Aula 5

Gerência de Configuração e Evolução

Profª Adriana Bastos da Costa

1

Conversa Inicial

2

Ciclo de vida do software

- O ciclo de vida de um software envolve não só as etapas que ocorrem durante o desenvolvimento, mas também as etapas que ocorrem durante todo o período de manutenção e suporte, enquanto o software estiver ativo e em uso

3

Ciclo de vida do software

- A gerência de configuração se aplica ao ciclo de vida do software, e não apenas ao ciclo de vida de desenvolvimento do software
- Ciclo de vida do desenvolvimento do software X Ciclo de vida do produto de software

4

- Esta aula está dividida em 5 temas principais, sendo eles:
  - O que é a Lei de Lehman
  - Como a Lei de Lehman está estruturada
  - Ciclo de vida do software
  - Envelhecimento e rejuvenescimento do software
  - Evolução do hardware e sua contribuição para a evolução do software

5

O que é a Lei de Lehman

6

Evolução do software

- De acordo com Sommerville (2019), existe uma necessidade constante e inevitável de evolução por parte de todo software que espera manter-se ativo por um longo período de tempo

7

Evolução do software

- A evolução do software ocorre de maneira natural, pois as necessidades do mercado e dos clientes mudam, além da evolução da tecnologia, que acaba exigindo novas funcionalidades e melhorias no software

8

Leis de Lehman

- Meir Manny Lehman foi um cientista e pesquisador na área da computação. De 1972 até 2002, ele foi professor e Chefe do Departamento de Computação no Imperial College de Londres. Junto com outros colegas de profissão, eles organizaram e difundiram os conceitos relacionados com a evolução do software, que ficaram conhecidos como Leis de Lehman

9

Leis de Lehman

- As leis foram definidas inicialmente para softwares *mainframe* da IBM, que utilizavam o sistema operacional OS/360
- Porém, como a Lei de Lehman é uma fonte histórica de bom senso em software, pode ser aplicada a outros contextos

10

Leis de Lehman

- As chamadas Leis de Lehman são generalizáveis e aplicáveis a diferentes contextos na tomada de decisão, planejamento, desenvolvimento e manutenção de software

11

Como a Lei de Lehman está estruturada

12

Leis de Lehman

- O foco das leis está na evolução natural que todo software passa
- Os 8 pontos identificados e organizados por Lehman passaram a ser vistos como "lei" porque eram generalizáveis e aplicáveis a muitos contextos de software

13

Leis de Lehman

- 1. Mudança contínua - um software deve ser continuamente adaptado, senão torna-se, aos poucos, cada vez menos satisfatório. O que se busca com essa lei é manter o software útil ao propósito para o qual ele foi construído

14

Leis de Lehman

- 2. Complexidade crescente - devem ser tomadas medidas para reduzir ou manter a complexidade de um software. O conceito ágil de refatoração está relacionado com esta lei. Refatoração é a ação de melhorar o código buscando soluções mais simples

15

Leis de Lehman

- 3. Autorregulação - sugere que o software possui uma dinâmica própria, o que decide as tendências da manutenção e limita o número de possíveis mudanças. O caminho saudável é priorizar as melhorias a partir da análise dos riscos. A substituição de recursos ou pessoas tem efeitos imperceptíveis na evolução do software. Essa lei requer processo e organização

16

Leis de Lehman

- 4. Conservação da estabilidade organizacional - as informações que são levadas em consideração para as tomadas de decisão e que levam à evolução de um software tendem a ser constantes. A estabilidade organizacional depende da efetiva manutenção e estabilidade do software, que precisa continuar atendendo o objetivo de negócio para o qual ele foi criado

17

Leis de Lehman

- 5. Conservação de familiaridade - à medida que o software evolui, podem ocorrer melhorias, que acarretam em mudanças comportamentais ou de limites de escopo do software. É preciso decidir se a mudança deve ser inserida no software em questão ou se um novo software deve ser projetado para atender a mudança necessária ao negócio

18

Leis de Lehman

- **6. Crescimento contínuo - o projeto inicial não consegue incluir tudo o que é necessário para atender o negócio, até porque novas tendências surgem no mercado e os clientes passam a exigir mais funcionalidades. É preciso identificar onde o aumento das funcionalidades pode impactar. Um requisito novo pode gerar alterações ou ajustes em requisitos já existentes**

19

Leis de Lehman

- **7. Qualidade diminuindo - o software precisa se manter útil, para que continue agregando valor para o negócio. É preciso analisar o todo, pois a qualidade do software tende a diminuir com a sua evolução. Devem ser utilizadas técnicas para garantir a rastreabilidade dos requisitos e sua consistências, além de técnicas de testes que busquem avaliar e executar testes de regressão.**

