MINISTÉRIO PÚBLICO
DO ESTADO DA BAHIA

Projeto de Padronização do Processo de Desenvolvimento de Software

Modelo de Gestão de Sustentação Ágil de Software

Versão 0.1

Histórico de Revisões

Data	Versão	Descrição	Autor
30/01/2018	0.1	Proposta Inicial.	UDS

Índice Analítico

1. Introdução

Devido à necessidade constante das organizações por softwares, é comum que estas desenvolvam suas próprias soluções, investindo, para isso, muito de seus recursos. Diante disso, é natural que estas organizações tenham a expectativa de que os softwares produzidos, durante sua operação, entreguem-lhes valor suficiente que justifique o investimento nele feito.

Neste cenário, é interessante para as organizações estabelecer mecanismos para que os softwares desenvolvidos continuem entregando valor por maior período possível, sendo a formação de equipes de manutenção de software (sustentação) uma forma bastante comum de tratar esta questão.

Ao mesmo tempo, metodologias ágeis de desenvolvimento de software têm demonstrado, nos contextos apropriados, bons resultados no que diz respeito à qualidade da entrega ao cliente e resposta rápida a mudanças no sistema. Estes resultados despertam o interesse das organizações na tentativa de melhor responder as mudanças, tanto nos sistemas em desenvolvimento quantos nos sistemas que estão em sustentação.

Este documento descreve o Modelo de Gestão de Sustentação Ágil de Software (MGSAS), instrumento da Diretoria de Tecnologia da Informação para abordar a sustentação de software no âmbito do Ministério Público do Estado da Bahia, considerando as questões de agilidade defendidas pelas metodologias ágeis.

2. Referências

- Modelo de Gestão de Desenvolvimento de Software (MGDS);
- Processo de análise de mudança em sistema em sustentação;
- Processo de atualização de sistema em sustentação;
- Processo de atualização emergencial de sistema em sustentação;
- Manifesto ágil para desenvolvimento de software, disponível em http://www.manifestoagil.com.br.

3. Definições, acrônimos e abreviações

Backlog. Conjunto de solicitações de mudança à espera de serem tratadas pela equipe de sustentação. São mantidas dentro do SGD.

DTI. Diretoria de Tecnologia da Informação do MPBA.

Equipe de Sustentação. Pessoas responsáveis por realizar a sustentação de sistemas.

MPBA. Ministério Público do Estado da Bahia.

Mudança. Alteração no comportamento ou funcionalidade atual do software ou inclusão de novas funcionalidades.

SGD. Sistema de Gestão de Demandas.

Sprint. Período de duas a três semanas de atividades em que um escopo de mudanças é selecionado e as atividades necessárias para implementá-la são executadas, culminando em uma nova versão do sistema.

Sustentação. Processo contínuo de monitoramento e análise de solicitações de mudanças, bem como a aplicação das mudanças (manutenção) através nos mecanismos e métodos apropriados.

4. Scrum e Práticas Ágeis

Embora constitua a base de várias práticas voltadas a agilidade, **Scrum não é ágil**, mas um conjunto simples de práticas para resolver problemas.

Com isso em mente, entre as principais práticas de mercado que apoiam o Manifesto Ágil (vide Apêndice A – O Manifesto Ágil), podemos citar **Kanban** e **XP**. Cada uma delas possui uma filosofia e vários pontos positivos de cada uma podem e devem ser reutilizados.

Caso o leitor já possua familiaridade com Scrum e as demais práticas, recomendamos seguir à seção Modelo de Gestão de Sustentação Ágil de Software, senão uma breve apresentação das práticas é exposta a seguir.

4.1. Scrum

Scrum é definido como *um framework no qual as pessoas podem abordar problemas complexos e adaptativos enquanto entregam produtos do maior valor possível produtiva e criativamente* [1].

As pessoas exercem os seguintes papéis no Scrum:

- Product Owner (PO) – é o dono do produto, o patrocinador, portanto quem determina o que tem maior **valor** e **prioridade** entre os itens do produto. Sua atividade principal é manter o **backlog de produto** atualizado (ainda que transfira a atividade, ele é o responsável!). Deve ser **uma só pessoa** (nunca um comitê), de modo que se saiba a todo tempo a quem se reportar sobre o produto.
- Equipe de Desenvolvimento – o conjunto de profissionais que desenvolve e **entrega incrementos** Prontos (definição abaixo) do produto. A equipe é **auto-organizada, auto-gerenciada e multifuncional**.

