Unidade 02

Introdução a RUP e sua estrutura

UniFil

Introdução da Unidade

Olá, querido aluno! Como vai? Nessa unidade vamos expandir os nossos conhecimentos sobre o processo de desenvolvimento de *software*, vamos compreender melhor como é a ferramenta RUP.

Na aula 01 iremos detalhar o RUP e seus conceitos, para que dessa forma seja possível aplicar esta ferramenta dentro do processo de desenvolvimento e assim entregar um *software* bem-sucedido.

Na aula 02, vamos compreender como funciona a estrutura de conteúdo do RUP dentro da modelagem de *softwares* orientado a negócios. Deste modo, formular e desenvolver sistemas que realmente atendam às necessidades do cliente, além disso trabalhar os dados dentro do *software* de modo que se torne possível a construção de um projeto de qualidade e de fácil manutenção de códigos.

Objetivos

- Compreender o funcionamento do RUP;
- Compreender a organização por conteúdo no RUP.

Conteúdo programático

Aula 01 – Introdução ao RUP.
Aula 02 – Organização RUP por conteúdo.

Você poderá, também, **assistir as videoaulas** em seu celular, basta apontar a câmera para os **QRCodes** distribuídos neste conteúdo.

Referências

COPYRIGHT IBM CORP. **Introdução ao RUP**. 2006. Introdução ao RUP. Disponível em: <http://walderson.com/IBM/RUP7/LargeProjects/#core.base_rup/guidances/supportingmaterials/introduction_to_rup_36B63436.html>. Acesso em: 25 fev. 2021.

PRESSMAN, Roger S. **Engenharia de Software**. 1. ed. São Paulo: Person, 2011. 1056 p.

UML**. What is UML**. 2005. Página sobre. Disponível em: <https://www.uml.org/what-is-uml.htm>. Acesso em: 25 fev. 2021.

Aula 01 - Introdução ao RUP

Olá aluno! Na aula anterior tivemos uma introdução sobre o que é um processo de desenvolvimento de *software*, entendemos a importância desse processo e compreendemos as notações UML juntamente com a necessidade de modelar um *software* com uma visão de negócios.

Para auxiliar nesse processo de desenvolvimento é que surge o RUP (*Rational Unified Process*) uma ferramenta que traz consigo três pilares centrais:

a) Conjunto de Filosofias e Princípios para o Desenvolvimento de um *software* bem-sucedido;

b) Uma estrutura de conteúdo e método reutilizável com blocos de construção de processo;

c) Método subjacente e a linguagem do processo.

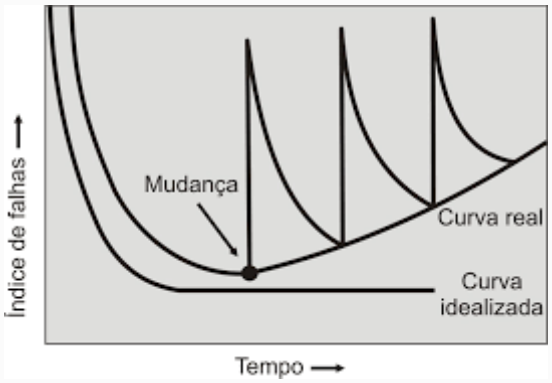
O RUP é uma plataforma *on-line* e interativa, desenvolvida para todos os profissionais que buscam entregar um *software* bem-sucedido e de qualidade. Uma das boas práticas aplicadas pelo RUP é o desenvolvimento interativo e incremental. Sendo assim, o *software* é entregue por partes, e o seu desenvolvimento é organizado de forma interativa.

Um ponto a ser destacado é que o RUP é flexível, podendo ser modificado de acordo com a proposta do projeto. Deste modo, para entender melhor como funciona a organização dentro da ferramenta RUP, primeiro é preciso compreender como funciona a organização baseada em conteúdo, para isso vamos dar uma olhada na arquitetura geral do RUP.

É importante lembrar que tanto o RUP quanto a UML são ferramentas que não estão vinculadas a nenhuma linguagem de programação, portanto, elas podem ser aplicadas a qualquer tipo de projeto de *software*. Com o RUP é possível organizar o processo de desenvolvimento de *software* ao longo do tempo, suas estruturas, seu ciclo de vida e suas iterações, permite a organização do processo baseado em seu conteúdo, tais como: atividades, papeis e artefatos, seguindo em um fluxo evolutivo do *software*.

O RUP sugere que as entregas devem ocorrer de forma incremental, dessa forma a cada tempo é entregue ao cliente uma parte do produto, o que facilita possíveis mudanças no *software* se necessário.

