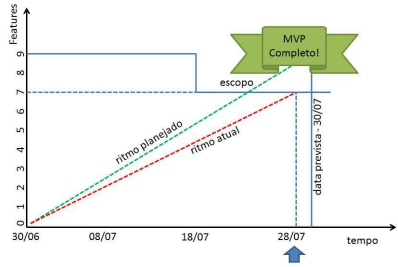


Gráfico 32 – *Burn-up* de *features* do MVP no dia 28/07

A sequência de gráficos mostra um exemplo de *burn-up* em momentos diferentes. Começando no dia 30/06, quando o *burn-up* foi criado com o planejamento de acordo com o ritmo esperado de construção das *features*. Seguindo com instantâneos do *burn-up* nos dias 08/07, 18/07 e 28/07, quando todas as *features* do MVP terminaram, e o mesmo foi entregue.

4.2.1 Eixos do *burn-up*

O eixo vertical é a quantificação das *features* para o MVP, e é medido em unidades (por exemplo: 9 *features*). O eixo horizontal

representa o tempo, normalmente medido em dias ou semanas.

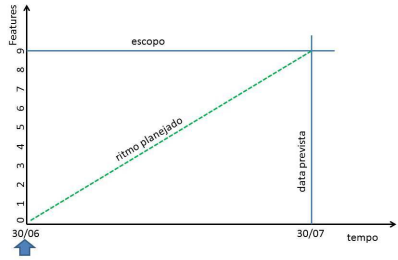


Gráfico 33 – Exemplo de *burn-up* de MVP iniciado em 30/06 e planejado para 30/07

4.2.2 No ritmo da construção do MVP

A vantagem da gráfico *burn-up* é a visão compartilhada do que deve ser alcançado. E isto fica claramente visível traçando uma linha horizontal de escopo e uma linha vertical de data de entrega do MVP, já que a intersecção dessas linhas representa o resultado no tempo esperado.

Ao desenhar uma linha diagonal a partir do ponto de partida (o início do tempo para a construção da primeira *feature* do MVP) para o resultado no tempo esperado, você tem uma indicação do ritmo de construção do MVP. No gráfico, este ritmo foi representado como a linha diagonal (planejado).

4.2.3 O progresso em gráficos

De tempos em tempos, você deve verificar a quantidade de *features* já construídas e a quantidade total de *features* planejadas para o MVP, sendo que a distância entre as linhas horizontais marcando as features atualmente prontas e a última *feature* a ser construída é a indicação da quantidade de *features* restantes.

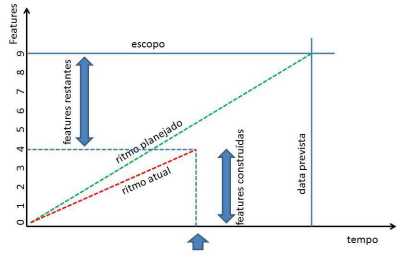


Gráfico 34 – *Features* construídas e restantes

Quando as duas linhas se encontrarem, todas *features* do MVP estarão completas. Atente então, que a distância entre essas linhas é uma medida poderosa de quão perto você está de completar o MVP.

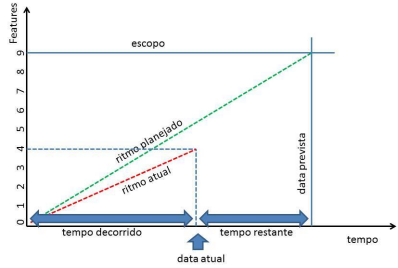


Gráfico 35 – Verificando progresso

Verificar regularmente o progresso é uma parte importante da gestão da construção do MVP, e há duas atualizações básicas para o *burn-up* de *features* do MVP:

- (1) o tempo mudou - a seta que representa a data atual deve ser movida para a direita até a posição correspondente, e a linha da data atual deve ser ajustada;
- (2) terminou a construção de uma feature - o total de *features*

completas deve ser alterado, e a linha de total de *features* completas deve ser ajustada.

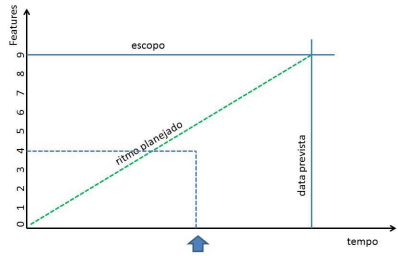


Gráfico 36 – Atualizações no *burn-up*: data atual e *feature* construída

Este mecanismo de atualizações de linha de escopo e linha do tempo permite identificar, de imediato, um desvio na duração esperada para a construção do MVP. Assim que constatado, este problema deve ser discutido e ações corretivas devem ser tomadas o quanto antes possível.

4.2.4 Linha de escopo de um MVP

Uma informação importante do gráfico *burn-up* é a linha de escopo do MVP: a linha horizontal contabilizando o total de *features* planejadas para o MVP. Esta linha define claramente se, e quando, novas *features* foram adicionadas ou removidas durante a construção do MVP. Ela também permite que você visualize a intersecção desta linha horizontal para a linha vertical, que representa a data planejada para a entrega do MVP.

Todas as partes a serem construídas para um MVP devem ser *features*. Se novas *features* surgem, elas devem ser adicionadas à lista de *features* e a linha de escopo deveria ser ajustada. Dessa forma, a nova linha permitiria identificar facilmente quando *features* estão sendo adicionadas, o que poderá afetar o tempo de

conclusão do MVP. O ato de adicionar uma nova *feature* é um sinal importante de que o tempo restante para a construção do MVP deve ser repensado. A linha de escopo também rastreia onde *features* estão sendo removidas para cumprir um prazo fixo. Este cenário é ilustrado no gráfico abaixo:

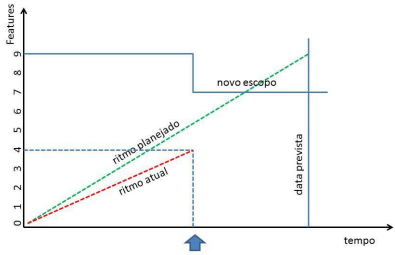


Gráfico 37 – Exemplo de *burn-up* de MVP após redução de duas *features*

Novamente, é importante entender como a remoção de uma *feature* do MVP vai afetar as outras *features*, e é algo que precisa e deve ser claramente discutido com todos.

5. Pontos de Função: uma medida de valor

Alan Albrecht trabalhou na IBM na década de 70. Lá, ele realizou experimentos relacionados à medição funcional de *software*. Utilizando principalmente métodos empíricos, seu estudo resultou na criação da "Análise de Pontos de Função" (APF).

A APF representou um grande avanço à Engenharia de *Software*, pois passou a utilizar a visão do usuário da aplicação como entrada do processo de medição, em detrimento à perspectiva física, que é orientada pela forma como o *software* é construído (ALBRECHT, 1979). Ela se tornou bastante popular a partir da década de 80, principalmente devido ao engajamento dos seus usuários, que organizaram-se sob uma organização sem fins lucrativos, batizada IFPUG (2010).

O IFPUG (*International Function Points Users Group*) assumiu a responsabilidade sobre a publicação e manutenção do *CPM - Counting Practices Manual*, o manual da APF. Ele se encontra atualmente na sua versão 4.3.1 e é sempre definido como a referência principal a respeito do processo e regras de contagem de pontos de função do IFPUG.

Na década de 90, o volume de utilizadores de APF tornou-se ainda mais significativo, e sua efetividade culminou na definição de um padrão de *facto*, através da ISO/IEC 20.926, norma também aderente à ISO/IEC 14.143-1, que trata do *framework* de medição funcional.

A ISO/IEC 14.143-1 distingue os dois subconjuntos de requisitos do usuário (RU): os "Requisitos Funcionais do Usuário" (RFU) e os "Requisitos Não-Funcionais" (RNF). Sendo assim, uma medição funcional atua apenas sobre o primeiro grupo de requisitos (IFPUG, 2010). Outros métodos de medição também foram criados conforme tais definições, como por exemplo, o SNAP (2015) e o COSMIC

(2015).

O SNAP (Processo de Avaliação Não Funcional) também é um método criado e mantido pelo IFPUG. Ele cobre a medição dos requisitos não funcionais do usuário e é utilizado como um complemento à APF (IFPUG). Embora atualmente em uso por algumas organizações, com relativo sucesso, cabe ainda alguma cautela ao utilizá-lo, pelos motivos que apresentaremos neste capítulo.

O COSMIC, também um método "mais jovem" no mercado, é considerado o primeiro integrante da segunda geração de métricas de *software*. Ele consegue ser ainda mais abrangente e maduro, tendo despertado muito interesse nas comunidades de usuários de métricas. Há, inclusive, muitos usuários experientes com amplo reconhecimento nos produtos do IFPUG apoiando a sua evolução.

O COSMIC já tem também seu próprio padrão: a ISO/IEC 19.761. E embora seja mais utilizado no mercado internacional (México, Holanda, Reino Unido, China, Índia, Colômbia, Itália e outros países), no Brasil, neste momento há apenas uma grande organização que o utiliza: uma instituição financeira cooperativa com atuação nacional, com sede no sul do país.

Agora que você está contextualizado, o guiaremos em relação à utilização das abordagens desenvolvidas: "IFPUG" e "COSMIC". Considere que embora ambas as abordagens, do façam referência à "Análise de Pontos de Função", chamaremos a primeira "APF-IFPUG" (ou simplesmente "APF") e a segunda "APF-COSMIC" (ou COSMIC), isso com o intuito de sermos mais objetivos nas citações.

5.1 Abordagem do IFPUG

A denonimação "abordagem do IFPUG" refere-se à utilização do método de medição funcional mantido pelo órgão IFPUG (*International Function Points Users Group*), assim como métodos relacionados e eventuais customizações.

5.1.1 O método APF

Como já referimos, a Análise de Pontos de Função (APF, ou ainda *FPA* - de *Function Points Analysis*, em inglês) é o método padrão do IFPUG para a medição funcional de *software*. Aplicando-o, é possível quantificar a funcionalidade solicitada e entregue sob o ponto de vista do usuário, tanto em termos de tarefas e serviços, como de suas estruturas de dados. Em outras palavras, isso provê a capacidade de quantificar requisitos funcionais do usuário, dimensionando as funções sobre as quais são construídos os sistemas de informação.

Como um método de medição funcional, a APF é totalmente independentemente da tecnologia ou mesmo de estratégias de projeto definidas para um desenvolvimento de *software*. Algumas de suas principais características são: não ser manipulável em sua essência; objetividade; e avaliação com base em "Funções de Transação" e "Funções de Dados" - há 5 tipos de função (Quadro 2) e 3 níveis de complexidade (Tabela 2, Tabela 3 e Tabela 4).

Funções	Tipos de Funções		
	Sigla	Nome	Conceito
De Dados	ALI	Arquivo Lógico Interno	Estrutura formada por grupos de dados relacionados e que armazena informações internas ao sistema.
	AIE	Arquivo de Interface Externa	Estrutura formada por grupos de dados relacionados e que armazena as informações externas ao sistema (mantida por outro sistema e apenas referenciadas pelo sistema em foco).
De Transação	EE	Entrada Externa	Processo elementar que primariamente mantém um ALI ou modifica o comportamento da aplicação.
	CE	Consulta Externa	Processo elementar que primariamente consulta dados de um ou mais ALI ou AIE e apresenta-os a um usuário.
	SE	Saída Externa	Processo elementar que primariamente consulta dados de um ou mais ALI ou AIE e apresenta-os a um usuário, mas adicionalmente executa cálculos, fórmulas, mantém um ALI e/ou altera o comportamento do sistema.

Quadro 2 - Tipos de Função

Em uma contagem de pontos de função, uma "Função de Dado" é classificada como "Arquivo Lógico Interno" ("ALI") ou "Arquivo de Interface Externa" ("AIE"). O estimador deverá distiguí-las e depois avaliar sua complexidade. Isso é realizado através de critérios estabelecidos no CPM, considerando a quantificação de dados únicos que são reconhecidos pelos usuários da aplicação e conforme

eles são dispostos nos agrupamentos de dados. Para este tipo de dado usamos a nomenclatura "Dado Elementar Referenciado" ("DER").

Os grupos lógicos que agrupam os tipos de dados nós denominamos "Registro Lógico Referenciado" ("RLR"). E com base na quantidade de "DER" e "RLR" podemos obter sua complexidade, utilizando para isso a Tabela 2 abaixo:

Qtde RLR	Qtde DER		
	De 1 a 19	De 20 a 50	51 ou mais
Apenas 1	Baixa	Baixa	Média
De 2 a 5	Baixa	Média	Alta
6 ou mais	Média	Alta	Alta

Tabela 2 - Complexidade de um Função de Dado na APF

Quando uma contagem envolve uma "Função de Transação", ela será classificada como "EE", "CE" ou "SE", a depender da sua intenção e características. Em seguida, serão identificados e quantificados os dados elementares informados e/ou disponibilizados na interface entre usuários e a aplicação, os quais também chamamos "DER". Este quantitativo, em conjunto com o total de "Arquivos Lógicos Referenciados" ("ALR") de uma transação sendo contada, define também sua complexidade, conforme critérios pré-definidos utilizados para funções de transação, os quais podem ser visualizados na Tabela 3 e na Tabela 4 a seguir:

Qtde ALR	Qtde DER		
	De 1 a 4	De 5 a 15	16 ou mais
0 ou 1	Baixa	Baixa	Média
2	Baixa	Média	Alta
3 ou mais	Média	Alta	Alta

Tabela 3 - Complexidade do Tipo de Função EE na APF

Qtde ALR	Qtde DER		
	De 1 a 5	De 6 a 19	20 ou mais
0* ou 1	Baixa	Baixa	Média
2 ou 3	Baixa	Média	Alta
4 ou mais	Média	Alta	Alta

** Uma CE não pode possuir 0 ALR.*

Tabela 4 - Complexidade dos Tipos de Função CE e SE na APF

Considerando um nível de complexidade e tipo de função, há um peso correspondente, que chamamos "contribuição". Ele representa uma valoração da função em pontos de função (ou seja, a quantidade de pontos conforme o tipo de transação e a complexidade correspondente - identificada com base nos critérios de "DER" e "RLR"/"ALR"). O relacionamento entre complexidade e contribuição de funções pode ser visualizado na Tabela 5.

	Complexidade		
Tipo de Função	Alta	Média	Baixa
ALI	15	10	7
AIE	10	7	5
EE	6	4	3
CE	6	4	3
SE	7	5	4

Tabela 5 - Complexidade versus Contribuição (IFPUG)

É importante esclarecer que o Manual de Práticas de Contagem de Pontos de Função (CPM) define as regras citadas e tem o intuito de orientar a medição do tamanho funcional de um projeto ou aplicação de *software*. Sendo assim, você encontrará nele tudo aquilo que precisa sobre os conceitos relacionados à aplicação do método, assim como detalhes a respeito do processo e respectivas regras e definições, algumas aqui omitidas devido a fins didáticos.

Você também poderá utilizar outros métodos e manuais para complementar o padrão do IFPUG, ou ainda, convenções criadas com base em analogias (abordaremos este assunto novamente mais tarde). Isso é uma prática comum, embora não constem nos docu-

mentos oficiais do IFPUG, visto que são necessidades especiais e que normalmente não tratam apenas aspectos de medição de tamanho funcional (tamanho) - muitas vezes contemplam derivações para elaboração de estimativas de esforço, prazo, custo e/ou qualidade de projetos de *software*.

5.1.1.1 Estimativas

Diferentemente de uma indicação ou estimativa de tamanho, uma medição de tamanho funcional exige maior detalhamento dos requisitos do usuário, mas é mais confiável, pois viabiliza uma quantificação exata dos itens do escopo de um projeto ou aplicação de *software*. No entanto, nem sempre há informações e/ou tempo suficiente para a realização de uma medição, como por exemplo, quando nas fases iniciais do ciclo de vida.

Contudo, é também possível realizar aproximações de tamanho com o método de pontos de função do IFPUG, mesmo que durante fases iniciais. Há de se avaliar, no entanto, o quanto isso é necessário e/ou agrega à avaliação. Normalmente, por estes motivos, uma Estimativa NESMA costuma ser aplicada nestes casos, sendo este método mantido pela organização NESMA.

Como a Estimativa NESMA foi concebida com base na própria APF, é totalmente compatível à APF (IFPUG). Ela funciona da seguinte forma: atribui-se a complexidade baixa da APF para todas as funções de dados (repositórios, definidos como ALI ou AIE) e a complexidade média para todas as funções de transação (definidas como EE, CE ou SE), resultando assim, no total de pontos de função de uma função específica (contribuição conforme definida na Tabela 6).

Tipo de Função	Complexidade
ALI	7
AIE	5
EE	4
CE	4
SE	5

Tabela 6 - Tipos de Função *versus* Contribuição - Estimativa NESMA

O método de Estimativa NESMA foi especialmente projetado para ser utilizado em situações onde ainda não conseguimos visualizar o detalhamento das funcionalidades do escopo em análise. E desta forma, a sua aplicação resume-se basicamente na identificação de funções e respetiva classificação. Destaca-se ainda, que o desvio entre uma Estimativa NESMA e uma contagem IFPUG detalhada costuma ser bastante aceitável e é perfeitamente passível de predição através do estabelecimento de fatores de escala estabelecidos a partir de análises históricas.

Na prática, especialmente em relação a desvios, um fator de escala pode ser obtido facilmente visualizando-se a relação existente entre o tamanho funcional inicial estimado e o tamanho funcional final medido em alguns projetos. Como exemplo: ao estimar um projeto 100 PF no seu início, sendo constatados 120 PF ao seu final, identificou-se uma razão similar durante a execução de outros N projetos. Desta forma, poderíamos definir que o fator de escala a ser considerado numa estimativa inicial futura seria de aproximadamente 1,2, ou seja, tendo um novo projeto de 50 PF e considerando o fator de 1,2, o produto resultante a ser utilizado como estimativa teoricamente mais confiável, ficaria na ordem de 60 PF.

5.1.2 O método SNAP

O Processo de Avaliação Não Funcional de *Software* (SNAP) do IFPUG está atualmente disponível em inglês na versão 2.3 (no idioma português, estamos na versão 2.2).

O SNAP é um modelo matemático paramétrico que pode ser

utilizado para avaliar e quantificar de forma objetiva os requisitos não funcionais de *software*. Ele tem como unidade de medida o "Ponto SNAP" e possibilita avaliar tanto requisitos técnicos como de qualidade. Permite ainda, que as organizações construam repositórios de dados históricos para suportar o processo de tomada de decisão.

Contudo, o objetivo principal do projeto que criou o SNAP, era garantir que um *framework* não funcional pudesse ser utilizado para estabelecer uma ligação entre o tamanho não funcional e o esforço necessário para prover os requisitos não funcionais. E para isso, foram definidas algumas características do método:

- O *framework* como um todo é uma avaliação do tamanho dos requisitos não funcionais;
- O *framework* é composto de categorias e subcategorias de avaliação;
- As subcategorias são avaliadas utilizando critérios específicos;
- A avaliação utiliza ambos os critérios de avaliação e/ou de medição.

É importante ressaltar também, que o SNAP foi especialmente projetado para obter os seguintes resultados após realizada uma avaliação não funcional:

- Poder ser utilizado em conjunto com o tamanho funcional e auxiliar na explicação da variação do esforço de desenvolvimento e da produtividade;
- Juntamente com o tamanho funcional, poder ser utilizado como entrada para modelos de estimativas;
- Ser determinado a partir de uma visão não funcional do usuário entendida e acordada entre o usuário e as equipes de desenvolvimento.

Conforme o IFPUG (2015), uma avaliação não funcional deveria poder auxiliar as organizações a prover informações relevantes sobre projetos e aplicações, e também apoiá-las na realização de suas estimativas e análises de qualidade e produtividade. Isso tornaria viável aos profissionais de *software*:

- Planejar e estimar melhor os projetos;
- Identificar áreas de melhoria de processos;
- Auxiliar na determinação de estratégias técnicas futuras;
- Quantificar os impactos das estratégias técnicas atuais;
- Prover dados específicos sobre questões não funcionais na comunicação com diversos públicos.

O modelo de referência definido para o SNAP é estruturado em 4 categorias e 14 subcategorias de medição, sendo que os requisitos não funcionais são mapeados para sub-categorias relevantes.

O conjunto de categorias e subcategorias avaliadas no SNAP é relacionado abaixo (IFPUG, 2015):

- 1. Operações de Dados
  - 1.1. Validações na Entrada de Dados
  - 1.2. Operações Lógicas e Matemáticas
  - 1.3. Formatação de Dados
  - 1.4. Movimentações de Dados Internos
  - 1.5. Entregar valor agregado aos usuários por configuração de dados
- 1. Projeto de Interface
  - 2.1. Interfaces do Usuário
  - 2.2. Métodos de Ajuda
  - 2.3. Múltiplos Métodos de Entrada
  - 2.4. Múltiplos Métodos de Saída
- 1. Ambiente Técnico
  - 3.1. Múltiplas Plataformas
  - 3.2. Tecnologia de Banco de Dados
  - 3.3. Processos *Batch*

- 1. Arquitetura
  - 4.1. *Software* baseado em componentes
  - 4.2. Múltiplas Interfaces de Entrada / Saída

O SNAP é um método recente no mercado, e com base nas experiências que tivemos, sugerimos alguma cautela ao utilizá-lo. Desta forma, o nosso recado é: tente não fixar um custo ou produtividade junto a um fornecedor de *software* sem possuir alguma base histórica anterior (por exemplo, ao escalar o modelo com *outsoucing*). E se por ventura isso for feito, considere a adoção de índices de produtividade separados para cada categoria ou subcategoria do modelo.

Outro ponto a destacar, é que caso você opte por utilizar o SNAP, em se tratando de projetos de manutenção, ainda numa abordagem de *outsourcing* escalável, seria importante definir num guia local de métricas as orientações e condições para utilização do método apenas em situações onde uma funcionalidade é afetada exclusivamente por requisitos não funcionais, tornando assim a aplicação dos métodos mutuamente exclusiva para uma mesma função e facilitando suas análises.

Mesmo que alguns conceitos possam ser deturpados por customizações na APF quando a utilizamos por analogia, avalie também a utilização de modelos de cálculo de tamanho não funcional com base em derivações realizadas a partir do método APF, como o Roteiro de Métricas do SISP. Indicamos isso pois é muito mais fácil trabalhar com uma única unidade de medida ao realizar a gestão de produtividade e o planejamento e acompanhamento das demandas de *software*.

5.1.3 Considerações

Conforme o SISP (2015b), a remuneração do fornecedor de desenvolvimento de *software* depende da adoção de uma métrica de faturamento que seja objetiva e justa, dentro da legalidade e que não onere a contratante ou mesmo a contratada. Contudo, para projetos

com métodos ágeis, existem alguns desafios, como por exemplo, a remuneração de refinamento de requisitos entre iterações.

Segundo DEKKERS (2015), integrante ativa do IFPUG e que decorre sobre o assunto em questão, ao aplicar pontos de função em um contexto de utilização da metodologia ágil devem ser avaliadas algumas questões, tal como se o propósito é obter uma medição para comparar a produtividade de desenvolvimento ou se o objetivo é contratar um fornecedor utilizando um modelo de *outsourcing*. DEKKERS (2015) identifica também, que há um dilema sobre a consistência de aplicação da técnica entre os modelos tradicional e ágil, visto que dependendo da abordagem utilizada o tamanho funcional pode ser distorcido, e portanto, diferente ao compará-los, algo que não parece adequado em relação ao que rege o IFPUG (2010) - uma vez que a APF deve ser independente da forma de implementação. Assim, note que este aspecto inclui também a estratégia e a metodologia adotada pelo projeto, e não apenas a tecnologia utilizada na construção do produto.

Em sua proposta, DEKKERS (2015), ressalta ainda, o cuidado necessário ao considerar a definição de funcionalidade completa e defende que as funcionalidades já tratadas devem ser consideradas alteradas apenas quando existirem *releases* diferentes do *software* sendo entregues, ou seja, em um cenário onde uma mudança ocorre sobre algo já desenvolvido e potencialmente entregue (no *Lean* MVP, isso seria de certa forma, análogo ao nível de MVP). Por fim, ela ressalta que a funcionalidade deve, ou ao menos deveria, ser sempre medida de forma independente de como ela foi desenvolvida.

De qualquer forma, cabe ressaltar que a decisão sobre qual método utilizar ou do que, como e onde medir, relaciona-se diretamente aos propósitos de medição presentes no contexto da sua organização.

5.1.3.1 Sobre *Benchmarking*

Dentre os propósitos de uma contagem de pontos função, podemos elencar "tornar um projeto mensurável e comparável em relação a outros projetos da organização ou entre organizações e/ ou contextos similares". Isso pode ser feito como um padrão aplicável às estimativas de tamanho e/ou avaliações de esforço, prazo e custo dos projetos, considerando também os aspectos que tangem à qualidade apurada (ISBSG, 2013).

Na década de 90, foi fundado o *ISBSG* (*International Software Benchmarking Standards Group*), uma empresa sem fins lucrativos e que hoje consolida informações sobre cerca de 6.000 projetos de *software*, os quais são oriundos de diferentes países e organizações (contempla tanto projetos desenvolvidos por métodos "tradicionais", como por intermédio de práticas ágeis).

Atualmente, o principal produto do ISBSG é um relatório de *benchmarking* utilizado para avaliação de produtividade de *software*. Ele é denominado *Reifer Consultants LLC and ISBSG Joint Report: Software Productivity Benchmarks* (*ISBSG*, 2013) e é composto por análises matemáticas e estatísticas vinculadas a milhares de projetos de *software* medidos com a APF e/ou convertidos a partir de outros métodos.

O ISBSG representa uma excelente fonte de *benchmarking* às empresas que querem comparar seus dados, tais como "produtividade" e "qualidade".

No relatório, além de medidas de comparação de produtividade entre diferentes metodologias, também há dados por setores específicos da indústria ou famílias de tecnologias (geração de linguagem). Abaixo, uma lista de setores da indústria considerados:

- Finanças;
- Bancos;
- Manufatura;
- Eletrônicos/Computação;
- Governo;

- Seguros;
- Serviços.

Sobre as tecnologias, destaca-se ainda, que segundo o ISBSG (2013, pág. 51 e 57), no contexto dos cerca de 6000 projetos de *software* analisados, existem diferenças de produtividade mais significativas apenas considerando os distintos setores e gerações de linguagem analisadas (exemplo: terceira, quarta e quinta gerações – mas principalmente esta última em relação às demais). Além disso, a unidade base para análise de produtividade, no relatório citado, é a quantidade de pontos de função não ajustados produzidos por uma pessoa em um mês (UFP/ PM), referência comumente utilizada no exterior e que no Brasil é mais popular na forma de "índice de produtividade" (que adotamos utilizando horas por ponto de função).

5.2 Abordagem do COSMIC

A "abordagem do COSMIC" engloba o método de medição funcional mantido pela organização COSMIC (*Common Software Measurement International Consortium*), contemplando também algumas customizações e estimativas derivadas.

O método COSMIC atua sobre projetos de desenvolvimento e manutenção de *software*, onde os requisitos funcionais da aplicação são criados ou impactados. Além disso, dentre suas principais características, podemos destacar: o tamanho flutuante, sua maior precisão e a existência de diferentes níveis de granularidade ou decomposição dos requisitos funconais.

5.2.1 O método COSMIC

O método COSMIC (2015) de medição de tamanho funcional é um padrão internacional de domínio público projetado para atender tanto as necessidades de dimensionamento de aplicações de negócio, quanto de *software real-time* ou híbridos destes. Sua

aplicação ocorrer através de uma medição ou aproximação do tamanho funcional, que é realizada à luz dos requisitos funcionais do usuário. O resultado desta avaliação é expresso em unidades de Pontos de Função COSMIC (PFC).

Assim como o método do IFPUG, o COSMIC pode ser estendido através da definição de diretrizes locais, porém, nativamente, ele já reconhece que *software* é algo desenvolvido conforme um determinado domínio de aplicação. Existem também, diversos documentos oficiais do COSMIC que detalham a abordagem específica a ser utilizada para cada um dos domínios, o que facilita sua utilização.

Basicamente, uma medição deve ocorrer com base num processo de medição, que conforme definido no Manual do COSMIC, é executado pelo "Estimador" em três fases:

- **Estratégia de Medição**: uma avaliação dos objetivos da medição, incluindo a definição do propósito da medição e a determinação do escopo de cada pedaço de *software*, isso através do Modelo de Contexto de *Software*, que contempla a identificação de camadas e fronteiras impactadas, assim como dos usuários funcionais, o nível de granularidade e o nível de decomposição dos requsitos;
- **Mapeamento**: uma avaliação dos requisitos funcionais do usuário do *software* aplicando o Modelo Geral de *Software*, ou seja, organizando os requisitos em processos funcionais, para que seus movimentos sejam convertidos em um tamanho funcional (em "Pontos de Função COSMIC");
- **Medição**: associação dos movimentos de dados identificados aos processos e fronteiras identificadas, resultando num total de unidades de "Pontos de Função COSMIC (PFC)" por processo e total medido, no contexto de um pedaço de *software*, projeto ou aplicação em análise.

5.2.1.1 Fase de Estratégia

Nesta fase, avalia-se um ou mais propósito(s) da medição, como exemplo:

- Prover uma estimativa de tamanho ao projeto de *software*;
- Realizar análise a respeito da produtividade do(s) projeto(s) de *software*.

Quando realizada uma medição COSMIC (2015), o propósito da contagem determina o escopo a ser analisado em um projeto ou sistema específico, sendo este escopo algumas vezes subdividido em várias partes como pedaços de *software* ou componentes da arquitetura de *software*. Tais partes (ou pedaços) de *software*, por sua vez, interagem com os usuários funcionais da(s) aplicação(ões) envolvida(s). Sendo assim, o Estimador COSMIC deverá listar todos os tipos de usuários que interagem com cada parte do escopo em análise.

Normalmente será mais apropriado selecionar um "Nível de Granularidade" do "Processo Funcional" para realizar uma medição funcional, porém dependendo da qualidade e da origem dos requisitos funcionais disponíveis para a avaliação, o estimador poderá optar por definir um nível ainda mais alto, fazendo com que sua estimativa considere por exemplo, a visão de "Macro-Processo" ou de "*Feature*". Da mesma forma, dependendo do propósito e dos insumos disponíveis, poderá optar também por decompor a parte de software em componentes menores, utilizando a abordagem de "Decomposição". Como você provavelmente pode perceber, o método COSMIC é bastante flexivel e completo!

5.2.1.2 Fase de Mapeamento

Os usuários funcionais são comunicados sobre eventos do mundo real e disparam os processos funcionais. Esses processos movimentam grupos de dados vinculados aos objetos de interesse do usuário (físicos ou conceituais), seja com o objetivo de entrar, apresentar, gravar ou ler informações.

Desta forma, identificar eventos e respectivos processos disparados, assim como os objetos de interesse e os grupos de dados, são as principais questões a serem resolvidas e/ou analisadas durante esta fase.

5.2.1.3 Fase de Medição

Após a Fase de Mapeamento, o estimador deverá avaliar cada processo funcional, sempre considerando o propósito, escopo, fronteiras e os usuários identificados, além é claro, dos movimentos de dados relacionados aos grupos de dados identificados (regras aplicáveis encontram-se no manual do método).

Existem os seguintes tipos de movimentos de dados, para os quais deve-se contar 1 PFC para cada movimento impactado em um dado processo funcional:

- Entradas ("*E*");
- Saídas ("*X*");
- Leituras ("*R*");
- Escritas ("*W*").

Durante uma avaliação COSMIC, é importante destacar que um processo funcional novo contribuirá com no mínimo duas unidades de PFC, uma para cada tipo de movimento encontrado, sendo no mínimo 1 Entrada ("E") e 1 Saída ("S") ou 1 Escrita ("W"). Entretanto, não há um limite superior pré-estabelecido ou um *range*, o que caracteriza e distingue o método da maioria dos demais padrões de medição funcional, tornando-o mais versátil em alguns contextos (também tratando melhor a correlação entre o tamanho e o esforço, por exemplo).

Contudo, no caso de modificações em um processo existente, haverá no mínimo 1 PFC relacionado, também sem um teto estabelecido. Perceba que esta estratégia é mais elegante que a soleção padrão que adotamos ao utilizar o método do IFPUG, onde normalmente consideramos deflatores para tratar o cenário de alteração, por exemplo.

É importante ressaltar também, que devem ser registrados na contagem, eventuais defeitos de avaliação, assim como premissas utilizadas, como aquelas associadas às estimativas antecipadas e que trazem impacto ao planejamento de projetos de *software*.

Sendo assim, o resultado da fase de Medição, é o total de PFC medido ou estimado para cada processo funcional, considerando cada pedaço de *software* presente no escopo de contagem.