- Scrum Master – é o guardião principal do Scrum, e possui diversas responsabilidades, entre elas: **auxiliar** o PO a saber como manter o backlog de produto atualizado, **facilitar** as atividades do processo e promover treinamento, **apoiar** a Equipe para obter itens de backlog claros.

O elemento central do Scrum é o **Sprint**, período de um mês ou menos no qual um incremento "Pronto" (definição de quando um item do produto está completo), usável e potencialmente implantável é criado. Dentro do sprint, algumas atividades essenciais são realizadas.

4.1.1. Planejamento da Sprint

Aqui todos participam visando determinar **o que pode ser entregue** na Sprint e **como** isto será feito. A partir do backlog do produto, selecionam-se itens para serem desenvolvidos. Esta atividade deve durar, no máximo, **oito horas**.

- O PO **determina o objetivo da Sprint** e que itens do backlog o satisfazem;
- A equipe **indica quantos e quais itens dentre os determinados** podem ser entregues para alcançar o objetivo;
- A equipe **estabelece a meta da Sprint** (os objetivos a serem alcançados pelo time) a qual tende a ser menor que o objetivo da Sprint.
- A equipe **define a forma de trabalho**, esclarecendo dúvidas com o PO (e outras pessoas com conhecimento para tal).

4.1.2. Reunião Diária

Em até quinze minutos, **todos os membros do time devem responder**:

- O que eu fiz ontem?
- O que farei hoje?
- Existe algum obstáculo para a minha atividade?

4.1.3. Desenvolvimento

É conduzir as atividades para alcançar o objetivo da sprint.

4.1.4. Revisão da Sprint

É um encontro informal, de até quatro horas, entre o time, o PO e pessoas de interesse do projeto para **apresentar e discutir o que está Pronto**, bem como reavaliar o Backlog do Produto.

4.1.5. Retrospectiva

A sprint é revista em até três horas, discutindo **o que foi bem e o que pode melhorar** em relação às pessoas, comunicação, processos e ferramentas.

Os artefatos produzidos no Scrum são o **Backlog do Produto**, o **Backlog da Sprint** (itens do Backlog do Produto escolhidos para compor a sprint) e o **Incremento** (conjunto de itens Prontos de uma sprint).

Por fim, recomendamos a leitura de [1] para aprofundar o conhecimento sobre Scrum.

4.2. Quadro Kanban

O termo Kanban, do japonês “sinal”, é um método de trabalho que privilegia [2]:

1. Trabalho – **visualizá-lo**, **limitar** sua quantidade em progresso, gerenciar seu **fluxo**;
2. Políticas – **expô-las**;
3. Feedback – **promover** em ciclos;
4. Processos – **melhorar e evoluir**.

Nisto, o “quadro Kanban” (vide Figura 1) serve como artefato principal para demonstrar uma unidade de valor e seu desenvolvimento.