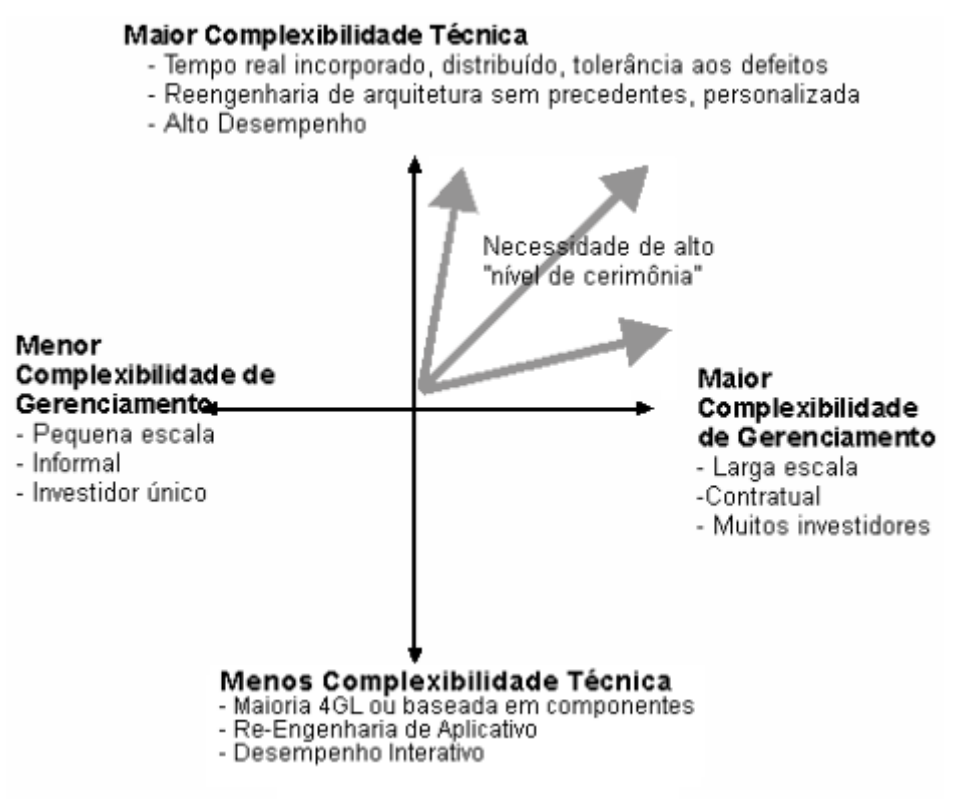
Figura 01 - Curva real de falhas em um *software*



Fonte: Pressman (2011)

Com a Figura 01, podemos observar como mudanças no *software* podem gerar falhas que não estavam previstas no início do projeto. Dessa forma, a cada incremento entregue do produto é possível enxergar com maior agilidade a possibilidade de falhas do sistema. Portanto, a entrega de incrementos com qualidade mitiga a quantidade de possíveis falhas. Sendo assim, para gerar um *software* com qualidade, é necessário se esforçar para entregar incrementos com propriedade.

Figura 02 - Matriz de complexidade



Fonte: RUP

Consultando a Figura 02 podemos visualizar que o RUP deve ser utilizado em projetos onde é necessário um maior nível de complexidade técnica e de gerenciamento, pois auxilia diretamente em processos formais, com pensamento em larga escala, no qual o *software* exige um pensamento complexo e qualificações técnicas da equipe de desenvolvimento.

LEITURA OPCIONAL

Leia todo o capítulo 01 do livro intitulado "**Engenharia de Software"** de Roger S. Pressman, ele aprofunda os conteúdos e está disponível na sua biblioteca Virtual, conforme link abaixo: Disponível em: https://www.academia.edu/40370740/Engenharia_de_Software_Uma_Abordagem_Profi_ssional

Para dar seguimento na nossa disciplina, nesta aula vamos aprofundar nossos conhecimentos sobre a ferramenta RUP, e visualizar como as propostas de negócio afetam o processo de desenvolvimento.

VIDEOAULA 01

Utilize o QRcode para assistir!

Assista a videoaula para compreender melhor o conteúdo.

A ferramenta RUP traz a proposta de ser orientada pelo negócio, desse modo, propõe princípios chaves para o desenvolvimento orientado a negócios, sendo eles:

a) Adaptar o processo;

b) Equilibrar prioridades de Investidores de Competição;

c) Colaborar através de equipes;

d) Demonstrar valor iterativamente;

e) Elevar nível de abstração;

f) Focalizar continuamente na qualidade do *software*.

