UniFil
EDUCAÇÃO A DISTÂNCIA

Unidade 2

Metodologias Ágeis Lean, Kanban e Extreme Programming

Abertura

Olá, amigo(a) discente! Seja bem-vindo(a)!

Com a existência de uma grande demanda por desenvolvimento de softwares nas empresas, é necessário compreender e organizar os passos que devem ser seguidos durante esse processo. Seguindo essa visão, nesta unidade vamos compreender e discutir os principais conceitos sobre o processo de desenvolvimento de software.

Na aula 01, vamos estudar os conceitos e práticas da metodologia *Kanban*, visualizar que esse método está ligado ao *Lean* e como esses conhecimentos podem auxiliar durante o desenvolvimento de softwares.

Na aula 02, vamos entrar em detalhes sobre o funcionamento do *Extreme Programming* (XP) uma metodologia altamente aplicada no mercado de desenvolvimento de sistemas, vamos conhecer como ela pode ser aplicada no cotidiano, além do impacto dessa metodologia na entrega final de um software.

Objetivos

- Conhecer os conceitos das metodologias *Extreme Programming*, *Kanban* e *Lean;*
- Conhecer como aplicar na prática os conceitos do *Extreme Programming*, *Kanban* e *Lean*

Conteúdo Programático

Aula 01 – *Lean* e *Kanban* conceitos e aplicação prática

Aula 02 – *Extreme Programming* fundamentos e aplicação prática

Quer assistir às videoaulas em seu celular? Basta apontar a câmera para os **QR Codes** distribuídos neste conteúdo.

Caso necessário, instale um aplicativo de leitura QR Code no celular e efetue o login na sua conta Gmail.

Lean e Kanban conceitos e aplicação prática

Olá, estudante, como vai? Vamos começar agora um estudo muito importante sobre metodologias ágeis. A partir desse momento vamos conhecer as metodologias ágeis mais utilizadas no mercado de Tecnologia da Informação no cenário atual.

A boa aplicação de uma metodologia pode modificar totalmente a experiência de construção de um software. Com o método correto sendo aplicado as chances de entregar um software com qualidade e que gere valor para o usuário é muito maior.

Assim, nesta aula vamos conhecer melhor a metodologia *Kanban,* sua história, conceitos e como podemos aplicar essa metodologia na prática.

Videoaula 1

Utilize o QR Code para assistir!

Assista o vídeo para entender melhor o assunto.

O *Kanban* teve seu início no século XX mais exatamente na década de 50, por volta de 1940 a Toyota começou a aplicar melhorias no seu processo de construção, tendo como foco otimizar o processo e reduzir custos. Essas melhorias se tornaram o que atualmente se conhece como *Lean*.

A ideia era sempre trabalhar com o material necessário, ou seja, cada etapa da construção recebia somente a quantidade de material necessário para construir a sua demanda diária, ou seja, se um carro precisa de 16 parafusos para travar as suas rodas e cada linha de produção fábrica 10 carros por dia, então cada linha deve receber por dia 160 parafusos, com isso o desperdício e os custos são reduzidos.

Dentro desse processo conhecido como *Lean* ou como alguns chamam *Lean Manufacturing*, existem alguns objetivos que são fundamentais que quando se aplica o processo *Lean* se espera alcançar, são eles:

- Otimização e Integração do Sistema de Produção;
- Qualidade;
- Flexibilidade do Processo;
- Compromisso com clientes e fornecedores;
- Produção por demanda;

- Redução de custos.

Como surgiu dentro de uma fábrica, o processo foi aceito e aplicado inicialmente somente na indústria, com o foco de reduzir os desperdícios e com isso melhorar a capacidade de produção e lucro dentro das indústrias.

Além disso, na época foi criado um quadro no chão de fábrica da Toyota, para que assim fosse visível em qual etapa da construção se estava e assim conseguir fornecer o material necessário para concluir esta etapa. Dessa forma, todos visualizavam a necessidade de cada linha de produção, criando transparência dentro do processo de construção.

Mas como é possível aplicar isso durante o desenvolvimento de um software? Bom, durante a construção de um software é necessário conhecer as necessidades do cliente, e organizar um time que conheça a sua capacidade para desse modo conseguir aplicar o *Kanban* e se organizar.