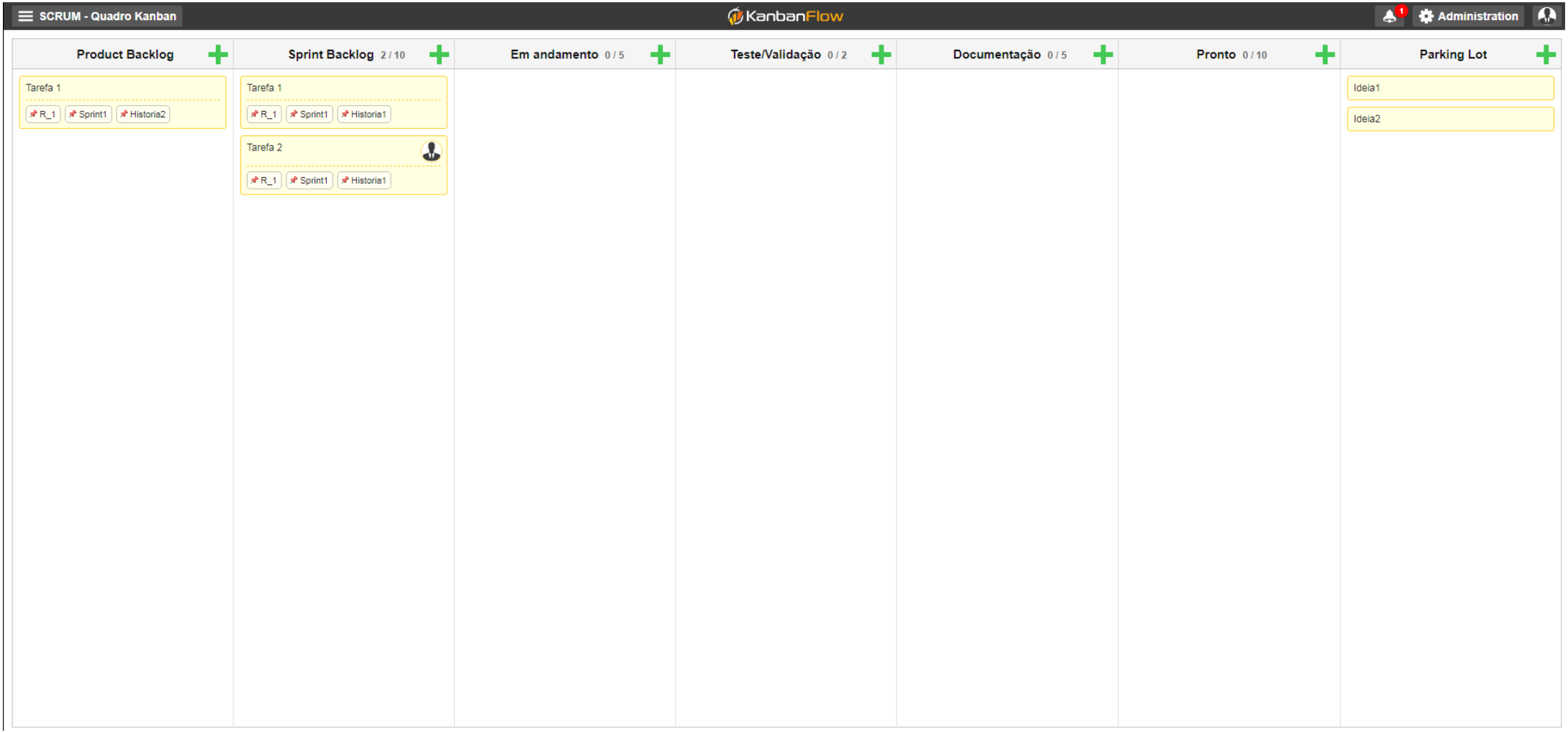

Figura 1: Exemplo de Quadro Kanban.

No modelo adaptado aqui, as raias representam os seguintes estágios das estórias identificadas:

- **Product Backlog** – representa o backlog de produto, que contém todas as estórias que o PO deseja ver implementadas.
- **Sprint Backlog** – representa o backlog da sprint planejada, o qual contém as estórias selecionadas para serem evoluídas na sprint.
- **Em andamento** – contém as estórias que estão sofrendo algum tipo de evolução pelo time.
- **Teste/validação** – abriga as estórias que foram concluídas pelo time e aguardam ou teste interno ou validação do PO / pessoas de interesse.
- **Documentação** – estágio transversal em que a estória é documentada, seja para melhorar o entendimento, seja para adicionar/ajustar critérios de aceitação.
- **Pronto** – é o conjunto de estórias prontas para serem entregues como incremento do produto.
- **Parking Lot** – em tradução literal, "estacionamento", no qual as ideias surgidas nos vários momentos de uma sprint aguardam para serem discutidas no planejamento da próxima sprint. Essas ideias podem se referir a novas estórias, melhorias de processo e impedimentos.

4.3. Estórias de Usuário

O termo estória de usuário tem origem no método **XP (Extreme Programming) [3]**, que foi idealizado para responder a problemas cujos **requisitos mudam constantemente**, como pode ser visto em seus princípios e técnicas. Uma delas, a estória de usuário, é um artefato simples que **sintetiza o requisito do usuário por meio de uma afirmação e de um conjunto de critérios de aceitação** que ratificam a afirmação. Exemplo:

> **Como** um gestor de tecnologia, **eu desejo** alocar os melhores desenvolvedores nos projetos mais complexos **para** garantir o maior retorno de investimento.
>
> Critérios de aceitação: **dado que** um novo projeto seja aceito, **quando** nenhum desenvolvedor considerado experiente esteja disponível, **então** será alocado o desenvolvedor livre com mais anos de experiência.