Vejamos abaixo (Figura 20), um exemplo de como as movimentações de dados poderiam ter sido identificadas supondo um novo requisito funcional que expressa a necessidade de "Cadastrar Pedido" (um requisito do Sistema de Pedidos de uma organização):

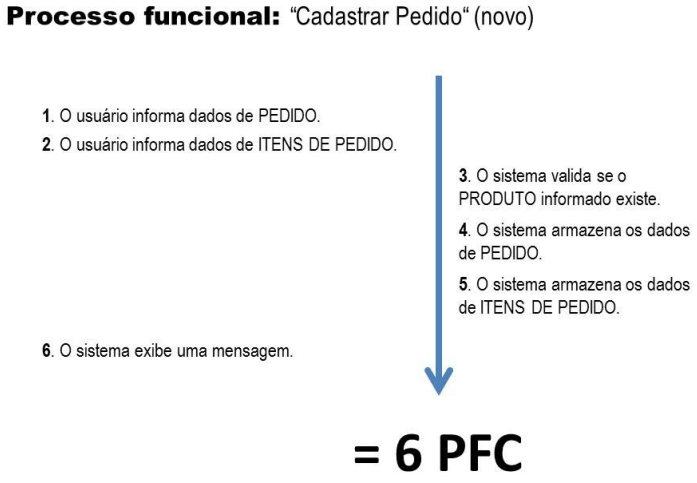


Figura 20 – Exemplo de inclusão de uma nova funcionalidade

E aqui um outro exemplo (Figura 21), considerando uma melhoria no mesmo processo, onde há a inclusão de 1 novo atributo no grupo de dados de "Pedido":

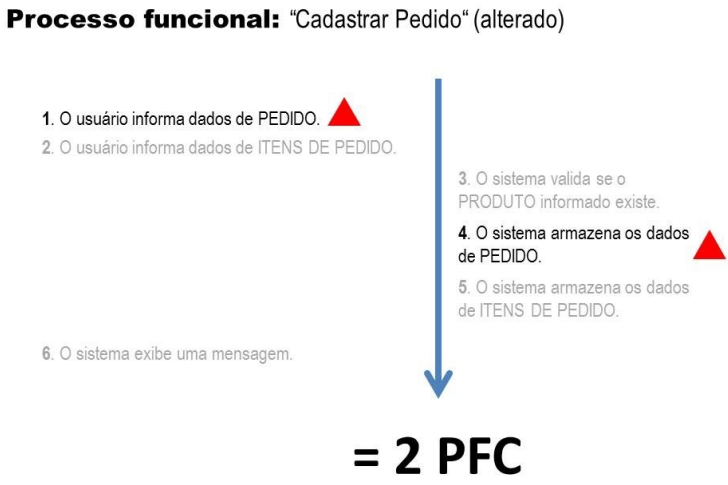


Figura 21 – Exemplo de uma alteração sobre uma funcionalidade

5.2.1.4 Estimativas

O método COSMIC (2015) foi desenvolvido para prover uma medição de tamanho funcional, mas por vezes o estimador precisará realizar apenas uma aproximação do tamanho, como por exemplo, nas situações citadas abaixo:

- Durante as fases iniciais do ciclo de vida de um projeto, onde os requisitos funcionais do usuário ainda não possuem um nível de detalhe suficiente para a realização de uma medição;
- Quando necessita-se obter uma ordem de grandeza de *software*, porém o tempo disponível para a sua avaliação é reduzido (é preciso uma maneira mais rápida para estimar);
- Quando a qualidade dos insumos disponíveis para a contagem não é suficiente para garantir o nível de precisão esperado.

Diferentemente da APF do IFPUG (IFPUG, 2010), onde um processo funcional pode ter no máximo três níveis de contribuição

os quais são atribuídos através de uma aferição de complexidade, no COSMIC o tamanho funcional de cada processo não define um limite superior de pontos de função possíveis, tampouco se relaciona com o termo "Complexidade", independente do seu significado.

Assim, abordagens como a da NESMA (2015), onde há uma contribuição generalista (referindo-se à atribuição da complexidade "baixa" para as funções de dados; e "média" para as funções de transação), não são utilizadas da mesma forma no COSMIC como o são na comparação de resultados entre uma Estimativa NESMA e o método APF do IFPUG.

O COSMIC possui um conjunto de diferentes abordagens de estimativa antecipada/rápida. Elas nos possibilitam calcular ou arbitrar sob um quantitativo de Pontos de Função COSMIC (PFC). No entanto, cada uma delas deve ser analisada conforme o propósito e a necessidade identificada pelo estimador. Consulte as orientações específicas no guia relacionado, na Base de Conhecimento mantida pela organização COSMIC.

A maioria das abordagens de estimativa antecipada (ou rápida) utilizam métodos de escala e/ou de classificação. Onde, a saber: os fatores de escala consistem de valores atribuídos a escalas (normalmente calibrados com base em dados locais); as classificações, por sua vez, são categorias onde vinculamos tais fatores de escala. É recomendável definir critérios de classificação objetivos para as categorias e classificações, pois isso permite que o método COSMIC seja aplicado consistentemente através das diferentes contagens realizadas. Exemplificando: para uma estimativa antecipada existem categorias pré-calibradas com dados locais, onde um valor *default* de PFC foi atribuído a cada categoria; ao avaliar um requisito qualquer nós o classificamos conforme a categoria, e sendo assim, um total de PFC terá por consequência sido atribuído.

Em suma, ao selecionar uma abordagem, faz-se necessário atentar:

- Para o nível de granularidade, a completude e a qualidade dos requisitos;

- Para a obtenção do maior nível de detalhe possível;
- Para os aspectos locais.

É necessário entender que esses métodos de estimativa tentam, na prática, calibrar valores médios para os processos funcionais (ou outro nível de granularidade), conforme os dados históricos gerados através das medições e estimativas realizadas no passado.

Todavia, você pode ainda se perguntar quanto ao que fazer para derivar estimativas de esforço e produtividade onde há alterações de processos já existentes na aplicação, por exemplo. Em resposta, se desejar fazer isso, o COSMIC sugere tratar médias calibradas diferentes para o tamanho de funções incluídas e alteradas, embora não exista uma proposta padrão com esse propósito.

Sugerimos ainda, que você analise seus próprios dados, lembrando que para derivar o esforço será preciso um índice de produtividade, que idealmente, será calibrado com base nos seus dados históricos.

5.2.2 Considerações

Segundo TRUDEL e BUGLIONE (2010), num mundo ideal, os times ágeis poderiam estimar e se comprometer com as suas próprias estimativas durante uma iteração. Contudo, na prática as organizações precisam estimar seus projetos de forma mais ampla e consistente, principalmente para viabilizar suas projeções de *budget*. Desta forma, estimativas ágeis não suportam adequadamente a necessidade de comparar e analisar o desempenho de projetos, um desejo comum aos gestores - seja em relação ao mercado (*benchmarks*) ou mesmo num âmbito interno. Há também uma referência detalhada sobre o assunto elaborada por PHAN (2011), onde o autor elenca os motivos para os quais os métodos como Planning Poker já não funcionam tão bem, principalmente nas culturas orientais. Leia mais sobre isso no CAPÍTULO 7.

TRUDEL e BUGLIONE (2010) ressaltam que comparações podem ser realizadas ao utilizarmos uma mesma unidade base ao

longo do tempo, sendo esta definida como um padrão de medição como este, que adere ao *Functional Size Measurement* (*FSM*), mantido e publicado pela ISO. Isso é requerido para que o processo de estimativas seja considerado consistente e repetível, além de compatível com os diferentes propósitos, de modo que possa ser derivado e avaliado inclusive na forma de indicadores, tais como: taxa de entrega, velocidade, produtividade ou custo, os quais são resultantes de atividades de projetos (sejam eles ágeis ou não). Todavia, comparações também podem ser realizadas, seja entre projetos de uma mesma organização ou em diferentes operações, considerando-se seus devidos domínios e particularidades.

Podem ser criadas referências de custo unitário para o desenvolvimento de *software*, com base em médias estabelecidas ou mesmo pela formação de parâmetros para derivação de esforço, prazo e qualidade. Elas podem ser comparadas aos dados atuais e contra o planejamento de um projeto, isso para também criar e acompanhar indicadores para monitoramento da eficiência do processo de desenvolvimento, propiciando a sua melhoria contínua.

Diversas organizações contribuem ou utilizam-se de *benchmarking* com o propósito de comparar custo ou desempenho para produzir projetos similares em um ramo da indústria, muitas vezes independente da metodologia utilizada. Desta forma, poderia ser adequado com base num propósito pré-definido, considerar uma iteração ou mesmo uma release como um projeto de desenvolvimento ou melhoria, desde que o seu incremento seja potencialmente disponibilizável em produção - o que é justificado como uma analogia aos métodos tradicionais de desenvolvimento de *software*. Nesta proposição, podem ser vislumbrados alguns potenciais candidatos a pontos de estimativa/medição:

- Estimativa Inicial – realizada no início do projeto (por aproximação), com base em uma lista de US ou requisitos, desde que possuam um nível de detalhe suficiente para a identificação das funcionalidades requeridas;
- Estimativa da Iteração ou MVP – opcionalmente, no início ou

durante cada iteração/MVP, pode ser realizada uma estimativa, para fins de planejamento da iteração/MVP;
- Medição da Iteração/MVP – ao final de cada iteração/MVP e considerando o total de movimentos de dados incluídos, alterados e excluídos durante a mesma realiza-se a medição do *software* entregue ao usuário, avaliando assim a evolução e o retrabalho associado a cada funcionalidade pronta em uma Iteração/MVP anterior e novamente envolvida no processo de construção;
- Medição do Projeto – considera o tamanho total do *software* efetivamente entregue ao usuário ao final do projeto, eliminando o retrabalho associado.

O guia "*Guideline for the use of COSMIC FSM to manager Agile projects*" (COSMIC, 2011) foi criado para consolidar o entendimento sobre a aplicação do método COSMIC em um contexto ágil. Neste documento, descreve-se uma abordagem para fornecer aos usuários do método, um direcionamento de acordo com seu propósito. É um guia bastante completo, com uma proposição de contagem de US (*User Stories*), orientando inclusive um estilo de documentação que poderia ser gerado e mesmo indicadores a serem analisados (comumente chamadas de "métricas" nas comunidades ágeis).

5.2.2.1 Sobre *Benchmarking*

Conforme publicado pela organização COSMIC (2016), existe uma ação junto ao ISBSG para realização de uma análise conjunta a respeito da "compatibilização" de projetos de *software* da base do ISBSG para *benchmarking*. Neste momento inclusive, já há resultados parciais publicados e disponibilizados por um relatório técnico do próprio ISBSG. Nele, consta que cerca de 10% da base citada anteriormente, com 6000 projetos medidos e/ou convertidos para APF - IFPUG foi também medida em Pontos de Função COSMIC. O relatório, cuja leitura é sugerida para membros de organizações que estão iniciando com o método COSMIC, contempla uma análise detalhada sobre como a produtividade (tamanho/esforço) e a veloci-

dade (tamanho/duração) dos projetos *software* variam, dependendo do domínio, tipo de projeto e linguagem de programação.

O referido relatório também contempla:

- Uma análise de como o esforço é distribuído sobre as atividades de projeto mais significativas;
- Uma análise das características das medições de tamanho funcional como COSMIC (como por exemplo: a proporção de cada tipo de movimentação e como ela varia de acordo com o domínio do *software*);
- Exemplos de como dados de *benchmarking* existentes de projetos medidos com a abordagem do IFPUG podem ser comparados com os resultados obtidos com dados de projetos medidos com o COSMIC.

6. Contratações de *Software* à Brasileira

Os modelos de desenvolvimento de *software*, na perspectiva das organizações que contratam serviços desta natureza, são vislumbrados através das seguintes estratégias básicas de *sourcing*: o *insourcing* e o *outsourcing*, ou ainda, modelos mistos.

Enquanto no *insourcing* os serviços de desenvolvimento de *software* são executados por um braço especializado da própria organização onde colaboradores ou prestadores de serviços estão sob a sua gestão direta, no *outsourcing*" (ou "terceirização"), segundo NETO (2014), há a transferência de responsabilidade por um determinado serviço, operação ou fase de um processo de produção ou comercialização de uma empresa para outra (ou outras), que é denominada "terceira". Sendo assim o termo em inglês "*outsourcing*" e o termo em português "tercerização" podem ser utilizados como sinônimos.

Para NETO (2014), a terceirização é um método administrativo que possibilita o estabelecimento de um processo gerenciado de transferência a terceiros das atividades acessórias e de apoio das empresas, permitindo-as concentrar-se no seu negócio principal (*core business*).

As organizações que utilizam a estratégia de *outsourcing* para aquisição de produtos e serviços de TI, principalmente quando contratantes em quantidade significativa, por vezes possuem um processo estruturado para viabilizar a gestão de fornecedores e contratos de tecnologia, suportando e monitorando as necessidades de aquisições no contexto que lhe foi atribuído.

Ressalta-se, que em uma relação de terceirização gerenciada são aplicáveis custos de transação (NETO, 2014), os quais consistem em trocas (transações) entre empresas ou mesmo dentro destas, que podem ser classificadas da seguinte forma:

- Custo de informação – relacionados à busca de informações a respeito de produtos, preços, insumos e compradores ou fornecedores;
- Custos de negociação – surgem do ato da transação, tal como a negociação e a elaboração de contrato ou o pagamento a um intermediário;
- Custos de monitoramento – são custos que surgem após a troca ter sido negociada, por exemplo: custos relativos ao monitoramento da qualidade de atendimento do fornecedor, para garantir que o acordo contratual está sendo seguido.

Existem ainda, alguns elementos da atividade econômica produtiva pressupostos pela abordagem de custos de transação, que são:

- Racionalidade limitada – a capacidade de decisão das pessoas é limitada e nem sempre racional;
- Oportunismo: em algumas situações, uma empresa ou outra buscará atuar em seu proveito próprio, mas isso não significa que o tempo todo assim será;
- Especificidade de ativos – onde uma empresa investiu na operação e a outra tenta apropriar-se do retorno obtido;
- Informação assimétrica – uma das partes possui informação mais qualificada e a utiliza eventualmente de modo oportunista.

As relações de custo são intrínsecas ao relacionamento entre contratante e contratada, e seus aspectos devem ser avaliados e tratados no sentido de estabelecer uma relação economicamente saudável e justa entre as partes. Tentando mitigar riscos associados, diversas empresas criam mecanismos de governança e gerenciamento de TI para proteger seus interesses, assim como a transparência de seus processos. Veremos a seguir, devido à extensa documentação disponível sobre o assunto e qualidade do material gerado, como isso funciona no Governo Brasileiro.

Para um melhor entendimento de como as contratações de *software* são realizadas no Brasil, apresentaremos um conjunto de práticas definidas e amplamente utilizadas por diversos órgãos que respondem ao Governo Federal, mas especificamente vislumbraremos orientações providas pelo SISP (Secretaria de Logística e Tecnologia da Informação, ligada ao Ministério do Planejamento), visando formarmos uma opinião e uma visão suficientemente clara sobre o assunto.

Destacamos que quanto ao "Regime de Execução" via terceirização, o SISP (2014) cita a execução indireta sob qualquer dos seguintes modelos e portanto, nos limitaremos inicialmente a estes formatos:

- Empreitada por preço global - quando se contrata a execução da obra ou do serviço por preço certo e total;
- Empreitada por preço unitário - quando se contrata a execução da obra ou do serviço por preço certo de unidades determinadas;
- Tarefa - quando se ajusta mão-de-obra para pequenos trabalhos por preço certo, com ou sem fornecimento de materiais;
- Empreitada integral - quando se contrata um empreendimento em sua integralidade, compreendendo todas as etapas.

A seguir, apresentaremos mais detalhes sobre o modelo brasileiro de contratações de *software*.

6.0.0.1 Contratando pontos de função

É importante citar que para realizar contratações utilizando pontos de função, uma prática que é recomendada no governo brasileiro, pode ser necessário estabelecer convenções locais e deflatores de produtividade para cada tipo de manutenção, visando estabelecer uma relação de linearidade entre as medidas de tamanho e as estimativas de esforço e/ou custo. No Brasil, um exemplo de guia de orientações de métricas (portanto, uma convenção ou manual local) bastante conhecido é o que suporta os contratos de prestação

de serviços de desenvolvimento de *software* mantidos pela Caixa Econômica Federal (CEF) com seus fornecedores, o qual apresenta diversas adequações em relação ao método original publicado pelo IFPUG.

Aliás, o Governo Federal do Brasil, através do Ministério do Planejamento, publicou um guia com diversas necessidades consideradas comuns aos órgãos públicos, o Roteiro de Métricas do SISP. Ele é aderente à Instrução Normativa 04/2010, que orienta que os contratos de software sejam mantidos com base em critérios objetivos, critérios pelos quais a APF se destaca. Com a Instrução Normativa, o Governo passa a ter um maior controle sobre o que está sendo contratado, um processo maduro, auditável e repetível (considerando o mesmo escopo e diferentes profissionais procedendo uma avaliação de tamanho, pode-se chegar a um mesmo quantitativo de pontos de função).

Destaca-se ainda, que no Brasil, dentre as organizações mais conhecidas que utilizam a Análise de Pontos de Função (APF) podemos citar: Banco Central (BACEN), Bradesco, Oi, TAM, Santander, Caixa Econômica Federal (CEF), Embratel, Porto Seguro, SERPRO, Sicredi, DATAPREV, Infraero, Petrobras, Correios, Tribunal de contas da união (TCU), Ministério das Cidades, Aeronáutica, Marinha do Brasil, Centro de Desenvolvimento da Tecnologia Nuclear, Comissão nacional de energia nuclear, INEP, Ministério da Cultura, Ministério do Trabalho, Ministério da Educação, além de muitas outras empresas dos setores público e privado.

6.0.0.2 *Agile* no governo: aspectos de contratação

No âmbito público, em 2013, o Tribunal de Contas da União (TCU), realizou um estudo amplo sobre a contratação de serviços de *software* ágil nas principais organizações públicas que naquela época utilizavam tais métodos de trabalho. São elas: Banco Central do Brasil (BACEN), o Tribunal Superior do Trabalho (TST), o Instituto do Patrimônio Histórico e Artístico Nacional (IPHAN), o Instituto Nacional de Estudos e Pesquisas Educacionais Anísio

Teixeira (INEP) e o Supremo Tribunal Federal (STF).

Segundo o TCU, no período em que foi realizado o estudo as empresas citadas haviam licitado alguns objetos. E neste contexto, destaca-se que o TST utilizou a unidade base de contratação em Horas de Serviço Técnico (HST) e os demais órgãos contrataram utilizando pontos de função.

Sobre o planejamento, gestão de demandas, aceitação de produto e forma de pagamento, foram elencadas algumas constatações da pesquisa executada pelo TCU (2013), conforme citado abaixo:

- Uma demanda para construção de um produto é precedida por um planejamento, que pode ser realizado apenas pela instituição contratante ou em conjunto, ou também em conjunto, entre essa e a empresa contratada;
- As instituições visitadas pelo TCU (2013) emitem uma ordem de serviço por ciclo, iteração ou *sprint*, ou por *release* de *software*, sendo mais comum o primeiro caso;
- A aceitação do produto entregue pela contratada, embora no *framework Scrum* seja preceituado que ocorra na reunião de revisão da *sprint*, essa prática não é executada nos contratos estudados, sendo que algumas instituições apenas verificam se os artefatos exigidos foram entregues, caracterizando o recebimento provisório (devido a questões legais);
- Constatou-se que algumas instituições remuneram os serviços de planejamento (quando realizados), enquanto outras remuneram apenas os serviços de construção do software;
- Algumas instituições, em um determinado período de tempo, enquanto a empresa contratada executava a construção do *software* em um ciclo, iteração ou sprint, a contratante preparava os itens do *backlog* do produto, para serem implementados no próximo ciclo e homologava o produto entregue no ciclo anterior;
- A forma de pagamento não segue um padrão, e cada órgão trata este assunto de uma forma, como exemplo:

  - No IPHAN, são considerados os ciclos de projeto (planejamento, implantação e encerramento), destinando um percentual da quantidade estimada de pontos de função para remuneração do Planejamento, fixa em 5% e prevendo uma revisão ao final, uma vez que a quantidade final de pontos de função pode não corresponder aos pontos inicialmente estimados. Para as fases de Execução e Encerramento, respectivamente, são destinados 80 e 15% do que foi efetivamente entregue.
  - No STF, o pagamento das ordens de serviço é calculado com base no tamanho funcional do produto homologado em pontos de função brutos e na efetiva execução do serviço. Ele é feito em duas etapas: a primeira, correspondente a 90%, é realizada após o recebimento definitivo da OS, referente à última sprint de cada produto; a segunda, correspondente a 10%, é realizada somente ao final do período de garantia, estipulado em 180 dias (peça 25, p. 21 e 39).
- Em relação a níveis de serviço, observou-se que quase todas as instituições visitadas instituíram vários indicadores a serem atendidos pela contratada e que o não atendimento a critérios mínimos aceitáveis (nível mínimo de serviço) implicaria na redução do valor a ser pago à contratada, são alguns exemplos de indicadores constatados:
  - No TST, foi instituído o Índice de Atraso na Entrega de OS, o qual tem por objetivo garantir a entrega da OS no prazo estimado para a sua conclusão, tolerando atrasos inferiores a 10% do prazo para a entrega da OS como aceitável, o que se afasta dos preceitos do *framework Scrum*, no qual uma sprint deve ser executada no prazo estabelecido; o BACEN também instituiu um indicador referente ao Atraso na Entrega da OS de projeto de construção, tolerando, assim como o TST, pequenos atrasos;
  - O TST criou o indicador Desconformidade na OS (DOS)

para assegurar a qualidade da OS, bem como instituiu nas listas de verificação para a avaliação dos artefatos produzidos, e caso alguma desconformidade seja detectada, o produto, e, consequentemente, a Ordem de serviço, é rejeitada.
  - O Bacen definiu o Índice de Defeitos por Pontos de Função da OS e o Índice de Defeitos Impeditivos por Pontos de Função da OS, onde para o primeiro indicador há tolerância para a aceitação da OS, enquanto para o segundo a tolerância é zero;
  - No IPHAN foi definido o Índice de Pontos Executados (IPE), um indicador que se destina a avaliar a qualidade dos produtos entregues, medindo a quantidade de pontos de função brutos executados e aceitos da ordem de serviço. Caso o IPE seja inferior ou igual a 80%, multas podem ser aplicadas à contratada – caso algum produto construído no ciclo não seja aprovado, este poderá ser considerado perdido, devendo um novo ciclo ser iniciado para adequação desse produto;
  - No STF, por sua vez, o Índice de Desconformidade do Produto afere a qualidade dos produtos entregues pela contratada.
- Sobre os valores ágeis, constatou-se que:
  - A preocupação de todas as instituições com relação à entrega de artefatos de documentação associados ao *software* produzido a cada iteração, facilitando, por exemplo, futuras manutenções por terceiros alheios ao processo de desenvolvimento;
  - A relação contratual prevalece sobre a possível colaboração entre as partes.
- Ao final da análise realizada pelo TCU (2013), foram mapeados os seguintes riscos relacionados às contratações utilizando métodos ágeis no âmbito público, os quais se constituem de pontos de atenção:
  - Contratação de desenvolvimento de software com adap-

tação de metodologia ágil que desvirtue sua essência;
- Alteração da metodologia ágil adotada no instrumento convocatório no decorrer da execução contratual;
- Alteração da metodologia ágil adotada no instrumento convocatório no decorrer da execução contratual;
- Exigência de artefatos desnecessários ou que se tornam obsoletos rapidamente;
- Utilização de contrato para desenvolvimento de *software* por metodologias tradicionais para desenvolvimento por métodos ágeis;
- Falta de comprometimento ou colaboração insatisfatória do responsável indicado pela área de negócios (*Product Owner*) no desenvolvimento do software;
- Falta do conhecimento necessário do indicado pela área de negócios (*Product Owner*) para o desenvolvimento do *software*;
- Excessiva dependência da visão do indicado pela área de negócios (*Product Owner*);
- Equipe da empresa contratada não ter *expertise* em desenvolvimento de *software* com métodos ágeis;
- Dificuldade de comunicação entre a equipe de desenvolvimento da contratada com o indicado pela área de negócios (*Product Owner*);
- Alteração constante da lista de funcionalidades do produto;
- Iniciação de novo ciclo sem que os produtos construídos na etapa anterior tenham sido validados;
- Falta de planejamento adequado do software a ser construído;
- Pagamento pelas mesmas funcionalidades do *software* mais de uma vez, em virtude de funcionalidades impossíveis de serem implementadas em um único ciclo, ou em virtude da alteração de funcionalidades ao longo do desenvolvimento do *software*;
- Não disponibilização do *software* em ambiente de pro-

dução para a utilização e avaliação dos reais usuários;
– Forma de pagamento não baseada em resultados.

6.0.0.3 Gestão de fornecedores

Segundo o *PMBOK* quarta edição (*PMBOK*, 2008), gerenciar aquisições em um projeto é usar os processos sugeridos que se fazem necessários conforme o caso, para a aquisição de um ou mais produtos ou serviços como resultados externos à equipe do projeto. Na publicação, as tarefas relacionadas às contratações foram organizadas nos seguintes quatro processos:

- Planejar as aquisições;
- Conduzir as aquisições;
- Administrar as aquisições;
- Encerrar as aquisições.

Os processos citados, além de se inter-relacionarem, são idealmente executados ao menos uma vez em cada projeto, ou ciclicamente (se for um projeto organizado por fases). Eles também podem ocorrer em outras áreas de conhecimento.

Ressalta-se ainda, que os seguintes processos do PMBOK podem ser aplicados na administração das aquisições:

- Orientar e gerenciar a execução do projeto: autorizando o trabalho do fornecedor na ocasião apropriada;
- Reportar o desempenho: monitorando o escopo do contrato, os custos, o cronograma e o desempenho técnico;
- Realizar o controle da qualidade: inspecionando e verificando a adequação do produto do fornecedor;
- Realizar o controle integrado de mudanças: garantindo que as mudanças sejam aprovadas de forma adequada e certificando que todas as pessoas envolvidas estejam cientes dessas mudanças;
- Monitorar e controlar os riscos: garantindo a sua mitigação

Em complemento, segundo o MPS-BR (SOFTEX, 2013, p. 22 e 23), que também apresenta processos para gestão de contratos e fornecedores, é possível quantificar a evolução do projeto, visualizando se um produto está em conformidade ou como ele evoluiu em termos de:

- Atendimento aos requisitos;
- Custos e prazos;
- Riscos envolvidos;
- Nível de problemas que estão sendo enfrentados;
- Aderência ao processo que foi contratado;
- Níveis de retrabalho encontrado.

Sendo assim, relacionando os itens estudados aos processos e métricas que permitam tal controle, identificam-se componentes para a tomada de decisão, tais como:

- Revisão de prazos e requisitos;
- Alocação de recursos;
- Descarte de recursos;
- Interrupção de atividades;
- Aceitação de artefatos;
- Rejeição de artefatos;
- Aplicação de penalidades;
- Aplicação de prêmios;
- Solicitação do envolvimento de interessados (stakeholders);
- E até mesmo a interrupção do contrato...

Salienta-se, contudo, que este é um tópico bastante abrangente e abordado apenas para efeito de introdução ao conteúdo, visto que possui forte relação com o contexto geral do trabalho.

6.1 Planejamento, Execução e Gestão de contratos

A seguir, serão abordados assuntos relacionados ao planejamento das contratações de *software*, conforme sua afinidade com o tema sendo tratado.

Como citado anteriormente, os contratos de desenvolvimento e manutenção de *software* são peculiares em termos das suas características, formas e normas. Sendo assim, visando apresentar alternativas para melhor operá-los e geri-los, veremos algumas regras e práticas constatadas em guias e contratos mantidos por órgãos do Governo do Brasil, os quais disponibilizam publicamente e de forma integral os artefatos utilizados, facilitando o acesso a um conjunto rico de informações e que em outras organizações (da iniciativa privada) seriam muito provavelmente considerados confidenciais.

6.1.1 Planejamento da contratação

De acordo com o SISP (2014, pág. 56), um guia aplicado às contratações públicas (em um conjunto específico de órgãos), no seu contexto de aplicação, a definição do objeto da contratação deve ser precisa, suficiente e clara, sem considerar especificações excessivas, irrelevantes ou desnecessárias, que limitem ou frustrem a competição ou o fornecimento dos serviços. Desta forma, na visão do SISP (2014), um objeto de contratação deve contemplar aspectos tais como os seguintes:

- Não tecnológicos (SISP, 2014, pág. 56), como:
  - Requisitos de negócio;
  - Capacitação;
  - Requisitos legais;
  - Requisitos de manutenção;
  - Requisitos temporais;
  - Requisitos de segurança;

  - Requisitos sociais;
  - Requisitos ambientais;
  - Requisitos culturais.
- De contratação (SISP, 2014, pág. 57 e 58), dentre os quais:
  - Arquitetura tecnológica, composta de *hardware*, *software*, padrões de interoperabilidade, linguagens de programação, interfaces, dentre outros;
  - Projeto e implementação, que estabelece o processo de desenvolvimento de *software*, técnicas, métodos, forma de gestão, de documentação, dentre outros;
  - Implantação, que define o processo de disponibilização da solução em ambiente de produção, dentre outros;
  - Garantia e manutenção, que define a forma como será conduzida a manutenção e a comunicação entre as partes envolvidas;
  - Capacitação, que define o ambiente tecnológico dos treinamentos a serem ministrados, os perfis dos instrutores, dentre outros;
  - Experiência profissional da equipe que projetará, implementará e implantará a solução de tecnologia da informação, que define a natureza da experiência profissional exigida e as respectivas formas de comprovação dessa experiência, dentre outros;
  - Formação da equipe que projetará, implementará e implantará a solução de tecnologia da informação, que define cursos acadêmicos e técnicos, formas de comprovação dessa formação, dentre outros;
  - Metodologia de trabalho;
  - Segurança da informação; e
  - Requisitos aplicáveis.

O SISP (2014) orienta ainda, que devem ser avaliadas contratações separadas para os itens que possam ser divididos, em tantas parcelas quantas se comprovarem técnica e economicamente viáveis, de forma a obter o melhor aproveitamento dos recursos

disponíveis. Prevê-se, ainda, que cada serviço ou produto do lote, deverá ser discriminado em itens separados nas propostas de preços, de modo a permitir a identificação do seu preço individual.

6.1.2 Homologação de fornecedores

O SISP (2014, pág. 111) também orienta os órgãos conveniados, a descrever os critérios técnicos que serão utilizados para habilitação dos concorrentes, seguidos das respectivas justificativas que levaram à sua exigência, considerando o objetivo da contratação e a legislação pertinente. Recomenda ainda, que para cada papel a ser desempenhado pela contratada, exista uma indicação dos requisitos de capacitação necessários para execução do contrato.

Sendo assim, a habilitação deverá ser realizada conforme o conjunto de critérios técnicos (pontuáveis), no caso de uma concorrência do tipo "técnica e preço", utilizando parâmetros como os que seguem:

- Critério técnico pontuável: descrição do critério pontuável a ser contabilizado;
- Pontuação: descrição da pontuação relacionada ao critério escolhido;
- Percentual (%): definição do percentual que o mesmo representa em relação à pontuação total;
- Justificativa: justificativa motivada para a escolha do critério relacionado.

No referido modelo, deverá ser considerada também a descrição dos critérios de julgamento, conforme os requisitos habilitatórios definidos, os quais não devem contrariar as normas e regras previamente estabelecidas.

Ressalta-se que o SISP (2015b) dispõe que não é suficiente ter a contratante preparada, com pessoas capacitadas e processos definidos para gestão e fiscalização de projetos de *software* com práticas de métodos ágeis, pois quando se utiliza terceirização

do desenvolvimento de *software*, devem ser definidos também, critérios de seleção de fornecedor que levem à seleção de empresa capacitada e com preço exequível para a prestação desses serviços de desenvolvimento.

6.1.3 Execução da contratação

Segundo o SISP (2014, pág. 60), o modelo de execução de uma contratação de *software* deve ser elaborado de forma a contemplar as condições para fornecimento da solução a ser contratada. E durante a sua elaboração, deve-se atentar para a fixação de rotinas de execução como: prazos, horários de fornecimento ou prestação de serviço, local de entrega, documentação mínima exigida para os padrões de qualidade. Também devem ser previstas a quantificação e estimativa prévia de volume de serviços, definindo mecanismos formais de comunicação para troca de informação, forma de pagamento, entre outros.