Dessa forma, como no exemplo, cada linha de produção de um carro recebe 160 parafusos por dia, então um time de desenvolvimento precisa saber de quantas pessoas ela precisa para entregar uma tarefa de desenvolvimento.

Assim, o *Kanban* tem seu foco na produtividade e otimização dos processos. Dentro do processo de desenvolvimento de um software o *Kanban* é facilmente aplicado, além de auxiliar na organização do fluxo, ele também descreve quais são as necessidades e qual deve ser o foco da equipe de desenvolvimento naquele momento.

Videoaula 2

Utilize o QR Code para assistir!

Assista o vídeo para entender melhor o assunto.

Existem quatro princípios básicos para a aplicação do método *Kanban*, são eles:

- Aceitar a busca por uma mudança evolutiva;
- Encorajar atos de liderança;
- Começar com o que você já faz;
- Respeitar os títulos / responsabilidades / funções atuais.

Bom, para aplicar o *Kanban* na prática é necessário possuir clareza de como o método se organiza. Para isso existem 2 tipos de quadros, o modelo físico e o modelo digital.

Figura 1. Exemplo quadro Kanban físico

Fonte: KANBANIZE.

O quadro *Kanban* físico é utilizado em algumas indústrias até hoje, pois facilita a locomoção dos cartões sem a necessidade de interagir com alguma plataforma, neste modelo é utilizado *post its* como os cartões além de demandar uma construção de material físico para a visualização.

Neste formato os participantes interagem movendo os cartões dentro das colunas e raias existentes no quadro. Algumas desvantagens são a letra de escrita, o tamanho do papel e até a forma de escrever os cartões. As vantagens são a transparência do processo de construção do quadro e a integração que é de modo livre entre todos do time.

Figura 2. Exemplo quadro Kanban digital

Fonte: KANBANIZE.

O quadro *Kanban* digital é muito utilizado por empresas de tecnologia e em indústrias que possuem recursos tecnológicos. Nesse caso cada integrante do time precisa possuir acesso a

plataforma onde está construído o quadro, cada cartão possui um número de identificação o que facilita nas conversas dentro do time.

Com tudo é necessário conhecer como um quadro *Kanban* é construído, sua estrutura é composta por:

- **Colunas**: cada uma possui um título que está vinculado a "fase" da construção que está sendo desenvolvida naquele cartão naquele momento.
- **Cartão**: cada cartão descreve uma demanda que possui informações de trabalho que precisam ser concluídas.
- **Raias**: alguns quadros *Kanban* possuem raias dentro do quadro que são divisões horizontais que ajudam a otimizar o fluxo de trabalho.
- **WIP (*Work In Progress*)**: Trabalho em Progresso é a quantidade de trabalho sendo iniciada ou a quantidade de cartões que estão em cada fase, sendo que estes também podem ser limitados para facilitar a organização do time.

Videoaula 3

Utilize o QR Code para assistir!

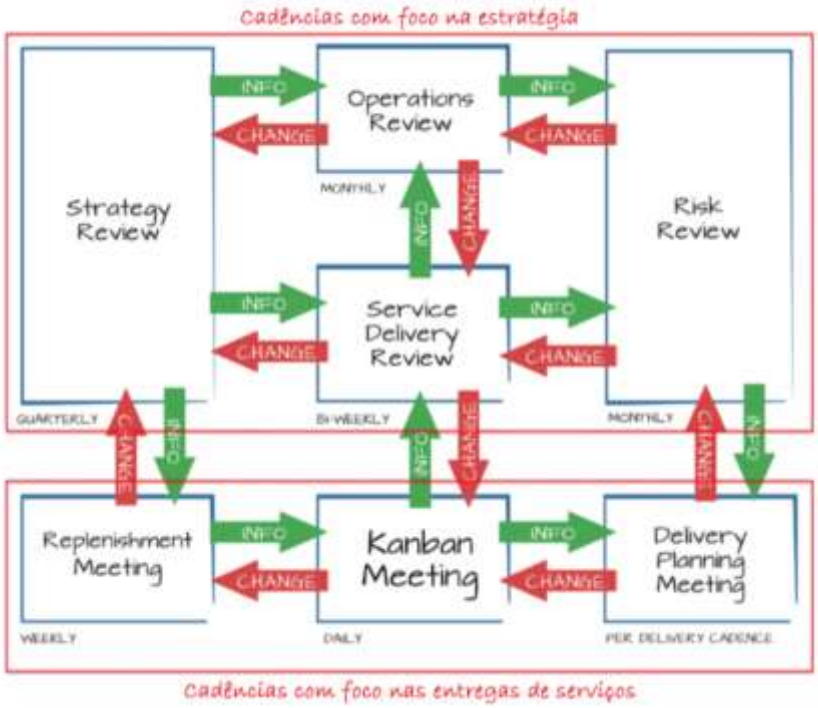
Assista o vídeo para entender melhor o assunto.