Esses princípios são na verdade boas práticas que ajudam a garantir que o processo de desenvolvimento ocorra com sucesso, visto que um *software* está relacionado fortemente com o negócio para qual ele está sendo criado. Pensando nesse contexto, nada melhor do que desenvolver o *software* orientado ao seu propósito. O RUP possui alguns conceitos, que quando bem estruturados facilitam a aplicação dessa ferramenta, são eles:

→ **Visão**

É necessário desenvolver uma visão clara sobre o *software* para que desse modo, seja possível entregar um produto que realmente resolva as necessidades do cliente, dentro do RUP a visão nos leva a capturar requisitos de um nível muito alto, e focar nas restrições de design para fornecer um entendimento completo sobre o sistema.

→ **Plano**

Gerencia todo o plano de desenvolvimento do *software* de modo que seja possível prever riscos e custos, além de avaliar seus requisitos, etapas, recursos financeiros e humanos.

→ **Riscos**

É muito importante identificar o mais cedo possível os riscos que envolvem o projeto. Desse modo, será possível mitigar esses riscos para que o projeto consiga ser implementado de maneira harmoniosa.

→ **Casos de Negócio**

Com a análise dos casos de negócio é possível desenvolver um projeto que auxilie o cliente em seu crescimento econômico, pois o projeto busca obter uma visão de lucratividade para o negócio do seu cliente.

→ **Arquitetura**

Para bem desenvolver um *software* é preciso definir uma estrutura de desenvolvimento. Dessa forma, no RUP é escolhido como uma dessas estruturas de arquitetura de *software*, para somente depois disso se iniciar o desenvolvimento propriamente dito. Uma arquitetura de *software* bem definida, afeta diretamente nas alterações futuras que um *software* pode sofrer.

→ **Protótipo**

A criação de um protótipo facilita o processo iterativo de construção do *software*, visto que a cada novo incremento a ser desenvolvido, é esboçado por intermédio do protótipo. Assim, o cliente pode visualizar como vai ficar o seu produto ao final de cada entrega.

→ **Avaliação**

Com uma comunicação clara e contínua, é possível realizar avaliações de cada parte do processo, a fim de gerar uma melhoria contínua, sempre avaliando o foco das entregas e o objetivo final do produto.

→ **Controle de Mudanças**

A cada entrega de incremento podem surgir diferentes necessidades que não estavam mapeadas no início do projeto. Com isso é necessário documentar, controlar e mapear os defeitos e alterações realizadas dentro do projeto.

→ **Suporte ao Usuário**

Quando se entrega um incremento para o usuário, na maioria das vezes é preciso realizar o suporte para a nova alteração ou funcionalidade que o sistema recebeu. O suporte ao usuário está relacionado diretamente com a melhoria contínua, pois é com o retorno do usuário que se torna possível avaliar melhorias no *software*.

→ **Processo**

É necessário que o processo seja flexível, pois um produto sofre diversas alterações até a sua conclusão, a adaptação do processo é um ponto fundamental para a disciplina de ambiente. Com os elementos essenciais bem trabalhados dentro do processo de desenvolvimento do *software*, é possível ter um meio de avaliar, contextualizar e identificar melhorias dentro do projeto a fim de garantir uma entrega bem-sucedida ao usuário final.

LEITURA OPCIONAL

Leia todo capítulo 02 do livro **"Engenharia de Software"** do escritor Roger S. Pressman, ele aprofunda os conteúdos e está disponível na sua biblioteca Virtual. Disponível em: https://www.academia.edu/40370740/Engenharia_de_Software_Uma_Abordagem_Profi_ssional

VIDEOAULA 02

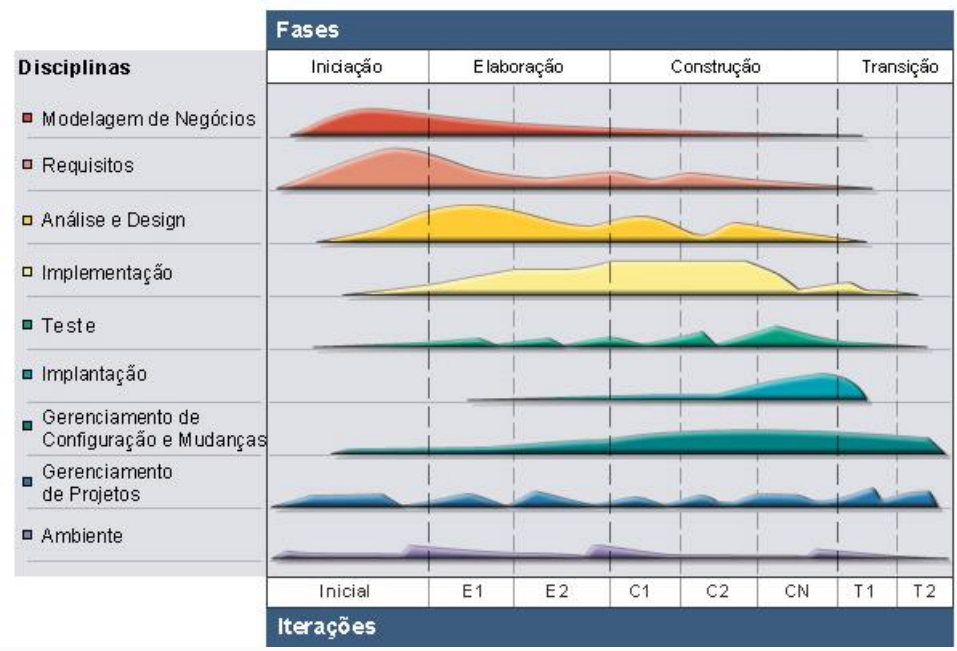
Utilize o QRcode para assistir!

