

Projeto de *Software* com Métodos Ágeis



Cruzeiro do Sul Virtual
Educação a distância

Material Teórico



Projeto – Engenharia da Aplicação e Controle

Responsável pelo Conteúdo:

Prof. Me. Artur Marques

Revisão Textual:

Prof.^a Esp. Kelciane da Rocha Campos



- Desenho de *Mockup* das Telas para IHC;
- Protótipo;
- Mapa Conceitual;
- Mapa Navegacional;
- Reuniões e Rituais *Scrum*;
- Controle Ágil de Projeto;
- Uso de Quadro *Kanban*.

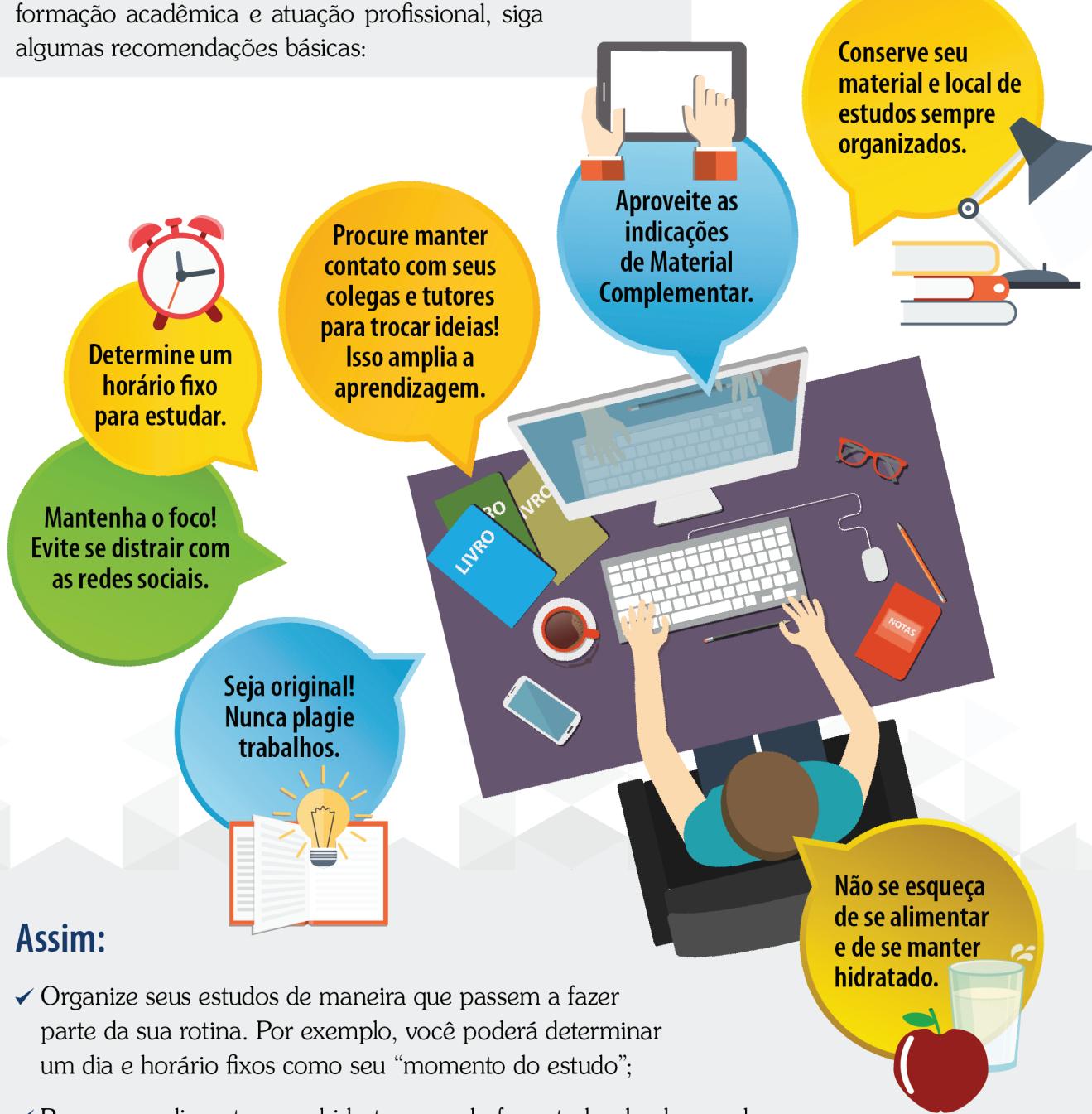


OBJETIVO DE APRENDIZADO

- Saber construir um quadro *Kanban* e realizar os gráficos de controle.