20

Leis de Lehman

- **8. Sistema de *feedback* - é preciso fazer uma análise de custo X benefício, buscando a melhoria contínua do software com foco em agregar cada vez mais valor ao negócio. Nessa lei é possível aplicar o conceito de PDCA (plan, do, check e act) que é usado na gestão dos negócios e se aplica perfeitamente bem à evolução de um software.**

21

Ciclo de vida de software

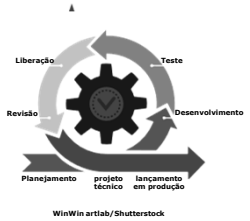
22

Ciclo de vida

- **Possui etapas abrangendo a vida do software, desde a definição de seus requisitos até o declínio de seu uso**
- **Possui etapas também durante seu suporte e manutenção, quando o software já está em produção**

23

Ciclo de vida de desenvolvimento de software



WinWin artlab/Shutterstock

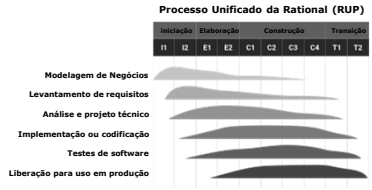
24

Ciclo de vida de desenvolvimento de software

KostiantynL/Shutterstock

25

Ciclo de vida de desenvolvimento de software



KostiantynL/Shutterstock

26

Ciclo de vida do produto de software

- O ciclo de vida do produto é um conjunto de etapas desde o seu projeto e concepção até o momento em que ele é descontinuado e retirado do mercado
- O conceito de ciclo de vida do produto foi desenvolvido pelo economista alemão Theodore Levitt. Segundo Levitt, nenhum produto "vive" para sempre

27

Ciclo de vida do produto de software

- Introdução - o software já está construído. Ele, então, é colocado em produção para ser utilizado pelos usuários e clientes. Essa é a fase na qual os usuários começam a conhecer o potencial do software, identificando pontos fortes e pontos fracos, que precisam ser melhorados. Aqui já podem ser identificadas melhorias evolutivas e corretivas

28

Ciclo de vida do produto de software

- Crescimento – é muito comum que o software continue passando por evoluções e melhorias, que precisam ser gerenciadas e priorizadas, para que sejam implementadas de maneira organizada. A gerência de configuração é fundamental para garantir a qualidade dos componentes do software, mantendo o software seguro e íntegro

29

Ciclo de vida do produto de software

- Maturidade – o software está mais estável. As melhorias evolutivas ainda podem surgir, mas cada vez em menor número, pois o software já está maduro o suficiente para atender ao negócio. É o momento no qual o cliente pensa se novas melhorias evolutivas deverão ser implementadas no software ou se vale mais a pena partir para o projeto de um novo software

30

Ciclo de vida do produto de software

- **Declínio - após atingir a maturidade e render bons resultados para a empresa, os softwares deixam de representar vantagens e precisam ser descontinuados. Pode ser que fique muito caro manter novas melhorias por conta do custo da manutenção. Nesse momento, é muito comum que a empresa comece a pensar em outras soluções para substituir o software**

31

Ciclo de vida

- **Vimos que existe um ciclo de vida para o desenvolvimento do software e outro para o produto de software**
- **Em ambos os casos, a gerência de configuração é fundamental para manter o controle e a segurança do que está sendo inserido no software**

32

Envelhecimento e rejuvenescimento do software

33

Envelhecimento do software

- **É um processo pelo qual a qualidade do código diminui ou se torna desatualizada, levando a vários problemas técnicos**
- **Qualquer software está sujeito a um ciclo de envelhecimento, que muda gradualmente suas características e desempenho para pior**

34

Envelhecimento do software - motivos

- **Atualizações de hardware e da tecnologia**
- **Acúmulo de erros ao longo do tempo**
- **Dados e falta de integridade de arquivo**
- **Memória "inchada e vazando"**

35

Rejuvenescimento do software

- **O rejuvenescimento do software está relacionado com as ações ou métodos usados para prevenir o envelhecimento do software, minimizando o impacto do tempo**
- **São ações que têm como objetivo remover os erros acumulados ao longo do tempo, liberar recursos do sistema e corrigir problemas de corrupção de dados**

36

Rejuvenescimento do software - ações

- **Reinicializar o sistema - primeira e mais simples ação para qualquer usuário**
- **Instalação limpa do sistema operacional - instalar uma cópia com configurações originais da versão mais recente do sistema operacional**

37

Evolução do hardware e sua contribuição para a evolução do software

38

Hardware e software

- **A história de TI mostra que a evolução do hardware e do software ocorrem de maneira bastante homogênea**
- **O hardware impulsiona a construção de softwares mais complexos e mais potentes, e a necessidade de softwares mais especializados pressiona a indústria do hardware a evoluir**

39

Tecnologias impulsionando o software

- **Acesso à tecnologia de sensores de baixo custo e baixa potência**
- **Conectividade**
- **Plataformas de computação na nuvem**
- ***Machine learning***
- **Inteligência artificial (IA)**

40

41