4

TEXTO BASE

ENGENHARIA DE SOFTWARE

Texto base

4

Crise de Software

Prof. João de Deus Freire Junior

Resumo

O processo de desenvolvimento de software é complexo e a demanda por software cresceu e ainda cresce exponencialmente. O software traz diferenciais competitivos para as organizações, serviços base e suportam as operações. Ele é essencial. Porém, há muitos problemas encontrados nos projetos de construção, melhorias e manutenção de software. De fato, vivemos uma crise do software.

1.1. Introdução

Por que tantos problemas são encontrados ao utilizar software? Porque os projetos de desenvolvimento de software falham tanto em estimativas de prazo e custo? Estamos em uma crise do software? Todas essas perguntas serão respondidas nesta aula. Será apresentada a crise de software, os principais problemas associados a ela e suas causas.

1.2. O que é a crise do software?

A Crise do Software foi um termo que surgiu nos anos 70 que expressava as dificuldades do desenvolvimento de software frente ao rápido crescimento da demanda por software, da complexidade dos problemas a serem resolvidos e da inexistência de técnicas de desenvolvimento de sistemas.

1.3. Problemas relacionados a crise de software

Há diversos problemas associados à Crise do Software que são percebidos pelos usuários, desenvolvedores e outros envolvidos com a atividade de criação, evolução e

manutenção de software. Esses problemas são as “dores” sentidas por todos envolvidos com software. Eles têm prejudicado muito as corporações, as empresas de tecnologia, entre outros. Segue abaixo quatro exemplos desses problemas:

1.3.1. As estimativas de prazo e de custo frequentemente são imprecisas

As estimativas de prazo e custo necessários para desenvolvimento de software geralmente falham e são muito maiores do que o previsto. Os prazos maiores do que os previstos afetam muito as corporações, pois, frequentemente o prazo pode estar associado ao início de uma nova operação de negócios, lançamento de novo produto e etc.; as corporações são afetadas também pelos custos maiores que os orçados causando cancelamento de projetos, prejuízos, perdas financeiras e etc. As falhas nas estimativas estão associadas às causas abaixo:

- As equipes de desenvolvimento de software não dedicam tempo para coletar dados sobre o processo de desenvolvimento de software.
- As equipes de desenvolvimento de software não tem nenhuma indicação sólida de produtividade, não podendo avaliar com precisão a eficácia de novas ferramentas, métodos ou padrões.

1.3.2. A produtividade das pessoas da área de software não tem acompanhado a demanda por seus serviços

O software é a tecnologia única mais importante no cenário mundial. O software se tornou uma tecnologia indispensável para negócios, ciência e engenharia; ele viabilizou a criação de novas tecnologias (por exemplo, engenharia genética e nanotecnologia), a extensão de tecnologias existentes (por exemplo, telecomunicações) e a mudança radical nas tecnologias mais antigas (por exemplo, indústria gráfica); se tornou a força motriz por trás da revolução do computador pessoal; já vemos pacotes de software sendo comprados pelos consumidores em lojas de bairro; o software evoluiu lentamente de produto para serviço, na medida que empresas de software ofereceram funcionalidade imediata (just-in-time), via um navegador Web ou aplicativos móveis; companhias de software se tornaram as maiores e mais influente companhias da era industrial criando a era digital; a Internet, iria evoluiu e modificou tudo: de pesquisa em bibliotecas a compras feitas pelos consumidores, incluindo discurso político, hábitos de namoros de jovens e de adultos não tão jovens. (PRESSMAN, 2011)

Os sistemas têm de ser construídos e entregues mais rapidamente devido a todo esse aumento exponencial da demanda; sistemas maiores e até mais complexos são requeridos; sistemas devem ter novas capacidades que antes eram consideradas impossíveis. Como os métodos de engenharia de software existentes não conseguem lidar com isso, novas técnicas de engenharia de software precisam ser desenvolvidas para atender a essas novas demandas. A produtividade das pessoas da área de software não tem acompanhado a demanda por seus serviços. (SOMMERVILLE, 2011)

1.3.3. A qualidade de software às vezes é menos que adequada

Com o aumento da demanda de desenvolvimento de software exposto no tópico anterior, as empresas do setor de Tecnologia da Informação têm sido pressionadas a oferecer soluções de maior qualidade em prazos cada vez menores.

Em contrapartida, uma pesquisa recente publicada em 2012 e feita com organizações nacionais revelou que 43% dos projetos de TI, em média, foram cancelados ou entregues com falhas comprometedoras no processo e/ou produto. (RIBEIRO et al., 2011).

A qualidade de software é comumente menor que adequada. Os problemas de qualidade entre outros fatores estão associados a falta de conceitos qualitativos sólidos de garantia de qualidade e utilização de práticas e ferramentas para assegurar a qualidade do software entregue.

1.3.4. O software existente é difícil de manter

De acordo com a IEEE a definição de engenharia de software é:

> "A aplicação de uma abordagem sistemática, disciplinada e quantificável no desenvolvimento, na operação e na manutenção de software; isto é, a aplicação de engenharia ao software." (IEEE, 1993)

Essa definição deixa claro a importância da manutenção de software no processo de engenharia de software. A manutenção começa logo no começo do uso do software. Logo que o software é liberado para os usuários finais, e em alguns dias, erros começam a ser relatados à equipe de desenvolvimento do software. Em adição, mudanças são solicitadas por grupos de usuários para adaptar o software às suas necessidades e para melhor atender suas operações de negócios e clientes. Os usuários sempre precisarão de algumas melhorias para fazer o software funcionar em seu mundo.