Orientações de estudo

Para que o conteúdo desta Disciplina seja bem aproveitado e haja maior aplicabilidade na sua formação acadêmica e atuação profissional, siga algumas recomendações básicas:



Assim:

- ✓ Organize seus estudos de maneira que passem a fazer parte da sua rotina. Por exemplo, você poderá determinar um dia e horário fixos como seu “momento do estudo”;
- ✓ Procure se alimentar e se hidratar quando for estudar; lembre-se de que uma alimentação saudável pode proporcionar melhor aproveitamento do estudo;
- ✓ No material de cada Unidade, há leituras indicadas e, entre elas, artigos científicos, livros, vídeos e sites para aprofundar os conhecimentos adquiridos ao longo da Unidade. Além disso, você também encontrará sugestões de conteúdo extra no item **Material Complementar**, que ampliarão sua interpretação e auxiliarão no pleno entendimento dos temas abordados;
- ✓ Após o contato com o conteúdo proposto, participe dos debates mediados em fóruns de discussão, pois irão auxiliar a verificar o quanto você absorveu de conhecimento, além de propiciar o contato com seus colegas e tutores, o que se apresenta como rico espaço de troca de ideias e de aprendizagem.

Desenho de Mockup das Telas para IHC

Se existe algo mal compreendido é a parte de *design* de interfaces humano-computador. Sim, aquela parte que todo mundo chama de interface de usuário, que é a representação para o mundo da “cara” do seu sistema.

Um *wireframe*, também conhecido como esquema de página ou plano de tela, é um guia visual que representa a estrutura do esqueleto de uma aplicação. Como atualmente a grande maioria é *web*, então trata-se da estrutura de uma aplicação *web*. O *wireframe* descreve o *layout* da página ou a organização do conteúdo, incluindo elementos de interface e sistemas de navegação pela aplicação e como eles funcionam juntos.

Um *wireframe*, portanto, é um *layout* de *design* de baixa fidelidade que atende a três propósitos simples, mas exatos:

- apresentar as informações que serão exibidas na página;
- fornecer um esboço da estrutura e do *layout* da página;
- transmitir a direção geral e a descrição da interface do usuário.

Os *wireframes* servem como um meio-termo entre os esboços em papel e caneta e seu primeiro protótipo, além de ajudarem a planejar o *layout* e os padrões de interação dos usuários sem se distrair com detalhes como cores.

A jornada do usuário proposta deve ser clara, sem a necessidade de cores, sombras ou menus sofisticados.

Eles são perfeitamente adaptados para projetos ágeis, porque devem ser alterados à medida que você coleta mais informações por meio da pesquisa do usuário ou da entrada de partes interessadas. Portanto, servem como uma linguagem comum entre *designers*, usuários, partes interessadas e desenvolvedores. Eles devem ser absolutamente simples e despojados.

Eles são úteis e devem ser feitos para gerar consenso e melhorias antes de sairmos fazendo. Por exemplo, aqui vão algumas dicas:

- Mantenha as cores em branco e preto ou tons de cinza;
- Use no máximo duas fontes genéricas, por exemplo *arial* e *calibri*, talvez uma com serif (*times new roman*) e uma sem serif (*arial*);
- Mostre a hierarquia de informações por meio da fonte, alterando o tamanho da fonte e se ela está estilizada (negrito, itálico etc.);
- Evite gráficos e imagens chamativas. Em vez disso, tente usar retângulos e quadrados simples como marcadores de posição com um “x” no meio da caixa para mostrar onde a imagem será colocada;
- Faça o mesmo com os vídeos, usando um triângulo como botão de reprodução no centro da caixa.

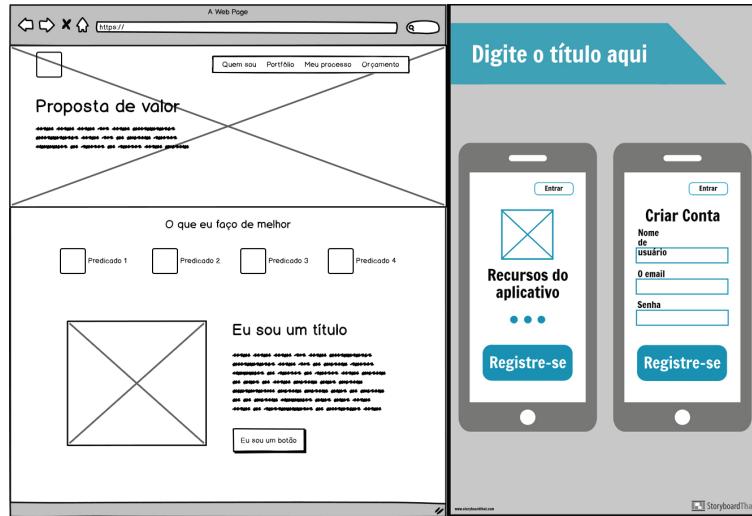


Figura 1 – Exemplos de *wireframe* para *Web* e *Mobile*

Fonte: *Designerd*, 2020

A próxima etapa do processo é uma maquete de alta fidelidade.