Assista a videoaula para entender melhor o assunto abordado.

Fases do RUP

O RUP possui sua arquitetura geral, onde estão descritas todas as etapas, conteúdos e iterações. Para melhor compreender, visualize essa arquitetura na Figura 03.

Figura 03 - Arquitetura Geral do RUP

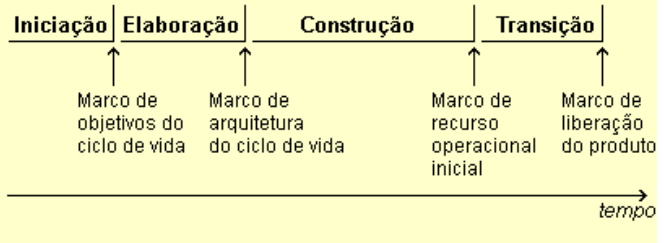


Fonte: RUP

Na Figura 03 é descrita a arquitetura geral do RUP, no ângulo X encontra-se o decorrer do tempo, suas fases e iterações, e no plano Y encontram-se os conteúdos trabalhados dentro do plano de desenvolvimento de *software*, as fases ou como alguns autores gostam de chamar o ciclo de vida do RUP são divididas em quatro fases: iniciação, elaboração, construção e transição.

Cada uma dessas fases é vista como etapas que quando concluídas geram um marco principal, sendo assim, cada fase do processo é um intervalo entre um marco principal e outro, no final de cada fase é realizada uma avaliação para determinar se os objetivos de cada fase foram concluídos com sucesso, somente com uma avaliação satisfatória é que se permite o avanço de fases.

Figura 04 - Fases e marcos de um projeto



Fonte: o autor (2021)

Como já citado, dentro do RUP existem quatro fases, sendo elas: Iniciação, Elaboração, Construção e Transição. Ao final de cada uma dessas fases é gerado um marco do RUP, conforme descreve a figura 04. Para compreender melhor como funciona cada fase, veja abaixo:

→ **Iniciação**

Nesta etapa, o ponto focal é conseguir efetivar o consenso entre os investidores e os objetivos dos ciclos de vida do projeto, nesta fase é necessária uma atenção maior quando os projetos são novos, pois a chance de surgir riscos para o produto é grande.

São alguns objetivos desta fase: gerar escopo do projeto de *software*, descrever casos críticos do sistema, demonstrar ao menos uma opção de arquitetura, estimar custo real do processo de desenvolvimento, calcular imprevistos e preparar o ambiente de suporte do projeto. É nessa fase, que se define o escopo total do projeto, seu contexto, requisitos, custo, planejamento e lucro.

→ **Elaboração**

Nesta etapa, o objetivo é conseguir criar uma linha de base para a arquitetura do sistema, para que ao entrar na fase de construção, o processo de desenvolvimento tenha uma base sólida.

Sendo que só é possível desenvolver essa base a partir da análise de requisitos mais significativos, pois são eles que impactam fortemente no *software*. Nesta etapa também é necessário produzir protótipos evolutivos dos componentes de qualidade do *software*, assim diminuindo os riscos de: troca de requisitos/design, reutilização de componentes, possível demonstração do produto para as partes envolvidas.

→ **Construção**

Dentro dessa etapa é quando são esclarecidos os requisitos restantes e concluído o desenvolvimento do sistema, seguindo a linha de base definida na etapa anterior.

Com isso, dando ênfase também no gerenciamento de recursos e o controle da operação a fim de reduzir os custos e atingir uma ótima qualidade do *software*, gerar as versões do projeto com eficiência, concluir a análise, desenvolvimento e testes de todas as funcionalidades necessárias no sistema. Além de desenvolver de modo iterativo e incremental um produto completo para que consiga realizar a transição para os usuários, avaliar e decidir se os usuários e o ambiente estão prontos para serem implantados.

