Conversa Inicial

Olá, caro aluno! Seja bem-vindo a mais um encontro da disciplina **Engenharia de Software**!

Em nossa quinta aula falaremos sobre **gerenciamento** de **projeto de *software***. Exploraremos, dentro do contexto **gerenciamento de projeto**, suas principais fases: planejamento, início, execução, controle e finalização. Como em qualquer tipo de projeto, o de *software* segue uma metodologia para garantir o sucesso do produto final e de seu processo.

Preparado para conhecer cada uma dessas etapas?

Então, acompanhe a introdução de nossa aula com a professora Maristela, que apresentará os conteúdos que serão vistos neste encontro.

Bons estudos!

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P01.mp4

Contextualizando

Gerenciar projetos de desenvolvimento de *software* é um ato desafiador. Um projeto de *software* tem por finalidade traduzir necessidades dos *stakeholders* em aplicações, sistemas para *web*, sistemas convencionais e outros tipos de *software*. Trata-se de um processo de tradução de necessidades humanas para um produto tecnológico que será utilizado, novamente, por pessoas.

Confira, a seguir, uma interessante animação que mostra as consequências do processo de desenvolvimento de um *software* sem o devido planejamento:

https://www.youtube.com/watch?v=QPiR8jTMLdI

Projeto de *Software* – Espectro de Gerenciamento

O projeto de *software* é um empreendimento com objetivo bem definido, que consome recursos e ocorre, geralmente, com prazos, custos e qualidade igualmente bem determinados. Projetos tornam-se cada vez maiores e mais complexos compostos por atividades multifuncionais. O perfil de um gerente de projetos deve ser mais integrador que especialista técnico.

Um projeto prevê etapas importantes: **planejamento**, **programação** e **controle de tarefas integradas**.

Para compreender melhor o que é o gerenciamento de projeto de *software*, é importante compreendermos como se dá o processo de gerenciamento de qualquer tipo de projeto, de modo geral. Na videoaula a seguir, a professora Maristela explica sobre o gerenciamento de projetos, o que é um projeto de *software* na prática e como funcionam suas etapas. Acompanhe:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P02.mp4

Estrutura, Metodologia e Requisitos

O gerenciamento de projetos possui tarefas com objetivos para benefício de todos os participantes. Uma boa gestão de projetos exige um bom planejamento e um excelente controle. A gestão de projetos é dita horizontal e traz consigo mais produtividade, eficiência e eficácia. Para que o gerenciamento seja eficiente e eficaz, ele deve abordar, de forma clara, três itens importantes: **pessoas**, **produtos** e **projeto**.

Vamos conhecer esses três itens?

PESSOAS: Segundo a People-CMM (People Capability and Maturity Model), toda organização necessita aprimorar continuamente suas habilidades para atrair, desenvolver, motivar, organizar e reter a força de trabalho necessária para se atingir objetivos estratégicos em um negócio.

PRODUTO: O desenvolvimento de qualquer produto necessita da sequência de algumas tarefas importantes: escopo, soluções, restrições técnicas e restrições de gerenciamento. Todo produto é desenvolvido para pessoas. No caso de gerência de projetos a engenharia de requisitos, pode ser um grande trunfo.

PROCESSO: Pode ser considerado uma metodologia que será utilizada para o desenvolvimento de um produto ou *software*. Processos envolvem atividades-tarefas, pontos de controle, artefatos de *software* e pontos de garantia de qualidade.

Por mais que tenhamos evoluídos nos últimos 30 anos, projetos de desenvolvimento de *software* continuam apresentando vários tipos de problemas. Entre 1998 e 2004 constatou-se, entre 250 grandes projetos, que apenas 25% deles são realizados com sucesso, enquanto 50% não cumpriram cronograma, custos e objetivos de qualidade. Enquanto isso, 35% não obtiveram problemas tão sérios, porém, houve muitas manutenções até seu pleno funcionamento.

Uma das grandes questões no gerenciamento de projetos de *software* encontra-se no fato de que produzimos capital intelectual, e para isso, precisamos de bons especialistas, uma seleção bem elaborada para constituição da equipe e um bom ambiente de trabalho. Outra forma de melhorar a garantia da qualidade de um projeto encontra-se no comprometimento de gerentes, líderes técnicos, programadores, clientes e usuários finais. Líderes devem trabalhar questões de **motivação**, **organização**, **ideias**, **inovação** e uma excelente **comunicação**.

Na videoaula deste tema, a professora Maristela faz uma análise da estrutura geral de um projeto de *software* e explica como devem ocorrer suas etapas. Acompanhe para compreender os requisitos a serem observados na realização do projeto:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P03.mp4

Projeto de Arquitetura

O projeto da arquitetura reflete a estrutura de dados e componentes do programa para construção de um sistema. A arquitetura não é o *software* operacional, mas a representação para análise, alternativas e redução de riscos para a construção do *software*. Ela facilita a comunicação entre as partes envolvidas e cria um modelo compreensível da estrutura do *software*.

