Resumo de Engenharia de Software

MATHEUS LOPES LOURENÇO – DSM2022

* O que é Engenharia de Software?

“É o curso que apresenta em **métodos**, as **técnicas para o desenvolvimento de software** que **abordam** **especificação**, **modelagem**, **arquiteturas**, **verificação** e **testes de software**, bem como para o **planejamento** e **gerenciamento do processo** de desenvolvimento. A disciplina de ES **se preocupa em sistematizar** o desenvolvimento através de **modelos, técnicas e ferramentas** para o produto e para o processo”.

* O que é Software e Qual a diferença de software genérico e software sob encomenda?

São **programas de computadores**, onde em suas diversas formas, temos a documentação associada. Sendo um **conjunto de soluções algorítmicas**, **codificadas** numa **linguagem de programação**, executado numa máquina real.

\*Software é um **produto conceitual e lógico**, podendo ser classificado em dois tipos de produtos os **genéricos (Produtos lançados para o mercado) e sob encomenda (Produtos previamente encomendados)**.

**Invisibilidade** – **Software é invisível** e invisualizável

**Complexidade** – Software é mais **complexo** do que qualquer outro produto construído por seres humanos

**Mutabilidade** – Existe sempre uma **pressão para se fazer mudanças em um software**

**Conformidade** – O software deve ser **desenvolvido conforme o ambiente**. **Não é o ambiente que deve se adaptar ao software.**

Se o software está conforme os requisitos (o ambiente) todo o suporte operacional deve se adaptar ao software.

* “**Bugs** e Glichts”

São mais conhecidos como: **Falhas residuais** (...) e não são erros, também são considerados **Falhas de Software (Falhas nos produtos)**. Eles são, **geralmente**, **causados por erros no próprio código-fonte**, mas também **podem ser causados** por algum **framework**, **interpretador**, **sistema operacional ou compilador**.”

* Defina os desafios para a engenharia de software no século 21

Os principais desafios que a ES terá de enfrentar nos próximos anos, será de entregar produto de qualidade em tempo relativamente curto, pois as empresas estão constantemente exigindo mais de seus programadores. Além disso, a exigência em relação ao currículo será mais exigidos aos desenvolvedores. Conhecimentos sobre língua estrangeira, linguagens de programação e estudo teórico são alguns tópicos em constante debate sobre a área. Contudo, o maior desafio atualmente é de atualizar e modernizar sistemas antigos **(Sistemas Legados)**.

* **Sistemas Legados**: (Exemplos: Sistemas Bancários)

Sistemas antigos, mas de extrema importância para uma organização e que funcionam bem.

Linguagens antigas – falta de pessoal

Específico de plataformas obsoletas

* Exemplo de Bugs ou Falhas de Sistemas em Empresas:

**Caso TSB Bank**

Milhões de clientes do Banco TSB, do Reino Unido, tiveram suas **contas bloqueadas após uma atualização de TI** que levou a uma **interrupção** **do banco on-line**. A **expectativa era de que uma atualização encerrasse os serviços bancários pela Internet e dispositivos móveis em um único fim de semana** de abril último, **mas acabou causando meses de interrupção**.

Os **problemas surgiram** com a **mudança do TSB** para uma **nova plataforma bancária** após sua divisão do Lloyds Banking Group. Imediatamente depois que o **novo sistema foi ligado**, muitos **clientes tiveram problemas ao fazer o login**, enquanto outros **visualizavam detalhes das contas de outras pessoas**. Os **clientes** permaneceram **bloqueados** mesmo após duas semanas da interrupção inicial.

Em julho, o TSB ainda estava trabalhando no **acúmulo de reclamações**, quando **outra interrupção ocorreu**, **bloqueando os clientes de suas contas online mais uma vez**.

O TSB alegou que o **problema foi resolvido** mais tarde naquele dia, **mas o problema poderá estremecer ainda mais a relação do banco com a empresa controladora Sabadell.** De acordo com o Financial Times, o grupo bancário espanhol está **considerando a venda do TSB**.

Outros casos em: <https://itforum.com.br/noticias/10-grandes-falhas-da-tecnologia-nos-ultimos-anos/>.

* Defina as Fases do Ciclo de Vida (Ciclo de Lançamento) de sistemas:

1 – Planejamento

2 – Desenvolvimento

3 – Aprimoramento

4 – Entrega

\*5 – Updates (O único dentre essas fases que não é um item obrigatório, porém na maior parte da entrega de softwares e de ciclo de lançamento – duração – o update se faz necessário para a entrega do Software).

* O que são Ferramentas Case?