Os **MOCKUPS** são uma representação mais visual do seu produto de *software*. Pense em uma maquete, esta é a parte em que você firma suas decisões visuais testando variações e começa a dar algum caráter ao seu *software*.

É uma simulação de alta fidelidade da aparência de uma aplicação ou um *site*. Mockups ou, se preferir, Maquetes combinam a estrutura e a lógica de um *wireframe*, mas com as imagens, gráficos e elementos de interface do usuário que o produto terá.



Figura 2

Fonte: *Getty Images*



Figura 3 – Exemplos de *Mockups*

Fonte: *Getty Images*

O Mockup é útil para:

- **Arquitetura da informação:** como você organiza as informações do seu *site* de forma lógica e hierárquica;
- **Fluxo do usuário:** como o usuário irá interagir com seu *site* e a UX de navegação;
- **Hierarquia visual e layout:** como todos os elementos aparecem em suas páginas da *web*, seus tamanhos e posições em relação uns aos outros e o espaçamento entre eles;
- **Cor:** quase sempre contém o esquema de cores que você usará no produto;
- **Tipografia:** contém os tipos de fontes, tamanhos e ousadia exatos que seu produto terá;
- **Imagens:** use imagens de alta qualidade que refletem exatamente as que você incluirá em seu produto;
- **Interação:** a forma com que o usuário interage com o sistema.

Os modelos de sistemas preenchem a lacuna entre equipes multifuncionais como um dispositivo de comunicação e garantem uma transferência de *design* suave. O processo de transferência é um momento vital no desenvolvimento de produtos. É quando você entrega o *design* final aos desenvolvedores, completo, com os recursos e as diretrizes de que eles precisarão para codificar sua maquete em um produto vivo e vibrante.

Uma maquete de sistema completa, idealmente combinada com um sistema de *design* que inclui um guia de estilo, especificações de *design*, padrões e componentes, ajuda os desenvolvedores, garantindo que não haja incertezas no *software*.

Haverá menos idas e vindas entre *design* e desenvolvimento porque todos terão as informações de que precisam e você evitará retrabalhos caros no futuro.

A etapa final antes de entregar seu *design* aos desenvolvedores é o protótipo. Fazer um protótipo é como vestir seu *MOCKUP* e torná-lo adequado ao público.

Protótipo

Um protótipo é um modelo de *design* de fidelidade média a alta da Interface do Usuário final de seu *software* ou aplicativo móvel. Além de oferecer uma visão mais detalhada dos atributos visuais de seu *design*, os protótipos geralmente incluem a primeira interação do usuário. A ideia geral de um protótipo de alta fidelidade é representar o mais próximo possível do produto.

Não é a versão final, mas é aceitável para mostrar a outras pessoas. Este é o ponto em que tudo o que você precisa é fazer pequenos ajustes antes de enviar seu modelo para produção.

Há vários tipos de protótipo, por exemplo:

- **Navegacional:** apenas permite a navegação entre telas para ver o circuito todo. Não há interação, apenas navegação;
- **Funcional:** nesse caso, testam-se características específicas da aplicação; em telas ou locais específicos podemos acessar locais e interagir com eles; muitas vezes não há regras ou grandes processamentos embutidos exatamente para vermos como ficou;

- Temos várias outras correntes de tipos de protótipos; muitas vezes você verá sites ou youtubers colocando 10 tipos de protótipos, 12 tipos de protótipos e que, de certa forma, *wireframes* e *mockups* são protótipos e assim por diante. Para efeito de nosso curso, os dois tipos que passei, e deixando separados os *wireframes* dos *mockups*, são suficientes, sem precisar reinventar a roda ou árvores taxonômicas às vezes sem sentido prático.



Exemplo de protótipo. Perceba que há praticamente todas as telas da solução.
Disponível em: <https://bit.ly/3j9uj5q>

Mapa Conceitual

Um mapa conceitual é uma ferramenta que visualiza relações entre conceitos. É útil para desmascarar informações complexas em grande escala.