Ademais, nesta etapa é importante elevar o nível de paralelismo entre as tarefas, pois existem componentes que podem ser desenvolvidos em paralelo, de um modo que mais adiante todo esse processo ocorra de forma orgânica.

→ **Transição**

O Foco desta fase é disponibilizar o *software* para os usuários, aqui podem ocorrer muitas iterações, o que também inclui testar o *software* em *release* e realizar pequenos ajustes com os *feedbacks* fornecidos pelos usuários, além de finalizar o material de suporte para eles, executar os processos para implementação do sistema no ambiente desejado.

Além de realizar treinamento com os usuários, introdução a marketing, distribuição pela equipe de vendas. Enfim, dessa forma toda essa fase é voltada para implementação do produto ao usuário, obtendo o consentimento dos envolvidos e verificando se as *baselines* estão completas.

VIDEOAULA 03

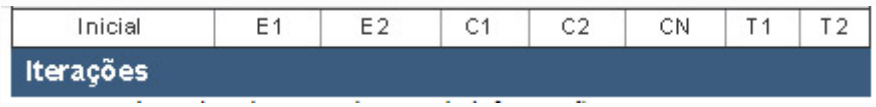
Utilize o QRcode para assistir!

Agora, assista à videoaula para ampliar seus conhecimentos a respeito do assunto.

Interações do RUP

Cada uma destas etapas tem suas iterações, porém é preciso entender o que é e como funciona uma iteração, para melhor compreender, vamos exemplificar:

Figura 05 - Aba de iterações



Fonte: RUP.

Iterações estão presentes em todas as etapas, ela funciona como uma “micro fase” dentro das fases já descritas sobre o RUP, possui suas entregas, planejamento, avaliação e correções conforme a necessidade.

O foco da iteração muda conforme o ciclo do *software* e tem sua duração definida por diversos fatores, como: o tamanho da organização, o tamanho do projeto, familiaridade com o projeto, técnica do projeto (se o projeto é complexo ou simples).

Somente após observar todos os pontos é que será possível calcular o tempo de execução de uma iteração e definir o prazo de entrega das iterações com seus custos e recursos para serem utilizadas dentro de cada uma delas.

Para ponderar o valor de tempo e peso sobre cada iteração é aplicado a regra de *Thumb*, dividindo em iterações com peso, baixo, médio e alto, usando de três a seis iterações por etapa. Para exemplificar:

Tabela 01 - Tabela com regra de Thumb

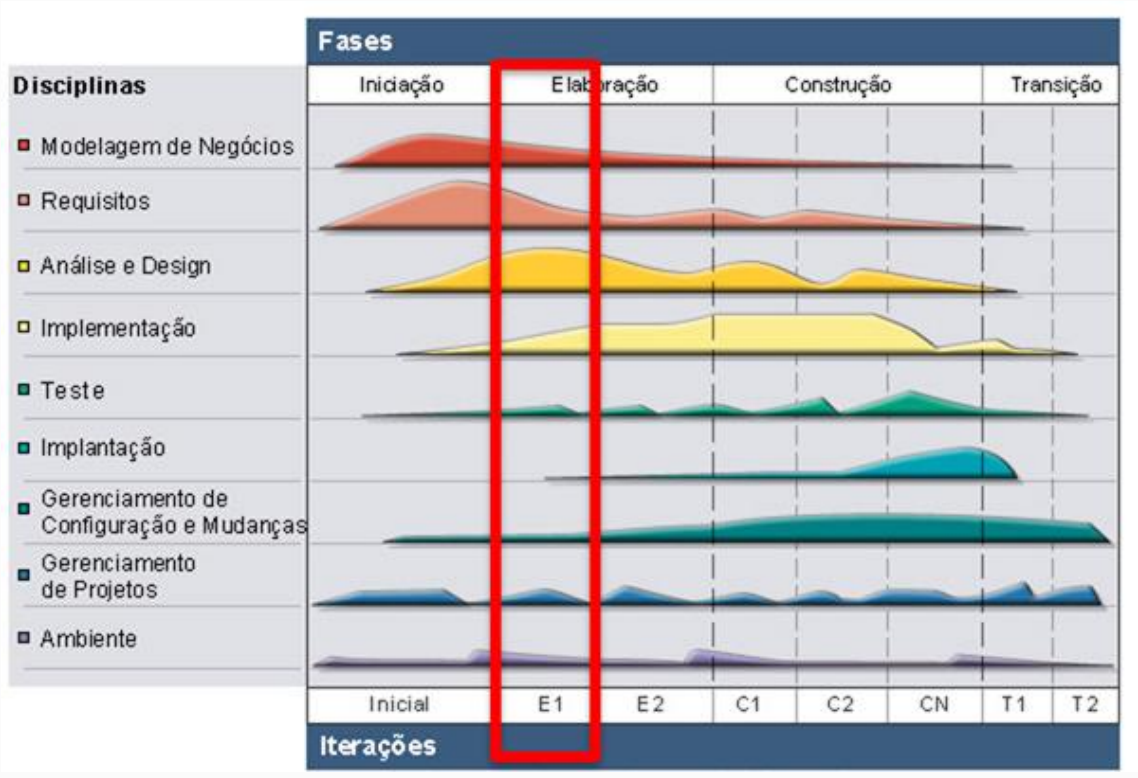
Fase	Baixo	Médio	Alto
Iniciação	0	1	1
Elaboração	1	2	3
Construção	1	2	3
Transição	1	2	2
TOTAL	3	7	9