Neste modelo, durante a execução contratual, a contratante encaminha Ordens de Serviço (OS), sendo que este instrumento tem por atribuição, formalizar a execução dos serviços.

Segundo o SISP (2014), uma Ordem de Serviço (OS) deve conter, idealmente:

- Definição e a especificação dos serviços a serem realizados ou bens a serem fornecidos;
- Volume de serviços a serem realizados ou a quantidade de bens a serem fornecidos, segundo as métricas definidas;
- Cronograma de realização dos serviços ou entrega dos bens, incluídas todas as tarefas significativas e seus respectivos prazos; e
- Identificação dos responsáveis pela solicitação do bem ou serviço na área requisitante da solução.

O SISP (2014) orienta que depois de recebidos todos os itens constantes da OS, deve-se elaborar um "Termo de Recebimento

Provisório", um instrumento que deve ser entregue ao Preposto da Contratada, informando que os itens constantes da Ordem de Serviço foram entregues para a contratante, para sua avaliação.

O requisitante então avaliará a qualidade dos itens recebidos, de acordo com os critérios de aceitação definidos no contrato, identificando, sendo o caso, as não conformidades. E caso não sejam identificadas inconformidades, o mesmo deverá emitir o "Termo de Recebimento Definitivo".

6.1.4 Monitoramento e gestão

Após definir o modelo de execução do contrato, deverá ser elaborado o modelo de gestão do contrato, que contemplará as condições de gestão e fiscalização do contrato de fornecimento da solução de TI (SISP, 2014). Nesse Modelo, devem ser planejados os seguintes aspectos:

- Fixação dos "Critérios de Aceitação" dos serviços,
- Procedimentos de teste e inspeção que servirão de base para os termos de "Recebimento Provisório" e "Definitivo";
- Fixação de valores e procedimentos para retenção ou glosa de pagamento;
- Definição clara e detalhada das sanções administrativas;
- Procedimentos para emissão de nota fiscal e pagamento, descontados os valores oriundos da aplicação de eventuais glosas.

6.2 Processos Relacionados

Intrinsecamente ligados às contratações ocorridas, os processos de Gestão do Portfólio e Desenvolvimento de *Software*, assim como o de Gerenciamento de Projetos, são vislumbrados de forma a adequar-se às necessidades de controle e execução de projetos de *software*. Na sequência, veremos um pouco mais sobre eles.

6.2.1 Portfólio de projetos

Segundo o SISP (2013), um portfólio é uma coleção de projetos, programas e outros trabalhos, em andamento ou planejados (sendo eles relacionados de alguma forma ou não), os quais estão agrupados com o propósito de facilitar o gerenciamento efetivo das ações para atender aos objetivos estratégicos organizacionais (Figura 22).

Deve-se destacar, que enquanto projetos e programas são temporários, os portfólios são contínuos. Sendo assim, uma organização pode possuir mais de um portfólio, cada um tratando de áreas (departamentos) ou objetivos específicos (em última instância, deve haver um portfólio abrangente para toda a organização).

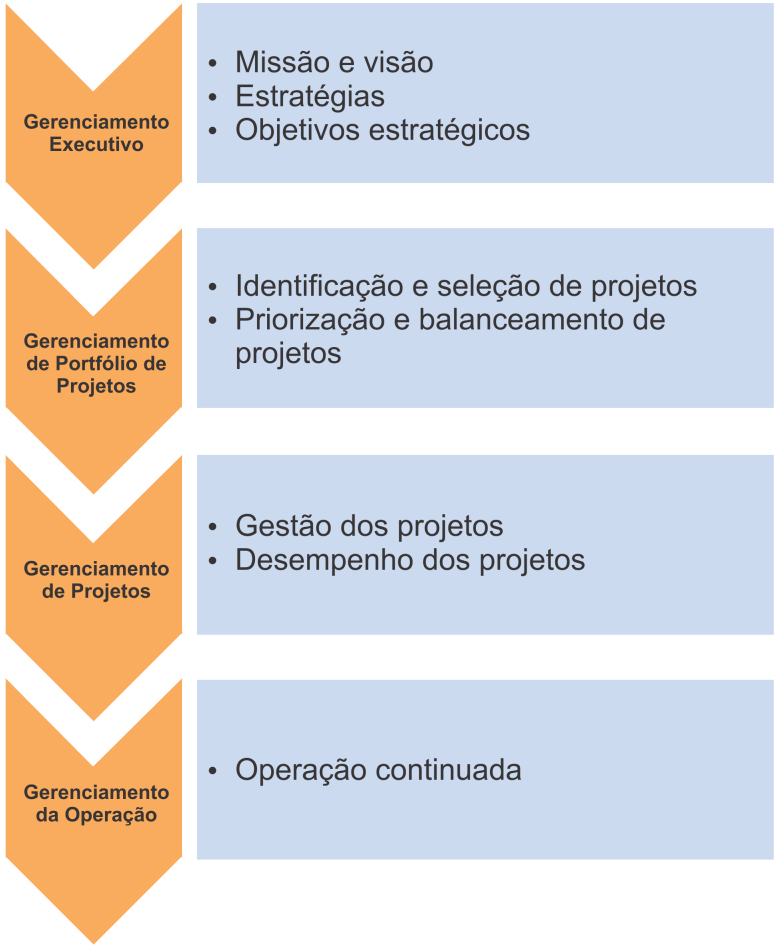


Figura 22 – Portfólio, Projetos e Demais Processos

O gerenciamento de portfólios de projetos refere-se ainda, à gestão centralizada de um ou mais portfólios (Figura 23), o que inclui a identificação, priorização, autorização, monitoramento e controle de projetos, programas e outros trabalhos relacionados. O gerenciamento de portfólios se concentra em garantir que os projetos e programas sejam analisados a fim de priorizar a alocação de recursos e que sejam consistentes e integrados às estratégias

organizacionais (SISP, 2011, pág. 22).

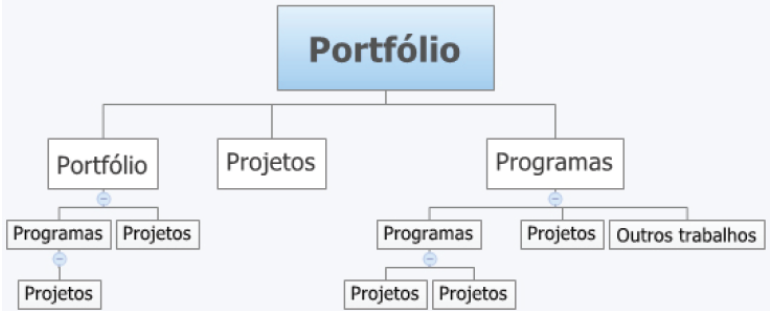


Figura 23 – Projetos, Programas e Portfólio

Em um portfólio (SISP, 2013), todos os componentes exibem certas características comuns:

- Representam investimentos feitos, em curso ou planejados;
- Estão alinhados com as metas e objetivos estratégicos;
- Têm algumas características que, tipicamente, os distinguem, o que permite agrupá-los para o gerenciamento efetivo;
- São quantificáveis e, portanto, podem ser medidos, classificados e priorizados.

O SISP (2013) define dois tipos de portfólio, sob duas perspectivas diferentes, conforme abaixo:

- Portólio de Projetos Departamental - inclui projetos ou programas de um determinado departamento ou unidade organizacional, em uma visão de projetos limitada ou restrita;
- Portfólio de Projetos Corporativo - são projetos ou programas que englobam toda a organização, sendo que a alta administração é patrocinadora desses projetos e possui a visão do todo (secretarias, departamentos, coordenações).

É importante ressaltar que o gerenciamento de portfólios de projetos provê um gerenciamento coordenado dos componentes do

portfólio, o que tende a auxiliar no alcance de objetivos maiores na instituição. Dessa forma, oferece benefícios para a tomada de decisão baseada em priorizações e informações estratégicas. Existem ainda, inúmeras vantagens na adoção dessas práticas, como por exemplo:

- Alinhar os projetos ou programas com as estratégias organizacionais;
- Prover a gestão das prioridades, mantendo o foco em projetos de maior valor para a organização;
- Permitir a alocação eficiente de recursos e investimentos;
- Possibilitar a redução do número de projetos redundantes ou sem alinhamento estratégico;
- Permitir uma visão clara das interdependências entre os projetos;
- Favorecer o envolvimento da alta administração com a gestão dos projetos.

Ressalta-se também, que o gerenciamento de portfólio tende a assegurar que a organização está desempenhando um trabalho alinhado às necessidades, evitando os problemas abaixo listados:

- Quantidade excessiva de projetos desenvolvidos simultaneamente;
- Recursos preciosos utilizados em projetos de baixa prioridade;
- Projetos ou programas sem relação com os planos estratégicos;
- Seleção de projetos que trazem pouco valor à organização ou com grandes riscos e poucos benefícios.

Em geral, um projeto nasce na área de negócio e passa pela gestão estratégica e de portfólio de projetos, onde sua viabilidade é avaliada. Posteriormente são realizados os critérios de priorização e balanceamento, permitindo que projetos de maior valor para a

organização sejam aprovados. Depois de aprovado, o projeto entra na etapa de execução, ou seja, inicia-se o seu desenvolvimento (SISP, 2013). A seguir, são elencadas as principais atividades inerentes ao processo de Gestão de Portfólio:

- Identificar Projetos – identificar projetos de TI em andamento e os novos projetos a serem executados, conforme o planejamento estratégico vigente;
- Selecionar Projetos – selecionar e categorizar os projetos e programas que comporão o portfólio, de acordo com suas características e relevância para a organização;
- Priorizar Projetos – priorizar os projetos de acordo com os critérios pré-estabelecidos (vide exemplo na Tabela 7), de forma a possibilitar distinguir àqueles com maior potencial de entrega de valor à organização;
- Balancear Portfólio – definir quais projetos serão executados (emitindo-se o Termo de Abertura do Projeto) e quais aguardarão na fila para execução (exemplo na Tabela 8);
- Monitorar Portfólio – monitorar e controlar as atividades de modo a assegurar a realização do gerenciamento de portfólio de projetos de TI de forma alinhada aos objetivos estratégicos da organização;
- Gerir Portfólio – realizar a gestão do portfólio de projetos, afim de garantir que os recursos estão sendo investidos conforme a sua importância estratégica para a organização;
- Gerenciar Mudanças e Estratégias – acompanhar a evolução das estratégias e acompanhar mudanças que causam impacto no portfólio de projetos.

Número	Critério	Avaliação	Valor	Projeto
1	Importância estratégica	Alta	2000	
		Média	1000	
		Baixa	500	
2	Abrangência dos Resultados do Projeto	Traz mudança para muitas áreas	1000	
		Traz mudança para poucas áreas	250	
		Traz mudanças para uma área somente	125	
3	Urgência do Projeto	Alta	1000	
		Média	500	
		Baixa	250	
4	Tempo Estimado do Projeto	Curto	500	
		Médio	250	
		Longo	125	
5	Conhecimento da Equipe	Profundo	500	
		Razoável	250	
		Pouco	125	
6	Fator Político	Alto	1000	
		Médio	500	
		Baixo	250	
7	Tempo Aguardando a Execução	Menos de 6 Meses	125	
		De 6 meses a 1 Ano e 6 meses	250	
		Mais de 1 Ano e 6 meses	500	
			TOTAL	

Tabela 7 – Exemplo de Critérios de Priorização de Portfólio

ID	Nome do Projeto	Área de Negócio	Patrocinador	Solicitante	Categoria	Riscos	Prioridade	Balanceamento	Status do Projeto	Status Gráfico	Observação
1									Em execução		
2									Adiado/Suspenso		
3									Em execução		
4									Priorizado		
5											
6											
7											
8											
9											
10											

Tabela 8 – Exemplo de Planilha de Acompanhamento de Portfólio

São resultados esperados de um Gerenciamento de Portólio de Projetos, conforme especificação do Nível F do modelo de qualidade MPS-BR (SOFTEX, 2013):

- As oportunidades de negócio, as necessidades e os investimentos são identificados, qualificados, priorizados e selecionados em relação aos objetivos estratégicos da organização por meio de critérios objetivos;
- Os recursos e orçamentos para cada projeto são identificados e alocados;
- A responsabilidade e autoridade pelo gerenciamento dos projetos são estabelecidas;

- O portfólio é monitorado em relação aos critérios que foram utilizados para a priorização;
- Ações para corrigir desvios no portfólio e para prevenir a repetição dos problemas identificados são estabelecidas, implementadas e acompanhadas até a sua conclusão;
- Os conflitos sobre recursos entre projetos são tratados e resolvidos, de acordo com os critérios utilizados para a priorização;
- Projetos que atendem aos acordos e requisitos que levaram à sua aprovação são mantidos, e os que não atendem são redirecionados ou cancelados;
- A situação do portfólio de projetos é comunicada para as partes interessadas, com periodicidade definida ou quando o portfólio for alterado.

6.2.2 Desenvolvimento de *software*

Segundo o SISP (2015b, pág. 16), muitas instituições iniciam o uso de métodos ágeis por meio da adoção do *framework* fornecido pelo *Scrum*, porém isso é apenas uma parte do que é necessário para oferecer soluções sofisticadas para seus *stakeholders*, já que, inevitavelmente, as equipes precisam olhar para outros métodos para preencher lacunas que o *Scrum* propositalmente não considera no seu arcabouço.

Desta forma, a proposta do SISP (2015b) para a metodologia ágil, ilustrada na Figura 24, considera que as atividades de desenvolvimento do projeto envolvem basicamente três fases (que por sua vez, englobam um conjunto de métodos):

- Planejamento – fase que define um roadmap do produto com base na concepção de um Documento de Visão;
- Construção do *release* – fase que utiliza diversos métodos para desenvolver *software* de forma iterativa e incremental;
- Transição – fase para implantação definitiva do produto desenvolvido, assim como aspectos de suporte e avaliação.

Figura 24 – *Framework* do SISP

Estas fases são compostas por grupos de atividades de gestão de ordem de serviço, acompanhamento do projeto e gestão dos ambientes de TI, onde os ciclos de entrega envolvem os grupos de atividades de planejamento, construção da *release* e transição, um para cada *release*. E após a finalização do grupo de atividades de construção do *release*, é feita a transição dos produtos para o ambiente de produção.

O grupo de atividades de "Gestão de Ordem de Serviço" reúne

mecanismos de controle, fiscalização e monitoração da execução e também dos produtos entregues pela contratada de desenvolvimento de *software*. Por meio deste instrumento é possível mitigar riscos da contratação, tais como falta de qualidade técnica e funcional, pagamento sem produto de *software* funcional, pagamento sem métrica objetiva, entre outros (SISP, 2015b).

O grupo de atividades de "Acompanhamento do Projeto" contempla atividades para monitorar os projetos em execução pela contratada, identificando e mitigando riscos e desvios, tratando impedimentos e descumprimento de cronograma, entre outros aspectos.

E finalmente, as "Atividades de Gestão de Ambientes de TI" contemplam tarefas para viabilizar a criação de ambientes e facilitar a sua disponibilidade, para construir e implantar cada *release* do produto de *software*.

6.2.2.1 Papéis

Na metodologia proposta pelo SISP (2015b), elencou-se o seguinte conjunto de papéis para a equipe da empresa contratante (o texto foi adaptado para contemplar alguns papéis comuns nas organizações, mas não originalmente declarados):

- Analista de Infraestrutura de TI – representante da área de infraestrutura de TI para um projeto específico, sendo uma de suas responsabilidades principais a disponibilização dos recursos de *hardware*, *software* e serviços necessários, relacionados aos ambientes de Testes, Homologação e Produção, além de outras atividades correlacionadas;
- Analista de Qualidade – representa diferentes perfis ligados a avaliação de qualidade de *software*, tais como: Analista de Teste, Arquiteto de *Software*, Analista de Banco de Dados, Analista de Configuração e Mudança e outros, cuja responsabilidade é avaliar e definir processos de verificação da qualidade dos produtos entregues pela empresa contratada para o desenvolvimento – esta atividade pode ser parcial-

mente delegada a uma outra empresa contratada que possa exercer esta função, o que para algumas funções (Analista de Testes/ Testador, por exemplo) é comum em algumas empresas públicas;

- Analista de Métricas – é o responsável pelos serviços de medição de *software*, utilizando a métrica definida no contrato firmado com a empresa de desenvolvimento de *software* – esta atividade também pode ser parcialmente delegada para outra empresa contratada que pode cumprir esta função, algo bastante comum nas contratações públicas;
- Cliente (da área de negócio) – representante da área de negócio e responsável pela solicitação e homologação dos serviços de desenvolvimento de *software*;
- *Coach* (este papel pode ser exercido pelo Líder de Projeto) – profissional experiente e com sólidos conhecimentos nas práticas do processo ágil, devendo zelar pela sua correta execução, com vistas a guiar os outros envolvidos no projeto a executar o processo e práticas de forma adequada;
- *Product Owner* – é o representante da área de negócio (cliente) com conhecimento suficiente para definir e priorizar requisitos do negócio e responder aos questionamentos da equipe de desenvolvimento;
- Líder de Projeto – é responsável pelo projeto e se relaciona com todos os envolvidos e atividades do projeto;
- Usuário – tipo de *stakeholder* que usa o produto de software, seja interno ou externo à organização;
- Outros (legais, no contexto dos órgãos públicos) – devido à atribuições legais ou organização interna da contratante, podem ser atribuídos outros papéis, tais como um Gestor do Contrato e Fiscal de Contrato.

Complementarmente, os papéis da equipe contratada, no modelo de referência do SISP (2013), são os seguintes:

- Preposto – um responsável técnico pelos serviços prestados, atua como um gestor técnico e ponto focal do time da contratada;
- Scrum Master – um dos responsáveis pelo time da contratada, este papel trabalha para o sucesso do projeto através da entrega de valor para o cliente, viabilizada pela atuação eficaz da equipe de desenvolvimento e apoio ao dono do produto;
- Time de Desenvolvimento – é um grupo de cinco a nove integrantes da contratada de desenvolvimento de *software*, que deve possuir uma característica multifuncional, auto-organizados, contando com analistas, arquitetos, programadores, testadores, administrador de banco de dados entre outros;
- Gerente de Relacionamento (opcional) – realiza o acompanhamento de todo o ciclo de atendimento da ordem de serviço garantindo as entregas conforme compromissos firmados.

6.2.2.2 Fase de "Planejamento"

Conforme o SISP (2015b), esta fase contempla a concepção do Documento de Visão, artefato em que são descritos os objetivos de negócio e as características-chave (*features*) do produto. E após a concepção deste, inicia-se então o planejamento de cada entrega (*release*), que consiste no agrupamento de objetivos de negócio e aspectos do produto definidos através de critérios de priorização e importância. Ao final do Planejamento, obtêm-se, portanto, um *roadmap* do produto.

Desta forma, são consideradas como principais atividades do Planejamento:

- Construir a Visão do Produto;
- Planejar o *Roadmap*.

6.2.2.3 Fase de "Execução"

A fase de construção do produto contempla o planejamento, especificação e implementação dos objetivos de negócio e das

características-chave do produto sendo desenvolvido, contidas em *releases* e construídas iterativamente (*Sprints*). Segundo o SISP (2015b), estas atividades são definidas a partir de algumas práticas de métodos ágeis, tais como: *Scrum*, *XP* e *Lean*.

Dada a natureza ágil de incentivo à mudanças constantes durante a construção, o SISP (2015b) cita dois tipos de mudanças em funcionalidades:

- Refinamento – resulta da evolução natural dos requisitos, como um aprimoramento do entendimento das histórias de usuário (telas/funcionalidades), com o aprofundamento, detalhamento e complementação de requisitos durante o processo de desenvolvimento ágil;
- Mudança de Escopo – resulta da evolução das políticas, estratégias, legislação, objetivos específicos de negócio e necessidades ou processos de negócio da empresa.

Também visando fornecer um melhor entendimento acerca do fluxo de atuação, é importante destacar na "Construção do Produto" as principais atividades de construção da *release* e desenvolvimento de iterações:

- *Release*
  - Elaborar o *Backlog* do Produto;
  - Planejar o *Release*.
- Iterações
  - Reunião de Planejamento da Iteração;
  - Executar a Iteração;
  - Reunião de Demonstração da Iteração;
  - Reunião de Retrospectiva;
  - Validar Incremento de *Software*.

6.2.2.4 Fase de "Transição"

A fase de "Transição do Produto" contempla as atividades que visam garantir a implantação de cada *release*, contemplando

também a avaliação dos resultados obtidos e condições gerais de entrega, com suporte ao produto do projeto.

Para o SISP (2015b), cada *release* deve passar por um processo de homologação, com execução das atividades de transição do *software* para produção. Desta forma, as atividades de transição visam garantir que clientes e usuários tenham condições de usar este produto de *software*.

São atividades de Transição do Produto:

- Homologar a *release*;
- Preparar e realizar treinamentos;
- Realizar reunião de encerramento da *release* ou do projeto.

7. Enxugando a Máquina: fazendo mais com menos

Com base nas referências apresentadas, vamos agora "enxugar a máquina"! Interprete isso como "manter a operação funcionando de forma sustentável, mensurável e otimizada", para que possamos cumprir nossos propósitos específicos, tais como:

- planejar e/ou acompanhar as entregas, estratégias e os processos da organização;
- identificar as fontes de desperdício vinculadas ao processo produtivo;
- oportunizar o autoconhecimento nas equipes de desenvolvimento e na organização como um todo;
- gerar os insumos necessários para alimentar o processo de melhoria contínua;
- gerar informações confiáveis para viabilizar a tomada de decisão.

Neste contexto, entenda "desperdício" como as "atividades ou coisas que não agregam valor ao negócio, ao produto ou sequer aos processos da organização", e que assim sendo, representam a energia e os recursos utilizados em vão enquanto tentamos alcançar nossas metas. Logo, ele é um obstáculo à eficiência!

Buscando na *web* ou num dicionário, talvez você encontre ainda os seguintes significados para "desperdício":

- despesa ou gasto exagerado;
- esbanjamento, uso sem proveito;
- todas as coisas que não se aproveitam; sobejos, sobras.

Todos os conceitos citados se aplicam perfeitamente à proposta desta obra.

7.1 Objetivos & Benefícios

Com reduzido investimento e menor impacto na operação em relação às propostas mais tradicionais, a fórmula "Enxugando a Máquina" proporciona aos times e gestores que atuam com desenvolvimento de *software* ágil e enxuto, a capacidade de planejar e/ou acompanhar processos e a cadeia de entrega na organização. Isso foi viabilizado ao encaixarmos métodos de estimativa e medição funcional como "plugins" do processo de desenvolvimento, tornando-o mensurável ao longo do tempo.

Processos mensuráveis são obviamente vistos como mais transparentes e consistentes, sendo que tais características são potencializadas pela utilização de métricas passíveis de repetição, objetivas e independendentes de uma solução técnica específica (ou seja, "de implementação"). Esse tipo de avaliação fomenta uma busca constante por fontes de desperdício, garantindo um "padrão mínimo" mais eficiente enquanto processo de desenvolvimento. Isso é estruturante e valoroso, inclusive como direcionador estratégico para a TI.

Todavia, é importante destacar que esta proposição "NÃO" foi criada como uma estratégia para "CONTROLE" dos times de desenvolvimento, assim como também não tem a intenção de compará-los diretamente, o que provavelmente não seria interpretado corretamente pelos *stakeholders* e inclusive prejudicaria a sustentabilidade do modelo ágil e enxuto. O seu objetivo principal então, é prover capacidade de "ACONPANHAMENTO" que seja considerada confiável e segura na perspectiva dos gestores e do mercado, a respeito do trabalho a ser feito, em andamento ou concluído.

Desta forma, "Enxugando a Máquina" viabiliza uma avaliação abrangente e sensata dos métodos utilizados frente aos resultados obtidos, através de esforços e recursos despendidos pela organização num viés principal de eficiência, embora também relacionando aspectos de eficácia, efetividade, maturidade e clima das equipes. Portanto, diferentemente da maioria das propostas em voga no

mercado, não usa a capacidade de estimar e/ou medir *software* apenas para derivação ou análise de esforço, mas para fornecer o aparato completo necessário para uma medição de desempenho da operação. Neste cenário, entenda o termo "medição" como a quantificação precisa de objetos e funcionalidades do escopo, a qual é realizada com base em métodos objetivos e passíveis de repetição.

No contexto ágil atual, então cabe uma nota a respeito de *Story Points*, comumente referenciada no mundo ágil, visto não ser esta uma métrica ou medida de tamanho, que tampouco pode ser considerada "objetiva" e/ou "repetível", pois sua função é apenas estabelecer uma ordem de esforço através de percepções de indivíduos de um determinado grupo diretamente envolvido com o desenvolvimento. Contudo, embora considerada eficaz para o propósito de planejamento, a sua utilização não se mostra suficientemente precisa e segura para balizar a tomada de decisão estratégica, já que é norteada por opiniões e não por análises incontestáveis. Com ela, sequer há como realizar algum *benchmarking* ou prover qualquer tipo de racional passível de auditoria.

O uso de métricas guiadas pela visão do usuário representa um fator normalizador vinculado a entrega de valor na perspectiva do produto, e reforça o entendimento de que a TI normalmente é "o principal meio" e na maioria das vezes "não o próprio negócio". Assim sendo, é importante que uma linguagem comum possa ser estabelecida entre os parceiros, fomentando a transparência a partir da máxima que tanto o negócio deveria ser capaz de acompanhar o desempenho da TI ao longo do tempo em termos de suas entregas, como a TI também deveria conhecer melhor a si para certificar-se de que está situada dentro dos parâmetros esperados, sejam internos e/ou de mercado, provando que os investimentos têm sido traduzidos em resultados otimizados, através de processos orientados por critérios de eficiência, logo, com menor incidência de desperdícios.

Em abordagens como essa, os executivos valem-se de meios de avaliação e projeção do negócio ao longo do tempo, antecipando-se a eventos que dependem ou impactam suas operações, sem

especulações. Este diferencial competitivo pode ser considerado um importante mecanismo de eficiência e *compliance*, dado o cenário de redução de custos e riscos, com elaboração de provas de valor mais consistentes para *business cases*. Este tipo de necessidade está em evidência, principalmente nas grandes corporações; e assim tende a permanecer.

7.2 Princípios & Valores

Assim como o "Manifesto Ágil" e o "*Lean*", a abordagem "Enxugando a Máquina" também possui princípios e valores que orientam os seus utilizadores em direção aos objetivos que foram delineados.

Sendo assim, são considerados princípios gerais desta obra:

- buscar o auto-conhecimento dos times, com aprendizado contínuo, tanto para nortear as discussões sobre o passado, como para projetar as decisões no presente e para o futuro da organização ("usar o ontem para trabalhar o hoje, viabilizando o amanhã");
- atentar para o que está ocorrendo na operação, assim como aos respectivos impactos na estratégia delineada para o produto, time, organização ou mercado;
- buscar "não controlar", mas "acompanhar", contribuir e facilitar o trabalho dos times, provando o valor das intervenções e evoluções realizadas;
- focar na identificação e na eliminação de desperdícios, conscientemente e como prova da eficiência "do todo";
- valorizar as pessoas e o que elas sabem fazer mais do que o detalhamento excessivo dos processos, porém garantindo o "padrão mínimo" que mitiga o caos;
- incentivar o uso de boas práticas, sem fundamentalismo e considerando SEMPRE a proposição de valor para "o todo".

Além dos princípios elencados, há valores fundamentais a serem desenvolvidos pelos indivíduos, para que eles se sintam ver-

dadeiramente engajados em uma busca particular pelos melhores resultados coletivos. São estes valores:

- ética (acima de tudo);
- respeito ao próximo;
- engajamento (o "fazer acontecer");
- valorização do bem comum e busca pela relação ganha-ganha (nas relações pessoais e comerciais);
- empatia (colocar-se no lugar do outro);
- colaboração (sem interesses ou formação de feudos culturais, funcionais e/ou políticos);
- transparência (em todas as ações realizadas).

7.3 Funcionalidades

"Enxugando a Máquina" provê um conjunto de "ferramentas", onde cada uma vincula-se à propósitos específicos; e é assim que você monta a sua própria caixa de ferramentas!

7.3.1 *Backlog*

O *backlog* é a relação de afazeres em espera (itens "a fazer" ou que estou "preparado para fazer") ou cujo desenvolvimento está sendo executado (itens "fazendo"). Ele pode assumir diferentes níveis de abstração/ visualização, tais como:

- Do "Produto" - contemplando todos os Épicos, *Features*, *User Stories* e "Tarefas" que envolvem ou impactam o seu escopo;
- Do "Épico" ou "Bloco de Trabalho" - definido a partir das *Features*, *User Stories* e/ou "Tarefas" que são planejadas para um intervalo de tempo, normalmente um período de 3 à 8 meses;
- Do "MVP" - considerando as *Features*, *User Stories* e/ou "Tarefas" de uma entrega que tenha vinculado um determinado propósito de valor;

- Da "*Sprint*" - contemplando as *Features*, *User Stories*, *Tarefas* e/ou registros de Melhorias que estão sendo ou serão trabalhadas na *sprint* (perspectiva da cadência de trabalho da TI).

Antes de adotar um nível de trabalho, avalie-o em relação às necessidades, assim como frente aos "sistemas de informação" disponíveis na organização e o modelo de trabalho que será executado. Desta forma, é possível inclusive que uma mesma pessoa trabalhe simultaneamente com uma visualização de MVPs e Épicos ou ainda na perspectiva de "*Sprints*".

Veja abaixo, uma ilustração sobre a formação de *backlogs* num contexto fictício, para um melhor entendimento desta estratégia:

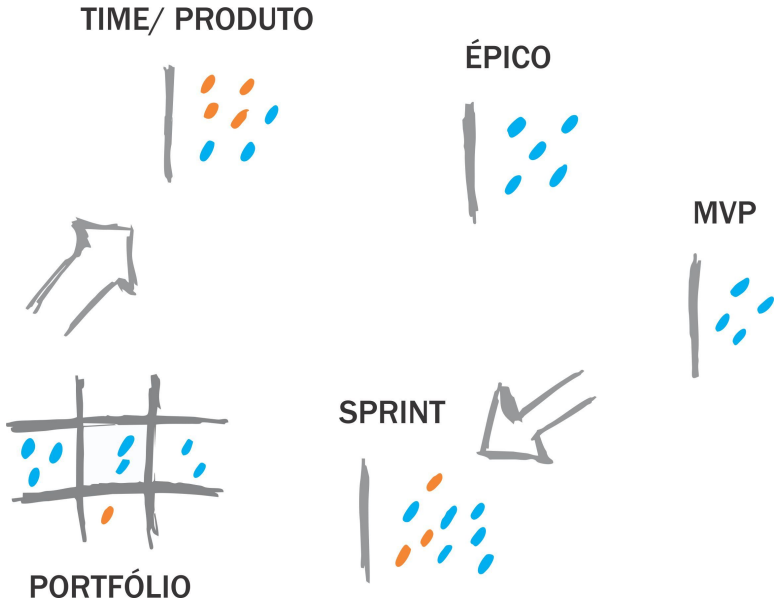


Figura 25 – Níveis de *Backlog*

7.3.1.1 Do produto

O "*Backlog* do Produto" engloba itens ordenados e priorizados no âmbito do produto, como o próprio nome refere. Isso possibilita

avaliações do que há de ser feito e do trabalho que já está em andamento no time, considerando que normalmente há 1 time para 1 produto - embora exista possibilidade de segregação de visões do time e de produto. Sendo assim, estamos trabalhando com um cenário onde o *backlog* do produto e do time compreendem os mesmos itens, e portanto, apresentam a mesma lista.

Embora único, o *backlog* do produto pode ser segregado em linhas de "Evolução do Produto" e "Sustentação do Produto" (Figura 26), uma estratégia que mostra-se útil para prover visibilidade sobre linhas orçamentárias e/ou prioridades diferentes, como também para o planejamento da alocação dos membros do time.

Figura 26 – Sub-backlogs

Há uma tendência natural de identificarmos maior volume e esforço relacionado aos itens de "Evolução do Produto" quando comparados aqueles que são dedicados à "Sustentação do Produto", visto que os primeiros originam-se das necessidades normalmente vinculadas às ações estratégicas de negócio, as quais são trabalhadas na forma de MVPs ou Melhorias. Os itens de "Sustentação do Produto", por sua vez, constumam ser tratados como "Incidentes" ou "Problemas", tipicamente demandas de manutenção corretiva, adaptativa ou perfectiva, e que por sua origem e impacto, nem

sempre são passíveis de serem planejados com antecedência. Eventualmente, ocorrem Melhorias oriundas do próprio time e que não precisam ser priorizadas por um Comitê externo, logo, este tipo de demanda também pode ser classificado como atividade de "Sustentação do Produto".