Com isso, o desafio da manutenção do software começou. As equipes de sustentação e desenvolvimento do software terão que trabalhar paralelamente na correção de bugs, solicitações de adaptação e melhorias que devem ser planejadas, programadas e, por fim, executadas. Logo, a fila já cresceu muito e o trabalho ameaça devorar os recursos disponíveis. Com o passar do tempo, sua organização descobre que está gastando mais tempo e dinheiro com a manutenção dos programas do que criando novas aplicações. De fato, não é raro uma organização de software despender de 60% a 70% de todos os recursos com manutenção de software. (PRESSMAN, 2011)

Os motivos de muito trabalho na manutenção de software são muitos. Osborne e Chikofsky fornecem uma resposta parcial:

> "Muitos softwares dos quais dependemos hoje têm em média de 10 a 15 anos. Mesmo quando esses programas foram criados, usando as melhores técnicas de projeto e codificação conhecidas na época [e muitos não foram], o tamanho do programa e o espaço de armazenamento eram as preocupações principais. Eles então migraram para novas plataformas, foram ajustados para mudanças nas máquinas e na tecnologia dos

> sistemas operacionais e aperfeiçoados para atender a novas necessidades dos usuários – tudo isso sem grande atenção na arquitetura geral. O resultado são estruturas mal projetadas, mal codificadas, de lógica pobre e mal documentadas em relação aos sistemas de software, para os quais somos chamados a fim de mantê-los rodando."(OSBORNE, 1990, p.10-11)

As más práticas de desenvolvimento e má elaboração ou inexistência de documentação de software levam a grande dificuldade na manutenção do software.

1.3.5. Resumo dos problemas associados à crise de software

A tabela abaixo apresenta um resumo dos problemas associados à crise de software.

Tabela 1.1. Variáveis a serem consideradas na avaliação de técnicas de interação.

Problema	Status na Crise
Prazos e Custos em relação às estimativas	Mais altos
Produtividade das Pessoas	Baixa
Qualidade de Software	Baixa
Dificuldade de manter software	Alta

1.4. Causas da Crise de Crise de Software

As causas associadas à crise de software são diversas. Vamos ressaltar neste tópico as três principais.

1.4.1. Natureza do software

A própria natureza do software é uma das causas da crise do software devido às suas próprias características. Segue essas características:

- O software é um elemento de sistema lógico e não físico (produto intangível). Consequentemente, o sucesso é medido pela qualidade de uma única entidade e não pela qualidade de muitas entidades manufaturadas.
- O software não se desgasta, mas se deteriora!

À medida que o tempo passa, os componentes de um hardware sofrem os efeitos cumulativos de poeira, vibração, impactos, temperaturas extremas e vários outros males ambientais ele começa a desgastar-se.

O software não é suscetível aos males ambientais que fazem com que o hardware se desgaste. Portanto, ele não se desgasta. Mas, o ambiente de negócios, a tecnologia, os processos, as organizações, sociedades, leis, regras mudam e todos esses aspectos podem fazer com que o software fique obsoleto e se deteriore.

1.4.2. Falhas das pessoas responsáveis pelo desenvolvimento de um software

As falhas das pessoas responsáveis pelo desenvolvimento de um software é uma das causas da crise do software. Isso acontece porque:

- Alguns gerentes de TI não tem nenhum background em software;
- Os profissionais da área de software têm recebido pouco treinamento formal em novas técnicas para o desenvolvimento de software;
- Alguns profissionais e gestores de TI são resistentes a mudanças.

1.4.3. Mitos de Software

Os mitos de software são inverdades sobre o processo de engenharia de software que propagam desinformação e confusão e eles são também uma das causas da crise do software. Os mitos podem ser:

- Administrativos;
- De clientes;
- Profissionais.

1.5. Você quer ler?

Segue uma indicação de leitura para estudo complementar. Trata-se de um artigo que apresenta e discute pontos da crise de software e de uma crise mais contemporânea a de significado: Da crise do software à crise de significado: a importância de aprender a compreender. Disponível em:

<https://medium.com/software-zen/da-crise-do-software-%C3%A0-crise-de-significado-a-import%C3%A2ncia-de-aprender-a-compreender-5ff8fb920e37>. Acesso em: 07 jan. 2021

1.6. Referências

PFLEEGER, Shari Lawrence. Engenharia de software - teoria e prática . 2 .ed. São

Paulo: Pearson (livros universitarios), 2003

RIBEIRO, A.; PRADO, D.; ARCHIBALD, R. Pesquisa sobre Maturidade e Sucesso em Gerenciamento de Projetos de Sistemas de Informação (software). Relatório TI 2010. [S.l]: Maturity by Project Category Model (MPCM), 2011. Disponível em: <http://www.maturityresearch.com/novosite/2010/downloads/PesquisaMaturidade2010_Relatorio-T I _Completo_V3.pdf>. Acesso em: 21 mar. 2012.

OSBORNE, W. M., and E. J. Chikofsky, “Fitting Pieces to the Maintenance Puzzle,” IEEE Software, January 1990, pp. 10–11.

PRESSMAN, R. S.(2011) Engenharia de Software: uma abordagem profissional. 7.ed. Porto Alegre: Bookman, 2016.