Fonte: o autor (2021)

Na tabela 01 está descrito como pode ocorrer a divisão de iterações por peso dentro da regra de *Thumb,* sendo assim, é possível tornar visual a quantidade de iterações por fase, dessa forma facilitando a demonstração e compreensão de toda a equipe. A mesma iteração pode estar presente em diversas disciplinas, porém não na mesma fase.

Vale pontuar que uma mesma iteração pode existir em diversas disciplinas, com a própria arquitetura geral do RUP, conforme demonstra a figura 06.

Figura 06 - Uma iteração em diversas fases



Fonte: RUP

Observando a Figura 06 é possível analisar visualmente que a iteração E1 está presente em diversas disciplinas, sendo assim, na **disciplina de implantação** essa iteração tem início e esforço baixo, porém possui grande esforço na disciplina de **análise e *design***. Com isso uma iteração pode ter peso diferente em diversas disciplinas.

Todos esses pontos devem ser considerados no momento de ponderar o peso sobre cada iteração, bem como dados de esforço, tempo, custo, qualidade, além da posição desta iteração dentro do contexto global do projeto. Somente depois de analisar todas essas informações é possível definir um peso para a iteração em questão.

Aula 02 - Organização RUP por conteúdo

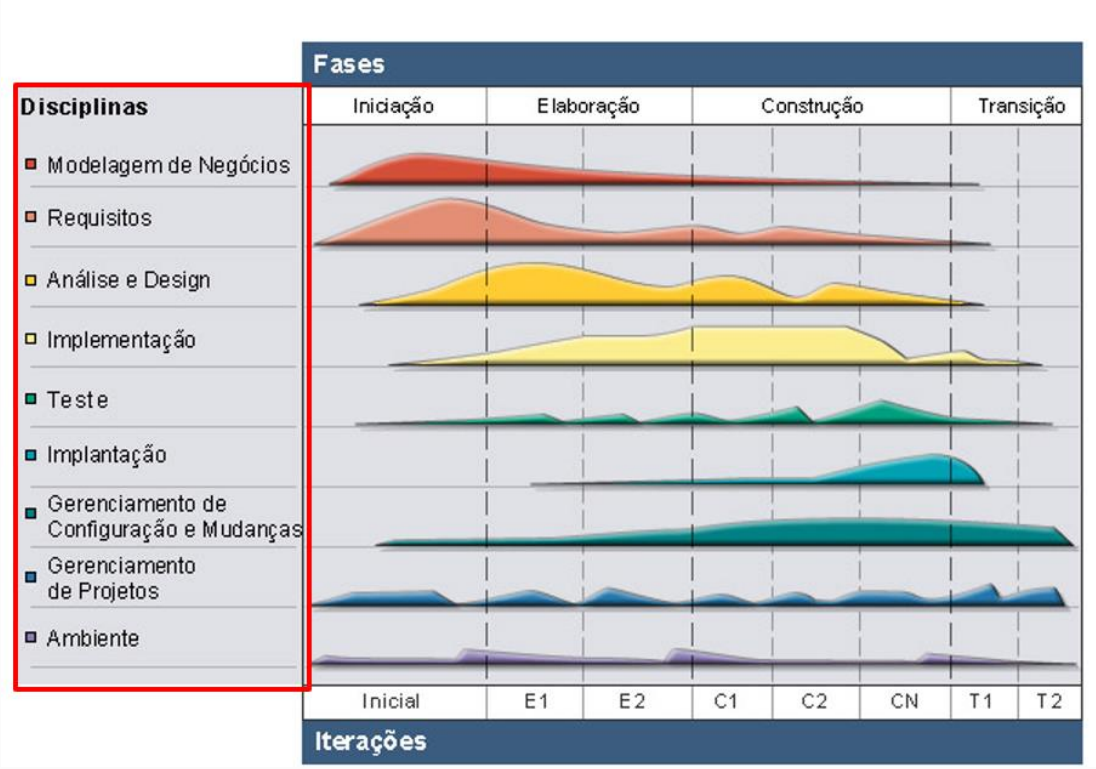
Na aula anterior vivenciamos uma base sobre o RUP e como ele organiza suas iterações, propostas de desenvolvimento e qual a sua arquitetura geral, além de demostrar como deve funcionar a ferramenta RUP. Nesta aula vamos entender como se estrutura a organização do RUP por conteúdo.