Há vários estilos de arquitetura, dentre eles:

1- Centrada em dados.

2- Centrada em fluxo de dados.

3- Centrada em chamadas e retornos.

4- Orientadas a objetos.

5- Em camadas.

6- Em padrões.

Agora que pudemos nos aprofundar a respeito dos requisitos e do escopo para o projeto, é hora de compreendermos melhor como funciona a arquitetura de *software*. Acompanhe a videoaula a seguir, em que a professora Maristela explica esse processo e sua importância para o sucesso deste trabalho:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P04.mp4

Projeto de componentes

O projeto de componentes é um conjunto completo de códigos (de programas) dentro de uma visão orientada a objetos (serviços), visão tradicional (elementos funcionais e estruturas de dados) ou visão orientada a processos.

Uma vez definido o projeto na área arquitetural e sua estrutura, é hora de passarmos para a realização do projeto de componentes. Na videoaula a seguir, a professora Maristela explica como funciona esta etapa. Confira:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P05.mp4

Projeto de interfaces

O projeto de interfaces, tão importante quanto o código de programa bem implementado, deve seguir algumas premissas básicas:

1- Deixar o usuário no comando do *software*.

2- Reduzir a carga de memória do usuário.

3- Criar interfaces consistentes.

Outros dois aspectos importantes na criação das interfaces são **usabilidade** e **acessibilidade**. A usabilidade é uma medida do quanto um sistema facilita o aprendizado, ajuda os aprendizes a se lembrarem do que aprenderam, reduz a probabilidade de erros, permite que sistemas se tornem eficientes e criam satisfação de uso do sistema.

Assim como em qualquer fase do projeto, o projeto de interfaces congrega processos, técnicas e ferramentas como apoio.

Hoje, há uma variedade de interfaces: *mobile*, *WebApps*, entre tantos outros. Nesta etapa, é necessária a definição dos perfis de usuários, aplicação de casos de uso, elaboração de tarefas, objetos e análise de fluxos de trabalho para suporte a uma construção eficiente de interfaces.

Na videoaula a seguir, a professora Maristela explica a importância de uma interface adequada para o segmento do *software*, e de um projeto muito bem definido para essa etapa. Agora falamos da multidisciplinaridade de nosso projeto. Como escolher a interface ideal? Saiba mais a seguir:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P06.mp4

Padrões e *WebApps*

Padrão é uma regra de três partes que expressa a relação entre contexto, problema e solução. São três tipos principais:

CRIACIONAIS:

a. Fábrica abstrata.

b. Métodos de fábrica.

c. Construtor.

d. Protótipo.

e. Único.

ESTRUTURAIS:

f. Adaptador.

g. Agregação.

h. Ponte.

i. Composição.

j. Container.

k. Proxy.

l. Tubos e filtros

COMPORTAMENTAIS:

m. Cadeia de responsabilidades.

n. Comandos.

o. Escutador de eventos.

p. Interpretador.

q. Iterador.

r. Mediador.

s. Visitante.

t. Visitante atendimento único.

u. Visitante hierárquico.

As tarefas no projeto de padrões são:

1- Exame do modelo de requisitos.

2- Desenvolvimento da hierarquia de problemas.

3- Determinação de padrões conforme linguagem e domínio do problema.

4- Adoção de critérios de qualidade.

Dentre as tarefas desta etapa, é importante salientar a necessidade da documentação dos padrões, que pode ser de forma descritiva.

Em padrões para *WebApps*, é importante desenvolver uma arquitetura de informações, de navegação, interação, de apresentação e de funcionalidades.

Tal tipo de aplicação requer, ainda, a preocupação com segurança, disponibilidade das interfaces e tempo, escalabilidade e tempo para colocação no mercado. Projetos para Web devem ser também simples, consistentes, possuir identidade, robustez e excelente navegabilidade.

Na videoaula a seguir, a professora Maristela explica o conceito de padrão e seu uso aplicado a *softwares* desenvolvidos para a *web*. Confira e entenda os tipos de padrão que estamos estudando neste tema:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P07.mp4

Estimativas e Riscos

Estimativas de custo e esforço de *software* não fazem parte de uma ciência exata. Envolvem fatores humanos, técnicas, questões ambientais e políticas das organizações. São baseadas em:

1- Projetos similares completos

2- Técnicas de decomposição para geração de estimativas de custos e esforço, que envolvem:

a. Dimensionamento lógica fuzzy.

b. Dimensionamento de pontos por função.

c. Dimensionamento de componentes-padrão.

d. Dimensionamento de alteração.

3- Modelos empíricos para estimativa de custo e esforço.

O importante a ser observado não é a sofisticação da técnica, mas a verificação e cruzamento com outras abordagens. Bom senso e experiência sempre prevalecem.

Trabalharemos, agora, alguns conceitos das estimativas LOC (linhas de código com medida-chave) e FP (Orientação à função ou *Functction Point*).