Neste cenário, note que as demandas críticas e que impactam no adequado funcionamento do produto ou do negócio atendido, são priorizadas e classificadas no *sub-backlog* de "Sustentação do Produto", recebendo assim uma atenção especial em relação aos itens de "Evolução do Produto". Então, para garantir o foco do time nas suas entregas, sugere-se escalar um ou mais membro(s) ou percentual médio de alocação do time na *sprint* para atender atividades específicas de "Sustentação do Produto".

A reserva de alocação é considerada uma boa prática, cujo objetivo é manter o planejamento e o orçamento dos *Targets* (projetos e demandas) sob controle. Oportunamente, mostraremos opções para separar o *budget* consumido dentro de um MVP ou *Sprint*, pois você pode ter que eventualmente trabalhar com diferentes *Targets* em paralelo, como projetos e pequenas melhorias.

7.3.1.2 Do épico

Para acompanharmos as necessidades planejadas e/ou em andamento que foram dedicadas a um time de desenvolvimento específico ("Blocos de Trabalho" compostos pelos *Targets*), ou ainda, conforme o impacto de um "*Target*" no time (portanto, um "Épico"), idealmente devemos considerar um horizonte compreendendo entre 3 e 8 meses de trabalho (normalmente, 6 meses), o que possibilita uma perspectiva mais realista do produto, conforme as evoluções alinhadas com os representantes do negócio e o seu respectivo tempo de validade em relação ao mercado. Pode-se dizer então, que um "Épico" visa atender os objetivos do negócio em um período finito de tempo, através da entrega de parcelas mínimas de valor (os "MVPs").

De forma complementar, abaixo está representada a "Cebola

do Planejamento", um conceito que foi adaptado nesta obra para demonstrar os diferentes níveis de informação utilizados e/ou providos pela abordagem "Enxugando a Máquina":

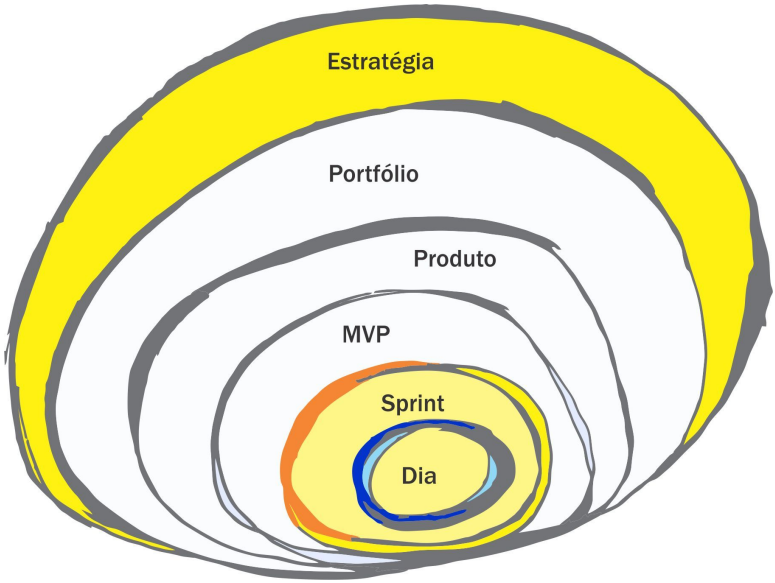


Figura 27 – Cebola do Planejamento

Posteriormente, também mostraremos como os "Blocos de Trabalho" ou os "Épicos" podem ser segregados em "MVPs" e como garantir um fluxo de entrega de valor gradual e constante durante o ciclo de vida do produto.

7.3.2 Eventos

As cerimônias ágeis fomentam o engajamento e a colaboração entre os integrantes dos times de desenvolvimento e/ou outros *stakeholders*. Elas afloram a criatividade, incentivando a cocriação, seja durante uma Concepção ou ao longo da execução dos projetos.

Esse recurso é muito bem-vindo também numa perspectiva enxuta, pois oportuniza melhorias, reduz o retrabalho e viabiliza

uma atuação mais efetiva para eliminação dos impedimentos e gargalos identificados. Há ainda, maior garantia de atendimento às necessidades mapeadas e priorizadas, algo indispensável num ambiente caracterizado por mudanças frequentes.

7.4 Portfólio & Esteira de Entregas

A "Esteira de Entregas" é um mecanismo criado com o objetivo de facilitar a comunicação dos planos de trabalho relacionados aos subportfólios, seja para acompanhamento pelo PMO e/ou demais *stakeholders*. Seu propósito é melhor entender, descrever e/ou discutir os assuntos relacionados aos blocos de valor produzidos, em espera ou em construção, para habilitar maior transparência na gestão na operação.

Com esta proposta, os *stakeholders* visualizam os "Targets", "Épicos", "MVPs", "Melhorias", "Problemas" e "Incidentes", além de eventuais dependências entre as entregas vislumbradas; e a TI, por sua vez, consegue obter uma visão do plano de *sprints*, também com suas respectivas interdependências, que são confrontadas com as cadências dos times.

Nos tópicos a seguir, veremos detalhes da "Esteira de Entregas" e de como ela pode ser utilizada numa operação.

7.4.1 Portfólio de projetos

O processo de "Portfólio de Projetos" utiliza alguns meios simplificados, mas bastante eficazes, para viabilizar a gestão de demandas de *software* da organização. Assim, habilita-se o acompanhamento do portfólio no nível de MVP, ou seja, com base na perspectiva de valor e considerando uma linguagem comum entre TI e negócio.

Considerando que o portfólio é normalmente segregado em subportfólios por linha de negócio e que o seu acompanhamento e gestão é implementado num quadro *Kanban*, o monitoramento de Projetos, Épicos e/ou MVPs torna-se menos oneroso e provê uma

forma de manter a rastreabilidade necessária entre os prospectos/demandas ("Targets"), o que é potencializado pelo plugin de medição escolhido.

Um quadro *Kanban*, embora seja também um recurso muito simples, é uma poderosa ferramenta de gestão, e em conjunto com a utilização de sistemas de registro proporciona meios de acompanharmos o atingimento dos objetivos definidos e priorizados junto ao negócio.

7.4.2 Projetos e épicos

Um *Target* pode impactar um ou mais produtos, assim como um produto pode evoluir através de um ou mais *Targets*, inclusive executados simultaneamente. Ocorre que, se a organização é projetizada, o *budget* provavalmente não será segregado por produto, mas por projeto (*Target*). E sendo assim, as preocupações sobre as fontes de recursos devem ser amenizadas, para que não venha a prejudicar o trabalho em progresso nos times.

Há várias formas de minimizarmos o impacto da conturbada visão de "Projetos" *versus* "Produtos". Entre elas, a segregação do orçamento poderia ocorrer por item de uma contagem de pontos de função, por *Feature*, ou ainda, por MVP (quando trabalhando com "Épicos"), já que estes elementos representam o cumprimento de necessidades parciais do produto, ou seja, configuram unidades mínimas de valor.

Pode ocorrer também, uma situação onde um *Target* demande de um time específico apenas uma pequena porção de funcionalidades, sendo este conjunto considerado pequeno demais para a realização de uma Concepção ou conhecido demais para isso em termos de escopo. Neste caso, podemos avaliar o uso de meios alternativos de descoberta ou confirmação de escopo, como o processo de *Design Thinking* e até uma simplificação da receita "Direto ao Ponto".

É igualmente relevante avaliar o contexto de um Épico abrangente com impacto em outro(s) time(s), pois talvez não seja preciso rodar um método conceptivo para cada time envolvido, conside-

rando que times diferentes podem colaborar numa mesma cerimônia. Assim, uma Concepção pode envolver mais de um produto, com interdependências e membros de mais de um time.

Lembre que um *Target* poderá ser subdividido em Épicos por Produto/Time, os quais são trabalhados na forma de MVPs, e ainda, que o time atuará em Melhorias, Incidentes e Problemas. Espera-se então, que o time tenha foco em uma única entrega por vez, com o intuito de doutrinar e praticar uma organização *just-in-time*.

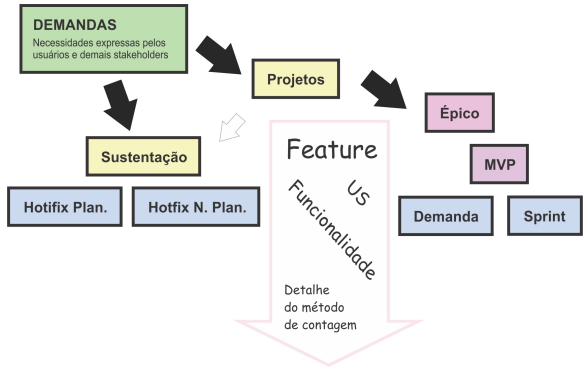


Figura 28 – Subunidades

Contudo, considerando o propósito de acompanhar o desempenho dos times (no caso de *outsourcing*) pode fazer mais sentido incluir no seu modelo de estimativa e medição, avaliar métricas gerais apenas de etapas específicas, tais como de "Execução" e "Transição", visto que as demais atividades estão relacionadas ao levantamento, planejamento e suporte ao desenvolvimento.

Projetos e épicos podem ser avaliados em diferentes níveis de granularidade nos quadros *Kanban*, onde podemos analisar o que estamos preparados para fazer, o que está em andamento e/ou o que já está feito. E o trabalho dos times também pode ser visualizado num diagrama de fluxo cumulativo (*cumulative flow*), que viabiliza o acompanhamento da saúde do sistema. No *link" abaixo, veja, por exemplo, como o SAFe trata isso:

http://www.scaledagileframework.com/portfolio-kanban/

7.4.3 Esteira de produtos mínimos

Espera-se que uma entrega adicione valor ao produto/ negócio, sendo esse o principal objetivo do desenvolvimento de um MVP ou Melhoria. Normalmente este valor será derivado das necessidades estratégicas e/ou diretas do usuário.

Os MVPs e Melhorias são executados num sistema de produção puxada, onde os times de desenvolvimento dedicam-se à prover valor gradualmente, de blocos com tamanhos similares ao longo do tempo. Isso é uma boa prática, pois o valor tende a ser melhor percebido pelo usuário, que terá o tempo necessário para compreender as mudanças realizadas.

Neste contexto, considerando que um MVP pode corresponder ou não a uma entrega em produção, adotaremos o conceito de "Entrega" como correspondente à "MVP" ou "Melhoria" na maioria das nossas explanações, já que ao menos em teoria, neste momento haveria a construção de um incremento pronto e potencialmente entregável em produção como uma porção de valor completa e significativa ao usuário. Contudo, isso não impede que exista entrega de valor parcial inúmeras vezes, dentro ou ao final das *Sprints* ou um conceito próprio de "feito" ou "pronto".

Os MVPs devem ser dispostos sequencialmente na esteira, mesmo que contribuam com diferentes projetos da organização. Esta forma de visualização facilita a compreensão, discussão e argumentação, inclusive proporcionando reavaliação de prioridades estratégicas, o que não costuma ser percebido tão naturalmente pelo negócio quando a visibilidade se dá simplesmente através de um mapa de iterações ou no nível de projetos.

Denominamos este mecanismo "Esteira de Produtos Mínimos", já que serve para prover o acompanhamento do sistema de produção no nível de "Portfólio de Projetos", conforme os objetivos delineados pelo negócio e acordados com os times. Logo, provavelmente o Escritório de Projetos, apoiado por uma equipe de POs da organização poderia encarregar-se de alinhar expectativas e prioridades com o negócio (ou seu Comitê), visando garantir a execução do plano de

trabalho.

Para melhor elucidar o conceito e demonstrar a utilidade do mecanismo, em outras palavras: a "Esteira de Produtos Mínimos" provê um sequenciamento de entregas para facilitar o seu monitoramento, pela TI e pelo negócio e ainda, para acompanharmos as entregas ordenadas e planejadas numa linha de tempo, conforme sua proposição de valor. Isso ocorre devido ao MVP ser a unidade granular padrão de valor para uma organização *just-in-time*, sendo eles parcelas das metas estratégicas.

Para visualizar as conexões existentes entre os projetos de um mesmo programa/portfólio ou mesmo entre diferentes produtos/times ligados a um mesmo projeto, pode ser conveniente visualizar uma esteira também na perspectiva de *Sprint*, com o intuito de apoiar o planejamento das equipes de desenvolvimento de *software* e dos líderes de projetos envolvidos. E ainda, ao utilizar uma janela de tempo padrão das iterações (*time-boxes*), vislumbra-se que, em tese, isso otimizaria a cadeia de produção, principalmente caso seja necessário enquadrar-se a períodos específicos de liberação de pacotes em produção, o que evitaria a formação de estoques pré-produção (notadamente um tipo de desperdício). Esta estratégia de sequenciamento e paralelidade de *Sprints* é conhecido como *Train* e mostra-lhe útil como um plano de trabalho abrangente da TI para os MVPs, Épicos, Produtos, Times e *Targets*. Perceba que essa é apenas uma dica, já que *Train* é uma abordagem que costuma ser empregada apenas em estruturas onde muitos times atuam paralelamente (conheça detalhes desta abordagem no *Framework* do SAFe).

Abaixo, confira um exemplo de plano de trabalho com diferentes iterações relacionadas a um Projeto, onde tornam-se visíveis os respectivos Épicos e MVPs:

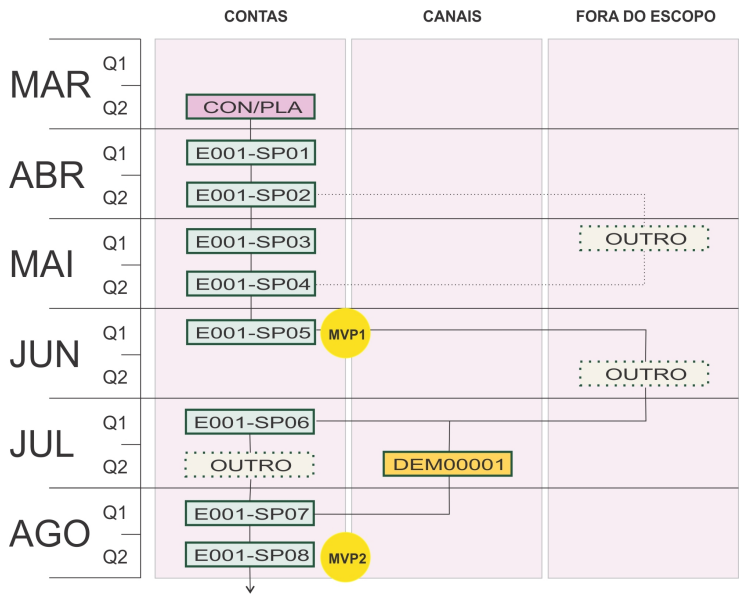


Figura 29 – Plano de Trabalho por Sprint

Como você pode perceber, a principal função do acompanhamento por iterações é seu uso pela TI, para que ela possa melhor executar, acompanhar e organizar o seu plano de trabalho, além de realizar a previsão de alocação dos times. Serve também de insumo às cerimônias de *status* de projetos e de ações relacionadas.

Pelos motivos citados, a abordagem foi batizada "Esteira de Produção", pois é um mecanismo que se preocupa primariamente com a montagem e evolução de "peças" do que de um produto 100% acabado (já que o MVP entrega parcelas de valor do produto).

O mais interessante é que não precisamos restringir-nos ao estabelecimento de um plano de trabalho, visto que há outros recursos vislumbrados, como o monitoramento da saúde do sistema, que se torna evidente com base nos dados coletados, nas ferramentas de apoio ou mesmo no quadro *Kanban*. Isso ajuda a acompanhar o trabalho em progresso com o intuito de adaptá-lo à capacidade da esteira. Veja abaixo, um exemplo de monitoramento visual, sendo que mais informações constam na sessão Monitoramento Visual.

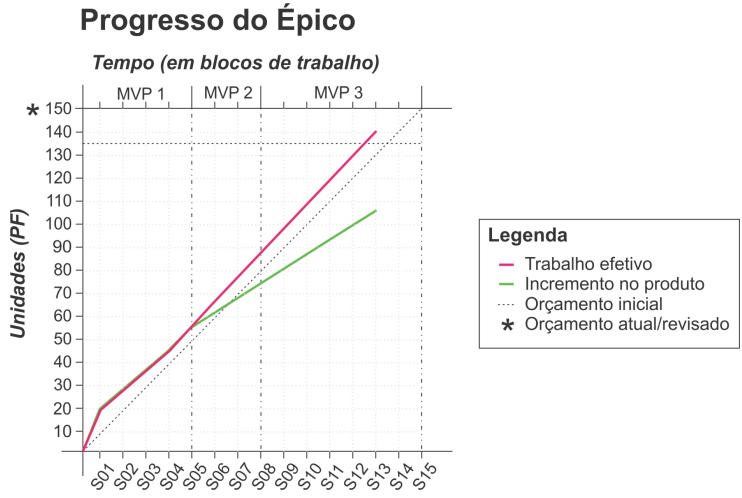


Gráfico 38 – Gráfico de *burn-up* de progresso: realizado e efetivo

7.4.4 Controle do fluxo de produção

O "Controle de Fluxo de Produção" é composto por 2 ou mais "Esteiras de Produtos Mínimos", considerando um subportfólio ou toda a operação. Ele pode ser visualizado no nível de "Time", "Produto" ou "Projeto" e sua função é garantir o modelo *just-in-time*, que otimiza o fornecimento frente à capacidade produtiva.

7.5 Desenvolvimento Lean MVP: adaptado

Construir produtos enxutos não parece tarefa fácil ao migrar de um modelo de desenvolvimento tradicional, pesado e com procedimentos rígidos associados. E note: realmente não é!

Um processo de *software* "enxuto" é formado por um conjunto de procedimentos versáteis e normalmente exige que as pessoas estejam engajadas e dispostas a serem mais flexíveis. Depende ainda, do estabelecimento de uma cultura orientada à princípios

e valores bem definidos, compreendidos e praticados; isso como forma de garantir foco nos objetivos dos *stakeholders*.

Para obtermos sucesso nas ações de aculturamento, tente organizar "clínicas de conhecimento" dinâmicas e cativantes. E dependendo do público, avalie a realização de ciclos de TEDs, *workshops* e/ou trilhas de palestras, inclusive convidando pessoas externas à organização para falar a respeito da sua experiência corporativa com processos ágeis e enxutos. Engajar pessoas em eventos colaborativos e multiculturais também ajuda a quebrar o gelo, melhora o clima e fomenta a empatia; além disso, contribui efetivamente para difundir conhecimento a respeito de métodos e conceitos fundamentais.

Lembre-se que o envolvimento das áreas de negócio e/ou de seus representantes é fundamental, assim como de todas as áreas de TI envolvidas ou impactadas. Isso deve ocorrer sempre, e desde cedo. Portanto, envolva-os e não esqueça de que são eles os patrocinadores daquilo que você faz, logo, na prática eles são os seus clientes diretos!

Agora você conhecerá detalhadamente as etapas do ciclo de desenvolvimento *Lean MVP*, que foi adaptado para viabilizar uma abordagem mensurável.

7.5.1 Iniciação

A etapa de "Iniciação" não integra o processo de Desenvolvimento de *software*, mas o de Gestão de Porfólio, que por sua vez, é vinculado ao processo de Gerenciamento de Projetos. Contudo, é fundamental para explanarmos os recursos de "Enxugando a Máquina, demonstrarmos como ocorre a sua conexão com o desenvolvimento.

Neste contexto, o termo "Target" ("alvo", em português) foi criado para generalizarmos as diferentes nomenclaturas utilizadas no mercado ao referir demandas de *software*, sendo esta a forma que encontramos para deixarmos claro o uso como uma mesma unidade granular. Assim, um *Target* pode referir-se à termos tais

como: “Demanda”, “Prospecto”, “Projeto”, “Demanda”, “Melhoria”, entre outros.

Em termos de requisitos iniciais, a definição do objetivo do *Target* é tratada durante a etapa de “Iniciação”. Neste momento, também esclarecemos a justificativa para o desenvolvimento e/ou a descrição preliminar do que ele se propõe na perspectiva corporativa. Ele é “o que a organização espera receber em termos de produto, serviço ou resultado” e por isso sua execução deve ser priorizada junto aos representantes do negócio.

Figura 30 – Processo de Desenvolvimento Ágil e Enxuto

A “Iniciação” possibilita elencarmos hipóteses iniciais sobre o negócio e/ou solução, servindo de insumo para as estimativas de altíssimo nível, as quais são idealmente realizadas com base num histórico de execução do processo de Desenvolvimento de *software*, com base na realidade de cada time envolvido.

Também são definidos nesta etapa os “Épicos” de um *Target* (ou “Blocos de Trabalho” formados por diversos *Targets*). Eles formam uma perspectiva inicial do escopo utilizada nas estimativas do “Portfólio de Projetos” e/ou seus subportfólios.

Como o SAFe sugere, visando representar a descrição do Épico,

uma estória em altíssimo nível pode ser utilizada para descrever o que o negócio pretende obter de valor para si (algo como a missão dada ao time), definindo já os critérios de aceitação relacionados. Cabe observar, que a visão de produto será refinada durante a Concepção de cada Épico ou Bloco de Trabalho, e sendo assim, logo ficará mais completa.

Para realizar estimativas de *Target*, assim como para obter detalhes adicionais, leia o tópico "7.6 - Estimativa e Medição", neste capítulo.

7.5.2 Concepção

Como vimos nos capítulos iniciais deste livro, a "Concepção" (ou "Concepção Enxuta") é um *workshop* composto por uma série de dinâmicas, cujo objetivo é identificar os itens certos a serem desenvolvidos no momento adequado pelo time, conforme sua expectativa de valor, prioridade e recursos disponíveis; para que sejam atendidos os objetivos dos *stakeholders*.

Normalmente, a Concepção contempla um único *Target*, mas dependendo da estrutura da organização e do propósito envolvido, por ventura poderá abranger vários deles no seu escopo; geralmente, mas não necessariamente, contando com um único "Time" ou "Produto". Caso a Concepção seja executada com múltiplos *Targets*, chamamos o seu objeto "Bloco de Trabalho". Caso contrário, e ocorra no âmbito do produto*, chamamos "Épico".

Para relembrarmos, basicamente, as seguintes atividades (alto nível) compõem o *workshop* de até uma semana que o ajudará a conceber o *roadmap* do produto:

- Dinâmicas do *workshop* conforme os passos explicados e demonstrados no livro "Direto ao Ponto";
- Atividades que documentam a passagem de conhecimento (dependendo do modelo de *sourcing* e definição de estratégia de manutenção de conhecimento sobre o produto e suas demandas);

- Estimativa inicial do Épico (detalhamento no tópico "7.6 - Estimativa e Medição");

Uma Concepção pressupõe a participação do negócio, dos membros dos times e dos profissionais *cross* da TI e negócio. Se possível, também devem participar eventuais fornecedores de *software* envolvidos (prestadores de serviço). Ao final deste trabalho, como entregáveis teremos: o "Sequenciador de *Features*" e eventualmente, o "*Canvas MVP*".

Atente que uma semana é o período ideal para tratar os Épicos ou Blocos cujo desenvolvimento compreenda algo entre 3 e 8 meses de duração, sendo que tanto este método simplificado ou o processo de *Design Thinking* podem ser aplicados aos *roadmaps* com menos de 3 meses ou com escopo realmente muito conhecido, com alto nível de certeza de negócio e técnico. Períodos maiores que 8 meses devem ser evitados num Épico, pois há alto risco dos requisitos mudarem, até mesmo devido ao tempo de mercado - neste caso, o ideal é realizar outra Concepção.

Também realizamos pequenas adaptações no modelo para viabilizar uma estratégia escalável de *outsourcing*, com transição de conhecimento da organização cliente para um fornecedor de *software* e vice-versa (quando da finalização do desenvolvimento). Isto é muito útil desde que tanto o negócio como a TI participem ativamente do *workshop* e continuem a manter uma interação constante com a outra parte durante a execução do trabalho. Assim, indica-se a transferência de conhecimento através de uma "Concepção Assistida", na qual, ao final de cada dia do *workshop* um vídeo é gravado com o propósito de consolidar os principais assuntos abordados. Convenhamos que independente da forma de trabalho ou estratégia de *sourcing*, isso seria uma forma de manter o conhecimento relevante sobre os requisitos do produto acessível aos desenvolvedores e demais responsáveis pela realização das entregas, servindo inclusive como histórico para eventuais necessidades de auditoria (cenário aplicável à realidade de instituições financeiras e órgãos públicos, por exemplo).

7.5.3 Planejamento

A etapa de "Planejamento", executada após a Concepção, engloba a elaboração do plano de trabalho e a preparação da infraestrutura necessária aos times, além de eventuais necessidades de aquisição de componentes de *software* ou *hardware*, ou mesmo a contratação de pessoas, etc.

Ao utilizar o LEAN MVP, a etapa de Planejamento contempla as seguintes atividades:

- Criação ou atualização do "Plano de Trabalho";
- Planejamento e preparação da infraestrutura;
- Estimativa inicial do Épico (detalhamento no tópico "7.6 - Estimativa e Medição"), que pode ser revisada em relação à Concepção;
- Aquisição de recursos de *software*/ *hardware* ou contratação de pessoas (se necessário);
- Contratação do Épico (caso se aplique essa estratégia):
  - Planejamento da contratação;
  - Contratação do Épico.

O "Planejamento" deve garantir os habilitadores que o time necessita para desenvolver suas entregas. E para isso, nas grandes organizações, os profissionais *cross* normalmente são envolvidos pontualmente durante o ciclo de vida do produto, embora não deficados à uma equipe específica. São exemplos de atividades que estes profissionais executam:

- Criação de ambientes;
- Avaliação de requisitos de arquitetura ou relacionados à segurança da informação;
- Negociações de contratos;
- Aquisições de *software* e *hardware*;
- Contratação de fornecedores/ pessoas para compor a equipe, etc.

A alocação dos profissionais *cross*, sempre que possível deve ser planejada com antecedência, sendo bastante provável que isso possa ocorrer em torno de 1 à 2 semanas anteriores à necessidade, já que o final da Concepção é o marco inicial do Planejamento e que cada etapa tem uma duração sugerida de 1 semana.

7.5.3.1 Escalando

Como já abordado, num regime de "fábrica de *software*" (*outsourcing*), vídeos da Concepção poderiam ser elaborados diariamente ("Concepção Assistida"), contemplando um conteúdo consolidado de cerca de 10 minutos/dia. Sendo assim, ao final da semana do *workshop*, teríamos um total de 5 vídeos para transferência de conhecimento, totalizando 50 minutos de informações relevantes sobre a origem dos requisitos.

Estratégias de escala devem ser evitadas, mas eventualmente podem ser necessárias, por exemplo, devido à sazonalidade de demandas. Ocorrendo isso, é importante que a passagem de conhecimento seja suficiente para dar início à etapa de Execução, guiando os desenvolvedores rumo à Transição, para viabilizar a entrega de um produto de qualidade e conforme as expectativas para manutenção pelo time perene - que deve ser o responsável pelo aceite junto ao negócio e pelo *merge* dos fontes.

Sem dúvidas, esta é uma estratégia a considerar quando novas demandas já não podem mais ser absorvidas pelo time ou quando não representam volume continuo e suficiente a ponto de justificar adição de membros ou mesmo um novo time.

Seguindo esta abordagem, é importante esclarecer que um ou mais membros do time devem acompanhar a execução da demanda escalada, participando das cerimônias de modo a garantir que as pessoas estão realmente envolvidas e contextualizadas a respeito do que precisa ser feito.

Como resumo e visando facilitar o entendimento desta proposta, veja a ilustração abaixo, que nada mais é do que uma analogia com os conceitos de *trunk* e *branches*, vinculados à disciplina de

gerenciamento de configuração de *software*, mas agora aplicados aos processos de aquisição e desenvolvimento:

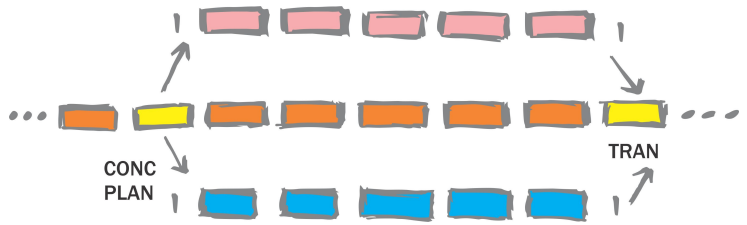


Figura 31 – Modelo de Escalada

Na ilustração, perceba que o tronco deriva galhos, que por sua vez, representam o aumento de abrangência e capacidade do time na linha do tempo. No caso fictício em questão, num dado momento, uma Concepção foi realizada e o time consolidou seu conteúdo como passagem de conhecimento; realizou ainda, uma estimativa de pontos de função (vide tópico relacionado). Já na etapa de Planejamento, foi contratada uma ordem de serviço para o desenvolvimento de MVPs, considerando uma concorrência indireta entre os fornecedores homologados, para os quais já havia sido fixado um preço ou uma faixa de preço unitário do ponto de função, o que foi definido em contrato prévio estabelecido entre as partes. Ao final da etapa, a solução foi contratada e contou-se o prazo para alocação dos profissionais do fornecedor e liberação de acessos, iniciando o desenvolvimento.

7.5.4 Execução

A "Execução" é uma etapa do ciclo onde os metódos de desenvolvimento e as disciplinas de engenharia de *software* são conectados com o intuito de prover maior agilidade da TI e efetivo valor ao negócio.

Nesta etapa, faz-se o uso de práticas provenientes do *Scrum*, *Kanban* e *Lean*, conceitos apresentados nos capítulos iniciais e cuja configuração pode ser facilmente customizada. Como aqui não há

uma "receita de bolo", então "a melhor solução é aquela que melhor atende aos propósitos de cada organização".

Basicamente, a "Execução" engloba as seguintes opções metodológicas:

- Práticas de requisitos, análise, projeto, codificação e testes;
- Iterações (*sprints*), incluindo atividades e cerimônias do *Scrum*, além de práticas oriundas do *Kanban* e do *Lean*;
- Atividades relacionadas ao gerenciamento de projetos e gestão de portfólio, incluindo *reports* e tarefas de acompanhamento;
- Contagens de pontos de função (vide detalhamento no tópico "7.6 - Estimativa e Medição"):
  - Contagem inicial (opcional), do MVP ou de cada *Sprint*;
  - Contagem final, do MVP ou de cada *Sprint*.
- Atividades que viabilizem e matenham ativo o processo de melhoria contínua.

A "Execução" pode ser "quebrada" em MVPs (na perspectiva do negócio) e também em *Sprints* (na perspectiva da TI). Isso normalmente contribui para a visibilidade do fluxo de produção, que pode ser controlado de forma a interconectar diferentes times, viabilizando uma melhor gestão das mudanças.

Complementarmente, indicamos um *time-box* uniforme das *Sprints* (como já referimos antes), considerando o período de 2 semanas para a realidade da maioria das organizações. Assim, na pior das hipóteses, ao menos quinzenalmente potencialmente haveria um novo pacote de mudanças disponíveis ao usuário. Não necessariamente este conjunto de valor será completo (como o previsto para o final de cada MVP), mas atenderá parcialmente os usuários ou viabilizará validação.

Um ponto de atenção é o replanejamento dos Blocos de Trabalho ou Épicos, que interconectados, podem ser complexos de gerenciar. Em contrapartida, o caminho crítico e a totalidade de atividades programadas mostram-se objetos mais claros, eliminando

ou mitigando a formação de estoques em um ambiente de múltiplos times (através do mapeamento das interdependências).

7.5.5 Transição

A motivação para definirmos um conjunto de atividades de "Transição" é a necessidade de disponibilização em produção de um código proveniente de terceiros, ou seja, contemplando então, as atividades de aceite do usuário (homologação) e liberação em produção (implantação) ao escalar uma abordagem de *outsourcing* ou quando da necessidade de instalação de pacotes/componentes terceiros.

A seguir, veja a relação de macro-atividades nesta etapa:

- Atividades relacionadas à homologação, como testes de aceitação da entrega;
- Gestão de mudanças e liberação, para implantação do *software* em produção;
- Passagem do conhecimento adquirido ao time responsável pelo produto mantido ou impactado.

No caso de aplicações complexas ou de domínio crítico, a manutenção de conhecimento provavelmente irá envolver algum tipo de registro dos recursos da aplicação, que explicitam em detalhes do que são constituídas as funcionalidades implementadas. Uma dica válida para tratar esta situação, é eliminar o fundamentalismo e avaliar o que gera valor ao negócio, ao produto e a quem dele depende. Procure olhar para o ecossistema como um todo com um viés de eficiência e não apenas para um elemento em particular do mesmo.