Este Modelo conta com um template de estória de usuário.

5. Modelo de Gestão de Sustentação Ágil de Software

Usando os conceitos definidos na seção Scrum e Práticas Ágeis, nosso modelo de trabalho baseia-se principalmente nos papéis e atividades do Scrum, aliadas ao quadro Kanban e o modelo de estórias de usuário do XP, já descritos.

Aqui definem-se portanto aspectos diversos das práticas comuns.

5.1. Estrutura básica da Sprint

Duração esperada: até 3 semanas

5.1.1. Distribuição das Atividades

Semana 1	Semana 2	Semana 3
Planejamento	Desenvolvimento	
Identificar e detalhar estórias	Testes	
Diagrama de classe	Documentação	
DER		Revisão
		Retrospectiva

5.1.2. Notas

A distribuição proposta não impede que atividades de semanas posteriores sejam antecipadas.

Uma release é composta de uma ou mais sprints.

Para momentos especiais (homologação, corrigir erros identificados na homologação, implantar uma versão) deve-se realizar uma sprint especial focada na atividade, com a duração mínima necessária para concluí-la.

Bibliografia

1: Ken Schwaber, Jeff Sutherland, Um guia definitivo para o Scrum: As regras do Jogo, 2017
2: Agile Alliance, Kanban Board, , https://www.agilealliance.org/glossary/kanban-board
3: Don Wells, When should Extreme Programming be Used?, 1999, http://www.extremeprogramming.org/when.html

Apêndice A – O Manifesto Ágil

As questões sobre agilidade no desenvolvimento de software foram postuladas por profissionais e acadêmicos renomados na engenharia de software em um documento chamado Manifesto Ágil de Desenvolvimento de Software. Este documento apresenta 4 valores que devem ser considerados no desenvolvimento ágil e 12 práticas ágeis que, adotadas, contribuem bastante para a agilidade.

O MGSAS foi concebido com base nas questões trazidas por este manifesto. A sintonia entre o modelo e o manifesto pode ser compreendida através da transcrição a seguir deste último, com os apontamentos, item a item, sobre o alinhamento de ambos.

Valores

1. **Indivíduos e interações** mais que processos e ferramentas.

   Os analistas de uma equipe de sustentação estão alocados no mesmo espaço físico, favorecendo a interação e a comunicação rápida e pessoal. Existe proximidade também com o cliente, que compreende sua responsabilidade na sustentação do sistema. Os processos, que são simples e de rápida assimilação, são executados por um grupo pequeno de pessoas e servem para orientar a execução das atividades e organizar as informações, dando suporte à operação da sustentação do sistema, da mesma forma que as ferramentas utilizadas pelas equipes.

2. **Software em funcionamento** mais que documentação abrangente.

   A documentação produzida pela equipe de sustentação é simples e é composta por casos de uso, quando se tratar de atualização de funcionalidades já existentes no sistema em questão, ou **histórias de usuários**, que são documentos simples de rápida confecção. O conhecimento a respeito do sistema também está materializado na código-fonte do mesmo, feito de forma padronizada, seguindo os princípios da legibilidade e simplicidade. Além disso, a sustentação do sistema conta com conhecimento armazenado de forma tácita pela equipe de sustentação. A respeito deste último aspecto, é relevante ressaltar que a **estabilidade dos analistas na equipe** é um fator fundamental na manutenção desse conhecimento tácito dentro da dela.

3. **Colaboração com o cliente** mais que negociação de contratos.

   A relação de trabalho entre a equipe de sustentação e o cliente (gestor do sistema) é de cunho administrativo, não comercial, favorecendo a colaboração entre os indivíduos, pois não há a necessidade de negociação de contratos.

4. **Responder a mudanças** mais que seguir um plano.

   Evoluir o sistema em sustentação deve seguir mais a **percepção atual do gestor a respeito do futuro do sistema** do que um planejamento feito no passado. Como este gestor recebe entregas rápidas e frequentes, esta percepção e planejamento podem se alterar de sprint para sprint, ou seja, a cada 2 ou 3 semanas.