VIDEOAULA 01

Utilize o QRcode para assistir!

Assista o vídeo para entender melhor o assunto.

Figura 01 - Arquitetura Geral do RUP (disciplinas)



Fonte: RUP

Na Figura 01, podemos observar em destaque no canto esquerdo da figura, com bordas vermelhas, temos as disciplinas as quais são elas: modelagem de negócios, requisitos, análise e design, implementação, teste, implantação, gerenciamento de configuração e mudanças, gerenciamento de projetos e ambiente. Conforme descreve a imagem, a organização dentro da disciplina se baseia no conteúdo do processo, tarefas, produtos de trabalho e funções.

Com isso, vamos detalhar cada disciplina apresentada dentro da arquitetura geral do RUP para que seja de fato compreensível como deve ser desenvolvida cada uma delas:

→ **Modelagem de negócio**

A modelagem de negócio possui diversas diretrizes, porém um modelo de análise para negócios define os casos de uso a partir de um ponto de vista interno e externo do sistema e como seus usuários interagem com o *software* e suas entidades de negócio.

Para descrever esses relacionamentos e entidades de negócio é possível utilizar de notações da UML que foram vistas na unidade 01. Dentro da modelagem de negócios são desenvolvidas notações de caso de uso, classes, atividades e de sequência.

INDICAÇÃO DE VÍDEO

Agora, assista ao vídeo no qual o conteúdo abordado será sobre as notações UML. Vale a pena conferir. Disponível em: <https://www.youtube.com/watch?v=aRVYgbiULgE>. Acesso em: 8 maio 2021.

→ **Requisitos**

Para conseguir descrever os requisitos é necessário entender os problemas que o *software* deve solucionar, com isso se torna possível limitar o desenvolvimento e compreender o escopo geral do projeto.

Dessa forma, fornece uma base para estimar o tempo de desenvolvimento do projeto e assim desenhar os prazos para as entregas das iterações, estimando custos e quais recursos são necessários em cada etapa do projeto.

Com a definição dos requisitos conseguimos estabelecer uma concordância entre os *stakerholders* do projeto, além de prover para a equipe de desenvolvimento maior clareza sobre as necessidades do cliente e suas metas, o que deve ser alcançado para satisfazer as partes interessadas.

Os requisitos são definidos claramente para descrever o que o *software* fará, a disciplina de requisitos está relacionada com outras disciplinas como: Análise e Design, Teste, Configuração e Gerenciamento de Alteração, Gerenciamento de Projeto e Ambiente.

→ **Análise e Design**

Na disciplina de análise e design é onde transformamos os requisitos elaborados e coletados na disciplina anterior, em um design para o sistema que será criado. Além disso, precisamos desenvolver uma arquitetura sofisticada para o *software*, pensando que ele deve possuir qualidade e ser de fácil manutenção. Um *software* que possui uma boa arquitetura facilita as

alterações futuras e favorece a equipe de desenvolvimento em desafios que podem surgir durante a produção.

Vale lembrar que o design do sistema e sua arquitetura deve corresponder ao ambiente de implementação, e assim projetando o *software* para obter seu melhor desempenho.

VIDEOAULA 02

Utilize o QRcode para assistir!

Assista à videoaula para compreender melhor o assunto.

→ **Implementação**

Dentro dessa disciplina é onde devemos desenvolver, organizar e testar as unidades projetadas com a finalidade de validar se as especificações desenvolvidas estão de acordo com as necessidades coletadas com o cliente, e com a arquitetura desenhada na disciplina anterior. Nesta disciplina, também definimos a organização do código em termos de subsistemas de implementação organizando em camadas.

Na disciplina de implementação testamos os componentes desenvolvidos como unidades, os testes de sistema e de integração são pontos trabalhados dentro da disciplina de teste. Dessa forma, limitando o seu escopo com classes individuais, e testando-as como forma de unidade.

→ **Teste**

Esta disciplina tem como foco aumentar a qualidade do *software* e age como fornecedor de serviços para outras disciplinas também dentro do RUP, é no teste que se documenta as falhas do *software*, e assim buscamos promover sugestões de melhoria, além de validar se o *software* realiza as operações propostas.

Com os testes de sistema e de integração podemos também validar se o *software* atende aos requisitos levantados, ao design proposto e se realmente resolve o problema do usuário.

