VIDE - Instrução Normativa STJ/GDG n. 13 de 15 de abril de 2024 (Alteração)

STJ | Secretaria de Tecnologia da Informação e Comunicação
Coordenadoria de Desenvolvimento de Soluções de Software

PDS - STJ Ágil

Processo Ágil de Desenvolvimento de Software do STJ

- Guia de Referência –
Janeiro 2022

Versão 2.0

Índice

1. Propósito deste Documento

A primeira versão deste documento surgiu em 2018, com o objetivo de ser um guia para o processo de trabalho da Coordenadoria de Desenvolvimento de Soluções de Software - CDES. Naquela época, estávamos descobrindo as práticas ágeis para o desenvolvimento de aplicações, como o *Scrum*, *Kanban*, Programação XP etc. e fizemos uma adaptação dessas práticas para tornar o trabalho da CDES mais ágil e enxuto. Desde então, esse Guia de Processo de Desenvolvimento de Software – PDS tem passado por atualizações e evoluções baseadas na nossa cultura de trabalho, no nosso relacionamento e no nosso dia a dia. O objetivo continua o mesmo: Descrever, de forma simples, as práticas adotadas na CDES para o desenvolvimento ágil de software.

2. Missão da CDES

A Missão é a declaração concisa da razão de ser daquela unidade, alinhada à missão da instituição. A CDES tem como missão:

"Construir, de forma ágil e com qualidade, produtos de software, que atendam e agregam valor às áreas de negócio do Superior Tribunal de Justiça - STJ e ao público jurisdicionado, visando sempre a melhoria contínua."

3. Princípios e Valores

Os princípios e valores a seguir são uma combinação dos princípios por trás do Manifesto Ágil combinado com a cultura de trabalho da CDES.

O sucesso depende das pessoas se tornarem mais proficientes em viver cinco valores: Compromisso, Foco, Abertura, Respeito e Coragem.

Satisfação do cliente por meio da entrega contínua de produtos de software.

Pessoas de negócio e times de desenvolvimento devem trabalhar em conjunto, diariamente, em torno de um objetivo comum para construir produtos cada vez melhores.

Software funcionando é a medida primária de progresso.

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

Simplicidade - a arte de maximizar a quantidade de trabalho que não precisará ser realizado – é essencial.

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

A boa comunicação e a transparência devem ser praticadas, diariamente, por todos.

Em intervalos regulares, o time reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

4. Tipos de Demandas

De acordo com a Instrução Normativa STJ/GP n. 13 de 28 de agosto de 2018, que disciplina o encaminhamento, a priorização e a aprovação de demandas de TIC, os tipos de demandas relacionados aos produtos de software na CDES são:

1. **Demandas de Novo Produto:** advém da solicitação de serviço de desenvolvimento de sistemas e consistem na construção de novos produtos de software. Essas demandas são priorizadas e aprovadas pelos comitês de sistemas e encaminhadas à CDES para execução;

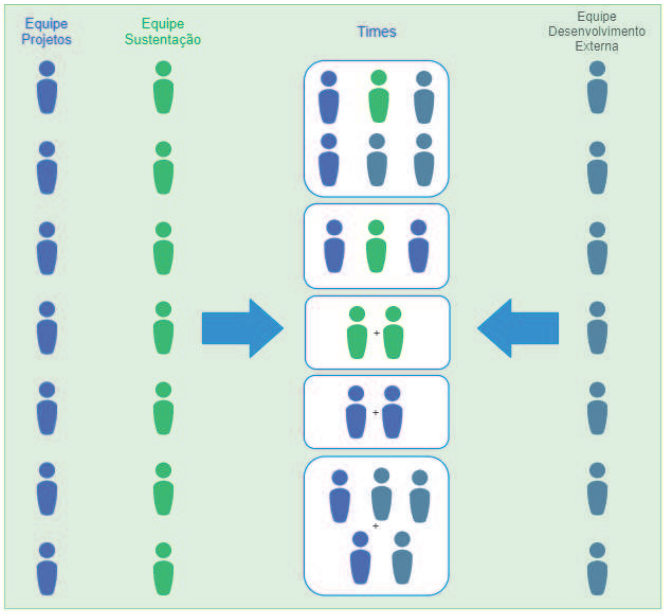
2. **Demandas Evolutivas:** advém da solicitação de manutenção evolutiva de sistema que surgem nos projetos do STJ ou na evolução dos processos de trabalho e procedimentos das áreas de negócio. Frequentemente elas requerem a evolução dos produtos de software do STJ, que foram desenvolvidos pela CDES. Essas demandas são priorizadas e aprovadas pelos comitês de sistemas e encaminhadas à CDES para execução;