Lembre-se ainda: projeto ágil não significa que o produto não deve possuir documentação, assim como isso em excesso também não garante a manutenção do conhecimento. Então procure manter um equilíbrio, certificando-se da existência de um "padrão mínimo", de forma que mesmo alguém alheio ao time possa compreender o

que foi feito e o que foi impactado, de um modo suficientemente claro e completo.

7.5.5.1 Escalada reversa

Ao retornar uma demanda ou MVP desenvolvido na modalidade de fábrica, é importante realizar novamente seu retorno, formalizando o recebimento e aceite do time e do negócio.

A "Escalada Reversa", como é chamado este retorno, é realizada aos moldes da que é executada no Planejamento, porém agora durante a etapa de "Transição". Nela, são produzidos os vídeos *demo* para o *showcase*, considerando aquilo que foi entregue em termos de *features* e/ou *user stories* e explicitando os novos recursos implementados, sempre com foco no conteúdo relevante aos envolvidos.

Independente desta abordagem ter sido criada para escalar o modelo de desenvolvimento, é considerado uma boa prática armazenar os artefatos gerados para fins históricos, mantendo assim o conhecimento adquirido de forma simplificada e que pode ser muito efetiva se bem organizada. Contudo, assuma a seguinte frase como premissa: "menos desperdícios, mais benefícios" e interprete-a para o seu contexto.

7.6 Estimativa e Medição: o "metro" do *software*

Nesta obra, nós trabalhamos com duas estratégias de estimativa e medição de tamanho de *software*, com funcionamento e propósito similar, as quais denominamos "plugins": a "Abordagem IFPUG" e a "Abordagem COSMIC".

Como vimos nos capítulos iniciais, ambas as abordagens quantificam tamanho de *software* em unidades referidas como "pontos de função", o que possibilita análises avançadas sobre dados correlacionados que estão vinculados ao desenvolvimento de *software*.

O uso de métricas funcionais apresenta uma série de vantagens, tais como:

- Objetividade e precisão;
- Baixo investimento de implantação e operacionalização;
- Configuram padrões ISO, totalmente repetíveis e auditáveis (ou seja: duas pessoas que aplicam os métodos chegam aos mesmos resultados);
- São orientadas pela visão do usuário, relacionando-se muito bem com o conceito de agregação de valor (ao produto);
- Fornecem ferramentas para o acompanhamento da eficiência dos processos em torno do desenvolvimento, ao longo do tempo (não é julgamento de valor para o negócio);
- Servem de entrada para a melhoria contínua e insumo para uma tomada de decisão mais acertiva;
- Imprimem maior transparência à operação, viabilizando *benchmarking* (como estamos?).

Independentemente do *plugin* escolhido, esta proposta pode ser aplicada tanto para a finalidade de planejamento como de acompanhamento dos times (ou ambos) e dos seus processos, aprimorando a visão de uma TI transparente e conhecedora de suas demandas e de um negócio conciente de limitações e que acompanha a evolução da eficiência "do todo" (da TI e do que está situado em torno dela), sentindo com isso, maior segurança.

Considerando estes aspectos, pode-se afirmar que os seguintes propósitos são cobertos em "Enxugando a Máquina":

- Fornecer apoio às estimativas iniciais;
- Acompanhar o andamento do trabalho (a fazer, em desenvolvimento e entregue);
- Acompanhar a performance dos times, seus processos e suas entregas;
- Fornecer insumos para a tomada de decisão;
- Fornecer insumos e garantir a melhoria contínua;
- Fornecer dados passíveis de *benchmarking* externo.

Sendo assim, com base nos propósitos elencados, são compatíveis os seguintes pontos de estimativa e medição:

- Estimativa inicial do *Target* - fornece estimativas iniciais para análise de viabilidade ou dotação orçamentária de um Projeto ou Melhoria, avaliando-o para formação do Portfólio;
- Estimativa inicial do Épico ou Bloco de Trabalho - apóia o planejamento de um time e/ou de um subportfólio, considerando o Épico ou simplesmente o trabalho a ser executado num período finito de tempo;
- Estimativa e medição do MVP (foco no planejamento dos Épicos e/ou acommpanhamento da operação por MVP - recomendado no cenário de *outsourcing*)
  - Estimativa do MVP - possibilita uma avaliação do trabalho que está previsto para ser entregue ao final do MVP, considerando as tarefas e o esforço a ser despentido (vinculado ao *sub-backlog* de "Evolução do Produto");
  - Medição do MVP - viabiliza o acompanhamento e a calibração dos parâmetros utilizados no sistema de estimativas considerando apenas os itens de "Evolução do Produto" (os ítens do *sub-backlog* de "Sustentação do Produto" são monitorados através de um percentual médio de alocação);
- Estimativa e Medição da *Sprint* (foco no planejamento dos times e/ou acompanhamento "*online*" da operação - ideal para cenários de *insourcing* ou com times mistos)
  - Estimativa da *Sprint* - viabiliza estimativas do time com maior acurácia (inclusive com possibilidade de acompanhamento diário) e inclui o acompanhamento dos itens de "Evolução do Produto" e de "Sustentação do Produto";
  - Medição da *Sprint* - possibilita confirmar e/ou avaliar a produtividade, taxa de entrega e a velocidade média do time, com capacidade máxima de acompanhamento

do trabalho, considerando os itens de "Evolução do Produto" e "Sustentação do Produto".

7.6.1 Plugin COSMIC

O "*Plugin* COSMIC" de estimativa e medição é indicado principalmente nos cenários onde há aplicações com alto nível de interação com o usuário (seja uma pessoa, *software* ou *hardware*) e/ou com muitas fontes ou destinos de dados. Também é indicada esta opção, quando a utilização do método do IFPUG já não responde adequadamente ao propósito por conta de suas limitações implícitas à definição de complexidade ("Baixa", "Média" e "Alta").

Além disso, o COSMIC é considerado bastante completo e moderno, aderindo perfeitamente aos objetivos e recursos de "Enxugando a Máquina". Certamente mostra-se como excelente ferramenta para medirmos o *software* desenvolvido em projetos ágeis, com fácil aplicação mesmo num cenário onde a especificação é provida em termos de *Users Stories* (US), já que não exige o conhecimento do estimador a respeito da funcionalidade inteira, como nos demais métodos funcionais.

Também bastante abrangente e detentor de uma completa Base de Conhecimento, o COSMIC trabalha muito bem os conceitos de camadas e de serviços, sem com isso perder os aspectos funcionais que o caracterizam como um método aderente ao padrão ISO 14.143-1.

O método ainda é pouco difundido no Brasil, muito por conta do sucesso que o seu concorrente do IFPUG tem alcançado nos últimos anos, porém tem despertado o crescente interesse de organizações públicas e privadas.

7.6.1.1 Estimativa inicial do *Target*

Considerando que o orçamento da organização é projetizado, durante a execução do processo de Portfólio de Projetos (que também pode incluir Melhorias), provavelmente será necessário realizar estimativas iniciais para avaliação dos investimentos ne-

cessários ao desenvolvimento de um *Target* e/ou para provisionar os custos associados. Neste momento, a disponibilidade e o *capacity* das equipes também costumam ser analisados.

Ocorre que, independente do método utilizado para aferir o tamanho dos *Targets*, sem uma base histórica formada, provavelmente você não possuirá boas referências para suas estimativas, então a primeira dica é: comece a montar uma, o mais cedo possível! E utilizando COSMIC com esta finalidade, é possível contar com uma extensão bastante completa e útil, chamada "*Guideline for Early or Rápido COSMIC Functional Size Measurement*" (COSMIC, 2015). O documento está disponível no site da organização COSMIC e descreve opções de escalonamento para a realização de estimativas iniciais, inclusive considerando os diferentes níveis de granularidade dos requisitos. Verifique esta fonte e escolha a abordagem que mais se encaixa à sua realidade!

Nas estimativas iniciais, o COSMIC trabalha com classificações de tamanho, que são resultantes de análises realizadas sobre um contexto de interesse. Elas fornecem parâmetros com base em tamanho, o que é muito útil para estimarmos *Targets* e Épicos! Sendo assim, a partir do tamanho podemos derivar outras estimativas, mas claro, sempre preferindo utilizar referênciais calibradas próprias.

Basicamente, os insumos para uma "Estimativa inicial na Iniciação" utilizando o *Plugin* COSMIC, podem compreender:

- Uma descrição do *Target* (como por exemplo, uma US que o defina);
- Uma relação dos Épicos ou Blocos previstos (se Épico, 1 para cada time de desenvolvimento e preferencialmente descrito na forma de US em alto nível);
- Entrevistas realizadas com os *stakeholders*, buscando classificar os Épicos ou Blocos vislumbrados (por exemplo, através de classificações P, M, G e GG);
- Os seguintes dados, como critérios de apoio à definição das classificações, conforme dados hitóricos calibrados:
  - Tamanho médio, em pontos de função;

  - Esforço e custo médio do ponto de função, por time de desenvolvimento;
  - Quantidade média de *sprins*, assim como a velocidade média (em pontos por *sprint*) ou ainda, a duração média.

Com nos parâmetros de entrada citados, a estimativa é então realizada, da seguinte forma:

- Durante o Portfólio de Projetos, o PMO, os POs e demais *stakeholders* tentam chegar a um consenso sobre uma estimativa para a execução do desenvolvimento ou evolução do produto de *software*, utilizando como referência as classificações previamente calibradas, euquadrando os Épicos, de forma aproximada conforme os critérios estabelecidos (conforme indicações obtidas pela avaliação histórica);
- Ao selecionar uma classificação, todos os parâmetros devem ser avaliados e se necessário revisados, sendo que a estratégia mais adequada é a de selecionar o cenário que mais se aplica dentre as classificações disponíveis;
- Ao final da classificação, todos os Épicos devem ser agrupados e o grupo discute a estimativa global para cada *Target*, avaliando sua ordem de grandeza;

Alguns resultados esperados para este tipo de estimativa são os seguintes:

- Tamanho, em pontos de função;
- Esforço, em horas;
- Duração, em quantidade de *Sprints*;
- Duração, em meses;
- Custo total.

Caso não disponha de tempo ou informações suficientes para construir sua base histórica, avalie uma ordem de grandeza proporcional à duração, ou seja, o tempo em que você imagina conseguir

finalizar cada Épico ou Bloco (assuma premissas). Assim, de acordo com o *time-box* definido do time e um percentual médio de alocação nos *Targets*, você pode avaliar variáveis correlacionadas, como o custo ou o tamanho. São exemplos de cálculos entre variáveis correlacionadas e que podem ajudá-lo neste momento:

- Se em 1 mês executamos duas *Sprints*, então um Épico com duração esperada de 5 a 7 meses, terá em média 12 *Sprints*;
- Se 1 *Sprint* tem um custo médio de "x" reais, logo 12 *Sprints* custarão aproximadamente 12x (porção relacionada à etapa de "Execução" do ciclo de desenvolvimento).

Outra alternativa, é considerar a velocidade média em pontos de função produzidos por *sprint* frente à expectativa de uma quantidade especifica de *sprints*.

Ao derivar dados históricos no nível de *Sprints*, atente ainda, aos cenários onde há Épicos concorrentes ou com vários *Targets*, ou ainda, como já referido, eventuais estimativas para reserva de alocação e/ou custo de atividades de "Sustentação do Produto". Outra questão importante, é que há atividades *cross*, ou seja, que não são exercidas por um time de desenvolvimento, assim como também existem atividades que não podem ser mensuradas através de pontos de função. Você deverá considerar isso e outras questões, talvez tratando um percentual aproximado. O mesmo se aplica à evolução natural dos requisitos ou mudanças de direcionamento, que podem se consideradas como "margem de erro" ao elaborar uma estimativa mais realista e completa (observe que isso também pode ser avaliado enquanto fator histórico e calibrado conforme a necessidade).

7.6.1.2 Estimativa inicial do Épico ou Bloco (Concepção ou Planejamento)

Ao realizar uma contagem de pontos de função, primeiramente reflita sobre o seu propósito. Deste modo, você provavelmente se deparará com uma ou mais das alternativas elencadas abaixo:

- Estimar o tamanho do Épico ou Bloco, como uma unidade ou segregado por MVP;
- Com base no tamanho estimado do Épico e do Bloco e num índice de produtividade histórico do time, derivar as estimativas de esforço;
- Com base no tamanho estimado do Épico/Bloco ou num esforço estimado, elaborar estimativas de custo de desenvolvimento, visando planejar e/ou contratar atividades de desenvolvimento do *software*;
- Com base no tamanho estimado do Épico, na velocidade média do time e/ou no percentual médio de alocação, elaborar uma estimativa de duração (em quantidade de *Sprints* e/ou meses).

Idealmente, um profissional especialista em métricas ou membro do time com conhecimento no método (no mínimo intermediário), é convidado para a Concepção, participando desde a dinâmica "Jornada do Usuário" (pois antes, talvez configure desperdício) até o final do *workshop*. Ele contribuirá na descoberta do escopo e com sua experiência obtida através das contagens anteriores do próprio time ou produto, ou mesmo de outros times. Elabora-se assim, uma estimativa mais realista e cujo racional e parâmetros são previamente definidos.

Todo o trabalho da Concepção é insumo para as estimativas e até mesmo uma dinâmica específica poderia ser utilizada para obter insumos complementares. No entanto, caso isso não seja possível ou a participação do especialista em métricas ou outro profissional conhecedor da disciplina seja inviável durante a Concepção, este tipo de estimativa poderá ser realizada também na etapa de Planejamento, considerando o apoio de uma pessoa de referência do time, idealmente o Scrum Master (estratégia que pode ser adaptada para cada contexto e necessidade).

No início, as estimativas iniciais tendem a ser pouco precisas pois os parâmetros e insumos podem não ser adequados, porém na medida que são coletados dados com melhor qualidade e em

maior volume, o modelo vai evoluindo e mostrando-se cada vez mais objetivo e matemático.

Nós podemos considerar como insumos de uma "Estimativa inicial do Épico (ou Bloco)", os seguintes elementos:

- Quaisquer dados formais ou informais gerados, assim como os artefatos derivados da Concepção e/ou do Planejamento;
- Dados calibrados no contexto de um time, tais como:
  - Custo unitário médio do ponto de função;
  - Índice de produtividade (médio), em horas por ponto de função;
  - Taxa de entrega, em quantidade de pontos produzidos por pessoa/mês ou pessoa/*sprint*;
  - Velocidade média, em termos de pontos de função produzidos por *sprint*;
  - Fator de retrabalho e/ou de evolução de requisitos, ou seja, considerando o fator de escala entre os requisitos iniciais e finais.
- Entrevistas com os participantes da Concepção e/ou Planejamento (ou dinâmica).

Ao realizar uma contagem inicial, considere que o especialista em métricas irá realizá-la já considerando eventuais convenções locais, registrando também suas premissas e observações, para fins históricos.

Assim, ao finalizar a contagem, espera-se que estejam disponíveis as seguintes informações:

- Tamanho total e por MVP vislumbrado, em pontos de função;
- Esforço em horas, conforme o tamanho identificado e a produtividade média do time;
- Quantidade de *Sprints*, com base no tamanho, na velocidade média do time e seu percentual de alocação médio para itens de "Evolução do Produto";

- Duração esperada, em meses, com base no *time-box* e velocidade média do time, assim como na sua estimativa de alocação média para itens de "Evolução do Produto" no período (uma aferição), ou mesmo, simplesmente com base na quantidade de *Sprints*;
- Custo total e unitário por ponto de função.

Caso você atualmente utilize APF do IFPUG na sua operação e tenha interesse em migrar para o COSMIC, de modo a continuar ainda utilizando suas referências atuais, é possível e viável, utilizar o método de Estimativa NESMA paralelamente e complementarmente, convertendo seus dados históricos também para a NESMA e comparando-os ao menos durante um período de transição. Isso pode ser utilizado como insumo para realizar uma prova do *business case*, que pode auxiliá-lo a mostrar o quão melhor tem sido o seu desempenho como uma organização ágil e enxuta em termos de eficiência. Observe que utilizando o COSMIC e a Estimativa NESMA em conjunto, é possível fazer esta transição de forma segura, mantendo-se ainda todas as suas fontes de referência atuais.

7.6.1.3 Estimativa do MVP

Ao planejar um MVP rumo a sua conclusão, pode ser necessário obter projeções atualizadas, seja para averiguar eventual necessidade de orçamento extra ou para fornecer insumos à execução do trabalho da TI. Nessas situações, sugere-se utilizar uma abordagem de contagem incremental, onde o histórico é mantido a cada iteração ou MVP concluído.

A linha de base considerará movimentos de dados revisitados entre as iterações, mesmo que vinculados a uma *User Story* (US) já considerada pronta anteriormente, pois este é um tipo de retrabalho que ocorre normalmente devido à evolução no entendimento de/do negócio.

Poderia ocorrer também uma incidência de retrabalho dentro de uma *Sprint*, quando de um mesmo movimento impactado por uma ou mais US, porém isso não deve ser considerado para efeito de

cálculo da produtividade ou outras medidas de base, pois a mexida no código poderia em tese ter sido realizada por uma mesma pessoa e talvez inclusive num mesmo intervalo de tempo - ou tratada por meio de componentização do produto, o que não é visível aos olhos do usuário e portanto, estaria fora do alcance de uma avaliação funcional pela métrica de tamanho utilizada, pois não é seu objetivo.

Considerando essa estratégia, três medidas adicionais poderiam ser derivadas (em relação às estratégias anteriores):

- Tamanho efetivo (ou simplesmente "Tamanho"), que representa o quantitativo de pontos de função realmente agregado ao produto;
- Percentual de ajustes, ou seja, o percentual obtido pela quantidade de pontos de função gerados por ajustes realizados num mesmo movimento de dados já contado dentro da *Sprint* em relação ao total de pontos contados (isso para acompanhar e mitigar o retrabalho, fornecendo ainda uma variável a considerar no parâmetro de escala para desvio de requisitos iniciais *versus* finais);
- Retrabalho, ou seja, o percentual ou quantidade de movimentos de dados ou processos funcionais novamente mexidos em razão de mudanças associadas à movimentos já antes implementados e entregues numa iteração ou MVP anterior (convencionar a situação aplicável atentando ao propósito vinculado).

Realizada normalmente com base em *Features*, *Users Stories* e/ou entrevistas para identificá-las, este tipo de estimativa possibilita obtermos dados a respeito do tamanho e trabalho previstos para execução até o final do MVP recentemente iniciado, isso de forma consolidada. Considera ainda: a duração aproximada, a alocação de recursos e os custos associados.

7.6.1.3.1 Estimativas com COSMIC *Planning Poker*

A técnica *Planning Poker*, segundo PHAM (2011), nem sempre funciona bem além das fronteiras dos Estados Unidos, como por

exemplo, nos países asiáticos. O autor defende que nestas culturas, o respeito às pessoas mais velhas e aos papéis em posição de liderança geralmente inibe os membros da equipe no momento de fazer uma estimativa diferente delas.

Embora enalteça os benefícios até aqui conquistados pelo método, PHAN (2011) sugere que procuremos uma abordagem mais objetiva, especialmente tendo em vista a necessidade de implantar e alocar recursos em escala corporativa para a metodologia ágil ou *Scrum*. Considerando suas convicções, propõe um método de estimativas que ele mesmo teria utilizado com sucesso durante muitos anos, em diversos projetos reais.

O método de PHAN (2011) utiliza os seguintes elementos:

- Usuários de Negócio (que interajam com o sistema);
- Regras de Negócio;
- Entidades de Negócio.

Os elementos são avaliados em termos do tipo de interação, regras de negócio, número de entidades manipuladas e dados lidos, criados, atualizados e excluídos. E ao final da avaliação, a estes movimentos atribui-se uma pontuação, cuja escala é pré-definida.

Na mesma linha, perceba que poderíamos utilizar um método mais objetivo, o método COSMIC, mas se preferirmos, ainda mantendo a visão lúdica do jogo *Planning Poker*. Cabe ressaltar que o COSMIC considera aspectos similares à proposta de PHAN, porém já é reconhecido como um padrão ISO/IEC (ISO/IEC 19.761), atendendo ao mesmo propósito: estabelecer um racional seguro e independente do "achismo" para as estimativas.

Ao utilizar COSMIC, é importante ressaltar que mediremos obtamanho ao invés do esforço, diferentemente de *Story Points*, que trata uma ordem de grandeza de esforço. Sendo assim, para derivar esforço você precisará considerar uma produtividade média histórica ou terá que valer-se de outras variáveis para este tipo de avaliação. Como a medida de tamanho possui um racional bem definido, as estimativas podem até mesmo serem confirmadas após

o desenvolvimento, através de uma medição da funcionalidade entregue.

Embora existam outras alternativas previstas pelo COSMIC (2015), o time poderia incorporar a técnica "COSMIC *Planning Poker*" (ou "CPP"), criada por Ismael Melo com base no "*Planning Poker*". Neste método, primeiramente os sistemas do escopo e seus objetos de interesse do usuário são identificados (do mundo real ou conceitual) - note que apenas nas primeiras estimativas haverá algum esforço para isso, podendo um especialista em métricas previamente identificar objetos, fronteiras e usuários na forma de uma *baseline* do produto. Em seguida, são também identificados os processos funcionais e relacionado a cada um deles, também a quantidade aproximada de movimentos de dados que precisarão ser desenvolvidos ou customizados para atender os requisitos funcionais do usuário, ou seja, as necessidades de movimentação de grupos de dados derivados dos objetos de interesse. Como essa aproximação será realizada através do método de *Planning Poker*, não se torna invasivo em relação aos métodos comumente praticados por equipes ágeis, podendo inclusive substituí-los.

Assume-se que no CPP nem sempre serão evidentes os processos impactados conforme prevê o método COSMIC no nível de processo funcional e que por isso, poderão eventualmente ser considerados outros níveis de abstração, tais como *features* ou *user stories*. O especialista em métricas da organização deverá apoiar o time, talvez indicando o nível de granularidade ideal para as estimativas. Essa é uma particularidade do COSMIC, visto que a identificação dos processos funcionais não é tão significativa em relação a quantidade de pontos identificados, em comparação ao método do IFPUG - ou seja, no COSMIC, a quantidade de movimentos de dados tem maior influência sobre o tamanho do que na APF do IFPUG, e eventuais diferenças significativas nos requisitos podem ser tratadas através da definição de Níveis de Granularidade acima do nível de Processo Funcional (padrão do método).

Ao utilizar *Story Points* o time atribui pontos como unidades

de esforço, no COSMIC, no entanto, temos unidades de tamanho de *software*, o que é uma medida objetiva e comparável ao longo do tempo. Utilizado em um momento de incerteza, ao invés de uma percepção ou expectativa sobre o esforço (o que convenhamos, pode ser desconfortável), avaliamos tamanho, isso com base no total de movimentos de dados contidos nos processos funcionais, *user stories* ou *features* do escopo e conforme a necessidade e implementação prevista.

Esta análise de tamanho (por aproximação) contribui para a geração de estimativas derivadas do esforço, tais como: prazo, custo, recursos e qualidade. Outra questão relevante, é que até mesmo os gráficos utilizados para acompanhamento do time (e para o time) poderão ser adequados para utilizar a unidade de tamanho. Sendo assim, ao invés de considerar *Story Points*, basta utilizarmos "Pontos de Função COSMIC". Considerando estas questões, provavelmente a absorção pelo time será mais orgânica.

Abaixo, confira um passo-a-passo para aplicação do CPP:

Figura 32 – COSMIC Planning Poker

Note que esta abordagem é uma opção para otimizar recursos e eliminar desperdícios, utilizando a própria equipe para a geração das suas estimativas, derivando dados que podem ser utilizados na esfera corporativa. No entanto, na impede que um especialista em métricas assuma a responsabilidade pelas estimativas com base em insumos pré-definidos ou conduza a equipe numa atividade de facilitação usando CPP - opções que obviamente tenderiam a resultados mais adequados.

7.6.1.3.2 COSMIC Canvas

O "COSMIC Canvas" é uma dinâmica de descoberta de escopo que serve também para derivarmos estimativas, de modo um muito simples e útil. esta técnica foi criada pelo Ismael Melo com o auxílio de Antônio Lacerda; e pode ser utilizada no início do planejamento de um MVP ou no início de uma *Sprint* (durante ou imediatamente após uma cerimônia de *Planning*).

Figura 33 – COSMIC Canvas

Segue abaixo, um passo-a-passo para a aplicação da dinâmica:

- 1º: Relembrar o contexto do escopo a todos os participantes.
- 2º: Identificar as personas (humanos) e demais usuários (sistemas ou coisas).
- 3º: Identificar objetos de interesse tratados (relação que vai sendo atualizada durante a dinâmica) - um objeto de interesse é um objeto do mundo conceitual ou físico do usuário.
- 4º: Identificar todos os sistemas impactados.
- 5º: Identificar as movimentações de dados de acordo com seu tipo, sejam "Entradas" (E), "Saídas" (X), "Gravaçôes" (W) ou "Leituras" (R), ou mesmo as "Entradas" (E) e "Saídas" (X) oriundas de interações via serviços, para cada US ou *Feature* e conforme os grupos de dados derivados dos objetos de interesse mapeados.
  - Deve-se incluir um *card/ post-it* para cada tipo de movimento (do lado esquerdo do Canvas) no local correspondente do quadro, conforme seu tipo e fazendo

constar a identificação da US e respectivos grupos de dados.

- 6º: Após finalizado o mapeamento do escopo, o facilitador da dinâmica deverá somar o total de itens de cada *card*, anotando no mesmo tal referência e transcrevendo ao lado a notação do tipo específico de movimento de dado.
- 7º: Após o passo 6 ser executado sobre todos os *cards* do canvas, os mesmos deverão ser movidos para o quadro de Sistemas (os quais representam as fronteiras) - *cards* de uma mesma US ou *Feature* deverão ser agrupados num *card* de tamanho maior, que receberá o nome da referida US/*Feature*.
- 8º: Não existindo mais movimentos de dados no lado esquerdo do Canvas, os "*cards* menores" de um mesmo "*card* maior" deverão ser somados, e isso representará a estimativa da US/ *Feature*, ou seja, o seu total de pontos de função COSMIC.
- 9º: Como último passo, o facilitador poderá escolher uma ou mais das opções elencadas abaixo:
  - Somar todos os *cards* relacionados a um sistema;
  - Somar todos os *cards* do escopo (do Canvas);
  - Calcular o produto dos quantitativos apurados e a produtividade de referência definida para o time ou para cada sistema (nos casos onde exista uma referência de produtividade diferente por sistema).

7.6.1.3.3 Diagramação COSMIC

Ao utilizar o COSMIC Canvas em conjunto com um *template* padrão de US, por exemplo, pode em conjunto, ser utilizada a seguinte notação, criada por Ismael Melo e derivada do Diagrama de Mensagem da UML:

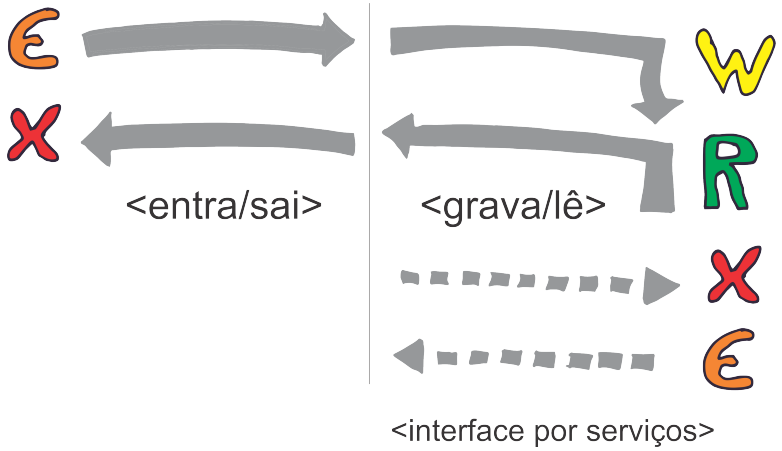


Figura 34 – Diagramação com COSMIC

Veja um exemplo de especificação de um processo funcional, representando todas as interfaces (movimentos de dados) do processo "Incluir Pedido":

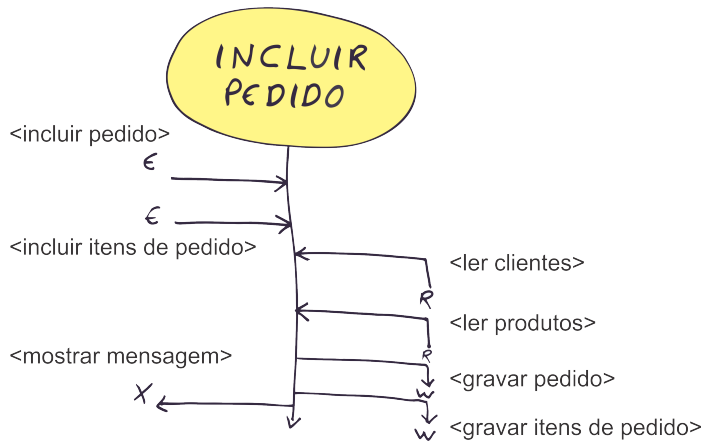


Figura 35 – Exemplo de Diagramação

7.6.1.4 Medição do MVP

Ao invés de realizar medições ao final de cada iteração, por conveniência ou devido à otimização dos custos com estimativas, pode ser executada uma medição final do MVP. Este cenário é aplicável quando não há necessidade de acompanhamento em "tempo real", seja da operação, dos processos ou do Portfólio.

Uma medição ao final do MVP também pode ser muito útil em um regime de contratação por fábrica de *software* (*outsourcing*), com execução dedicada do ciclo de desenvolvimento ágil ocorrendo num ambiente externo à organização. O fluxo de contratação tende a ser rápido e dinâmico (vide tópico neste capítulo), com preço do ponto de função ou preço fixo global pré-acordado, ou ainda, conforme uma quantidade de horas, recursos e valor horário, considerando também uma determinada quantidade de *Sprints* estimadas e acordadas para cada MVP vinculado - idealmente, prevendo um mecanismo contratual que estabeleça uma contagem inicial e outra final para cada MVP.

No nosso modelo, um MVP é um incremento que pertence a um Épico; e sua contagem pode ser inclusive uma atualização da referência do Épico, a qual considerando também uma medida do retrabalho executado (no nível de MVP). Sendo assim, para o caso de um MVP ser contratado com um fornecedor executando apenas parte das atividades do escopo, tome o cuidado apenas de separar a porção que lhe cabe da contagem.

Aos que preferem manter uma referência incremental do Épico, provavelmente armazenada em uma planilha eletrônica que é constantemente mantida, haverá algumas vantagens, como:

- O conhecimento sobre o tamanho é documentado em um único local, mantendo-se o histórico de mudanças;
- A contagem do segundo MVP em diante, torna-se mais fácil, pelo histórico agregado e pelo contexto implícito das contagens anteriores;
- O planejamento do Épico/Bloco e do Projeto mostra-se mais transparente;

- A contagem de mudanças de escopo ocorridas de um MVP para outro se tornam mais fáceis de serem gerenciadas, tanto pelo reaproveitamento dos itens da contagem, como por não ser necessário buscar informações em contagens passadas;
- Ao acessar um diretório versionado, as informações sobre um Épico, assim como dos seus MVPs e iterações relacionadas, podem ser armazenadas e recuperadas a qualquer tempo, integralmente e de forma consolidada;
- Diversos indicadores são passíveis de serem derivados para promover a gestão de entregas e do *Target*, e eles podem ser consolidados em uma única pasta/planilha.

No entanto, há também algumas desvantagens:

- É mais difícil isolar o contexto de um MVP específico, assim como dos indicadores relacionados quando eles são mantidos juntos;
- Se não houver Concepção e Planejamento (por algum motivo), provalvemente não haverá uma perspectiva de Épico (ou de um bloco de trabalho), e logo, não existiriam MVPs relacionados;
- Se houver desenvolvimento paralelo de MVPs, será muito complexo validar hipóteses e análises de sobre os seus dados.

Como insumos da medição, utilize os artefatos relacionados às *Features* e *US* entregues pelo MVP, em sua versão final. Além disso, entrevistas são muito úteis para complementar os requisitos, ou mesmo, vídeos que documentem a passagem de conhecimento ou um *showcase* apresentado ao negócio.

Como resultados da medição e das estimativas derivadas, é possível obter os seguintes indicadores (a relação de saídas pode variar conforme o propósito):

- Tamanho, em pontos de função, do MVP, das *Sprints* e dos processos funcionais;

- Esforço total realizado;
- Esforço de profissionais internos à organização (do time);
- Esforço de fornecedores de *software*;
- Esforço de fornecedores de qualidade;
- Percentual de alocação média (em média, quanto tempo das *sprints* foi consumido para tratar o MVP);
- Índice de produtividade (a razão entre o esforço e o tamanho apurado);
- Velocidade média, em pontos de função produzidos por *sprint*;
- Duração, em quantidade de *Sprints* do MVP;
- Duração, em meses;
- Custo total do MVP e médio (do ponto de função);
- Custo médio do ponto de função no fornecimento de serviços por fornecedor (custo de fornecimento);
- Qualidade, em inconformidades por pontos de função ou a cada 1000 pontos de função (defina sua abordagem).