A metodologia *Kanban* tem em seu escopo algumas reuniões cíclicas, que servem para guiar o processo de mudança e evolução de um produto e viabilizar sua entrega. Essas reuniões o *Kanban* chama de cadências, são no total 7, sendo elas:

- **Revisão de Risco** – mensal: é um encontro onde o time é convidado a revisitar os pontos de riscos presentes em suas demandas e até mesmo os prazos.
- **Revisão de Operações** – mensal: a ideia é revisar as operações aplicadas no último mês.
- **Revisão de Estratégia** – trimestral: reunião com o objetivo de esclarecer os pontos de estratégia que o time aplicou nos últimos 3 meses de desenvolvimento.
- **Revisão de Entrega de Serviço** – semanal: é a revisão de como foi o processo para entregar um determinado serviço, aqui são levantados pontos como o que deu certo e o que não deu dentro da entrega.
- **Reunião de Reabastecimento** – semanal: reunião tem como objetivo sempre alimentar o quadro *Kanban* com novas atividades, para que a equipe sempre possua insumo de demandas em seu quadro.

- **Reuniões de Atualização** – diária: encontro onde toda a equipe se reúne de modo rápido com o objetivo de manter todos atualizados de suas demandas onde também são todos informados de mudanças necessárias ou impedimentos que surgiram.
- **Reunião de Planejamento de Entrega** – variável de acordo com as entregas: reunião com a proposta de organizar como será o processo de entrega de uma determinada demanda desenvolvida pelo time.

Figura 3. Cadências Kanban



Fonte: disponível em: https://bit.ly/3tbhT3U. Acesso em: 14 mar. 2022.

As cadências são divididas em 2 grandes grupos, sendo eles:

- **Serviços**: são as cadências que estão relacionadas às entregas de demandas que impactam os serviços realizados no dia a dia.
- **Estratégia**: são aquelas que possuem seu foco no planejamento e estratégia de como o desenvolvimento do método pode gerar um novo caminho dentro da empresa e quais são os passos futuros olhando para os aprendizados do passado.

Além disso, a proposta é com esses encontros sempre desenvolver uma melhoria contínua dentro do processo da metodologia.

Dessa forma existem também os *feedbacks´s* que servem para representar as informações que circularam dentro dessas 7 cadências, com a aplicação desses *feedbacks´s* a equipe consegue

encontrar seus pontos fortes e fracos para assim aplicar seu esforço e evoluir esses pontos, melhorando não só o método *Kanban* mas os profissionais que o aplicam no cotidiano.

Extreme Programming

Olá, aluno, tudo bem? Nesta aula vamos conhecer melhor o *Extreme Programming* (XP), vamos descobrir juntos seus princípios e valores e como essa metodologia vem mudando a forma de desenvolvimento de software nos últimos 25 anos.

O XP tem sua origem em 1996 e foi criado por Kent Beck ao se deparar com uma proposta de desenvolver um software gigante que para substituir um software já existente (legado), com isso foi aplicado dentro deste projeto em 1996 os valores e princípios do XP e desde então empresas do mundo inteiro aplicam esses conceitos dentro das suas equipes.

Segundo Beck o XP é uma maneira leve, eficiente, de baixo risco, flexível, previsível, científico e divertido de desenvolver softwares, além de desenvolver um software o XP leva a proposta de tornar a construção de um sistema em algo legal para todos os envolvidos.

Videoaula 1

Utilize o QR Code para assistir!

Assista o vídeo para entender melhor o assunto.

Com isso esse tipo de metodologia é recomendado para:

- Projetos com requisitos vagos e que mudam frequentemente;
- Desenvolvimento Orientado a Objetos;
- Equipes Pequenas (não maior a 12 pessoas);
- Desenvolvimento Incremental – Iterativo, com versões intermediárias.