São as **“ferramentas auxiliares”** para o desenvolvimento de softwares, onde abrange uma classificação de todas as ferramentas baseadas em computadores que **ajudam** em atividades de **engenharia de software**, desde **análise de requisitos, modelagem até programação e testes**.

* O que é **Modelo de Processo de Software**?

Existem vários modelos de processo de software (ou **Paradigmas de Engenharia de Software**). Onde cada um representa uma tentativa de colocar ordem em uma atividade inerentemente caótica. Ou seja, modelo de processo significa a representação de processos de produtos.

* O que é **Iteração de Processo de Software**?

Também chamado de **Processo Iterativo (... é um tipo de Modelo de Processo que cerca os Paradigmas)** é aquele que **progride através de refinamentos sucessivos**. A equipe de desenvolvimento apresenta uma parte de um sistema, ciente de que está incompleto ou deficitário em algumas (talvez muitas) partes. A equipe então refina estas partes até o produto apresentar um nível satisfatório.

* O que é **Metodologia de Desenvolvimento** de Software?

É a **maneira ou modo** de **desenvolvimento** e **andamento** de **projetos** que consistem, basicamente, na **junção de abordagens** que podem ser **utilizadas para a criação de sistemas de processamento de dados**. O sucesso de qualquer projeto voltado à elaboração de software **depende diretamente da escolha da metodologia mais correto**.

* Faça um comparativo entre métodos (**metodologias**) Scrum e XP

**Scrum**

É reconhecida por sua **qualidade ao proporcionar uma gestão mais dinâmica** dos projetos, fazendo com que os **objetivos e prazos** sejam **cumpridos dentro do prazo pré-estabelecido**. Onde reuni todas as questões anteriores ao ***Backlog*** e **dá uma tarefa ordenada à sequência de sua criação**. Além disso, a metodologia pode ser usada em processos de ***Product Owner***, SCRUM Team e ***SCRUM Master***. O SCRUM também trabalha com o que chamamos de **monitoramento de resultados**, parte importante para qualquer projeto ou agência. Para isso, ele usa pontos que **medem o tempo de cada tarefa**, levando você a saber se o seu projeto vai ser entregue no prazo ou se você terá que atrasá-lo um pouco. Daí vem a grande popularidade dessa ferramenta.

**Scrum e a agilidade não são a mesma coisa**, **mesmo que o Scrum seja centrado na melhoria contínua**, **por mais que seja o princípio fundamental da agilidade**. No entanto, o **Scrum é uma estrutura para concluir tarefas**, enquanto a **agilidade é uma forma de pensar, de mudar e alterar seu modo de pensar**.

**XP**

Metodologia usada para **agilizar o desenvolvimento** de software. Sendo reconhecida pela **economia de tempo que proporciona**. Assim, é **possível criar sistemas de melhor qualidade e de modo bem mais rápido que os desenvolvedores estão acostumados**. Com o **Extreme Programming ou XP** as **interações entre indivíduos** **são mais que um processo** e conseguem ser **potencializadas**. Além disso, você conta com um software com **documentação ampla** e consegue responder mais rapidamente a mudanças.

**OBS: Termos e seus respectivos significados ...**

“***Backlog*** refere-se a um **log de acumulação de trabalho num determinado intervalo de tempo**. Backlog é uma espécie de estoque de folhas de requisições/encomendas relativas a produtos ainda não produzidos. Grosso modo, backlog é uma **"pilha de pedidos" em espera”**.

***“Product Owner é*** o dono do produto, pessoa ou empresa que requisita o desenvolvimento de um Software”

“***SCRUM Master*** é quem faz a facilitação do Scrum para equipes maiores garantindo que a estrutura seja seguida. Sendo a pessoa que possui o compromisso com os valores e práticas do Scrum, mas **também deve manter a flexibilidade e a abertura para oportunidades** de melhorar o fluxo de trabalho da equipe”.

<https://www.estudiosite.com.br/site/desenvolvimento-web/3-vantagens-das-ferramentas-scrum-kanban-e-xp>.

<https://www.estudiosite.com.br/site/sem-categoria/7-desvantagens-das-ferramentas-scrum-kanban-ou-xp>.

* Descreva dois critérios para utilizarmos RUP no desenvolvimento de software

**RUP**

**O Processo Unificado da Rational** conhecido como RUP (***Rational Unified Process***), é um **processo de engenharia de software criado para apoiar o desenvolvimento orientado a objetos**, fornecendo uma **forma sistemática** para se **obter vantagens no uso da UML**. Foi criado pela Rational Software Corporation e adquirido em fevereiro de 2003 pela IBM.