Práticas

1. Nossa maior prioridade é satisfazer o cliente, através da entrega adiantada e contínua de software de valor agregado.

   O cliente, o gestor de sistema, assume papel central no MGSAS. Sua proximidade com a equipe de sustentação faz com que as necessidades por mudanças sejam transmitidas com rapidez à equipe de sustentação. A utilização de sprints, que duram de 2 a 3 semanas, tem, por consequência, entregas rápidas, fazendo com que **respostas às mudanças** sejam rápidas. O fato de estas sprints ocorrerem sequencialmente, uma após outra, leva à entrega constante de valor, satisfazendo o cliente.

2. Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando vantagem competitiva para o cliente.

   Nos processos utilizados pelo MGSAS, as mudanças de requisitos são tratadas como mudança de escopo de uma sprint e são analisadas e tratadas conforme as prioridades definidas pelo gestor do sistema. Com isso, a mudança poderá ser encaixada tanto na sprint corrente quanto na próxima sprint ou ser criada uma sprint emergencial específica para a sua implantação.

3. Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo.

   A entrega de software funcionando ocorre frequentemente, sprint após sprint, que possuem tempo de vida entre 2 a 3 semanas.

4. Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto.

   As pessoas da área de negócios e os analistas trabalham, na maioria dos casos, no mesmo prédio, tendo a disponibilidade para reuniões presenciais, podendo trabalhar em conjunto quando necessário. Além disso, as pessoas das áreas de negócio compreendem seu papel nos processos de sustentação e sua importância, e possuem à disposição um canal telefônico e o Sistema de Gestão de Demandas para efetuar solicitações.

5. Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho.

   Os analistas das equipes de sustentação são servidores do Ministério Público do Estado da Bahia, regidos por um plano de desenvolvimento de carreira, baseado em suas competências, que os motivam a executar suas atividades de forma profissional e com qualidade. O ambiente de trabalho

também oferece os recursos apropriados para essa execução, tais como estações de trabalho modernas, climatização e móveis adequados.

6. O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de **conversa cara a cara**.

   Os analistas de uma equipe de sustentação estão alocados na mesma sala da DTI. Isso favorece que conversas “cara a cara” sempre que necessário, mesmo quando as informações contidas no processo de comunicação já estejam registradas em ferramentas, como o SGD e atas de reunião.

7. Software funcional é a medida primária de progresso.

   Com a utilização de sprints e processos de sustentação nos quais o gestor do sistema realiza uma homologação dos mesmos e recebe, frequentemente, uma nova versão no ambiente de produção, a percepção do progresso torna-se fácil, justamente pela mensuração das evoluções e correções no sistema, ou seja, software funcional.

8. Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente.

   A estratégia de utilização constante de sprints curtas e sequenciais, as quais seguem as definições de processos simples e de rápida assimilação, executados por equipes estáveis, favorece à sustentabilidade do modelo, entregando valor ao gestor de forma gradativa.

9. Contínua atenção à excelência técnica e bom design aumenta a agilidade.

   A equipe de sustentação segue padrões relacionados ao desenvolvimento de software estabelecidos pela DTI que tornam os artefatos de software impessoais e coletivamente compreensíveis. Além disso, utiliza-se uma arquitetura de referência, comum a todos os sistemas, pautadas em boas práticas de desenvolvimento, agilizando a construção de soluções.

10. Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser feito.

    A estratégia de utilizar sprints pequenas permite a entrega rápida de software e dá, antecipadamente, a oportunidade do gestor avaliá-lo, identificando, o quanto antes, os pontos no sistema que agregaram valor e os que precisam ser melhorados. Como o escopo destas sprints são pequenos, o risco de serem implementadas funcionalidades ou recursos desnecessários torna-se menor.

11. As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis.

    As melhores decisões sobre um problema vêm de pessoas próximas das causas e consequências delas. Partindo da hipótese de que os membros de um time ágil possuem habilidades diversas e conhecimento dos requisitos, arquitetura empregada e conjunto de tecnologias envolvidas, são eles as pessoas mais capacitadas para decidir e evoluir os componentes diversos da solução.

12. Em intervalos regulares, o time reflete em como ficar mais efetivo, então, se ajustam e otimizam seu comportamento de acordo.

    O manifesto não se encerra em si, tanto que oferece o último dos doze princípios para a melhoria dos processos e pessoas que adotam as práticas ágeis. Um exemplo disso é a reunião de retrospectiva do *Scrum*, na qual, entre outras coisas, se verificam pontos em que é possível melhorar a eficácia e eficiência da equipe.