Valores XP

Figura 4. Organização e vínculos XP

Fonte: imagem da internet.

A Figura 4 demonstra os fluxos de valores defendidos dentro do *Extreme Programming*, sendo eles 5 no total:

- **Comunicação**: com isso, o time conseguirá atuar em conjunto nas atividades, sempre alguém do time vai saber a solução do seu problema, mas você precisa avisar, em um time XP é necessário sempre avaliar a comunicação e possuir de forma clara que ela é uma parte do processo.
- **Simplicidade**: sempre saber para onde você gostaria de ir, quais objetivos quer alcançar e assim buscar propor soluções simples e ir evoluindo a solução, faça o simples primeiro.
- **Feedback**: se esforçar para dar um feedback rápido é uma característica bem forte em um time XP, com isso também é necessário opinar sobre novas ideias, qualidade do código fonte, dizer se os testes foram fáceis ou difíceis de implementar e quais problemas a equipe identificou.
- **Coragem**: Não é necessário ser um policial ou segurança para ter coragem, o time precisa estar preparado para tomar decisões críticas de modo rápido e com isso até mesmo jogar parte do código fora ou até mesmo escrever pouca documentação, mas é importante lembrar que coragem não é inconsistência é preciso ser equilibrado.
- **Respeito**: respeitar as pessoas envolvidas, o projeto, respeitar o que cada um faz e compreender que o desenvolvimento de software só é possível, pois existe respeito entre todos e assim um time comprometido e responsável.

Mudanças acontecem todos os dias em projetos de desenvolvimento de software, uma equipe com os valores do *Extreme Programming* deve possuir a inteligência e estar preparados para essas mudanças, para assim aplicar fortemente esses valores e dessa forma, realizar mudanças harmoniosas e sem erros, tomando as decisões corretas.

Videoaula 2

Utilize o QR Code para assistir!

Assista o vídeo para entender melhor o assunto.

Práticas XP

Além dos valores o XP também reúne um conjunto de práticas que são propostas, sendo elas:

- *Planning Game* – Jogo do Planejamento
- *Standup Meeting* – Reunião em pé
- *Pair Programming* – Programação em Pares
- *Test Driven Development* – Desenvolvimento Dirigido por Testes
- *Refactoring* – Refatoração
- *Shared Code* – Código Compartilhado/Coletivo
- *Coding Standards* – Código padronizado
- *Simple Design* – Design (Desenho) Simples
- *Metaphor* – Metáfora do Produto
- 40-*hour week* – 40h semanais / Ritmo Sustentável
- *Continuos Integration* – Integração Contínua
- *Short Releases* – Releases Curtos

Diferente de outras metodologias, no XP não são ritos, são práticas e propostas, desse modo nem toda prática é o resultado de uma reunião nem decisão técnica.

***Planning Game* – Jogo do Planejamento**

Nesta prática o foco é sempre agregar valor no software para o cliente, a proposta é sempre ocorrer no início de uma iteração ou *release*.

- *Releases* (8 semanas): Entrega de módulo do software que represente valor.

- Iteração (2 semanas): Implementação de conjunto de funcionalidades.

Nessa prática, o cliente é o responsável por priorizar as entregas da próxima *release* e suas iterações.

***Standup Meeting* – Reunião em pé**

Figura 5. Standup Meeting XP

Fonte: imagem da internet.

A prática *Standup Meeting* tem como seu objetivo atualizar todos os integrantes do time sobre os fatos que estão ocorrendo, dessa forma a frequência dessa prática deve ser diária. A reunião deve ser rápida e objetiva, por esse motivo precisa ser realizada de pé.

A dica para essa prática é sempre responder essas perguntas:

- O que fiz de ontem para hoje?
- O que irei fazer amanhã?
- Estou com alguma dificuldade?
- Qual o valor agregado da tarefa que estou realizando?

Ao responder essas perguntas, cada integrante do time consegue possibilitar a toda a equipe conhecer o que está sendo feito e como isso vai impactar a entrega final, lembrando que a comunicação é um dos valores do XP.

Fonte: imagem da internet.

Figura 6. Pair Programming XP

***Pair Programming* – Programação em Pares**

A prática de programar em pares é uma proposta do *Extreme Programming*, mas também é encontrada em outras metodologias, a ideia é programar com mais um colega ao seu lado sendo como um navegador ou copiloto.

