AULA 5:

MITOS E VERDADES NO DESENVOLVIMENTO DE SOFTWARE

Padronizações

ISO – International Organization for Standardization.

- Além de ser um acrônimo, é derivado da palavra grega que significa “igual”.
- Rede de institutos nacionais de padronização de 147, com sede em Genebra, Suíça.

Ética na Engenharia de Software

IEEE-CS ACM Software Engineering Code of Ethics and Professional Practice www.acm.org/serving/se/code.htm

- Atuar de acordo com o interesse do **público**
- Benéfica para o **cliente** e **empregador**
- Os **produtos** atendam aos padrões profissionais mais elevados
- Manter integridade e independência nas **avaliações** profissionais
- Enfoque ético no **gerenciamento** do desenvolvimento
- Integridade e reputação da **profissão**
- Ser justo com seus **colegas** de trabalho
- Postura de **aprendizagem** por toda vida

Diferentes Tipos de Desenvolvimento

- Desenvolvimento interno por demanda
- Desenvolvimento por contrato
- Desenvolvimento de linha de produto
- Desenvolvimento de F/OSS (Free and Open Source Software)
  - "Com um bom número de olhos, todos os bugs são superficiais" (Raymond, 2000) => Lance o produto logo e frequentemente.
- Desenvolvimento Web
- Desenvolvimento sistemas críticos
- Desenvolvimento de sistemas de tempo real
- Desenvolvimento de sistema embarcado
- Desenvolvimento de sistemas científicos
- Etc.

Treinamento do Desenvolvedor

- Lidar com incerteza e indefinição.
- Criatividade e inovação.
- Capacidade de avaliar processos, métodos e ferramentas.
- Capacidade de adaptar processos, métodos e ferramentas para cenários específicos.
- Trabalho em grupo.
- Comunicação.

Componentes do Software

- Um "sistema" de computador é formado por dois tipos de componentes:
  - Executáveis em máquina e.
  - Não executáveis em máquina.

- Componentes do software: mapear exigências do cliente em código executável.

Características

- **Reusabilidade** de componentes (como bibliotecas).
- Bom projeto de interface.
- Escolha da Linguagem.

Tipos Comuns de Software

- Básico: compiladores, editores simples, drivers, componentes do SO.
- Tempo Real: monitora/analisa/controla eventos do mundo real. Diferente de interativo.
- Comercial: controle de estoque, vendas etc. Acessam bancos de dados.

Tipos Comuns de Software

- Científico e de Engenharia: intenso processamento de números.
- Embutido ou embarcado: celulares, micro-ondas, injeção eletrônica.
- Pessoal: processador de texto, planilha, jogos, apresentações etc.
- Inteligência Artificial: sistemas especialistas, redes neurais e aprendizado.

Principais Problemas

- Estimativas de prazo (meses, anos) e custo imprecisas (uma ordem de magnitude a mais).
- Produtividade abaixo da demanda.
- Software de baixa qualidade (erros que tiram a confiança do cliente sobre o produto).

Mais Problemas

- Não dedicamos tempo para coletar dados sobre o processo de desenvolvimento de software. Com poucos dados históricos como guia, as estimativas têm sido “a olho”, com resultados previsivelmente ruins.
- Sem nenhuma indicação sólida de produtividade, não poderemos avaliar com precisão a eficácia de novas ferramentas, métodos ou padrões.

Mais Ainda

- A insatisfação do cliente com o sistema “concluído” ocorre muito frequentemente. Os projetos de desenvolvimento de software normalmente são levados a efeito apenas com um vago indício das exigências do cliente.
- A comunicação entre o cliente e o desenvolvedor de software frequentemente é muito fraca.

Ainda Não Acabou

- A qualidade do software frequentemente é suspeita.
- Só recentemente começamos a entender a importância dos testes de software sistemáticos e tecnicamente completos.
- Somente agora estão começando a surgir conceitos quantitativos sólidos de confiabilidade e garantia de qualidade de software.

Mais Um

- O software existente pode ser muito difícil de manter. A tarefa de manutenção de software devora a maioria de todos os dólares destinados a software.
- A capacidade de manutenção de software não foi enfatizada como um critério importante para a aceitação do software.

Causas

- Gerentes sem conhecimento (background) em software.
  - Mas um bom gerente não pode gerir qualquer processo?
  - Sim, se ele estiver disposto a aprender quais são os marcos (*milestones*) que podem ser usados para medir o processo, aplicar métodos efetivos de controle, não levar em conta a mitologia e tornar-se fluente numa tecnologia que se modifica rapidamente.

Mais causas

- Os programadores ou engenheiros de software têm pouca instrução formal em técnicas para desenvolvimento. **Natureza do Software**.
  - Cada pessoa aborda a tarefa de "escrever programas" com a experiência advinda de esforços passados. Algumas pessoas desenvolvem uma abordagem ordeira e eficiente, mesmo por tentativa e erro, mas muitas criam maus hábitos, que resultam em qualidade e manutenibilidade deficientes.