As citadas "saídas" devem ser armazenadas numa base histórica, para que possam ser calibradas e utilizadas com intuito de avaliarmos a performance de processos em torno do desenvolvimento. Assim, garantimos melhor poder de predição do modelo durante as estimativas iniciais.

7.6.1.5 Estimativa da Iteração

No início de cada iteração, utilizando COSMIC, o time tem algumas opções para a realização das estimativas:

- Utilizar o *paper* da organização COSMIC para estimativas ou ainda incluindo um guia de convenções locais;
- Utilizar o COSMIC *Planning Poker*;
- Utilizar o COSMIC Canvas.

Incorporando este padrão internacional, o time contará com um método objetivo e verá que suas as estimativas poderão ser

confirmadas ao longo do tempo, ao final de cada iteração. E adicionalmente, inclusive um eventual desvio médio identificado poderia ser incorporado às suas estimativas futuras, como uma composição de escala ou margem de erro.

Considere utilizar o COSMIC Canvas, no início da *Sprint*, já que com ele, ao mesmo tempo que se obtém o tamanho estimado por aproximação, é possível realizar uma análise de impacto bastante abrangente e detalhada no nível necessário sobre o escopo de atuação. É uma cerimônia rápida e sem desperdícios, que agrega valor ao trabalho do times.

Em geral, as estimativas executadas no início de cada iteração servem de mecanismo para prover visibilidade a respeito das entregas, inclusive podendo contemplar atualizações diárias dos itens num sistema de informação que possibilite o acompanhamento do Portfólio e dos subpotfólios com o viés de monitoramento da operação.

7.6.1.6 Medição da Iteração

Uma medição da iteração contempla a contagem do *software* produzido pelo time numa *Sprint* específica que está em análise. Idealmente, este tipo de análise é realizada por um especialista em métricas (um profissional devidamente qualificado como conhecedor do método), para garantir sua corretude.

O profissional executará a medição conforme os insumos disponíveis e acessará a equipe de desenvolvimento caso necessite esclarecer dúvidas a respeito das tarefas e serviços fornecidos à aplicação na *Sprint*.

Devem ser considerados como contáveis numa contagem final da iteração, os seguintes elementos:

- *Users Stories* finalizadas pelo time na *Sprint* (elas são ligadas às *Features* e, por sua vez, aos MVPs);
- Melhorias, que agregam valor ao produto, mas não estão relacionadas a projetos (são itens pequenos demais para compor

um projeto ou compreendem ações vinculadas à Sustentação do produto);
- Problemas e Incidentes, os quais representam o passivo da organização e/ou uma dívida técnica gerada pelo time.

Cada elemento contempla uma representação ou descrição do *DELTA* (a diferença), como expresso na ilustração abaixo:

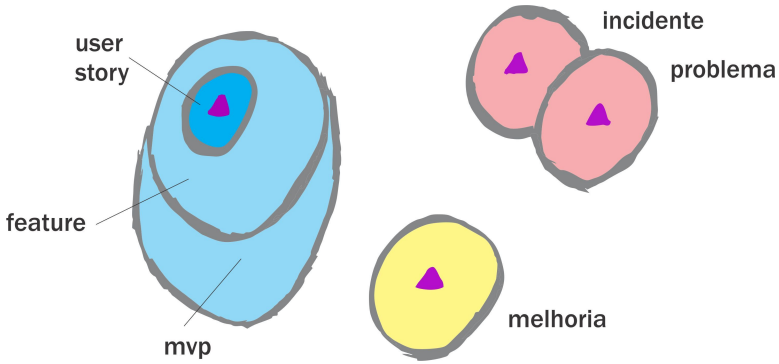


Figura 36 – Itens mensuráveis na iteração

Sendo assim, note que 100% do trabalho do time que é relacionado ao desenvolvimento de *software* será medido e contabilizado na produtividade, velocidade e taxa de entrega, sensibilizando também demais medidas de desempenho.

Neste contexto, podemos considerar também algumas derivações, como por exemplo, relacionadas à métrica de produtividade, tais como as seguintes:

- "Produtividade NOMINAL", utilizada para medir a eficiência do time em relação ao trabalho executado (100% do trabalho realizado);
- "Produtividade REAL", utilizada para medir a eficiência do time considerando uma estimativa de horas efetivamente trabalhadas;
- "Produtividade VALOR", utilizada para medir a eficiência do time considerando apenas os itens que agregaram valor à

organização de forma direta, ou seja, excluindo-se todos os Problemas e Incidentes, pois eles representam dívida técnica.

Observa-se que após uma contagem COSMIC, é considerado uma boa prática manter a *baseline* ou o mapa de processos já identificados, por sistema/fronteira e contemplando usuários, objetos de interesse, grupos de dados e movimentos de dados relacionados. Isso poderá ser útil no futuro, ao realizar outras contagens.

7.6.2 Plugin IFPUG

Para realizar uma comparação mais precisa das suas estimativas em relação ao histórico já existente na organização e/ou no mercado, e também caso você prefira não considerar o uso do COSMIC em conjunto com uma Estimativa NESMA, ou ainda, se por ventura entender que a característica de suas aplicações aderem melhor à proposta da APF IFPUG (*software* de negócio intensivo em dados), utilize o método do IFPUG adaptando-o para os seus diferentes propósitos. É uma abordagem muito utilizada no mercado, e que aplicada corretamente, certamente atende muito bem a expectativa.

Isso significa que podemos utilizar a "Análise de Pontos de Função" (APF) do IFPUG para avaliação do tamanho funcional, assim como opcionalmente o "Processo de Avaliação Não Funcional" (SNAP) para avaliação do tamanho não funcional; ou mesmo, podemos considerar orientações e diretrizes por analogia, visando abranger também aspectos de avaliação não funcional. Um exemplo de derivação por analogia é o Roteiro de Métricas do SISP (2015), um guia amplamente utilizado no setor público brasileiro.

Atente também que embora o SNAP utilize uma abordagem conceitual teoricamente mais sólida, as analogias derivadas da APF para medição de tamanho mostram-se mais adequadas para a maioria dos propósitos, pois num dado momento pode se tornar muito dificil trabalhar com duas unidades de medida cujo comportamento tende a diferir bastante (não há uma correlação ou modo de conversão entre APF e SNAP, por exemplo).

7.6.2.1 Deflatores de produtividade como fatores de impacto

Utilizando a APF do IFPUG, para melhor representar as variações de produtividade e/ ou preço em decorrência do tipo de desenvolvimento envolvido, convém e é comum utilizarmos deflatores de pontos de função, avaliando cada item do escopo conforme o seu tipo de manutenção (seja o processo elementar ou as estruturas de dados):

DEFLATORES	
Tipo de Impacto	**Valor do Fator**
Inclusão	-
Alteração	0,50
Exclusão	0,25
Testes	0,15

Tabela 9 – Deflatores de Pontos de Função

Note na tabela acima, uma sugestão para tratamento do escopo que envolve funções apenas testadas, o que é chamado "Pontos de Função de Testes" ("PFT"). Este não é um recurso nativo da APF, mas muitas vezes é interessante utilizar esta abordagem para remunerar ou avaliar a produtividade de testes de regressão manuais realizados sobre funcionalidades que não estão compreendidas dentro do escopo "mexido", mas que eventualmente podem ter sofrido algum impactado por mudanças indiretamente relacionadas. A saber, "Pontos de Função de Testes" é uma abordagem derivada do Roteiro de Métricas do SISP e alguns estudos que o antecederam.

7.6.2.2 Estimativa do Target

Sugere-se avaliar uma proposta semelhante àquela que sugerimos no tópico sobre COSMIC, ou seja, criar uma base histórica e escalonar o tamanho e/ou definir uma ponderação através de uma referência de duração a partir dos *logs* das execuções anteriores do processo (exemplo: as faixas de duração e respetivos custos). No entanto, quando o escopo for razoavelmente conhecido, considere realizar uma contagem utilizando o método de Estimativa NESMA,

assim como um fator calibrado de desvios entre as contagens de *Target* e o consolidado das medições finais executadas.

São parâmetros de entrada numa estimativa de *Target* as seguintes informações:

- A descrição do *Target*;
- As entrevistas realizadas com *stakeholders*;
- Os seguintes dados, agrupados em classificações que demonstram padrões encontrados na base histórica (dados calibrados):
  - Tamanho médio, em pontos de função;
  - Esforço e custo médio do ponto de função;
  - Quantidade média de *sprints* e tamanho médio em pontos de função por *sprint* (velocidade);
  - Duração esperada, em meses.

Com base nos insumos citados, a estimativa poderá ser realizada. E ao final, serão disponibilizados os seguintes resultados:

- Tamanho, em pontos de função;
- Esforço, em horas;
- Duração, em meses;
- Duração, em quantidade de *Sprints*;
- Custo total.

7.6.2.3 Estimativa do Épico ou Bloco

A contagem inicial do Épico (ou Bloco de Trabalho) é realizada de forma estimada, durante a Concepção ou Planejamento, e pode ser viabilizada com base nas funcionalidades demonstradas durante as cerimônias ou no Sequenciador de *Features*, assim como em entrevistas realizadas com o time de desenvolvimento. O seu resultado pode ser expresso como um total de pontos do Épico ou segregado por MVP, para facilitar o planejamento. Dependendo da sua realidade, considere prever o relacionamento do item da

contagem com um projeto específico ou mesmo sistema (útil para algumas empresas, dependendo do propósito vinculado).

Os parâmetros de entrada para realizar uma "Estimativa do Épico" são os seguintes:

- Insumos gerados durante as etapas de Concepção e Planejamento;
- Dados calibrados de:
  - Custo unitário médio do ponto de função;
  - Índice de produtividade (horas por ponto de função);
  - Taxa de entrega (quantidade de pontos produzidos por pessoa/mês);
  - Velocidade média (pontos de função produzidos por *sprint*);
  - Fator de retrabalho e/ou de evolução dos requisitos (para avaliar a escala entre requisitos iniciais e finais).
- Entrevistas com participantes da Concepção e/ou Planejamento, desde que estes tenham condições de sanar eventuais dúvidas relacionadas ao desenvolvimento do escopo.

Para realizar uma contagem inicial, considere também:

- Obter a referência da estimativa realizada durante o *Target*;
- Considerar convenções locais, documentando as devidas premissas e observações pertinentes ao processo de estimativas.

Desta forma, ao finalizar a contagem, espera-se que estejam disponíveis as seguintes informações:

- Tamanho, em pontos de função, total, por MVP e por processo elementar ou estrutura de dados;
- Esforço em horas, conforme tamanho e produtividade identificada;
- Duração em meses, com base no *time-box* padrão da organização, adicionando-se uma estimativa proporcional às classificações calibradas de Épicos;

- Duração, pela quantidade de *Sprints*, com base no tamanho e na velocidade média;
- Quantidade de recursos, com base no tamanho e na taxa de entrega;
- Custo total.

7.6.2.4 Estimativa do MVP

Outra contagem estimada, mas agora no nível de MVP, realizada com o direcionamento das *Features* e entrevistas. Ela define a estimativa de tamanho e o trabalho a ser considerado executado até o final de um MVP iniciado, considerando ainda uma duração aproximada para referência de alocação de recursos e custos associados.

A primeira contagem de MVP não necessariamente precisa ser realizada, visto que durante o Planejamento do Épico já foi realizada uma separação da quantidade de pontos prevista para cada MVP. No entanto, a medida que o projeto evolui, mudanças vão surgindo e as estimativas tem de ser atualizadas (sempre que possível).

Embora opcionalmente executável, este ponto de estimativa tende a ser particularmente útil aos líderes de projeto, que precisam saber com antecedência se será necessário revisar o orçamento e/ou planejamento do Projeto ou Épico (uma eventual necessidade de extra-orcamentação, por exemplo, que normalmente precisa ser levada à um Comitê de TI ou ao próprio negócio, para aprovação).

7.6.2.5 Medição do MVP

Uma contagem final nesta abordagem poderá, a depender das especificidades da organização, apresentar o conjunto de funcionalidades no escopo dos MVPs, de maneira incremental. Neste formato de trabalho, a partir do primeiro MVP entregue, todas as funcionalidades mensuradas em MVPs anteriores permanecem na contagem de referência do Épico em questão, conforme os exemplos que serão abordados a seguir. No entanto, isso não significa que você perderá informações sobre o que foi entregue em cada iteração,

pois na contagem consolidada dos MVPs (falaremos sobre isso mais tarde), é possível destacar em qual iteração foi finalizado o desenvolvimento da funcionalidade. Por consequência, cada uma terá um número de pontos associado e poderá derivar medidas de produtividade, por exemplo.

Fazendo a contagem incremental do Épico, indicada em casos onde se queira avaliar apenas MVPs e onde o trabalho é realizado através de planilhas, você terá algumas vantagens:

- O conhecimento sobre o tamanho e sobre demais estimativas derivadas pode ser documentado em um único local, mantendo inclusive o histórico;
- A contagem do segundo MVP do Épico em diante torna-se mais fácil, também pelo histórico agregado e contexto implícito;
- O planejamento do Épico é do Projeto tornam-se mais transparentes;
- A contagem das mudanças de um MVP para outro tornam-se mais fáceis de serem realizadas devido ao reaproveitamento e por não ser necessário buscar informações das contagens passadas de todos os MVPs anteriores em uma base histórica ou ferramenta que gerencia os atendimentos de métricas;
- Ao acessar um diretório versionado, as informações sobre um Épico, dos seus MVPs e das Iterações relacionadas podem ser armazenadas e recuperadas integralmente e de forma consolidada;
- Diversas medidas derivadas para a gestão de entregas e do projeto podem ser consolidadas em uma única planilha.

Abaixo, um exemplo de como uma contagem incremental poderia ser aplicada utilizando a abordagem de medição a cada final de MVP (resultados derivados podem ser escados ao nível de Iteração, caso necessário):

Exemplo 1

O MVP1 do Épico X envolveu o seguinte escopo mapeado (e respectiva estimativa):

- Cadastro de cliente (Inclusão);
- Incluir cliente (Inclusão);
- Excluir cliente (Inclusão);
- Emitir relatório de clientes (Inclusão).

Função	Tipo de Impacto	Tamanho (PF)
Cadastro de Cliente (ALI)	Inclusão	10
Incluir Cliente (EE)	Inclusão	4
Excluir Cliente (EE)	Inclusão	3
Emitir Relatório de Clientes (SE)	Inclusão	5
Total		**22**

Quadro 3 – Contagem no Exemplo 1

Exemplo 2

O MVP2 deste épico, por sua vez, envolveu o seguinte escopo mapeado (e respectiva estimativa):

- Alterar cliente (Inclusão);
- Consultar cliente (Inclusão).

Função	Tipo de Impacto	Tamanho (PF)
Cadastro de Cliente (ALI)	Inclusão	10
Incluir Cliente (EE)	Inclusão	4
Excluir Cliente (EE)	Inclusão	3
Emitir Relatório de Clientes (SE)	Inclusão	5
Alterar Cliente (EE)	Inclusão	4
Consultar Cliente (EE)	Inclusão	4
Total		**30**

Quadro 4 – Contagem no Exemplo 2

Exemplo 3

Consolidando a visão dos demais MVPs adicionados ao próprio, o MVP 3 envolveu então o seguinte escopo (e respectiva estimativa):

- Incluir cliente (devido à alteração da funcionalidade para inclusão de novas regras de negócio envolvendo a validação dos dados, sendo que esta necessidade foi identificada somente no MVP3).

Função	Tipo de Impacto	Tamanho (PF)
Cadastro de Cliente (ALI)	Inclusão	10
Incluir Cliente (EE)	Inclusão	4
Excluir Cliente (EE)	Inclusão	3
Emitir Relatório de Clientes (SE)	Inclusão	5
Alterar Cliente (EE)	Inclusão	4
Consultar Cliente (EE)	Inclusão	4
Incluir Cliente (EE)	Alteração *	2
Total		**32**

** Considerando um deflator de 0,50*

Quadro 5 – Contagem no Exemplo 3

São consideradas desvantagens da realização de uma contagem incremental cujo foco reside no MVP:

- Em um cenário de baixa maturidade e com muitos MVPs, provavelmente em algum momento eles serão executados em paralelo, o que difultará uma sobreposição;

- Caso não tenha sido executada Concepção e/ou Planejamento não haverá um Épico ou Bloco, e logo, também não existirá referência comum para múltiplos MVPs (algo como um agrupador);
- Dependendo do sistema de informação e/ou processo utilizado, pode ser complexo descobrir a Concepção que originou um MVP específico.

Como parâmetro de entrada para a medição, são úteis os artefatos iniciais de concepção, eventuais entrevistas e acesso ao sistema (uma demonstração no *show-case*, quando da entrega/transição do MVP, por exemplo). Contudo, há outras possibilidades a serem avaliadas no intuito de fornecer os insumos suficientes para a realização de uma contagem de pontos de função.

Após uma contagem de pontos de função, os seguintes resultados poderão ser obtidos:

- Tamanho, em pontos de função;
- Esforço total realizado;
- Esforço interno;
- Esforço de fornecedor de *software*;
- Esforço de fornecedor de *qualidade*;
- Índice de produtividade;
- Custo médio do ponto de função;
- Custo médio do ponto de função por fornecimento de serviços (custo de fornecimento);
- Duração, em quantidade de *Sprints* executadas no MVP;
- Velocidade média, em pontos de função produzidos por *sprint*;
- Duração, em meses;
- Qualidade, em inconformidades por ponto de função (ou em 1000 pontos de função).

É importante ressaltar que as "saídas" devem ser armazenadas em uma base histórica e periodicamente calibradas, visando evoluir

a capacidade de análise de performance dos processos em torno do desenvolvimento e também a capacidade de predição durante as estimativas iniciais.

7.7 Monitoramento Visual de Performance

Em geral, os gestores necessitam acompanhar os projetos e entregas que estão sob sua responsabilidade, sendo que este tipo de monitoramento baliza a tomada de decisão e possibilita análises mais consistentes sobre os processos utilizados, com o intuito de evoluí-los.

Considerando o uso de métodos de tamanho como *plugins* do processo de desenvolvimento, veremos a seguir como podemos implantar uma estratégia de monitoramento visual de performance.

Para fins didáticos e também para evitar repetição, utilizaremos o nível de abstração de "iteração" na abordagem do COSMIC, mas outro nível de abstração e/ou abordagem já referido poderia ser utilizado, inclusive englobando os mesmos recursos.

Os assuntos ou meios de acompanhamento que serão analisados durante este tópico são os seguintes:

- Diagnóstico;
- Progresso;
- Desempenho;
- Consolidado.

7.7.1 Diagnóstico

Realizar o "Diagnóstico" da operação é importante para realizarmos o plenajamento e o acompanhamento. Esta ação contribui assim, na elaboração das seguintes análises vinculadas ao escopo, trabalho e saúde do sistema:

- Capacidade;

- Complexidade;
- Saúde do Sistema.

7.7.1.1 Capacidade

Cada time possui uma "Capacidade de Entrega", assim como a organização também. Sendo assim, podem ser consideradas duas perspectivas principais de avaliação:

- Quantidade de pontos de função, em média, produzidos por mês pelo time;
- Quantidade de pontos de função, em média, produzidos pela organização, considerando todos os times e/ou por subportfólio.

O PMO, assim como as próprias equipes, deve utilizar esta referência para uma análise da força de trabalho da TI da organização.

7.7.1.2 Complexidade

Visando maior assertividade na elaboração das estimativas iniciais nos diferentes cenários de desenvolvimento aplicáveis, há meios de considerarmos uma medida de complexidade por produto/time para ponderá-los e analisá-los separadamente.

Isso normalmente não é um habilitador inicial (ao implantar o modelo de métricas), mas é considerado útil para obtermos uma aproximação para o processo de estimativas, ou ainda, para explicar eventuais diferenças de performance de um time (ou entre times).

A complexidade do ambiente onde o time está inserido ou de uma aplicação sob o seu escopo pode ser avaliada com base em alguns parâmetros. O COCOMO II, considerando principalmente os seus "multiplicadores de esforço", é muito útil para atender este tipo de necessidade, assim como dados da complexidade ciclomática da aplicação, que pode ser automaticamente obtida em ferramentas como o *Sonar Cube*.

7.7.1.3 Saúde do Sistema

O gráfico de avaliação da saúde do sistema é uma ferramenta extremamente útil para realizar o acompanhamento do trabalho ao longo do tempo. Ele possibilita demonstrar visualmente o trabalho a fazer, além do trabalho em andamento e do que já foi feito. Além disso, pontos de gargalo passam a ser identificados claramente quando de uma ocorrência onde ultrapassamos a capacidade máxima prevista para o trabalho em progresso.

Você deve utilizar outros indicadores, como o Tempo de Atravessamento, Taxa de Entrada e Taxa de Saída, para ajudá-lo a compreender o que este gráfico pretende lhe mostrar.

7.7.1.3.1 Por Portfólio e Subportfólio

Na perspectiva de um portfólio de projetos é importante avaliar os projetos a fazer, em andamento e aqueles concluídos em um determinado momento histórico, seja para averiguar o seu *status* ou mesmo para identificar eventuais impedimentos. Veja abaixo como podemos constatar isso através de um gráfico de fluxo cumulativo:

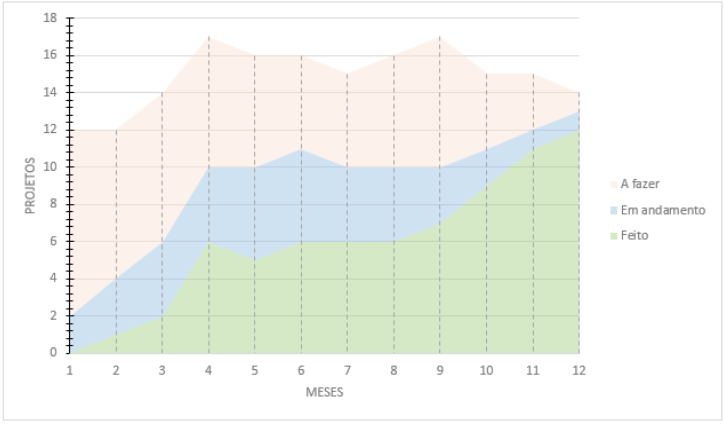


Gráfico 39 – Saúde do Sistema por Projetos no Ano

Outras visões do gráfico acima podem ser geradas, como exemplo, considerando a perspectiva de Épicos ou MVPs (unidades de

valor):

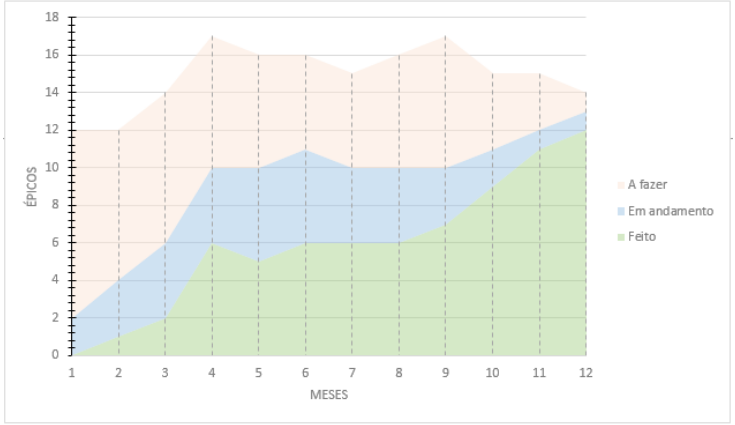


Gráfico 40 – Saúde do Sistema por Épicos no Ano

7.7.1.3.2 Por Time e Produto

Dependendo do nível de granularidade que você escolher e também de como efetivamente utilizará o modelo de estimativas e medição, você poderá ter um acompanhamento fino e até mesmo em tempo real do trabalho a fazer, que está sendo feito ou que já foi concluído pelo time. Você perceberá isso principalmente se preferir utilizar o *Plugin* COSMIC de estimativa e medição como um método de estimativas do time, ou seja, inclusive utilizando o COSMIC *Planning Poker* ou outra técnica compatível para avaliar o esforço dos itens de desenvolvimento.

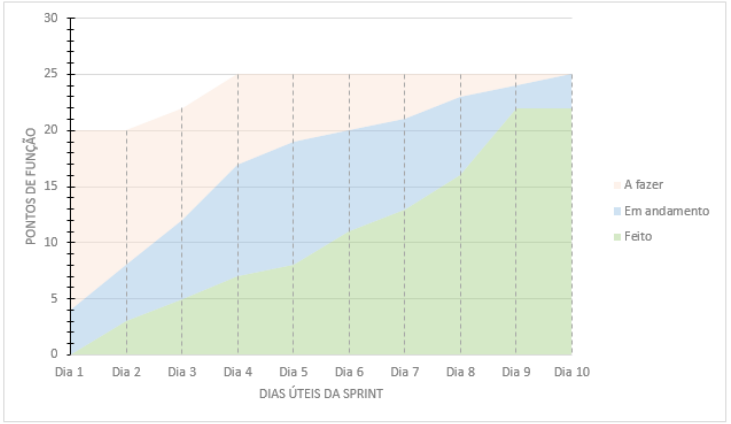


Gráfico 41 – Exemplo de Gráfico para uma *Sprint*

7.7.2 Progresso

Uma medida de progresso representa o quanto fizemos ou o quanto ainda nos falta fazer.

Para avaliar esta medida podemos utilizar as seguintes opções:

- Realizado;
- Completude;
- Escala de Evolução.

7.7.2.1 Realizado

Um gráfico que demonstra o trabalho realizado e/ou efetivo em relação aos marcos e metas é uma excelente forma de representar o progresso de uma Entrega, de um Épico ou de um Bloco de Trabalho. Ele também possibilita demonstrar o retrabalho através da perspectiva de quanto trabalho foi despendido ao revisitar ou desistir daquilo que há havia sido considerado pronto.

O gráfico *burn-up* abaixo demonstra claramente o trabalho efetivo realizado (unidades de tamanho que foram desenvolvidas ou customizadas), o incremento real no produto (o que realmente

agregará em termos de funcionalidade) e a avaliação de retrabalho (a diferença entre o trabalho realizado e o incremento). Veja só:

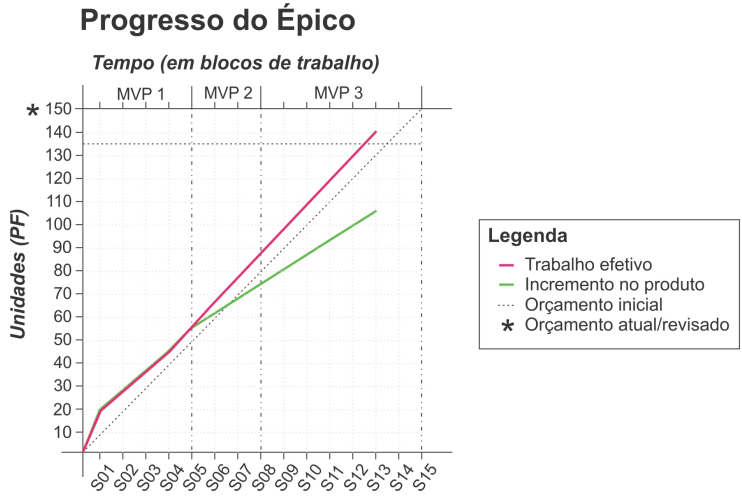


Gráfico 42 – Gráfico de *burn-up* de progresso: realizado e efetivo

Com base no gráfico, podemos derivar algumas conclusões ou criar indicadores dos blocos, para análises e/ou previsões futuras.

Contudo, a depender do propósito, pode ser conveniente visualizar o trabalho realizado em uma perspectiva de alto nível. Veja exemplos de algumas visões que podem ser utilizadas:

- Por Período;
- Por Projeto;
- Por Épico ou Bloco;
- Por MVP;
- Por *Sprint*;
- Por Dia.

7.7.2.2 Completude

Medidas de progresso são úteis para acompanharmos a evolução dos itens no *backlog*, seja numa *Sprint* ou MVP. Assim, é

possível acompanhar a quantidade de unidades de *software* construídas, em pontos de função, e o trabalho a ser realizado, através do percentual de completude.

Abaixo, você pode visualizar um gráfico relacionado:

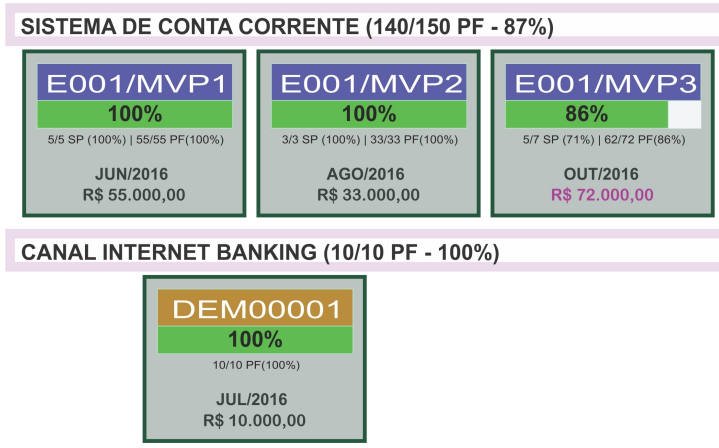


Gráfico 43 – Progresso de MVPs por Épico

Dependendo do grau de abstração, é possível optar pelo acompanhamento de completude no nível de:

- Projeto: para acompanhamento do portfólio na visão corporativa;
- Épico ou Bloco: para acompanhamento do subportfólio ou portfólio em um nível de macro-entregas;
- MVP: para viabilizar a esteira de produtos mínimos;
- *Sprint*: para prover um status de completude da *sprint*.

7.7.2.3 Escala de Evolução

O esforço, os custos e o total de unidades de *software* planejados é tomado como alvo, porém nem sempre corresponde ao efetivamente executado. Desta forma, referências podem ser observadas ao final de uma *sprint* (por exemplo), visando averiguar a eficiência das estimativas.

Os níveis onde são obtidas informações de estimativas que são confrontadas com o realizado são os seguintes:

- No Portfólio e Subportfólios;
- No Projeto;
- No Épico;
- No MVP;
- Na *Sprint*.

O fator de evolução obtido tem uma tendência central de *1,00*, que oscila para mais ou para menos. Esta referência pode ser utilizada como fator de escala para a realização das estimativas iniciais.

7.7.3 Desempenho

As medidas de "Desempenho" representam quão eficaz, eficiente ou efetivo o processo ou a equipe tem sido em relação ao seu histórico ao longo do tempo. Elas poossibilitam avaliar "onde estamos" e "onde queremos chegar" em relação às expectativas da organização. Isso torna-se viável ao medirmos as entregas realizadas, contribuindo para a evolução do modelo de estimativas e viabilizando inúmeras análises de tendências sobre os dados gerados na operação.

Alinhados ao objetivo e visando prover capacidade de acompanhamento à operação, dos subportfólios e dos times específicos, sugerimos a definição de 6 alternativas e respetivos critérios de avaliação, atuando sobre 5 domínios de informação: "Eficácia", "Eficiência", "Efetividade", "Maturidade" e "Clima". Assim, cada domínio será avaliado conforme as alternativas, em uma escala de níveis de 0 à 5.

Cada alternativa disponível para um nível será composta por um ou mais critérios, que podem ser combinados conforme suas propriedades:

- Medida: < MEDIDA >;

- Operador: “>”, “<”, “=”, “>=”, “<=”;
- Constante: < VALOR >;
- Agregador: “E”, “OU”.

Um critério vinculado a uma medida cria uma condição, que por sua vez, pode ser agregada, unindo outros critérios, forma um nível específico do domínio. Idealmente, cada time terá uma configuração, e níveis de abstração superiores também calibrarão suas próprias avaliações, as quais podem ser utilizadas em função de metas a serem seguidas.

7.7.3.1 Eficácia

O termo “Eficácia” remete ao significado “chegar ao objetivo proposto”, ou seja, cumprir uma função ou meta delineada. Representa ainda, a execução de algo que foi determinado ou a quantidade e/ou qualidade dos produtos e serviços entregues ao usuário. Por exemplo, podemos dizer: “o equipamento mostrou-se eficaz”.

O domínio de “Eficácia” pode ser avaliado com base em duas medidas:

- Atingimento de Metas;
- Qualidade do Código.