LOC – linhas de código com medida-chave:

- Não acomoda linguagens não procedurais.
- Nível de detalhe difícil.
- Penaliza programas bem projetados.

A figura 1 explicita um exemplo de uso de LOC:

Função	LOC estimado
Interface de usuário e recurso de controle	2.300
Análise geométrica bidimensional	5.300
Análise geométrica tridimensional	6.800
Gerenciamento de base de dados	3.350
Recursos de visualização da computação gráfica	4.950
Função de controle de periféricos	2.100
Módulos de análise do projeto	8.400
Linhas de código estimadas	*33.200*

Figura 1: LOC – Linhas de Código com Medida-Chave.

FP – Function Point:

- Orientada à função.
- Determina tamanho e complexidade do *software* sob perspectiva do usuário.
- Quantifica a funcionalidade proporcionada ao usuário a partir do desenho lógico.
- Oferece ferramenta para dimensionar aplicações.
- Quantifica custo, esforço e tempo.
- Calcula índices de produtividade e qualidade.
- Normalização para comparar *software*.

A figura 2 demonstra o uso da FP:

Fator	Valor
Backup e recuperação	4
Comunicações de dados	2
Processamento distribuído	0
Desempenho crítico	4
Ambiente operacional existente	3
Entrada de dados on-line	4
Transações de entrada em múltiplas telas	5
Arquivos mestres atualizados on-line	3
Complexidade dos valores dos domínios de informação	5
Complexidade do processamento interno	5
Código projetado para reutilização	4
Conversão/instalação no projeto	3
Instalações múltiplas	5
Aplicação projetada para alteração	5
Fator de ajuste de valor	**1,17**

Por fim, é obtido o número estimado de FP:

$$FP_{estimado} = \text{contagem total} \times [0{,}65 + 0{,}01 \times \Sigma(F_i)] = 375$$

Figura 2 – Function Point.

Além das técnicas LOC e FP citadas anteriormente, temos também:

1- Baseada em processos.

2- Casos de uso.

3- Modelos empíricos.

4- COCOMO II.

5- Orientada a Objetos.

6- Métodos Ágeis.

Até o momento, falamos muito sobre estimativas, técnicas, ferramentas e formas de melhorarmos nossas previsões e planejamento. Agora, trabalharemos com o tema **Risco**.

Riscos existem para quaisquer tipos de projetos. Sendo assim, com a engenharia de *software* não seria diferente. A gestão de riscos auxilia nas ações que suportam as equipes de *software* no entendimento de incertezas. O risco, independentemente de ocorrer ou não, deve ser previsto em qualquer tipo de projeto.

Riscos ameaçam o planejamento do projeto, tanto a nível de orçamento, quanto de cronograma, recursos humanos e materiais, clientes e requisitos. Riscos técnicos ameaçam a qualidade e a data de entrega do *software* a ser produzido. É muito importante que sejam observados problemas em potencial de projeto, implementação, interface, verificação e manutenção. Já os riscos de negócio ameaçam a viabilidade do *software* e a ser criado, bem como o projeto ou o produto.

Acompanhe a videoaula deste tema a seguir, em que a professora Maristela fala sobre a importância de estimarmos o custo no projeto de *software*, e de verificarmos os possíveis riscos em todos os âmbitos do projeto:

http://ava.grupouninter.com.br/videos/video2.php?video=http://vod.grupouninter.com.br/2015/OUT/MT180011-A05-P08.mp4

TROCANDO IDEIAS

O projeto de *software* é uma fase que ocorre imediatamente antes da implementação dos códigos (programação). Compartilhe com seus colegas o que cada um compreendeu e se alguém possui experiência em algum tópico que trabalhamos neste encontro. Senão, busque mais informações sobre padrões, projetos de interface, estimativa e riscos no projeto de *software*.

Participe, através do fórum desta disciplina, acessando o Ambiente Virtual de Aprendizagem (AVA)!

NA PRÁTICA

Agora que vimos as fases gerais do projeto de *software* e as características do gerenciamento deste projeto, veja, a seguir, o artigo *Uma abordagem para a gerência de projetos de software,* que aborda, entre outros fatores, a importância de se gerenciar um projeto:

http://www.uvv.br/edital_doc/GETPROJECT%20GERENCIADOR%20ONLINE%20DE%20PROJETOS%20DE%20SOFTWARE_516be976-99c4-4a38-a84c-84771096b300.pdf

SÍNTESE

Nessa aula tivemos uma visão geral sobre algumas fases do projeto de *software*, bem como suas características. Observamos que, após a fase de elaboração e identificação do domínio e problema em questão, podemos tomar diversos caminhos até chegarmos ao código (programas), de fato. O importante é adotarmos metodologias, técnicas e utilizarmos ferramentas que nos apoiem na construção de um projeto de *software* mais maduro, documentado e planejado. Isto evitará, no futuro, que construamos um *software* que não será utilizado por estar fora do escopo, das necessidades reais de nossos usuários ou, então, por conter falhas funcionais e de interface.

Até a próxima aula!