Outras Causas

- Histerese (resistência à mudança)
  - É irônico que enquanto o hardware experimenta enormes mudanças, as pessoas da área de software responsáveis pelo aproveitamento desse potencial, muitas vezes se oponham à mudança quando ela é discutida e resistam a ela quando ela é introduzida.

Mitologia do Software

- **Mitos administrativos:** Advém de gerentes sobre pressão de orçamento e tempo.
- **Mitos do cliente:** Advém de falsas expectativas e insatisfação com o desenvolvedor
- **Mitos do Profissional de desenvolvimento:** Advém de se considerar o software como uma forma de arte. Será que o software é uma arte ou uma engenharia?

Mito do Manual de Práticas

- Mito: Já temos um manual repleto de padrões e procedimentos para a construção de software. Isso não oferecerá ao meu pessoal tudo o que eles precisam saber?
- Realidade: O manual de padrões pode muito bem existir, mas será que ele é usado? Os profissionais de software têm conhecimento de sua existência? Ele reflete a moderna prática de desenvolvimento de software? É completo? Em muitos casos, a resposta a todas estas perguntas é “não”.

Mito do Computador Moderno

- Mito: Meu pessoal tem ferramentas de desenvolvimento de software de última geração; afinal de contas lhes compramos os mais novos computadores.
- Realidade: É preciso muito mais do que o último modelo de computador para se fazer um desenvolvimento de software de alta qualidade. As ferramentas de engenharia de software auxiliadas por computador (CASE) são mais importantes do que o hw para se conseguir boa qualidade e produtividade; contudo, a maioria dos desenvolvedores de software não as usa ainda

O Mito das Hordas de Mongóis

- Mito: Se nós estamos atrasados nos prazos, podemos adicionar mais programadores e tirar o atraso.
- Realidade: O desenvolvimento de software não é um processo mecânico igual à manufatura. Acrescentar pessoas em um projeto de software atrasado torna-o ainda mais atrasado. Gasta-se tempo educando os recém-chegados, o que reduz o tempo de desenvolvimento produtivo. Pessoas podem ser acrescentadas, mas somente de uma forma planejada e bem coordenada.

Mitos do Cliente: da Especificação

- Mito: Uma declaração geral dos objetivos é suficiente para se começar a escrever programas – podemos preencher os detalhes mais tarde.
- Realidade: Uma definição inicial ruim é a principal causa de fracasso dos esforços de desenvolvimento de software. Uma descrição formal e detalhada do domínio da informação, função, desempenho, interfaces, restrições de projeto e critérios de validação é fundamental. Essas características podem ser determinadas somente depois de cuidadosa comunicação entre o cliente e o desenvolvedor.

O Pior Mito do Cliente

- Mito: Os requisitos de projeto modificam-se continuamente, mas as mudanças podem ser facilmente acomodadas, porque o software é flexível.
- Realidade: É verdade que os requisitos de software se modificam, mas o impacto da mudança varia de acordo com o tempo em que ela é introduzida.

Mitos do Profissional: Terminar Mais Cedo

- Mito: Assim que escrevemos o programa e o colocarmos em funcionamento, nosso trabalho estará completo.
- Realidade: Alguém disse certa vez que “quanto mais cedo se começa a ‘escrever o código’, mais tempo demora para que se consiga terminá-lo”. Os dados da indústria indicam que entre 50 e 70% de todo o esforço gasto num programa serão despendidos depois que ele for entregue pela primeira vez ao cliente.

Mito da Qualidade

- Mito: Enquanto não tiver o programa "funcionando", eu não terei realmente nenhuma maneira de avaliar sua qualidade.
- Realidade: Um dos mecanismos mais efetivos de garantia de qualidade de software pode ser aplicado desde o começo de um projeto – a revisão técnica formal. As revisões de software são um "filtro da qualidade" que têm sido consideradas mais eficientes do que a realização de testes para a descoberta de defeitos.

Mito do Executável

- Mito: A única coisa a ser entregue em um projeto bem-sucedido é o programa funcionando.
- Realidade: Um programa funcionando é somente uma parte de uma configuração de software que inclui vários outros elementos. A documentação forma os alicerces para um desenvolvimento bem-sucedido e fornece um guia para a tarefa de manutenção do software.

Mito ou verdade?

- Software bem feito não sofre manutenção.
  - Software ruim é descartado, para software bom há trabalho de manutenção por anos.
- Se nos atrasarmos no cronograma, podemos adicionar mais programadores.
  - Adicionar pessoas a um projeto de software atrasado, atrasa-o ainda mais. [Brooks, 1975].
- Se eu terceirizar um projeto de software não preciso mais me preocupar.
  - Quando escrevemos um programa e o fazemos funcionar, nosso trabalho está completo.