Elas são complementares e por isso esperamos que faça sentido para a maioria das organizações considerar 2 critérios na condição (para cada nível/ alternativa). Sendo assim, um exemplo de condição aplicável ao NÍVEL 5 deste domínio é a seguinte:

```
([Indicador: ATINGIMENTO DAS METAS] >= [90%]) E ([Indicador: QUALIDADE DE CÓDIGO]) <= [3]
```

7.7.3.1.1 Atingimento de Metas

O "Atingimento de Metas* da *Sprint* é um valor percentual que avalia se foi entregue tudo aquilo com o qual o time se comprometeu.

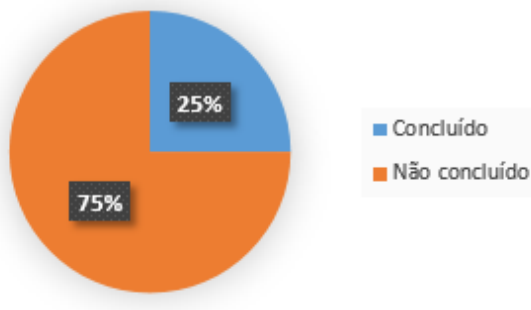


Gráfico 44 – Atingimento das Metas

Existem algumas alternativas para a apuração deste indicador, sendo as principais:

- Considerar a quantidade de pontos de função aceitos em relação à quantidade de pontos de função planejados na *Sprint*;
- Considerar a quantidade de pontos de função aceitos em relação à quantidade de pontos de função dos itens executados na *Sprint*;
- Considerar a quantidade de pontos de função dos ítens concluídos na *Sprint* (não necessariamente aceitos) em relação ao total de pontos de função planejados.

7.7.3.1.2 Qualidade de Código

O indicador de "Qualidade de Código" demonstra o número de inconformidades identificadas durante a homologação e/ou implantação referente a porção do produto que foi entregue na *Sprint*, considerando uma proporção por unidade de ponto de função ou a cada 1000 pontos de função implementados (usa-se esta unidade

para uma compatibilidade direta com referências de *benchmarking*). Assim, tanto o tipo da inconformidade como a unidade da unidade de medida que serão utilizados são aspectos que você precisará definir ao implementar esta medida.

Abaixo, você acompanha um gráfico que representa o número de inconformidades identificadas na entrega de uma *Sprint* fictícia:

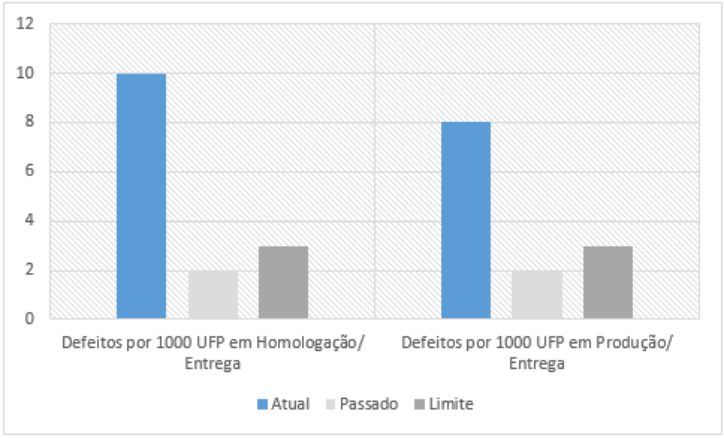


Gráfico 45 – Qualidade da Entrega

A saber, visando calcularmos o resultado do exemplo citado, utilizamos a seguinte fórmula:

> Índice de Qualidade de Homologação = Quantidade de Inconformidades na Homologação / Quantidade de PF (da *Sprint* ou MVP)

Se preferir, também podemos avaliar a qualidade considerando outros tipos de inconformidade, como de processo e/ou documentação. Outra alternativa, é fornecer visões em perspectiva executiva, em gráficos que demonstrem a visão consolidada dos times (vide

abaixo).

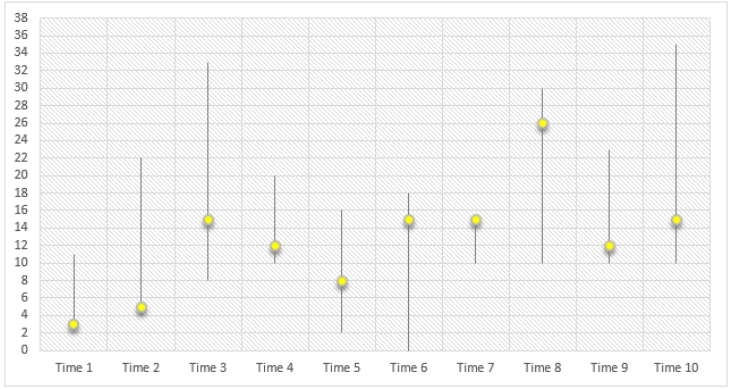


Gráfico 46 – Qualidade nos Times

7.7.3.2 Eficiência

O conceito de "Eficiência" remete à qualidade do que se faz com eficiência, que dá bom resultado e que produz o efeito desejado com competência. O termo também representa a relação existente entre os produtos/ serviços gerados (*outputs*) e os respectivos insumos utilizados, ou seja, vinculando aquilo que foi entregue com o que foi efetivamente consumido.

Em geral, dados sobre eficiência são utilizados para avaliar como estão sendo utilizados os recursos disponíveis (sua alocação), mas também são úteis como referência ao planejamento e prover capacidade de acompanhamento aos Projetos, Épicos, MVPs e *Sprints* que são executados pela organização.

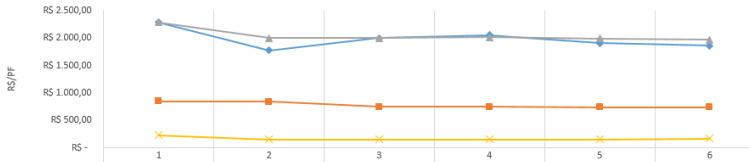
7.7.3.2.1 Custo

Uma análise de "Custo" inevitavelmente remete à eficiência, pois refere-se ao investimento que viabiliza as entregas. Nós o avaliamos na forma de "Custo Unitário" (do Ponto de Função), que pode representar:

- Os custos gerais do serviço;

- Os custos de fornecimento de *software* (caso aplicável);
- Os custos de qualidade do processo (se necessário avaliá-los separadamente).

O gráfico a seguir demonstra um acompanhamento fictício de medidas de Custos de um time:



	1	2	3	4	5	6
Individual	R$ 2.285,71	R$ 1.777,78	R$ 2.000,00	R$ 2.051,28	R$ 1.904,76	R$ 1.860,47
Acumulado	R$ 2.285,71	R$ 2.000,00	R$ 2.000,00	R$ 2.012,58	R$ 1.990,05	R$ 1.967,21
Acumulado SW	R$ 851,43	R$ 838,13	R$ 745,00	R$ 749,69	R$ 741,29	R$ 738,89
Acumulado QA	R$ 228,57	R$ 150,00	R$ 153,33	R$ 155,97	R$ 155,22	R$ 167,21

Gráfico 47 – Custo Unitário do PF

7.7.3.2.2 Produtividade

A "Produtividade" representa o total de horas as quais estimam-se necessárias ou que foram efetivamente utilizadas no desenvolvimento de uma unidade de trabalho. No contexto em análise, podemos dizer que ela é uma referência ao tempo médio necessário para produzir cada unidade de ponto de função.

Abaixo, veja um exemplo de gráfico onde a produtividade é apresentada, considerando tendências como a sua média e amplitude, assim como valores calibrados através da aplicação de métodos matemáticos e/ou estatísticos sobre uma base histórica:

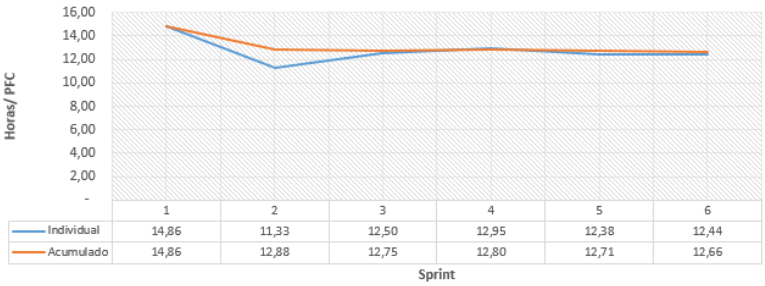


Gráfico 48 – Produtividade

Confira também no gráfico abaixo, como você pode apresentar os resultados consolidados dos times:

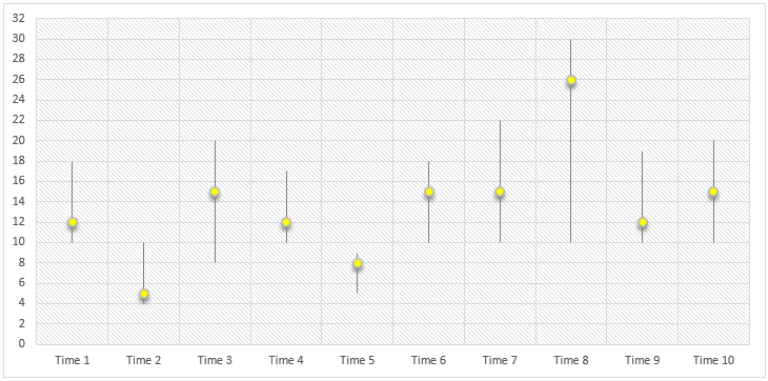


Gráfico 49 – Produtividade dos Times

Contudo, evite comparar diretamente a produtividade entre times de desenvolvimento, já que normalmente é possível conhecer suficientemente todas as variáveis de ambiente relacionadas, como aspectos de negócio e tecnologia, fatores humanos e outros. Note que embora isso pudesse ser aproximado, devido aos princípios e valores ágeis, considera-se que comparações tendem a não ser sustentáveis ao longo do tempo.

7.7.3.2.3 Taxa de Entrega

A "Taxa de Entrega" é uma medida padronizada da quantidade de pontos de função que em média são produzidos pelo trabalho de uma pessoa num período de tempo pré-definido, mas sem especificamente identificá-la. Esta referência responde à seguinte questão: "em média, quantos pontos saem da esteira devido ao trabalho de uma pessoa no período?". A Taxa de Entrega ainda pode ser definida como "prova real" da "velocidade" e/ou da "produtividade", quando não utilizada como medida principal para o cálculo de eficiência.

Abaixo, confira um gráfico com dados históricos de Taxa de Entrega, formados com base no trabalho realizado por 10 times de uma organização (fictícios):

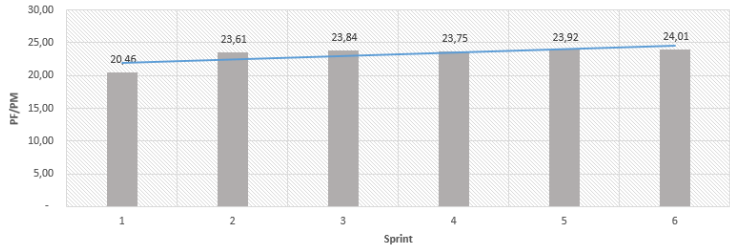


Gráfico 50 – Taxa de Entrega

Ao avaliar a capacidade média dos membros de um time ao longo do tempo, é possível realizar análises sobre eventuais mudanças na sua configuração. Todavia, ao simular retirar ou incluir um profissional no time, considere que este tipo de movimento tende a impactar também os demais integrantes, causando oscilações de desempenho do grupo no curto prazo. Isso significa que as comparações na linha do tempo devem ser avaliados a médio ou longo prazo após a ocorrência de mudanças nos membros de uma equipe.

Esta medida viabiliza ainda, a comparação da operação de desenvolvimento com o mercado, de forma similar ao que fazemos com o índice de produtividade dos projetos. Há *benchmarking*

relacionado e passível de obtenção através de diversas fontes como o ISBSG.

7.7.3.2.4 Tempo de Execução

O "Tempo de Atendimento" responde à seguinte questão: "em média, quanto tempo demoramos para concluir um item de trabalho?". Ele pode ser avaliado em termos de:

- Tempo de atravessamento;
- Tempo de saída;
- Tempo decorrido.

7.7.3.2.5 Atravessamento

Na perspectiva dos desenvolvedores, o "Tempo de Atravessamento" (*lead time*, em inglês) é calculado da seguinte forma: DATA FIM - DATA INICIO (do item de trabalho, ou especificamente uma US). Entretanto, na perspectiva de negócio, sugere-se considerar como DATA FIM, o aceite ou conclusão da US, e como DATA INÍCIO, a data inicial da *Feature*, que foi elaborada durante a etapa de Concepção.

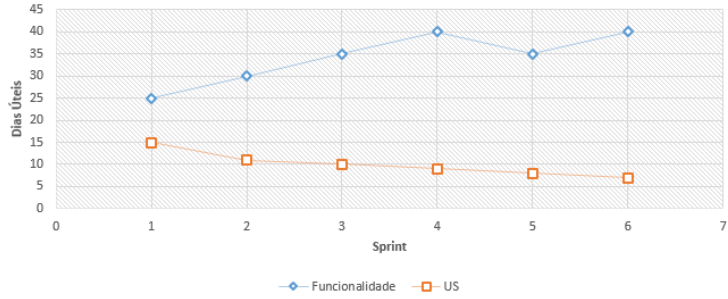


Gráfico 51 – Tempo de Atravessamento

Neste contexto, a primeira abordagem remete a um controle de produção estabelecido do time para o time, enquanto a segunda reflete o tempo que o negócio teve de esperar até a disponibilização de um novo recurso.

> Você também pode calcular o tempo de atravessamento de um processo funcional (considerando a visão do usuário) através do cálculo ***DATA FIM - DATA INÍCIO***, onde a primeira data é a de disponibilização da primeira versão da funcionalidade num MVP ou *Sprint* onde ela foi aceita, e a segunda é a data de início Épico, marco de quando o usuário confirmou pela primeira vez o escopo.

7.7.3.2.6 Saída

A "Taxa de Saída" (ou Taxa de Transferência) representa a quantidade média de itens de trabalho ou pontos de função concluídos diariamente. É ainda, um indicador fundamental para o acompanhamento do WIP (*work in progress*), que possibilita avaliar a estabilidade do sistema.

Cabe observar que um sistema ***estável*** é aquele onde a taxa de entrada é igual a taxa de saída. Desta forma, este é o cálculo que você deve fazer para avaliar se não está trabalhando com itens demais ou se não tem executado menos do que sua capacidade possibilitaria.

7.7.3.2.7 Decorrido

O "Tempo Decorrido" é utilizado para avaliar o tempo médio realizado na produção de um item de trabalho. É uma referência que pode ser obtida através do seguinte cálculo: ***ESFORÇO (HH) / ITENS ENCERRADOS***.

Dependendo da necessidade de acompanhamento, um item pode ser considerado um "Projeto", "Épico", "MVP", "*Feature*" ou uma *User Story* (US). O resultado do cálculo, por sua vez, serve principalmente para realizar a calibração da produtividade de referência que será utilizada nas estimativas iniciais do produto.

7.7.3.2.8 Velocidade

A "Velocidade Média" é a quantidade de pontos de função que um time é considerado ou já foi capaz de produzir, como valor médio, num determinado período de tempo (tipicamente, uma iteração). É uma medida comumente avaliada ao analisar o desempenho de times de desenvolvimento de *software* ágil.

Veja abaixo, um gráfico que pode ser utilizado para apresentar o histórico de velocidade de um time:

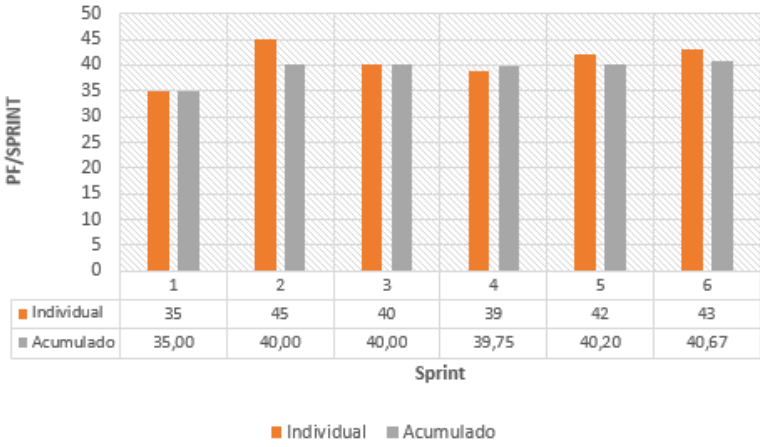


Gráfico 52 – Velocidade Média

A velocidade de um time, assim como destacamos em relação ao índice de produtividade, idealmente não deve ser comparada diretamente com outro time, já que temos um propósito de acompanhamento das entregas com viés decisório e não de controle funcional.

7.7.3.3 Efetividade

Segundo um bom dicionário, o termo "Efetividade" remete à qualidade daquilo que atinge o seu objetivo ou a capacidade de funcionar regularmente, satisfatoriamente, e como referência ao que é real e verdadeiro. Está também vinculado ao grau de

satisfação ou o valor agregado, à transformação produzida. Como exemplo, podemos dizer que: "o resultado da reunião foi muito efetivo".

Sendo assim, traduzimos para o nosso contexto que a efetividade das entregas representa uma análise do valor agregado ao trabalho do time de desenvolvimento, conforme a percepção dos usuários impactados e/ou seus requisitantes. Isso pode ser considerado através do Índice de Satisfação, que serve para ponderar quão bons temos sido no que fazemos, considerando a visão do negócio. Este indicador é complementado pela percepção de valor obtida junto aos *stakeholders*.

7.7.3.3.1 Satisfação

O aspecto de "Satisfação" do usuário pode ser avaliado durante uma *Review*, ou calibrado em um período a ser definido. Esta referência representa o quão satisfeito o negócio está com o produto entregue e/ou o trabalho executado (ou seja, o atendimento prestado).

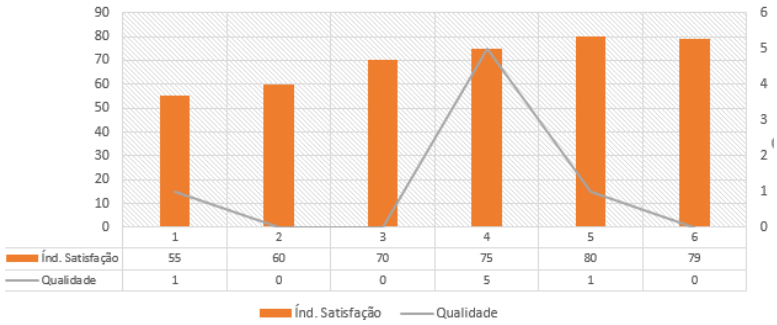


Gráfico 53 – Satisfação do Cliente

7.7.3.3.2 Percepção do Valor

A "Percepção de Valor" conforme a visão dos usuários e/ou requisitantes (*stakeholders*) pode ser incorporada à análise de "Satisfação" ou avaliada como um indicador separado, em uma perio-

dicidade a ser acordada com as partes interessadas.

Contudo, talvez o ideal seja obter ambas as medidas separadas, mas durante as cerimônias de *Review*.

7.7.3.4 Maturidade

Uma medida de "Maturidade" serve para demonstrar quão pronto o time está para trabalhar eficientemente com a agilidade. Também para visualizar os principais pontos que impactam a capacidade do time em suas entregas, como ferramentas e processos, por exemplo. Tais pontos precisam ser destacados para acompanhamento, fomentando a evolução contínua e possibilitando enxergar como o time tem melhorado durante sua vida.

O acompanhamento de maturidade é viabilizado através de num gráfico de radar, o que possibilita a avaliação da aptidão para a utilização de métodos ágeis e enxutos de forma facilmente entendível pelos interessados.

Para proceder com este tipo de análise, utilizamos uma relação de 7 níveis de maturidade (de 0 a 6) conforme escala definida no modelo de "Prontidão" de PHAM (2011), que você irá conhecer a seguir. Considere descrições completas e questionários como entrada para uma definição precisa do nível em que você se encontra. Este níveis podem ser customizados e adaptados à realidade de cada organização.

O gráfico de "Prontidão" serve ainda, para avaliar o momento de adoção ágil e enxuta, e contribui na identificação do que precisa ser feito para possibilitar o atingimento de metas, avaliando se estamos ou não no caminho certo.

Confira abaixo, um exemplo do gráfico de maturidade:

Gráfico 54 – Maturidade do Time

Esta medida mostra quão experiente e preparado o time está para encarar os desafios que lhe foram propostos, o que pode ser obtido através dos dados coletados de forma objetiva através de questionários. Conforme PHAM (2011) instrui, precisamos avaliar fatores que estão distribuídos na forma de "Dimensões de Ambiente" (DA), os quais no modelo original, são demonstrados como tão significativos que podem inclusive impactar as estimativas dos projetos. A seguir, são elencados tais dimensões:

- Organizacional;
- Infraestrutura de Desenvolvimento;
- Equipe;
- Tecnologia;
- Processo;
- Negócio.

Cada dimensão proposta pelo autor, é avaliada por quesitos ou alternativas, cuja pontuação é pré-definida entre 0 e 2 pontos, sendo que isso indica maior ou menor capacidade ou habilidade no quesito específico. Sendo assim, ao final da avaliação, teremos uma pontuação total entre 0 e 36 pontos, e dependendo do valor total, três cenários de classificação são possíveis:

- Se o valor de DA ficar entre 0 e 11, a avaliação indica que o ambiente facilita o trabalho da equipe.
- Se o valor de DA ficar entre 12 e 23, a avaliação implica que o ambiente não dificulta nem facilita o trabalho da equipe.
- Se o valor de DA ficar entre 24 e 36, a avaliação indica que o ambiente facilita o trabalho da equipe.

O autor cita ainda, que o principal objetivo desta abordagem é manter-se perto dos seguintes alvos:

- Entregas de *software* frequentes;
- Colaboração regular entre a empresa e a equipe de *software*;
- Cronometrar tudo ou todas as reuniões para evitar perda de tempo;
- Ciclos de inspeção e adaptação frequentes;
- Autogerenciamento e autonomia da equipe;
- Tudo isso a um ritmo sustentável (para que a equipe não se prejudique).

PHAM (2011) defende a utilização desta proposta para uma análise de prontidão da organização, considerando os seguintes níveis:

- Se o valor de DA for próximo a 36, suas chances de sucesso no uso de *Scrum* são maiores que a média
- Se o valor de DA for de 18, significa que suas chances de sucesso estão na média.

- Se o valor de DA for próximo a 0, suas chances de sucesso no uso de *Scrum* são menores que a média.

Para chegar às pontuações citadas, as seguintes questões são consideradas (relação por DA, com pontuação por questão variando entre 0 e 2 pontos):

7.7.3.4.1 Organização

Abaixo, segue a relação de itens avaliados para a dimensão "Organizacional":

- Os diferentes departamentos trabalharam em conjunto, com sucesso, em projetos *Scrum* anteriormente?
- Há alguma forte resistência dentro da empresa em relação ao *Scrum*?
- Existe um grande suporte ao *Scrum* entre os diferentes departamentos?

7.7.3.4.2 Processos

A dimensão "Processo" é avaliada, por padrão, conforme os estudos de PHAN (2011), através das seguintes questões:

- O *Scrum* é a estrutura de processos adotada pela empresa?
- Existe um bom suporte ao *Scrum dentro da empresa?
- Há uma forte resistência contra o *Scrum* dentro da empresa?

7.7.3.4.3 Infraestrutura

A "Infraestrutura" (de desenvolvimento), por sua vez, é avaliada com base nas questões a seguir:

- Os testes automatizados já estão a postos e são uma prática comum?
- Os testes de integração contínua já estão a postos e são uma prática comum?
- O ambiente de construção diária já está a postos e é uma prática comum?

7.7.3.4.4 Tecnologia

Em relação aos fatores da dimensão de "Tecnologia", PHAN (2011) sugere as seguintes questões:

- A equipe de desenvolvimento tem bastante experiência com a linguagem de programação?
- Os membros da equipe de desenvolvimento possuem bastante experiência a tecnologia a ser empregada?
- O ambiente de produção do *Scrum* já está pronto?

7.7.3.4.5 Equipe

Sobre a "Equipe", o autor sugere a avaliação dos itens abaixo:

- O *Scrum* é completamente novo para a equipe?
- Os membros da equipe já trabalharam juntos e com sucesso?
- Os membros da equipe se conhecem bem e apreciam uns aos outros?

7.7.3.4.6 Negócio

E finalmente, a dimensão de "Negócio" é avaliada através das seguintes questões:

- Existe um *Product Owner* completamente disponível e engajado com a equipe?
- O *Product Owner* está familiarizado com o *Scrum*, mas não possui experiência prática?
- O *Product Owner* já usou *Scrum* anteriormente com sucesso?

7.7.3.5 Clima

É muito importante acompanhar o "Clima" da equipe ao longo do tempo, pois isso propicia um ambiente estável e sustentável, com pessoas felizes no seu trabalho.

A estabilidade melhora a integração e o engajamento dos membros do time, o que por sua vez, contribui para o estabelecimento de um fluxo de entregas contínuo.

Em outras palavras, com um bom clima organizacional, os objetivos da organização tendem a ser satisfeitos mais facilmente.

7.7.3.5.1 Termômetro

Como proposta de avaliação do clima e a evolução do time ao longo do tempo (em aspectos gerais), podemos utilizar a dinâmica do "Termômetro", criada por Ismael Melo.

O Termômetro serve para facilitar restrospectivas, e viabiliza a melhoria continua através de planos de ação que regulam a temperatura do time e tratam itens gerais a evoluir no seu contexto.

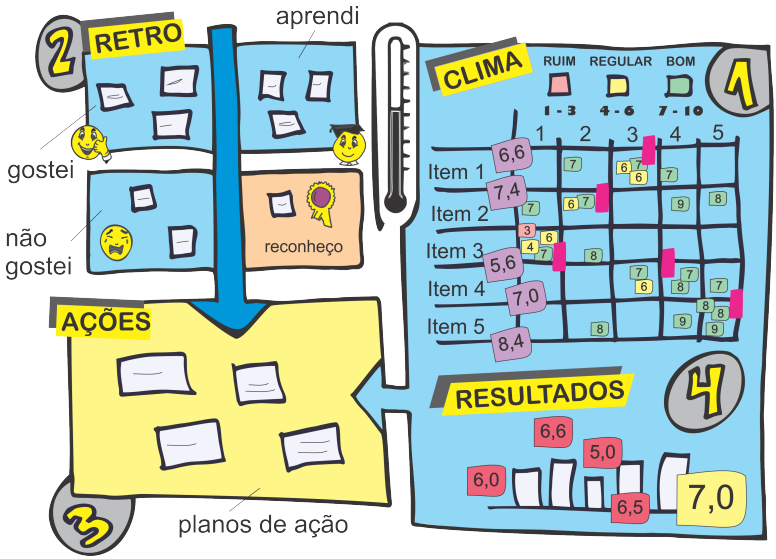


Figura 37 – Retrospectiva do Termômetro

Acompanhe a seguir um passo-a-passo de como a dinâmica funciona:

- 1º - Prepare a cerimônia de retrospectiva, respondendo às seguintes questões:
  - Qual a meta? Dica: item principal a ser tratado, o foco ("o clima", "a melhoria contínua dos processos", "a melhoria contínua do time", etc).
  - Quem vai participar? Dica: apenas o time deve participar, excetuando-se os gestores, sendo que idealmente o

facilitador não deve participar.
  - Quanto tempo? Dica: procure reservar entre 1,5 horas à 3 horas, dependendo do tamanho do time.
  - Onde será? Dica: Numa sala de reunião com espaço para todos.
  - Como a sala será preparada? Dica: agende uma sala com quadro branco ou *flipchart* e com espaço suficiente para a circulação dos participantes.
  - Qual a estrutura básica? Dica: decida em utilizar o desenho do canvas num quadro, flipchart ou outro meio (prepare previamente ao início), leve canetas para todos, *post-its* coloridos e outros recursos adicionais.
- 2º - Defina as características que gostaria de avaliar referentes ao clima e/ou motivação (isso deve ocorrer numa reunião separada, utilizando um modelo de dinâmica da sua preferência).
- 3º - Inicie a dinâmica, explicando como ela funciona (caso todos já conheçam, apenas relembre os itens fundamentais), em seguida, resgate planos de ação priorizados na última retrospectiva e veja se e como foram concluídos. Explique que as características elencadas podem ser revisadas sempre que isso for necessário, para melhor representar os fatores que impactam no clima da equipe.
- 4º - Siga sequencialmente as seções do Canvas.
  - *SEÇÃO 1 DO CANVAS*
    * Na SEÇÃO 1 do Canvas, explique ela tem duas funções: fornecer uma visão da característica que o time gostaria de priorizar em relação às ações para melhoria do clima; e também obter uma nota para cada característica para que possamos obter a temperatura no termômetro.
    * Cada participante deverá avaliar seu sentimento relacionado a cada característica presente no Canvas, atribuindo também num *post-it* (VERDE, AMARELO ou VERMELHO) e uma pontuação dentro

do *range* correspondente. Para cada participante é permitido votar apenas uma vez por coluna de prioridade e por linha de característica.
    * Depois da votação, marque com um post-it de outra cor, para cada coluna de prioridade, a característica que mais recebeu votos.
    * Destaque para a equipe a característica onde a prioridade ficou 1, e explique que ao criar nossos planos de ação, em relação ao clima, iremos priorizar esta característica.
    * Para cada característica, calcule a média dos *post-its* e coloque um novo *post-it* para este fim.
- *SEÇÃO 2 DO CANVAS*
    * Deixe livre para quem quiser participar da SEÇÃO 2, diferentemente da SEÇÃO 1, onde a participação de todos é necessária.
    * Peça para voluntariamente os participantes colarem *post-its* referente ao que gostaram, não gostaram e aprenderam. Destaque que eles devem considerar apenas coisas que julgam significativas ao grupo.
    * Peça ainda, que se alguém julgar que algo ou alguém merece reconhecimento, por um trabalho ou utilidade que mereça um prêmio, que coloque um *post-it* em "Reconhecimento". Votos adicionais para uma mesma opinião, podem ser marcados no mesmo *post-it*.
- *SEÇÃO 3 DO CANVAS*
    * Com base no ítem priorizado na SEÇÃO 1 e também nos ítens "Gostei", "Não Gostei" e "Aprendi", pergunte ao time se desejam estabelecer planos de ação relacionados, devendo aqueles que tiverem opiniões colocarem no quadro um *post-it* relacionado.
    * Agrupe itens similares ou iguais.

        * Estabeleça junto ao grupo, no máximo 3 planos de ação a serem tratados (vote).
        * Pergunte ao grupo, quem ficará responsável por cada item (voluntários).
    - *SEÇÃO 4 DO CANVAS*
        * Faça o cálculo da média considerando o resultado numérico de cada característica e considere que este é o resultado da avaliação do clima, ou seja, é a "temperatura do time".
        * Se preferir, faça um gráfico ou cite avaliações anteriores.
- 5º - Encerre a cerimônia relembrando os resultados, assim como os planos de ação. Em seguida, agradeça os participantes.
    - Dica: para registrar o momento, proponha uma *selfie* com o time!
- 6º - Tabule, armazene e mostre os resultados da dinâmica (aos gestores e para o grupo), destacando os planos de ação priorizados.
    - Preferencialmente, os itens pendentes devem ser pequenos o bastante para serem concluídos até o final da próxima *Sprint*).

7.7.4 Consolidado

Este tópico consolida os resultados que podem ser obtidos nos ítens anteriores, fornecendo uma visão alto nível criada para fornecer uma capacidade de acompanhamento necessária aos gestores, especialistas do Escritório de Projetos, analistas de métricas, *agile coaches* e/ou especialistas em metodologias de desenvolvimento de *software*.