A essência do teste é encontrar fraquezas dentro do *software*, assim desafiando-o a cenários previstos e imprevistos dentro das disciplinas anteriores, a fim de identificar e corrigir falhas.

→ **Implantação**

As iterações presentes dentro dessa disciplina têm o ponto focal de tornar o *software* disponível para seus usuários, dentro dela descrevemos três modelos de implantação do produto, são elas:

a) Instalação personalizada;

b) O produto facilmente oferecido;

c) Acesso do *software* pela internet.

Em cada um desses três modelos há a ênfase de testar o produto no local de desenvolvimento, seguindo de testes beta (testes com alguns usuários) antes de finalmente ser oferecido de maneira formal para o cliente.

VIDEOAULA 03

Utilize o QRcode para assistir!

Assista à videoaula para ampliar seus conhecimentos sobre o assunto abordado.

→ **Gerenciamento de Configuração e Mudança**

Quando se tem muitas pessoas envolvidas em um projeto é necessário gerenciar da maneira correta as suas configurações e como ocorrem as suas mudanças, para assim evitar confusões de valores (valor agregado ao produto) entre os integrantes, deste modo, assegurando que o produto entregue não resulte em conflitos entre as partes envolvidas, alguns problemas devem ser evitados, são eles:

a) Atualização simultânea: quando duas ou mais pessoas efetuam a mesma tarefa e acabam sobrescrevendo um a tarefa do outro, o que causa também retrabalho para aplicar correções;

b) Notificação limitada: quando uma alteração é realizada por um integrante e somente alguns integrantes são avisados e os demais não são notificados;

c) Múltiplas versões: quando se tem um *software* muito grande é comum possuir versões diferentes que estão em fases distintas, quando encontrada uma falha ela deve ser propagada entre as versões, evitando novamente retrabalho e conflitos entre os integrantes.

São para esses tipos de situações que se torna necessário o uso de *softwares* que organizam os repositórios de desenvolvimento com versões e mantém a integridade do sistema, fornecendo um ambiente de desenvolvimento estável restringindo as alterações de acordo com as políticas do projeto. Com isso fornecendo segurança aos integrantes.

→ **Gerenciamento de Projetos**

Esta disciplina é uma das mais continuadas durante as fases do RUP, pois é dentro dela que se apresenta abordagens para a entrega de um *software* bem-sucedido. Nesta disciplina, buscamos uma estrutura para gerenciar projetos de *software* de maneira intensiva, com orientações para planejar, formar equipe, executar e monitorar o projeto.

Com isso, o foco é gerenciar os riscos do projeto com o objetivo de mitigá-los, realizar um planejamento repetitivo, por intermédio do ciclo de vida e de uma iteração específica, e assim monitorar o projeto com métricas, tornando visível o avanço do projeto para as partes interessadas. Ela é considerada uma disciplina de suporte dentro do RUP, pois, é uma atividade presente em todas as fases do projeto.

→ **Ambiente**

A disciplina de ambiente, assim como a disciplina anterior, são um suporte para o desenvolvimento do projeto, tendo como finalidade fornecer para toda a equipe processos e ferramentas para que o *software* alcance seu objetivo com sucesso, é como um suporte contínuo presente em todas as etapas, sempre fornecendo insumos para a equipe de desenvolvimento.

Esta disciplina fornece as informações de necessidade da equipe, como: infraestrutura, casos de desenvolvimento, gabaritos, orientações e outras informações que possam ser necessárias dentro do processo de desenvolvimento de um *software*.

Encerramento

Chegamos ao fim de mais uma unidade, estimado aluno. Espero que tenha gostado do conteúdo que trouxemos para você. Nessa unidade, mostramos um pouco sobre o RUP e como essa ferramenta pode auxiliar no processo de desenvolvimento de *software*.

Na aula 01, você estudou como é formada a arquitetura geral do RUP e como isso fornece um norte para o desenvolvimento de um software, no pensamento de fases do desenvolvimento e como a entrega de iterações pode impactar no estado atual de um software. Além de visualizar os conceitos de iterações e fases.

Em nossa aula 02, aprofundamos os nossos conhecimentos dentro da organização por conteúdo no RUP, visualizando qual é o efetivo papel de cada disciplina dentro do processo de desenvolvimento e como elas estão presentes em diversas fases, tendo seus picos de acordo com as necessidades do projeto.

Com isso encerramos a unidade 02, com uma visão de que um software vai muito além do código e que possui diversos pontos que devem ser considerados dentro do processo de desenvolvimento de um software. Nos encontramos na unidade 03, até mais.

Bom trabalho!

Esperamos que este guia o tenha ajudado compreender a organização e o funcionamento de seu curso. Outras questões importantes relacionadas ao curso serão disponibilizadas pela coordenação.
Grande abraço e sucesso!