Pode-se associar essa prática ao de pilotos de avião, sendo o programador o capitão que é o responsável por realizar as operações e codificar e ao colega ao lado a posição de copiloto, aquele que auxilia o capitão e realiza observações e dicas para qual melhor caminho seguir.

Essa prática promove a iteração, respeito e o *feedback* rápido entre os participantes da mesma equipe, além de auxiliar no nivelamento do conhecimento, é uma prática bastante utilizada no mercado.

Videoaula 3

Utilize o QR Code para assistir!

Assista o vídeo para entender melhor o assunto.

***Test Driven Development* – Desenvolvimento Dirigido por Testes**

A ideia do *Test Driven Development* (TDD) é construir o teste antes mesmo de criar o código, essa prática favorece o conhecimento de negócio das demandas que estão sendo desenvolvidas, dessa forma deixando o desenvolvedor com mais domínio sobre as regras de negócio que precisam ser implementadas.

Dessa forma, a ideia desse modelo de desenvolvimento é sempre construir o teste e depois construir o código de uma forma que ao finalizar ele e executar o teste, ele retorne com sucesso.

***Refactoring* – Refatoração**

A prática de refatoração está totalmente interligada com o valor da simplicidade, com a ideia de sempre desenvolver soluções simples e depois ir implementando melhorias.

A melhoria contínua no código ocorre por intermédio da refatoração e assim mantendo sempre o software com o código limpo e de qualidade, dessa forma facilitando a sustentação do software no futuro.

***Shared Code* – Código Compartilhado/Coletivo**

A construção do software precisa ser compartilhada com toda a equipe, dessa forma todos conseguem se ajudar e ocorre o nivelamento do conhecimento entre todos do time, um integrante consegue revisar o código do outro além de programar em pares, essa prática fortalece a comunicação e o respeito entre toda a equipe.

Para essa prática é fortemente utilizada uma ferramenta de desenvolvimento chamada *GIT*, pois auxilia no versionamento e no compartilhamento de código.

Indicação de Leitura

GIT. Sobre. Disponível em: https://git-scm.com/about

***Coding Standards* – Código padronizado**

Para facilitar a organização do time e do software, a prática de *Coding Standards* consiste em padronizar o código e sua estrutura. Para auxiliar nessa padronização existem modelos já validados como de maneira de construir o código ou até mesmo de estrutura do sistema.

Para auxiliar os padrões de desenvolvimento existem os designs *pattern,* alguns são até utilizados pelos desenvolvedores sem saber, além deles também existem padrões de projetos como a arquitetura hexagonal, essa padronização não está vinculada à linguagem de programação específica, pois essa padronização pode ser agregada a qualquer tipo de projeto de software.

Indicação de Vídeo:

O que são Design Patterns? Disponível em: https://www.youtube.com/watch?v=8vq2QB4ogKM. Acesso em: 14 mar. 2022.

***Simple Design* – Desenho Simples**

Essa prática traz a proposta de sempre pensar no modelo simples antes, tanto em questão de experiência do usuário quanto na questão de codificação, ou seja, sempre desenhar um esboço do que precisa ser feito de maneira simples.

O foco é sempre agregar valor para o cliente de maneira simples e de fácil aplicação, com isso, antes mesmo de desenvolver é desenhado o que vai ser desenvolvido, isso ajuda o time inteiro a compreender melhor cada funcionalidade.

***Metaphor* – Metáfora do Produto**

A prática de construir uma metáfora de produto, consiste em compreender a necessidade do cliente e para auxiliar no entendimento do time, construir uma metáfora dessa necessidade com acontecimentos do cotidiano.

Para construir essa prática é preciso estar com a mente oxigenada e assim utilizar a criatividade para conseguir construir uma metáfora que não utilize de computadores, para assim levar o time a buscar outras referências durante o desenvolvimento.

40-*hour week* – 40h semanais / Ritmo Sustentável

Com a prática de ritmo sustentável, é sempre bom lembrar que o time precisa descansar para assim continuar a desenvolver um software de qualidade, para que isso ocorra o XP propõem que o time trabalhe no máximo 40 horas por semana e não apoia a prática de realizar horas extras.

Com isso estabelecer um ritmo de desenvolvimento que seja sustentável dentro da equipe e fazer com que os integrantes se sintam bem em estar dentro do projeto é essencial, pessoas boas constroem código bons.