3. **Demandas de Sustentação:** advém da solicitação de serviço de manutenção corretiva de sistema e consistem de demandas necessárias para a conservação dos produtos desenvolvidos pela CDES, mantendo sua importância e relevância na cadeia de valor do STJ. Podem ser classificadas como:
    a. **Incidentes**: Indisponibilidade do sistema, falhas diversas etc. reportadas por usuários durante o uso das aplicações.
    b. **Erros**: *bugs* ou erros nos produtos de software que podem prejudicar ou dificultar o seu uso. Demandas desse tipo são tratadas na execução das *sprints*.

4. **Demandas de Consultoria:** advém da solicitação de auditoria ou de serviços específicos (extração de dados, orientação técnica e prestação de informações) e que diz respeito aos produtos de *software* desenvolvidos pela CDES.

5. Papéis e Responsabilidades

Os times da CDES são organizados por área de negócio, de forma multidisciplinar e composto por membros das equipes de projeto e/ou sustentação, além de equipes de desenvolvimento externas, quando é o caso.



Organização dos Times da CDES

Cada time atua no desenvolvimento de um ou mais produtos de software. Eles são formados por:

- ***Stakeholder****:* pessoa da área de negócio, interessada no(s) produto(s) de software e responsável por demandar e inspecionar os produtos de software que estão sendo desenvolvidos.

- ***Scrum Master:*** responsável por promover o ambiente de desenvolvimento ágil resolvendo impedimentos e incentivando o time.

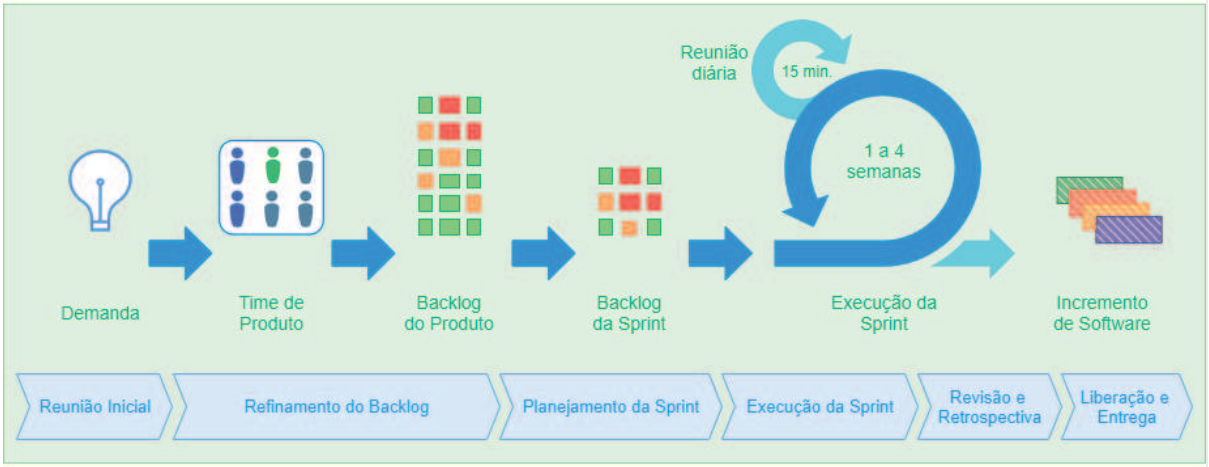
- **Dono do Produto (*Product Owner – PO*)**: ordena o trabalho a ser realizado pelo time, criando, mantendo e priorizando o(s) backlog(s) do(s) produto(s).

- **Analistas:** que auxilia o Dono do Produto no refinamento do *backlog*.

- **Desenvolvedores (Equipe de Desenvolvimento):** que constrói(em) o(s) produto(s) de software.