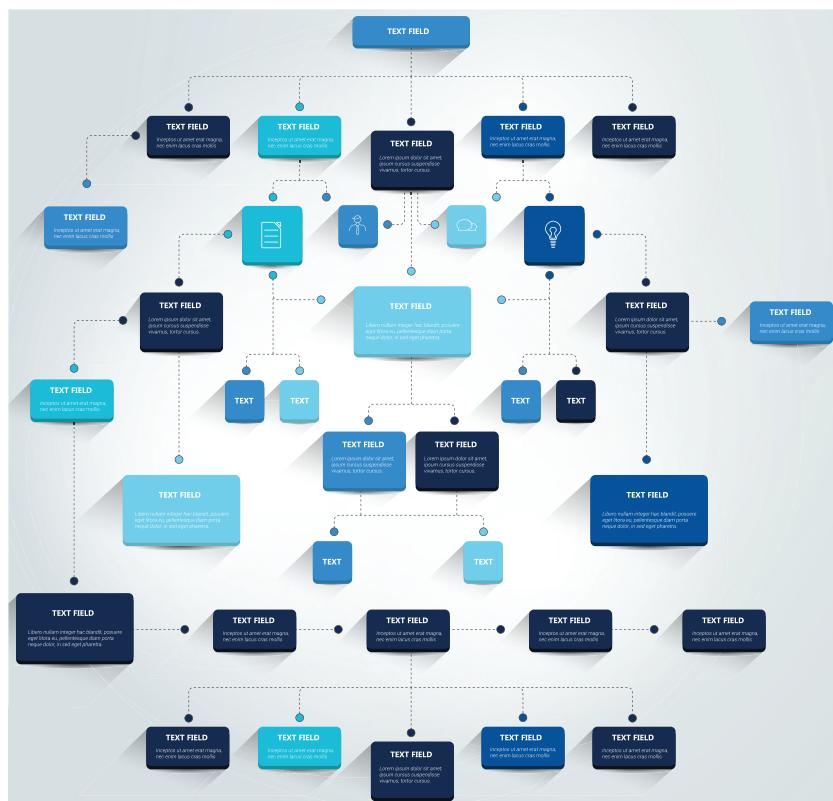


Figura 4 – Mapa conceitual de nosso Café *gourmet*

Fonte: Acervo do conteudista

Mapa Navegacional

Os mapas de navegação também são conhecidos como mapas de aplicação ou do site, são uma ferramenta de desenvolvimento muito importante e podem ajudar a manter o processo de desenvolvimento sob controle.

Eles são basicamente uma representação visual de como as páginas serão vinculadas. Em aplicações ou sites simples, os mapas de navegação são muito simples porque todas as páginas são vinculadas a todas as outras páginas, mas em grandes aplicativos ou sites com centenas de páginas vinculando todas as páginas a todas as outras páginas não são práticos, os mapas de navegação desses aplicativos ou sites são muito complexos.

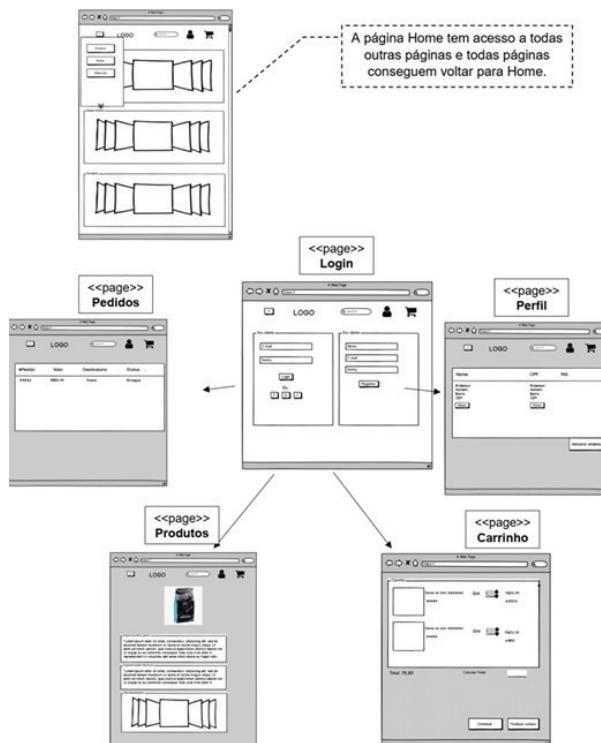


Figura 5 – Trecho de wireframe do Café gourmet com as setas de direcionamento para indicar a navegação

Fonte: Acervo do conteudista

Reuniões e Rituais Scrum

Em linhas gerais, estamos falando sobre *Sprint Planning*, *Daily stand-up*, *Sprint Review* e *Sprint Retrospective*. Do ponto de vista do conhecimento, eles são fáceis de entender simplesmente lendo o Guia do Scrum. Apenas alguns parágrafos e é isso..., certo? Não tão rápido! Colocar em prática é a parte complicada.

Do ponto de vista prático, essas reuniões são muito mais difíceis. Vamos começar entendendo a diferença entre um ritual e um evento.

- **Ritual:** é algo firmemente enraizado em uma tradição particular com um significado quase espiritual ou mágico;
- **Evento:** é algo que simplesmente ocorre (planejado ou não);
- **Scrum:** é ao mesmo tempo muito pragmático e profundamente enraizado em práticas estabelecidas por profissionais ao longo dos tempos.