***Continuos Integration* – Integração Contínua**

A integração contínua é uma necessidade durante a construção do software, pois como mudanças e refatorações são feitas a todo momento possui um ambiente de desenvolvimento onde a integração é eficaz, para dessa forma facilitar e manter a qualidade do software em desenvolvimento.

O processo consiste em adicionar uma parte do desenvolvimento em um ambiente comum e assim validar todos os testes do sistema, dessa forma cada integração não irá acrescentar erros ao sistema, para facilitar esse processo o mercado utiliza de vários softwares como: *CruiseControl*, Jenkins, Hudson, Bamboo, *BuildMaster*, *Teamcity*.

***Short Releases* – Releases Curtos**

Com essa prática a ideia é que o cliente sempre tenha acesso às novas integrações o mais rápido possível, e assim agregando valor ao cliente. Com pequenas entregas o cliente fica mais satisfeito e surgem novas alterações no software.

Essas entregas precisam também ser contínuas, ou seja, a cada 2 meses uma nova versão. Dessa forma favorecendo o *feedback* por parte do cliente de forma precoce. Diminuindo atrasos em entregas, aumenta assertividade, e aumenta a taxa de aproveitamento do produto.

Com a aplicação do *Extreme Programming* não resolverá todos os problemas, porém com essa metodologia aumentam as chances de que o software seja construído com sucesso e que o cliente irá ficar satisfeito, com as entregas de valor que o software oferece.

Aplicar o XP no dia a dia, é desafiador pois algumas empresas ainda estão acostumadas a trabalhar com o formato de metodologias tradicionais. Quando isso acontece é necessário ter cuidado com a forma de seguir o desenvolvimento do software, a proposta é ser leve e ao mesmo tempo saudável tanto para o cliente quanto para o time de desenvolvimento.

Seguir os valores e práticas apresentados nesta aula favorecem a construção de um sistema que gere resultado e demonstra o potencial de cada integrante do time de desenvolvimento, por isso, mesmo que a metodologia definida no projeto não seja o XP isso não quer dizer que cada pessoa não possa aplicar alguns de seus valores e práticas no seu cotidiano.

Fórum Avaliativo

Prezado(a) aluno(a)!
Agora convido você para realizar a primeira atividade avaliativa da disciplina: **Fórum de Discussões**.

Para participar, você deverá clicar em “**Discussões**” no “**Menu Lateral**” e acessar.
Lembre-se: Após a data estipulada o Fórum será encerrado e não será mais permitido participar.

Sua contribuição é muito importante. Bons estudos!

Encerramento

Chegamos ao final da nossa primeira unidade, onde você teve contato com os primeiros conceitos e ferramentas do *Lean*, *Kanban* e do XP. Com certeza, muito do que foi mostrado aqui você já tinha uma pequena noção. Espero que você, aluno, tenha aprendido um pouco mais sobre metodologias ágeis.

No mundo dos softwares há muitas metodologias de desenvolvimento que podem ser melhores ou piores, no final tudo depende da demanda e do produto que deve ser entregue. Na aula 1 visualizamos alguns conceitos e como funciona as metodologias *Lean* e *Kanban* e como essas metodologias surgiram e estão relacionadas.

Na aula 2, foi discutido e apresentado a metodologia *Extreme Programming* um método ágil fortemente utilizado no mercado de tecnologia, também foi analisado seus valores e práticas e como esses podem auxiliar no desenvolvimento de um software.

Juntando as duas aulas teremos conhecimento inicial sobre algumas metodologias ágeis. Caro aluno, pense um pouco como os conhecimentos desta unidade podem te ajudar em softwares que você desenvolve em seu dia a dia. Até a próxima.

Referências

PRESSMAN, Roger S. **Engenharia de Software**. 1. ed. São Paulo: Person, 2011.

KANBOARD. **Página inicial**. Disponível em: https://kanboard.org/. Acesso em: 30 de set. 2021.

Quer assistir às videoaulas em seu celular? Basta apontar a câmera para os **QR Codes** distribuídos neste conteúdo.

Caso necessário, instale um aplicativo de leitura QR Code no celular e efetue o login na sua conta Gmail.


UniFil
EDUCAÇÃO A DISTÂNCIA
UNIFIL.BR