6. Fluxo de Trabalho

De forma geral, o fluxo de trabalho na CDES segue um processo que tem como base o Scrum, com algumas adaptações. O foco é a entrega contínua de incrementos de software, por meio de sprints de 1 a 4 semanas.



Fluxo de Desenvolvimento da CDES

7. Reunião Inicial

A Reunião Inicial é uma reunião de conhecimento da demanda bem como de alinhamento de expectativas a respeito do que deve ser desenvolvido. Essa reunião pode levar até 3 horas.

8. Fase de Preparação

Após a reunião inicial, o Dono do Produto, junto com os analistas, prepara o *backlog* inicial do produto, priorizando os itens de maior importância. Eles podem também se reunir outras vezes com os *stakeholders* para refinar os requisitos para o planejamento da *sprint*.

Durante esse tempo o time também combina como será a arquitetura da solução, definindo em linhas gerais, quais tecnologias serão utilizadas para a solução do produto de software.

9. Planejamento da Sprint

Com base no *backlog* do produto priorizado, o time realiza o planejamento da *sprint*, selecionando os itens possíveis de serem desenvolvidos no tempo da *sprint* (geralmente de 1 a 4 semanas, previamente acordado), compondo o *backlog* da *sprint*. Eles combinam com o dono do produto a **Definição de Pronto**, ou seja, o que deve ser entregue ao final da *sprint*. O time, então, se compromete a entregar o incremento de *software*, ao final da *sprint*, de acordo com o combinado. A equipe também usa esse tempo para planejar como o trabalho escolhido será realizado. Esse planejamento pode levar até 4 horas, dependendo do tamanho da *sprint*.

10. Execução da *Sprint* e Reunião Diária

O time de desenvolvimento então inicia a execução da *sprint*, construindo o incremento de *software* a partir do *backlog* da *sprint*. Durante esse período, o time também se reúne diariamente, por 15 minutos, para responder a três perguntas:

1. O que eu fiz desde a última reunião?
2. O que eu vou fazer até a próxima reunião?
3. Há algum impedimento para o trabalho ser realizado?

11. Revisão da Sprint

Após o último dia da execução da *sprint*, a equipe se reúne para realizar a revisão do produto. Nesse evento, o time de desenvolvimento apresenta o incremento de *software* construído para o Dono do Produto e os *stakeholders*. Eles revisam o que foi realizado na *sprint* e colaboram sobre o que fazer a seguir. O Dono do Produto analisará a resolução de cada história apresentada e decidirá se está "Pronta" ou não, de acordo com o critério previamente estabelecido. Nesse momento, o *backlog* do produto pode ser atualizado para refletir as novas demandas e oportunidades. Caso alguma história seja reprovada, ela será reinserida no Backlog ficando disponível para uma próxima sprint.
Essa revisão pode levar até 4 horas para *sprints* de 4 semanas.

12. Retrospectiva da *Sprint*

Em seguida, o time realiza a retrospectiva da *sprint*. O último evento antes da conclusão da *sprint*. De acordo com o Guia do Scrum, "O propósito da retrospectiva da *sprint* é planejar maneiras de aumentar a qualidade e a eficácia do trabalho". O time discute como foi a última sprint em relação a indivíduos, interações, processos, ferramentas e a Definição de Pronto. Eles respondem basicamente a dois questionamentos:

1. O que deu certo?
2. O que pode ser melhorado?

O time, então, identifica as mudanças mais úteis para melhorar sua eficácia e procura aplicar essas mudanças na *sprint* seguinte. A retrospectiva pode levar até 4 horas para sprints de 4 semanas.
Finalmente, o time inicia o planejamento da próxima *sprint* e o ciclo se repete quantas vezes forem necessárias.

13. Liberação e Entregas

A entrega dos produtos de software ocorre após a finalização da *sprint*, quando o incremento de software é revisado e homologado. É um evento separado da *sprint*. Além disso, cada produto possui uma forma distinta de entrega. Para os novos projetos consulte o guia disponível no endereço https://gitlab.stj.jus.br/devops/templates/template-gitlab-ci.