7.7.4.1 Do Time

Commo vimos, o desempenho do time é avaliado com base em métricas, que devem ser organizadas em forma de gráfico. São

avaliados, portanto, as seguintes medidas de desempenho:

- Eficácia
  - Atingimento das Metas
  - Qualidade de Código
- Eficiência
  - Custos
  - Produtividade
  - Taxa de Entrega
  - Tempo de Execução
  - Velocidade
- Efetividade
  - Satisfação do Usuário
  - Percepção de Valor
- Maturidade
  - Organização
  - Processos
  - Infraestrutura
  - Tecnologia
  - Equipe
  - Negócio
- Clima
  - Termômetro

Veja um gráfico que contempla esta visão:

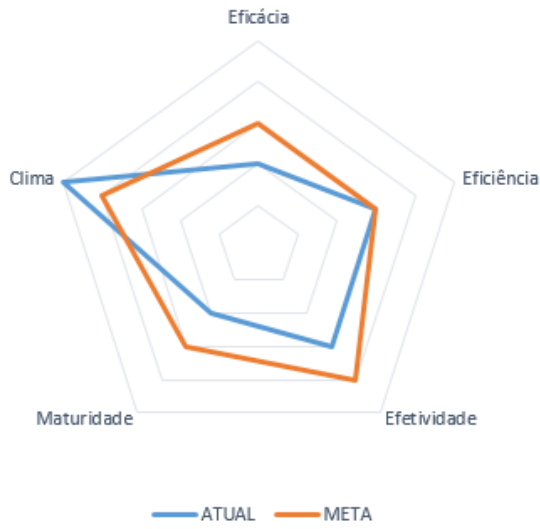


Gráfico 55 – Radar do Timme

Conforme já abordamos no início deste tópico e provavelmente você pode notar no gráfico acima, cada vértice é composto por 6 níveis de avaliação (0 à 5), onde os critérios são pré-estabelecidos conforme a necessidade de cada organização, com base nas medidas disponíveis. Sugere-se que o PMO ou uma área que defina estes direcionadores estratégicos em conjunto com a diretoria de TI e também apoiada por um especialista ou consultoria em métricas. Eles definirão em conjunto com os gestores os parâmetros a serem utilizados, assim como eventuais metas aplicáveis.

7.8 Estratégia de *Sourcing*

Uma organização pode utilizar uma ou mais estratégias de sourcing: *insourcing*, *outsourcing* e modelos híbridos destes.

Existem diversas abordagens de execução e contratação aplicáveis ao contexto em análise, porém consideraremos duas delas neste livro:

- times mistos de desenvolvimento, compostos pelo cliente e fornecedores, os quais compartilham responsabilidades, com um conjunto de atividades que podem ou não serem previamente atribuídas e definidas a cada um;
  - se um conjunto padrão de atividades de desenvolvimento é executado por padrão para um fornecedor, pode-se avaliar a atribuição de preço para a unidade de medida trabalhada;
  - se as atividades são compartilhadas, e por vezes as responsabilidades também, uma abordagem com remuneração horária, utilizando-se medidas de acompanhamento do time para as entregas e individual para comprometimento do fornedor com as entregas mostra-se mais adequada.
- times segregados, onde as responsabilidades e atividades de clientes e fornecedores são específicas de cada um
  - sugere-se avaliar uma precificação unitária, padrão ou definida por Projeto, Épico/Bloco ou ainda por Entrega (MVP).

Mostraremos a seguir, em linhas gerais e com base nas premissas citadas, como você pode definir uma estratégia de *sourcing* considerando a necessidade de composição de times internos com profissionais oriundos de fornecedores de *software*.

Também será abordado um modelo paralelo, visando prover escalabilidade para o trabalho dos times.

7.8.1 Gestão de fornecedores

A gestão dos fornecedores de TI deverá ficar a cargo de um processo de aquisição consistente e versátil, preparado para diferentes usos. No entanto, destacamos que como este estudo compreende basicamente atividades em torno de desenvolvimento de *software*, utilizaremos principalmente conceitos e definições relacionados como principais referências, e dentre estas fontes destacamos os guias do SISP, já abordados nos capítulos anteriores.

7.8.1.1 Cadastro e habilitação

Seja em um modelo de um ou múltiplos fornecedores, cabe a avaliação de requisitos para habilitação dos fornecedores/ parceiros os quais suportarão a operação. Tais requisitos referem-se à análise cadastral, da capacidade financeira e riscos associados.

7.8.1.2 Homologação e contratação

A homologação idealmente deve ser conferida a um ou mais fornecedores habilitados dado um contexto tecnológico e de negócio, por ventura cobrindo o escopo de um ou mais produtos. O detalhamento necessário deve ser fornecido pelo requisitante através de um Termo de Referência, documento a ser anexado ao contrato.

O fornecedor então precificará os perfis solicitados no termo de referência e dará aceite às condições, apresentando eventuais comprovações necessárias (por exemplo, atestados de capacidade técnica), e conferindo o aceite às políticas internas.

O próximo passo é a assinatura do contrato. No entanto, mesmo que homologado, não necessariamente um fornecedor formalmente contratado terá garantias de execução contratual para uma contratação específica, mas ele estará apto para que isso rapidamente possa ser feito e formalizado através de um instrumento contratual (uma Ordem de Serviço), agilizando a questão burocrática relacionada.

7.8.2 Execuções contratuais

O modelo de execução contratual pode ser bastante simplificado. Basicamente, ele é estruturado com base na execução de Ordens de Serviço, um tipo de instrumento que estende as definições de um contrato principal firmado entre as partes (contratante e contratada).

Tipos de ordens de serviço diferentes devem ser previstos para possibilitar a contratação de serviços pontuais, tais como aqueles

definidos para o transbordo e/ou que referem-se ao desenvolvimento ou consultoria em *software*.

Sugerimos, portanto, a elaboração de pelo menos dois tipos de OS:

- Contratação Pontual de Perfis;
- Contratação Pontual de Desenvolvimento.

Enquanto a primeira modalidade contempla a contratação de um ou mais profissionais em um modelo de alocação (independente de quem gestiona os recursos), a segunda destina-se à contratação via estimativa de custo da unidade base padrão, ou seja, o custo unitário do ponto de função.

7.8.2.1 Estabilidade com *Insourcing*

Embora possa limitar a quantidade de recursos disponíveis, o *insourcing* remete um ambiente mais econômico e estável, desde que bem planejado e gerido. Pode, no entanto, limitar a capacidade de atendimento das necessidades da organização, o que atualmente, dependendo da linha de negócio, poderia representar dificuldades em competir no mercado.

Em grandes empresas, uma estratégia de *sourcing* mista parece mais adequada para a maioria dos casos, considerando que times mistos serão compostos e pontualmente o que fugir à capacidade deste é remetido para execução externa.

7.8.2.2 Escalabilidade com *Outsourcing*

Uma estrutura balizada por *outsourcing* exige custos adicionais de controle e qualidade, mas em contrapartida, provê escalabilidade das soluções. Neste caso, até onde existirem recursos e capacidade de gestão, o modelo poderá ser escalado, desde que existam recursos perenes ao menos para realizar a transição e o acompanhamento necessário.

Se existirem ferramentas de apoio para a contratação de seus projetos, sugerimos inclusive, homologar os fornecedores habilita-

dos para cada termo de referência, estabelecendo um custo perfil para consultoria pontual e prevendo mini licitações (uma janela de 5 dias para a concorrência) com base na contratação vinculada ao termo de referencia homologado. Esta é uma opção plausível inclusive às organizações públicas, porém necessita ser avaliada em termos de operacionalização (a Homologação do Fornecedor talvez tenha que ser aprimorada).

7.8.3 Gestão de contratos

Após a execução dos serviços prestados num determinado período, normalmente em intervalos mensais, os mesmos devem ser declarados pelo fornecedor, contemplando medição do nível de serviço atingido e outros dados úteis para sua validação. O ciclo encerra com a efetivação do pagamento ao fornecedor e com a publicação de informações aos executivos e demais envolvidos na contratação. Deve ser possível, por exemplo: homologar, classificar e *rankear* os fornecedores contratados; isso através de mecanismos transparentes e alinhados às práticas de cada organização.

7.8.3.1 Declaração de serviços

A declaração dos serviços prestados pelo fornecedor deve conter informações num nível suficiente para possibilitar a sua validação, mas também, para que possa ser auditável pelas partes, ou ainda, por entidade idônea destacada para este fim (de modo que se possa comprovar que foram executados).

É considerado boa prática, considerar os serviços declarados como insumo da apuração de nível de serviço, o que otimiza esforços e diminui custos de transação relacionados.

A declaração de serviços para a contratação de perfis dos times perenes, deve idealmente ocorrer por Ordem de Serviço (OS), uma por *Sprint*. Também deverá conter a relação de perfis necessária e um respetivo esforço aproximado. Ao final de cada *Sprint*, ocorre a atualização destes parâmetros, com base no que foi efetivamente realizado.

Sobre a contratação de serviços no regime de fábrica, no entanto, sugere-se realizar uma estimativa inicial do MVP (que prevê parâmetros para o cálculo de custo e duração), que servirá para acordar uma linha de base de escopo, assim como uma premissa de desvio máximo do orçamento e de prazo máximo para a sua entrega, também através de Ordem de Serviço (OS). Ao final do trabalho, na entrega do MVP, realiza-se a atualização da Ordem de Serviço, agora com base na medição final e nos parâmetros comerciais pré-acordados.

O relatório de serviços da contratada, possuirá então, uma relação de times e os serviços aplicáveis ao mesmo, na forma de Ordens de Serviço (OS), com diferentes classificações.

7.8.3.2 Nível de serviço

A apuração do nível de serviço do fornecedor contratado dependerá diretamente do demonstrativo de serviços prestados e/ou da massa de dados que a organização utilizará para validação.

Devem ser previstos neste fluxo, mecanismos de ajuste e expurgo que apoiarão uma possível revisão; e também um relatório do mês e o histórico anual para publicação frente aos demais interessados por esta informação na organização.

Em relação aos indicadores contratados, é importante diferenciá-los para cada contexto e forma de trabalho, assim como para os tipos de serviços que serão prestados. Assim, no mínimo, sugere-se segregar conjuntos de indicadores para as seguintes situações:

- Na contratação de perfis para formação/ composição de times perenes (ex.: *Turnover*)
- Na contratação de fábrica de software, em modo "transbordo" (ex.: escopo planejado *versus* entregue, atendimento no prazo, qualidade das entregas, etc.)

Podem ser criados indicadores no contrato, tanto para efeito de penalização como para bonificação, ou ainda, indicadores informativos. Idealmente, eles devem ser publicados através de informativos

sobre a qualidade dos serviços prestados. E normalmente, eles são ponderados para a composição de uma nota final.

7.8.3.3 Prestação de contas

O pagamento executado aos fornecedores é sensibilizado pelos serviços executados e eventuais penalizações ou bonificações realizadas.

Este fluxo deve ser acionado no mês subsequente à execução dos dos serviços e é constituído por um relatório que especifica uma Nota Fiscal, as Ordens de Serviço (OS) e as despesas, bonificações e penas aplicáveis, contemplando ainda o seu vínculo com o requisitante e alçada competente pela aprovação do pagamento.

Ao final desta consolidação, o relatório em questão deve ser enviado para aprovação do pagamento pela alçada de aprovação. E desta forma, proceder-se-à o pagamento ao fornecedor.

8. Dicas & Macetes

Neste capítulo mostraremos o que você pode fazer para focar no seu objetivo mantendo os custos controlados.

8.1 Trabalhando com custos

Antes de aprimorar sua visão sobre custos, tenha em mente duas coisas:

- Despenda dinheiro no que gera valor, mas não olhe somente para o hoje (vislumbre também o amanhã);
- Saiba valorizar o que te dá visibilidade sobre a operação.

A dica básica aqui é não cair no conto do *business case*, onde você investe no fornecedor de *software* ou no projeto apenas olhando para o retorno financeiro do projeto. E a dica de ouro é encontrar o custo unitário para cada contexto de negócio e tecnologia, preferencialmente segregando visões sobre seus custos de desenvolvimento, qualidade e transação. Se conseguir fazer isso, suas estimativas iniciais e futuras tendem a ser mais precisas e seu controle de custos será maduro, eficaz, transparente e o mais interessante: comparável (ao longo do tempo e com o mercado).

8.2 Tercerizando a análise de qualidade

Tercerização da qualidade, a depender da criticidade do negócio em que sua empresa atua, pode ser uma boa pedida. Esta abordagem pode ser até mesmo econômica para ajudar a homologar as entregas de *software* e/ou artefatos gerados pelos fornecedores.

A dica básica aqui é manter ao menos um processo mínimo, auditado preferencialmente por um consultor externo. E a dica de ouro é: utilize ferramentas de varredura de código que façam

parte do trabalho de qualidade por você, para otimizar os custos do serviço (por exemplo: Sonar).

8.3 Tercerizando estimativas

Assim como a qualidade pode ser terceirizada, as métricas obtidas também podem ser avaliadas ou geradas por times externos, conforme a necessidade, como contratações pontuais ou com pessoas alocadas, englobando estimativas para o desenvolvimento de demandas futuras, medições do presente e passado recente, além de avaliações e calibrações matemáticas ou mesmo estatísticas do histórico.

A dica básica é ter alguém qualificado que lhe atenda, preferencialmente com certificações relacionadas ao método de dimensionamento e conhecimento em um ambiente ágil.

A dica de ouro é ter um profissional dedicado e altamente qualificado em métricas, que seja capaz de gerar visões analíticas e criar indicadores de performance com os dados que sua empresa produz. Um *Scrum Master* pode ser empoderado do conhecimento necessário em métricas, com o intuito de acompanhar e prover acompanhamento a quem necessita, inclusive com um viés de melhoria contínua.

8.4 Ferramentas para estudo dos dados

Caso você conheça a linguagem de desenvolvimento *R* (linguagem criada por estatísticos para estatísticos), aplicar métodos matemáticos e estatísticos e plotar seus gráficos em tela pode ser facilitado com a utilização da ferramenta RStudio (uma IDE). Considere também instalar os plugins *RServe* e *Shiny* (*web server*), disponíveis para instalação através do terminal de linha de comando do *RStudio.*

Desta forma, você poderá codificar soluções que contribuem para o desenvolvimento de um ambiente dinâmico de amostras e

análises científicas, realizando por exemplo:

- Correlação simples entre variáveis (exemplo: tamanho *versus* esforço);
- Dispersão de valores (exemplo: tamanho *versus* produtividade);
- Histograma de produtividade dos times (exemplo: produtividade do *MVP* por time);
- Regressão linear simples (exemplo: analisando o quanto se pode explicar o esforço através do tamanho dada uma fórmula matemática sugerida).
- Regressão linear múltipla (exemplos: usando APF IFPUG, para descobrir constantes a serem consideradas para cada tipo de deflator; usando APF IFPUG e SNAP, buscando definir uma constante de produtividade a ser utilizada como referência para cada um dos métodos).

Além disso, a capacidade de representar gráficos complexos e a capacidade de servidor adquirida através do *RServe* e Shiny*, permite a criação de *Dashboards* online, os quais poderão ser disponibilizados para acesso por outros usuários da organização.

9. Considerações Finais

Nós esperamos que este material realmente tenha sido útil para as suas pretensões. E para finalizar, gostaríamos de ajudá-lo a refletir sobre o que realmente significam os principais termos que citamos durante este livro:

9.1 Ser "Ágil"

No contexto que apresentamos, ser ágil é fazer algo com "eficácia" e preocupando-se em otimizar o uso dos recursos disponíveis em prol do trabalho ("eficiência") que realmente agrega valor a quem o espera e percebe ("efetividade"), permitindo-se errar para obter um melhor aprendizado ("maturidade", mas sempre corrigindo sua direção o mais cedo possível e mantendo o clima da organização estável ("clima").

9.2 Ser "Responsável"

Em se tratando de desenvolvimento de *software* e também seus projetos e contratações relacionadas, ser responsável é planejar e acompanhar onde você se enxerga, em relação a você mesmo e ao mercado. É saber também onde você pretende chegar e o que precisa mexer ou mudar para evoluir, ou ainda, para conseguir manter-se vivo.

Ser responsável também é ser imparcial, buscando manter relações profissionais e pessoais ganha-ganha. Seja com seu colega de trabalho ou na interface comercial com outra organização, você deve buscar sempre o melhor resultado para manter sua relação de pé.

Responsabilidade é ainda, não ser fundamentalista em relação a nenhum método em que você acredita, mas realmente saber

discernir de mente aberta o que realmente é bom e o que é ruim, e também aquilo que faz sentido e agrega valor ao usuário e para a sustentabilidade do ecossistema de desenvolvimento.

9.3 Ser "Enxuto"

Ser enxuto é focar no que agrega valor, sem fazer estoque; e fazer o trabalho certo, no momento onde ele se fizer necessário, e apenas nesse. É também saber avaliar o que faz sentido e o que não faz, eliminando deliberadamente qualquer tipo de desperdício. Para reconhecer um desperdício, no entanto, não basta percepção: você precisa demonstrar eficiência! E é aí que os nossos números são bem vindos! Contra a matemática e a estatística, não há argumentos...

Bibliografia

- ABRAN, Alain. *Software Metrics and Software Metrology*. IEEE, 2010.
- ALBRECHT, Allan J. "Medindo a Produtividade do Desenvolvimento de Aplicativos." FattoCS. 1979. http://www.fattocs.com.br/artigos/MedindoaProdutividadedo DesenvolvimentodeAplicativos.pdf (acesso em 13 de Dez de 2015).
- AMARAL, Daniel Capado et al. Gerenciamento ágil de projetos: aplicação em produtos inovadores. São Paulo: Saraiva, 2011.
- ARCHIBALD, R. D. *Managing High-Technology Programs and Projects*, 3ª Edição. New York: John Wiley & Sons, 2003.
- ASTELS, D. et. al. *eXtreme Programming* – Guia Prático. Rio de Janeiro: Campus, 2002.
- AUDY, Jorge L. N., e Rafael PRIKLANDNICKI. Desenvolvimento Distribuído de *Software*. Rio de Janeiro: Elsevier, 2007.
- AUGUSTINE, S. *Managing agile Project*. Virgínia: *Prentice Hall PTR*, 2005.
- BECK, Kent et al. *Manifesto for agile software development*. 2001. http://www.agilemanifesto.org/ (acesso em 08 de Nov de 2015).
- BECK, Kent et. al. *Planning Extreme Programming*. 1. ed. Boston: Addison-Wesley, 2001.
- BECK, Kent. *Extreme Programming explained: embrace change*. 1. ed. MA: Addison-Wesley, 2000.
- CAROLI, Paulo. Direto ao Ponto: criando produtos de forma enxuta. São Paulo: Casa do Código, 2015.
- CAROLI E CAETANO, Paulo, Tainã. *Fun Retrospectives*. LeanPub 2015.
- COHN, M. Agile Estimating and Planning. Massachusetts: Pearson Education, 2005.

- COSMIC. *Benchmark: measure of performance*. s.d. http://cosmic-sizing.org/ cosmic-fsm/benchmarking/ (acesso em 03 de Janeiro de 2016).
  - —. *The COSMIC Functional Size Measurement Method, Version* 4.0.1. *The Common Software Measurement International Consortium*, 2015.
- CRUZ, Fábio. *Scrum* e PMBOK: unidos no gerenciamento de projetos. Rio de Janeiro: Brasport, 2013.
- DEKKERS, Carol. "*Counting Function Points for Agile / Iterative Software Development*." IFPUG. s.d. http://www.ifpug.org/ Articles/ Dekkers-CountingAgileProjects.pdf (acesso em 02 de Janeiro de 2016).
- FRANCO, E. F. Um modelo de gerenciamento de projetos baseado nas metodologias ágeis de desenvolvimento de *software* e nos princípios da produção enxuta. Dissertação de Mestrado, São Paulo: Escola Politécnica da Universidade de São Paulo, 2007.
- GARTNER. *Gartner Says Adopting a Pace-Layered Application Strategy Can Accelerate Innovation*. 2012. Disponível em: http://www.gartner.com/newsroom/id/1923014/ (acesso em 10 de Nov de 2015).
- GIL, Antonio Carlos. Como Elaborar Projetos de Pesquisa. 5ª ed. São Paulo: Atlas, 2010.
- GIL, Antonio Carlos. Como Elaborar Projetos de Pesquisa. 4. ed. – São Paulo: Atlas, 2002
- GODOY, A. S. Introdução à pesquisa qualitativa e suas possibilidades. Revista de Administração de Empresas, v. 35, n. 2, p. 57-63, 1995.
- GOMES, Andre F. Desenvolvimento de *software* com entregas freqüentes e foco no valor de negócio. São Paulo: Casa do Código, 2013.
- HIGHSMITH, J. *Agile Project Management: creating innovative products*. Boston: Addisson - Wesley, 2004.

  - —. *Agile Software Development Ecosystems*. Boston: MA: Addison, 2002.
- IFPUG. *Function Point Counting Practices Manual*(CPM)". Release 4.3. *New Jersey: International Funciotn Point Users Group*, 2010.
  - —. Processo de Avaliação Não Funcional (SNAP)". *Release* 2.2. *New Jersey: International Function Point Users Group*, 2015.
- ISBSG. *Reifer Consultants LLC and ISBSG Joint Report: Software Productivity Benchmark*. Técnico, Victoria (AUS): ISBSG, 2013.
- JONES, C. *Estimating Software Costs: Bringing Realism to Estimating*, 2ª *Edition. New York*: McGraw-Hill, 2007.
- LEVINE, H.A. *Project Porfolio Management*, Jossey-Bass. San Francisco (USA), 2005.
- PHAM, Andrew. Scrum em ação: gerenciamento e desenvolvimento Ágil de projetos de *software* / Andrew Pham e Phuong-Van Pham [tradução Edgard B. Damiani]. São Paulo, Novatec Editora: *Cangage Learning*, 2011.
- PMI. *A Guide to the Project Management Body of Knowledge – PMBOK® Guide 2000 Edition.* Pennsylvania-USA: *Project Management Institute* (PMI), 2000.
- POPPENDIECK, Mary et al. Implementando o Desenvolvimento *Lean* de *Software*: do conceito ao dinheiro. Porto Alegre: Bookman, 2011.
- PRESSMAN, Roger S. Engenharia de *Software*. 6. ed. Porto Alegre: McGraw-Hill, 2006.
- RIES, Eric. A *Startup* Enxuta. São Paulo: Lua de Papel, 2012.
  - —. *The Lean Startup*. USA: Crown Business, 2011.
- ROESCH, Sylvia Maria Azevedo. Projetos de Estágio e de Pesquisa em Administração: guia para estágios, trabalhos de conclusão, dissertações e estudos de caso. 2 ed. São Paulo: Atlas, 1999.

- SABBAGH, Rafael. *Scrum*: gestão ágil para projetos de sucesso. São Paulo: Casa do Código, 2013.
- SATO, Danilo. *DevOps* – na prática: entrega de *software* confiável e automatizada. São Paulo: Casa do Código, 2013.
- SCHWABER, Ken. Guia do *Scrum. Scrum Alliance*, 2009.
- SILVA, L.V., L. MACHADO, A. SACCOL, e D. AZEVEDO. Metodologia de Pesquisa em Administração: uma abordagem prática - Coleção EAD. São Leopoldo: Editora Unisinos, 2012.
- SISP. Guia de Boas Práticas em Contratação de Soluções de Tecnologia da informação, V 2.0. Brasília: Ministério do Planejamento, Secretaria de Logística e TI, 2014.
  - —. Metodologia de Gerenciamento de Portfólio de Projetos do SISP. Brasília: SLTI, 2013.
  - —. Metodologia de Gerenciamento de Projetos do SISP, versão 1.0. Brasília: MP-SISP, 2011.
  - —. Roteiro de Métricas de *Software* do SISP: versão 2.1. Brasília: Ministério do Planejamento, Orçamento e Gestão, 2015.
- SOFTEX. MPS-BR - Melhoria de Processo do *Software* Brasileiro: Guia de Aquisição. SOFTEX, 2013.
- TCU. TC 010.663/2013-4: conhecimento acerca da utilização de métodos ágeis nas contratações… Levantamento de Auditoria, Brasília: Tribunal de Contas da União, 2013.
- TERRIBILI FILHO, Armando. Indicadores de Gerenciamento de Projetos: monitoração contínua. São Paulo: M. Books do Brasil Ltda, 2010.
- VAZQUEZ, C. E., G. S. SIMÕES, e R. M. ALBERT. "Análise de Pontos de Função: Medição, Estimativas e Gerenciamento de Projetos de *Software*. 9ª Edição. São Paulo: Editora Érica, 2010.
- VERGARA, S. C. Métodos de pesquisa em administração. São Paulo: Atlas, 2010.
- VIANNA, Maurício et al. *Design thinking*: inovação em negócios. Rio de Janeiro: MJV Press, 2012.

- YIN, Robert K. Estudo de Caso: Planejamento e Métodos. Tradução Daniel Grassi. 2 ed. Porto Alegre Bookman, 2001
- YIN, Robert K. Estudo de Caso: Planejamento e Métodos. Tradução Daniel Grassi. 3 ed. Porto Alegre Bookman, 2005
- WOMACK, J., Jones, D. A mentalidade enxuta nas empresas (*Lean Thinking*), 1ª Edição. São Paulo: Editora Campus, 2004.

Termos e Definições

- *Adhoc* - Solicitação e eventos para manutenção ou criação de funcionalidades.
- Agilista - alguém que pratica e/ou promove a metodologia ágil.
- AIE (Arquivo de Interface Externa) - Agrupamento de dados mantido externamente a um sistema, mas referenciados por seus processos para fins de consulta e/ou validação (vide também o significado de APF).
- ALI (Arquivo Lógico Interno) - Agrupamento de dados que são mantidos por uma aplicação, conforme a percepção do usuário e de acordo com as regras definidas pela APF.
- APF (Análise de Pontos de Função) - Um método para medição funcional de aplicações de *software*.
- *Burn-Down* - Tipo de gráfico onde são apresentados dados e tendências de progresso, na perspectiva do que falta ser feito. Em outras palavras, ele representa a quantidade de trabalho que falta ser feito no eixo vertical (y) *versus* o tempo no eixo horizontal (x).
- *Burn-Up* - Diferentemente do gráfico *burn-up*, este tipo de gráfico representa o que já foi feito em relação ao que havia planejado. Desta forma, a linha sob o eixo vertical (y), que representa as unidades de trabalho, inicia na marcação mínima e avança em direção aos valores planejados.
- Calibração - Ação que possibilita a obtenção de dados de referência a partir de análises matemáticas e/ou estatísticas.
- CE (Consulta Externa) - Tipicamente uma funcionalidade (processo elementar) responsável por enviar dados para fora da fronteira da aplicação, contada na APF desde que seja reconhecida pelo usuário e que não possua geração de dados derivados ou cálculos, e também desde que não mantenha

quaisquer dados não técnicos na aplicação durante sua execução.

- Ciclo de Desenvolvimento - Conjunto de etapas ou fases e respectivas atividades vinculadas e que são executadas para produzir *software*.
- *Change Request* - Mudança de requisitos realizada em um item de *software* previamente iniciado, em decorrência de mudanças nos requisitos.
- CPM (*Counting Practices Manual*) - É o manual que define e convenciona as regras de contagem que devem ser aplicadas durante uma contagem de pontos de função (APF).
- Cronograma - Artefato relacionado à disciplina de gerenciamento de projetos, o qual representa o plano de trabalho de um projeto.
- Custo - Valor despendido por alguma coisa.
- *Dashboard* - Um quadro de gráficos e/ou indicadores que facilitam a visibilidade e o acompanhamento sobre a operação.
- Demanda - Uma necessidade que precisa ser desenvolvida e entregue ao seu solicitante.
- Deflator - Algo como um índice usado para o cálculo do impacto, de acordo com o tipo de mudança sofrido por uma função (na APF do IFPUG).
- Diretrizes de Contagem - Regras que norteiam questões não esclarecidas totalmente em modo nativo, para a aplicação de métricas de *software* em contextos especificos tecnológicos ou de negócio.
- Duração - O prazo, ou tempo necessário para o desenvolvimento de uma ou mais necessidades.
- EE (Entrada Externa) - Um processo elementar cuja intenção é manter dados dentro de uma aplicação (vinculado ao termo APF).
- Equipe - Um time ou grupo de pessoas organizadas para trabalharem em conjunto ou sob regras/contextos similares.

- Escritório de Métricas - Uma estrutura física ou conceitual especializada em métricas de *software*.
- Escritório de Projetos - Estrutura física ou conceitual especializada em gestão de projetos.
- Esforço - Uma quantificação em horas necessárias para produzir um determinado resultado.
- Especialista em Métricas - Um profissional que atua com métricas de *software*, por ventura certificado em um ou mais métodos através de exames reconhecidos internacionalmente.
- Estória (ou *User Story*) - Uma descrição do que precisa ser feito pela perspectiva da necessidade do usuário.
- Fábrica de Métricas - Estrutura física ou conceitual para atendimento remoto de demandas de contagem ou consultoria em métricas.
- Fábrica de Qualidade - Estrutura física ou conceitual para atendimento remoto de demandas de testes ou consultoria em testes.
- Fábrica de *Software* - Estrutura física ou conceitual para atendimento de demandas de desenvolvimento ou consultoria em desenvolvimento de *software*.
- *Feature* - Recurso ou macrofuncionalidade do produto.
- Fornecedor - Empresa contratada para um serviço e/ou para prover/habilitar um produto ou resultado.
- Fronteira - No contexto da APF, uma interface conceitual entre o que é externo e o que é interno à uma aplicação.
- Funcionalidade - No contexto da APF, um processo elementar.
- IFPUG (*International Function Points User Group*) - Grupo de usuários de pontos de função que constitui um órgão internacional que é responsável pelos métodos APF e SNAP, definindo suas regras de contagem.
- *Insourcing* - Estratégia de balizada pela contratação de recursos humanos internos.

- IP (Índice de Produtividade) - Estimativa de horas que são necessárias para produzir uma unidade de tamanho.
- Iterativo-Incremental - Modelo de desenvolvimento orientado pela execução de entregas parciais, com entregáveis completos a cada rodada.
- *Kanban* - Nome do método e também do quadro de tarefas com foco em um sistema de produção puxada.
- *Just-In-Time* - Sistema de produção adequado à demanda.
- *Lead Time* - O tempo de atravessamento, ou seja, o tempo deste que algo começa a ser produzido até a efetiva entrega.
- *Lean* - Modelo enxuto de produção, utilizando o conceito de *just-in-time*.
- Marco - Momento da linha do tempo significativo para os *stakeholders*, seja oriundo de uma iniciativa ou projeto.
- Metodologia - De acordo com o dicionário Aurélio, no campo da literatura, metodologia é definida como o conjunto de técnicas e processos utilizados para ultrapassar a subjetividade do autor e atingir a obra literária. Já o dicionário *Webster*, no ramo da computação, conceitua metodologia como um conjunto organizado de procedimentos e guias documentados para a execução de uma ou mais fases do ciclo de vida do *software*.
- Métodos Ágeis - Métodos que privilegiam a entrega frequente com proposição de valor ao negócio do usuário.
- Métricas de *Software* - Métodos de dimensionamento utilizadas para viabilizar estimativas e a medição de *software*.
- Minuta Contratual - Rascunho ou versão inicial de um contrato, e portanto, de onde origina-se o mesmo.
- MVP (*Minimum Viable Product*) - O produto mínimo que é viável para o negócio, ou seja, o conjunto de *Features* que serão construídas ou customizadas para a entrega de valor ao seu produto e usuário.
- NESMA - Organização holandesa que define diretrizes de dimensionamento.

- NMS (Nível Mínimo de Serviço) - Nível mínimo de serviço acordado numa relação gerenciada de terceirização.
- *Outsourcing* - O sinônimo de terceirização.
- PMBok - O guia que consolida boas práticas para gerenciamento de projetos conforme propostas pelo órgão PMI.
- Portfólio de Projetos - Um conjunto de projetos e/ou itens de entrega de uma ou mais áreas da TI.
- Prazo - A duração de uma atividade ou Projeto.
- Projeto - Um conjunto de atividades coordenadas para obtenção de um ou mais resultados.
- *Release* - Uma versão de um conjunto de código que forma o *software*.
- Retrabalho - Trabalho adicional sobre algo já antes considerado completo ou definido.
- RFI (*Request for Information*) - Uma requisição formal de informações de uma organização para o mercado.
- RFP (*Request for Proposal*) - Uma requisição para obtenção de propostas comerciais ou técnicas e comerciais do mercado.
- RFQ (*Request for Quotation*) - Uma requisição ou tomada de preços de empresas do mercado para um determinado serviço.
- *Scrum* - *Framework* ágil para gestão de projetos de *software*.
- *Sprint* - Iteração.
- SE (Saída Externa) - Um processo elementar cuja intenção é enviar dados para fora da fronteira da aplicação e que possui derivação ou manutenção de dados em sua lógica de processamento, conforme definição fornecida pelo IFPUG no método Análise de Pontos de Função (APF).
- SNAP (*Software Non Functional Process*) - Métrica utilizada para medição do tamanho não funcional, conforme definição das regras elaboradas pelo órgão IFPUG.
- Tamanho Funcional - A quantificação dos requisitos funcionais do usuário.

- Tamanho Não Funcional - Uma medida não funcional, ou seja, a quantificação dos requisitos não funcionais do usuário.
- Time - Uma equipe.
- Visão de Produto - O que o produto deve possuir em termos de recursos.
- *Waterfall* - Uma das mais conhecidas metodologias de desenvolvimento de *software*, termo também referido como cascata (tradução literal), clássico ou linear.
- WIP (*Work in Progress*) - O trabalho em andamento, ou o limite definido para adequá-lo a capacidade de execução.