Profissionais altamente qualificados e experientes, frequentemente no SCRUM, interagem de uma maneira que é difícil de entender para pessoas que não estão

diretamente envolvidas no processo. Algumas palavras são pronunciadas e todos os presentes entendem o que está acontecendo e o que precisa ser feito. Os gerentes ficarão frustrados se quiserem influenciar. Nesse sentido, ritual é um nome muito bom para utilizarmos;

- **Sprint Planning:** o planejamento de cada *sprint* inicia o próprio *Sprint* e é onde o Dono do Produto e a equipe de Desenvolvimento discutem quais Itens do *Backlog* do Produto serão incluídos no *Sprint*. Embora o *Product Owner* tenha o direito de priorizar cada PBI para possível inclusão no *Sprint*, a equipe de Desenvolvimento é incentivada a responder, levantar questões e retroceder quando necessário. A Equipe de Desenvolvimento, então, prevê quantos itens do *backlog* pode entregar no *Sprint*, dado seu conhecimento de velocidade, recursos e quaisquer fatores que possam influenciar o tempo e os recursos que a equipe tem disponíveis. O resultado da Reunião de Planejamento do *Sprint* é obter um Objetivo e um *Backlog* do *Sprint* com o qual todos concordem, que seja realista e alcançável.

Se não ficou claro para você, vamos ver o que ocorre:

- » um gerente ou representante do cliente apresenta o que deve ser entregue no final de um *sprint* de quatro semanas;
- » a equipe de desenvolvedores planejará em quatro horas quais prioridades serão entregues no *sprint* e como o farão;
- » os gerentes devem se preocupar apenas com que um plano (*backlog*) seja feito com os itens certos;
- » no processo, eles só podem ajudar removendo impedimentos ou facilitando o trabalho.

- **Daily Meeting:** tem um tempo de 15 minutos. Levantar-se não é obrigatório. No entanto, muitas equipes consideram esta uma técnica útil para manter a reunião curta e direta. O *Daily Meeting* é uma oportunidade para a Equipe de Desenvolvimento fazer *check-in*, avaliar o progresso no sentido de atingir a Meta do *Sprint* e revisar e planejar suas atividades para as próximas 24 horas. Lembre-se das 3 perguntas mágicas e mantenha o foco:

- » o que você realizou desde a última reunião?
- » no que você está trabalhando até a próxima reunião?
- » o que está atrapalhando ou impedindo você de fazer seu trabalho?

- **Sprint Review:** em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e, em seguida, sintoniza e ajusta seu comportamento de acordo. Esse evento acontece ao final do *Sprint*. Ele permite a você a oportunidade de mostrar o Incremento de software pronto às partes interessadas, como: clientes, gerência e qualquer pessoa considerada relevante e interessada. Além de demonstrar recursos de trabalho produzidos durante o *Sprint*, você também está atrás de feedback útil que pode ser incorporado ao *Product Backlog*, que pode ajudar a orientar o trabalho para *sprints* futuros;

- **Sprint Retrospective:** é quando a equipe Scrum analisa o que pode ser melhorado em *Sprints* futuros e como deve ser feito. Não importa o quanto bom seja o time Scrum, sempre haverá oportunidade de melhorar e essa reunião dá ao time um tempo dedicado para identificar, discutir e planejar isso. Todos devem participar: Time de Desenvolvimento, o *Scrum Master* e o *Product Owner*.

Controle Ágil de Projeto

O controle de projeto ágil é preponderantemente feito utilizando-se ferramentas visuais como quadros e *dashboards*.

Depois que a reunião de planejamento do *sprint* é concluída e a equipe assume um compromisso, a equipe começa a monitorar seu progresso usando radiadores de informação altamente visíveis. Esses radiadores incluem o gráfico de *burndown* e o quadro de tarefas.

Elaboração dos Gráficos de Controle

Burndown

É uma representação gráfica do trabalho restante em função do tempo. É frequentemente usado em metodologias ágeis de desenvolvimento de software, como *Scrum*. No entanto, os gráficos queimados podem ser aplicados a qualquer projeto que contenha um progresso mensurável ao longo do tempo. Normalmente, em um gráfico desse, o trabalho notável geralmente está no eixo vertical, com o tempo ao longo da horizontal. É útil para prever quando todo o trabalho será concluído. No *Daily Scrum*, a equipe de desenvolvimento atualiza o gráfico e plota o trabalho restante do dia. Um quadro de *burndown* é praticamente mandatório para um time *Scrum* pelos seguintes motivos principais:

- Monitorar o aumento (quando tem mais coisa para fazer que o inicialmente levantado) do escopo do projeto;
- Manter a equipe funcionando dentro do cronograma (veja bem, cronograma em *Scrum* é relativo; se o trabalho foi fatiado em 10 *sprints* de 15 dias, é o tempo que temos de projeto independentemente do trabalho adicional que entre e que a equipe *Scrum*, inclusive, tem o direito de barrar e construir um *road map* pós-entrega, se for o caso, e em comum acordo com o dono do produto e negociações com os clientes);
- Comparar o trabalho planejado com a progressão da equipe.