14. Ferramentas e Práticas Recomendadas

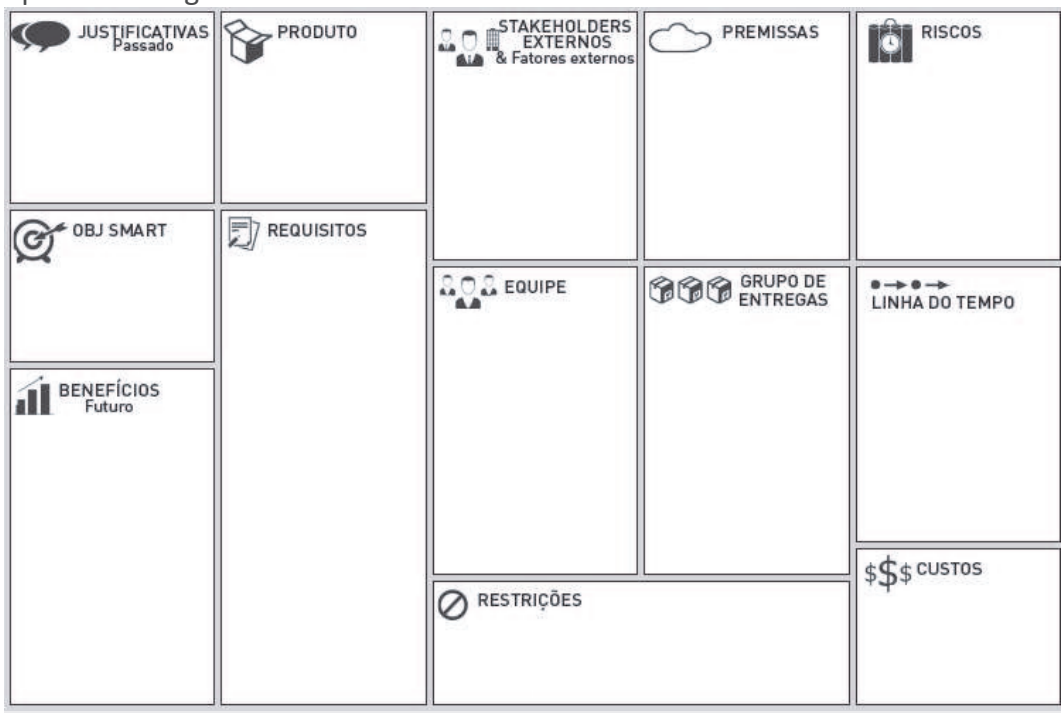
***Backlog* da Sprint:** reunir todas as histórias de usuário e técnicas selecionadas pelo Dono do Produto para a implementação na *sprint*.

***Backlog* do Produto:** agrupar todas as histórias de usuário e técnicas que fazem parte do produto. Priorizar as estórias em ordem de importância do projeto.

Backlog Grooming: é o refinamento do *backlog*. O Dono do Produto, junto com a equipe de análise, quando houver, devem se reunir para refinar os itens de *backlog* do produto que serão priorizados nas *sprints* futuras.

***Bugfix*:** quando é encontrada uma falha no produto de software que ainda não está em produção, ou seja, ainda não está sendo usada pelos usuários, classificamos essa falha como *bugfix*. O Dono do Produto deve analisar e priorizar a correção do *bugfix* de acordo com as prioridades da *sprint* e negociar com o time de desenvolvimento quando a correção será realizada.

Canvas: o *canvas* é um documento que permite, de forma visual, expor a todos os envolvidos a visão geral do produto a ser construído. O documento, de página única, é composto dos seguintes itens:



Documento de Arquitetura: documentar, de forma visual, objetiva e clara, a arquitetura em que o projeto será construído. O objetivo e o teor do documento de arquitetura do projeto devem estar alinhados ao Documento de Referência de Arquitetura instituído no

Histórias de Usuários: uma das formas mais simples e eficientes de descrever a necessidade de um usuário sobre uma aplicação ou sistema é por meio da história de usuário, levantando quem são os atores, a funcionalidade a ser executada e o motivo negocial pelo qual ela deve ser implementada.

Histórias Técnicas: quando não há, necessariamente, um usuário envolvido em um item do *backlog*, é possível descrever a necessidade por meio de uma história técnica. Desse modo, em uma integração de sistemas, por exemplo, o sistema "cliente" torna-se um "usuário" do sistema provedor de informações.