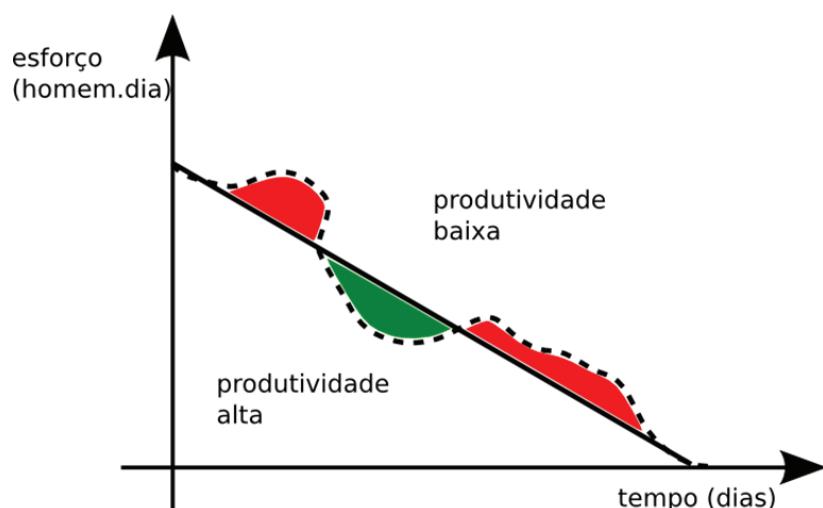


Figura 6 – Gráfico de *Burndown* com dois momentos de baixa produtividade (vermelho) e um momento de alta produtividade (verde)

Fonte: Reprodução



O Gráfico de *Burndown* facilita o acompanhamento da produtividade da equipe durante a iteração. Se um desvio acima da reta ideal ocorre, significa que a equipe está produzindo menos do que deveria e que, por isso, pode não acabar as tarefas designadas para a iteração até o seu final. Por outro lado, se o desvio ocorre abaixo da reta ideal, a equipe está produzindo a mais do que o previsto, o que pode significar uma subestimação da produtividade da equipe ou uma superestimação de complexidade das tarefas designadas para a iteração. Disponível em: <https://bit.ly/3oErniD>

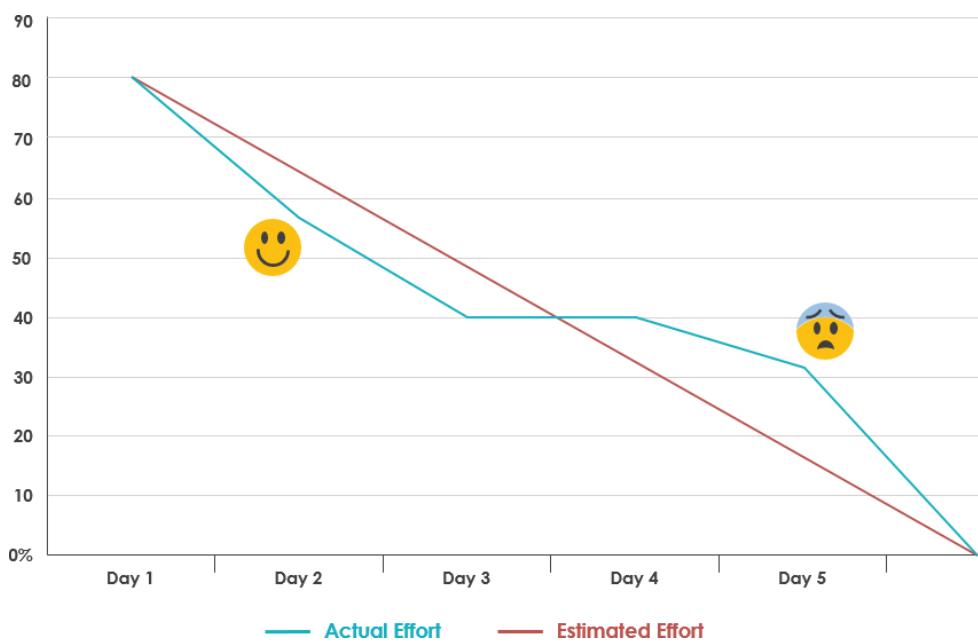


Figura 7 – Outro tipo de gráfico de *burndown*

Fonte: Reprodução



Ele consiste em: “1. Marca-se no eixo Y (vertical) o somatório das estimativas de esforço das atividades do *Sprint*; 2. Marca-se no eixo X (horizontal) o total de dias de trabalho do *Sprint*; 3. Traça-se uma linha reta ligando estes dois pontos (linha tracejada da figura ao lado). Ela representa a meta diária de avanço; 4. Dia após dia, verifica-se no quadro *Scrum* as atividades que foram concluídas, e marca-se no Gráfico *Burndown* a quantidade de esforço restante até o fim do *Sprint*; 5. A meta é chegar a zero no final do *Sprint*. Entretanto, o maior desafio acaba sendo a estimativa do tempo para completar cada tarefa. Segundo a própria teoria do *Scrum*, o ser humano é ruim em estimar tempo. Por isso, originalmente no *Scrum*, a estimativa do “tamanho” do projeto é feita em termos do esforço despendido, por meio de uma medida relativa e não absoluta! No *Scrum*, utilizamos um comparativo entre as entregas do projeto para determinar o esforço de cada uma delas.

Disponível em: <https://bit.ly/3j8mop3>

Burnup

Um *Burnup Chart* é uma ferramenta usada para rastrear quanto trabalho foi concluído e mostrar a quantidade total de trabalho para um projeto ou iteração. É usado por vários métodos de engenharia de *software*, mas esses gráficos são particularmente

populares no gerenciamento de projetos de *software Ágil* e *Scrum*. O trabalho concluído e o trabalho total são mostrados no eixo vertical em quaisquer unidades que uma equipe de projeto considere que funcionam melhor, ou seja, horas de trabalho, dias de trabalho, pontos de história ou qualquer outra unidade de trabalho. O acesso horizontal exibe o tempo, geralmente em dias, semanas ou iterações/*sprints*.

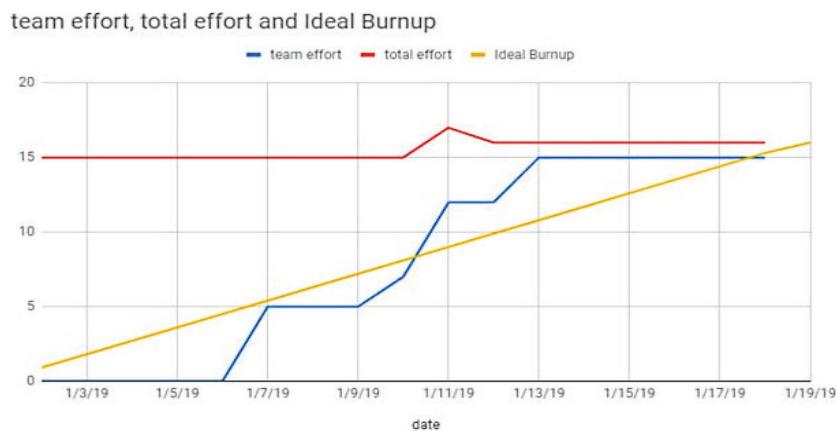


Figura 8 – Exemplo de gráfico de *burnup*

Fonte: Reprodução



Linha Vermelha: esforço total ou linha de marco trata do esforço total necessário para atingir a meta da equipe (um *sprint* ou lançamento); **Linha Azul:** esforço da equipe ou linha de trabalho concluído, mostra o progresso da equipe em direção à meta, ao longo de um determinado período (normalmente, pontos de história concluídos); **pôr fim a linha amarela:** Linha ideal, ou seja, o ritmo de trabalho ideal necessário para cumprir o prazo do projeto.

Disponível em: <https://bit.ly/2MMKr05>

O gráfico de *burnup* é atualizado ao final de cada unidade de tempo. Depois de atualizado, o gráfico mostra o progresso real da equipe. A linha de esforço da equipe sobe de baixo para cima em direção à linha de esforço total. Quando as duas linhas se encontram, o objetivo da equipe é alcançado.

Uma equipe pode ver quanto trabalho resta em seu projeto observando a distância entre a linha de esforço da equipe e a linha de esforço total. O gráfico simplifica o rastreamento, a modificação e, de preferência, melhora o processo de trabalho de uma equipe.

Diferenças: um gráfico *burndown* mostra a quantidade de trabalho restante em um projeto (o esforço restante), enquanto um gráfico *burnup* mostra quanto trabalho foi concluído e o escopo total do projeto.

Em um gráfico de *burndown*, a linha vai de cima para baixo à medida que a equipe progride, enquanto em um gráfico de *burnup* a linha sobe de baixo para cima. Ambos os gráficos usam os mesmos eixos.