Gestão de requisitos: consiste na gestão do ciclo de vida do(s) requisito(s) utilizado(s) para a construção e evolução do produto objeto do *backlog*. No Scrum, o requisito é formalizado por intermédio das histórias de usuário, as quais contêm as características inerentes à necessidade que será atendida pelo produto. Nesse contexto, são diretrizes da gestão de requisitos no âmbito desse PDS:

- o registro e histórico da comunicação de requisitos entre PO e a equipe de desenvolvimento;
- o emprego de boas práticas para confecção de histórias de usuário;
- o armazenamento consistente dessas histórias em ferramenta(s) capaz(es) de promover a indexação, consulta, pesquisa de conteúdo e histórico de alterações;
- a rastreabilidade entre história (requisito) e funcionalidade do produto e descarte.

Hotfix: o *Hotfix* consiste em uma solicitação de correção urgente de uma falha identificada na versão do produto de software que se encontra implantada em produção, ou seja, está sendo utilizada pelos usuários. Nestes casos, a correção dessa falha deve ser imediata, considerando que gera alto impacto para os usuários. Portanto a equipe deve ser mobilizada para fazer a correção e lançar uma versão de correção antes mesmo do final da sprint em execução.

***Kanban*:** o *kanban* ajuda a tornar a visualização do trabalho mais clara e objetiva. Portanto a recomendação é que tanto o *backlog* do produto como o *backlog* da *sprint* sejam organizados em um ou mais quadros *kanban*. Ademais, devem ser usadas marcações coloridas que permitam visualizar de forma rápida e eficiente o que é urgente, o que está duplicado etc.

Métricas: na CDES os produtos desenvolvidos na nova arquitetura são validados por meio de Ferramenta de Análise Estática de Código. Nessa ferramenta são avaliadas questões técnicas relativas ao código fonte da aplicação. Cada item recebe uma pontuação e um peso e, dependendo item, ele pode ou não passar no *Quality Gate*. O *Quality Gate* é configurado de forma que o incremento de software só pode ser promovido para o ambiente de testes, caso atenda aos requisitos técnicos especificados para o código que está sendo entregue. Caso o código não atenda esses critérios, a equipe de desenvolvimento deve fazer as devidas correções apontadas pela ferramenta.

***Mockups*:** apresentar graficamente (em forma de protótipo não funcional) as funcionalidades presentes nas estórias de usuário, a fim de facilitar o entendimento dos requisitos por todas as partes envolvidas e a construção dos critérios de aceitação.

Nota de *Release*: documentar as funcionalidades implementadas no *build* entregue, a fim de manter a rastreabilidade entre a versão implantada e os requisitos.

Revisão em Pares: É importante que os times se organizem para realizar a revisão em pares dos produtos que estão em desenvolvimento durante a *sprint*. Um membro da equipe, ao finalizar uma tarefa, repassa a outro membro da equipe para fazer a revisão do item desenvolvido, de forma que a eventual presença de erros seja resolvida ainda durante a *sprint*.

Roadmap do Projeto: detalhar a linha do tempo elaborada no *Canvas*, estipulando prazo para, no máximo, duas entregas.

Segurança: deve-se observar os princípios, diretrizes e boas práticas disciplinados no Processo de Desenvolvimento Seguro de Software e no Guia para Desenvolvimento Seguro de Software do STJ, anexos a este Processo.

Testes: os testes são fundamentais para o sucesso de uma entrega de software. A recomendação é incorporar práticas reconhecidas do mercado para aumentar a qualidade dos produtos de *software* desenvolvidos na CDES.

15. Referências Bibliográficas

BECK, Kent; Et. Al. *Manifesto para Desenvolvimento Ágil de Software. Disponível em https://agilemanifesto.org/iso/ptbr/manifesto.html. Acesso em 09/11/2021*

SCHWABBER, Ken; SUTHERLAND, Jeff. *O Guia do Scrum.* 2020. Disponível em https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Portuguese-European.pdf. Acesso em 09/11/2021.

CRUZ, Fábio. *Scrum e Agile em Projetos - Guia Completo: Conquiste sua certificação e aprenda a usar métodos ágeis no seu dia a dia*. Editora Brasport, 2015.

SUTHERLAND, Jeff. Scrum. *A arte de fazer o dobro do trabalho na metade do tempo*. Editora Leya, 2016.

SUTHERLAND, Jeff J. *Scrum, Guia prático.* Editora Sextante, 2020.