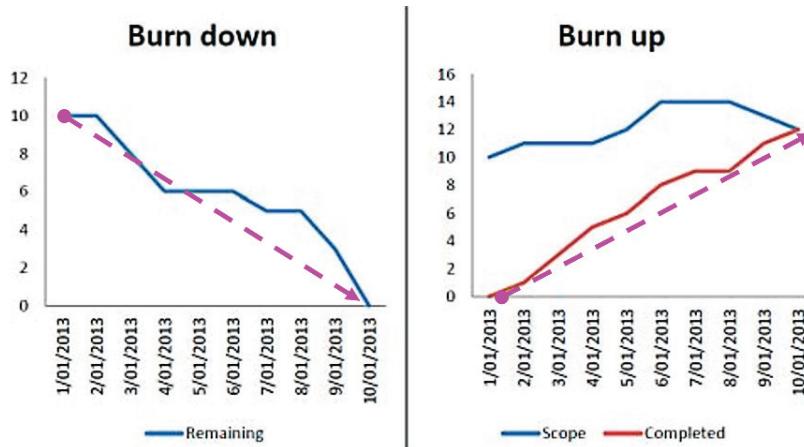


Figura 9 – Comparativo entre gráficos de burndown x burnup

Fonte: Reprodução



Burn-down são simples e representam uma única linha correndo para o zero à medida que o projeto é concluído. Qualquer pessoa pode entender este gráfico e não precisa de explicação. No entanto, pode ocultar informações importantes – por exemplo, os efeitos da alteração do escopo. O *Burn-up* rastreia o trabalho concluído e o trabalho total com duas linhas separadas, ao contrário de um gráfico *burn-down*, que os combina em uma única linha. A linha de trabalho total comunica informações importantes – por exemplo, o projeto ainda não foi concluído porque o trabalho é lento para ser feito ou muito trabalho novo está sendo adicionado. Essas informações podem ser cruciais para diagnosticar e retificar problemas com um projeto. Disponível em: <https://bit.ly/3pClt1q>

Uso de Quadro Kanban

Uma abordagem visual para gerenciamento de projetos em que as equipes criam representações físicas de suas tarefas, geralmente usando notas adesivas em quadros brancos ou por meio de aplicativos *online*. As tarefas são movidas por estágios predeterminados para rastrear o progresso e identificar obstáculos comuns.

O quadro de tarefas ou quadro KANBAN é usado pela equipe para acompanhar o andamento das tarefas de cada recurso. As colunas mínimas usadas são Tarefas a fazer, Tarefas em progresso e Tarefas concluídas.

Eu, particularmente, gosto muito de colocar as seguintes colunas (história, a fazer, em progresso, a verificar, a validar e pronto). As equipes terão sua reunião scrum diária no quadro de tarefas e moverão os itens ao longo do quadro ao declarar o que fizeram ontem, o que planejam fazer hoje e contra quais obstáculos estão lutando.

Histórias	A Fazer	Em Progresso	Para Verificar	Para Validar	Pronto
HU1	COD1 TST9 DOC5 DSG3 BD12 COD5	COD7 COD9	TST3	HML5 HML1	HML6 COD21 TST2 COD26 TST1 COD18
HU25	COD8 TST8 TST11 COD4	COD3	TST13	HML51	TST31

Figura 10 – Exemplo de quadro de tarefas para um projeto de desenvolvimento de software

Fonte: Acervo do conteudista

Você percebe que estamos desenvolvendo duas histórias e a decomposição delas em tarefas em “A FAZER”.

Um conceito importante em um quadro *Kanban* é o de *WIP – Work in Progress* ou Trabalho em progresso. Ele representa o número de tarefas sendo trabalhadas por um indivíduo ou equipe ao mesmo tempo. Os limites de *WIP* são indicados em um quadro e podem ser definidos para um processo *Kanban* completo ou por etapa. Implementar limites de *WIP* é uma prática essencial e isso permite que as equipes estabilizem o trabalho e aumentem a previsibilidade, elementos cruciais para estabelecer um sistema baseado em *pull* ou puxado. Os limites de *WIP* são identificados pela equipe responsável pelo fluxo de trabalho.

São úteis para:

- Ensinar as equipes a se concentrarem em fazer as coisas;
- Evitar que as tarefas se acumulem em qualquer etapa do processo;
- Permitir que as equipes conheçam sua verdadeira capacidade;
- Expor bloqueadores, gargalos e ineficiências;
- Ajudar a evitar que as equipes fiquem sobrecarregadas ou relaxadas;
- Forçar as equipes a trabalhar apenas no que podem.

Ao iniciar sua prática, seus limites de *WIP* seriam mais frequentemente uma estimativa. Embora não haja uma regra rígida para definir os limites, a ideia básica é que cada indivíduo trabalhe em uma tarefa por vez. Não se estresse muito ao definir os limites iniciais, já que você está apenas começando a adoção; o limite inicial que você definiu pode não representar com precisão a capacidade da sua equipe e isso é bom. Quanto mais você faz seu trabalho e segue religiosamente seus limites, melhor pode determinar qual limite funciona melhor para sua equipe. Recomendamos que você acompanhe as métricas de desempenho de sua equipe para que possa obter o limite de *WIP* que reflete a capacidade de sua equipe.

Material Complementar

Indicações para saber mais sobre os assuntos abordados nesta Unidade:

▶ Vídeos

Kanban no Jira 2020: como configurar o KANBAN BOARD no JIRA SOFTWARE em 2020 (Tutorial completo)

<https://youtu.be/tsCLad7i17A>

Gestão ágil de projetos com os superpoderes do Trello

<https://youtu.be/1o9BOMAKBRE>

Gráfico burndown Scrum Excel

<https://youtu.be/LVNZiD5kX7w>

Agile Minute # 27 – Previsibilidade com BurnUp chart

<https://youtu.be/PqRa0I9-RiA>



Cruzeiro do Sul
Educacional