**O principal objetivo do RUP é atender as necessidades dos usuários garantindo uma produção de software de alta qualidade que cumpra um cronograma e um orçamento previsíveis**. Assim, o RUP **mostra** como **o sistema será** **construído na fase de implementação**, gerando o modelo do projeto e, opcionalmente, o modelo de análise que é utilizado para garantir a robustez. O RUP **define** **perfeitamente quem é responsável** pelo que, como as coisas deverão ser feitas e quando devem ser realizadas, **descrevendo** todas as metas de desenvolvimento especificamente para que sejam alcançadas.

O RUP organiza o desenvolvimento de software em **quatro fases**, onde são tratadas **questões sobre planejamento, levantamento de requisitos, análise, implementação, teste e implantação do software.**  Cada fase tem um papel fundamental para que o objetivo seja cumprido, distribuídos entre vários profissionais como o Analista de sistema, Projetista, Projetista de testes, entre outros.

Diagrama

Descrição gerada automaticamenteGráfico, Tabela

Descrição gerada automaticamente

**Fase de Concepção / Iniciação**: Esta fase do RUP **abrange as tarefas de comunicação** com o cliente e planejamento. É feito um **plano de projeto avaliando os possíveis riscos**, as **estimativas de custo e prazos**, estabelecendo as **prioridades**, **levantamento dos requisitos do sistema** e **preliminarmente analisá-lo**. Assim, haverá uma anuência das partes interessadas na definição do escopo do projeto, onde são examinados os objetivos para se decidir sobre a continuidade do desenvolvimento.

**Fase de Elaboração**: Abrange a **Modelagem do modelo genérico do processo**. O objetivo desta fase é **analisar de forma mais detalhada a análise do domínio do problema**, **revisando os riscos** que o projeto pode sofrer e a arquitetura do projeto começa a ter sua forma básica. Indagações como "O plano do projeto é confiável?", "Os custos são admissíveis?" são esclarecidas nesta etapa.

**Fase de Construção**: Desenvolve ou Adquire os componentes de Software. O principal objetivo desta fase é a **construção do sistema de software**, com foco no **desenvolvimento de componentes** e outros recursos do sistema. É na fase de Construção que a maior parte de **codificação** ocorre.

**Fase de Transição**: Abrange a **entrega do software** ao usuário e a **fase de testes**. O objetivo desta fase é **disponibilizar o sistema**, tornando-o disponível e compreendido pelo usuário final. As atividades desta fase incluem o **treinamento dos usuários finais e a realização de testes da versão beta** do sistema visando garantir que ele possua o nível adequado de qualidade.

<https://www.infoescola.com/engenharia-de-software/rup/>.

* Quando são utilizados métodos formais para especificação dos requisitos de software? O que são especificações formais? Quando as técnicas de especificação formal são mais adequadas?

**Termo**

* “**Especificação de requisitos de software** é a **etapa do desenvolvimento** que **determina o sucesso** de um software. É o que define os **objetivos e funções que um software precisa executar**, bem como as que ele não pode ter (**restrições**)”.
* **“**Ao **satisfazer tais expectativas**, você garante que o **sistema** seja **funcional e confiável**, bem como apresente um bom desempenho.”
* “**Métodos Formais é parte de uma coleção de técnicas de requerimentos**, onde são baseadas em **representações matemáticas** e na **análise de software**. Incluem **métodos** de **Especificação Formal**, **Análise e provas de especificação**, **Desenvolvimento Transformacional**, **Verificação de programas.** Apenas utilizamos os métodos quando queremos a **redução do número de defeitos no sistema**, e conseguintemente sua aplicação é na **engenharia de sistemas críticos** e sua utilidade neste setor é bastante aplicada pois **altos custos de manutenção** **devem ser evitados”**.

**Resposta**

* Especificação Formal é expressa em uma determinada linguagem cujo vocabulário, sintaxe e semântica são formalmente definidos e rígidos onde temos dois tipos específicos:

–Formais: A notação utilizada possui uma sintaxe e uma semântica totalmente precisa.

–Informal: Escritas em linguagem natural.

A mais utilizada é a Formal, porém temos outros tipos de Especificação.

* <https://www.monitoratec.com.br/blog/especificacao-de-requisitos-de-software/#:~:text=A%20especifica%C3%A7%C3%A3o%20de%20requisitos%20de%20software%20%C3%A9%20a%20etapa%20do,n%C3%A3o%20pode%20ter%20(restri%C3%A7%C3%B5es)>.
* <http://disciplinas.lia.ufc.br/es10.1/arquivos/CAP10.pdf>.
* <https://silo.tips/download/metodos-formais-ian-sommerville-software-engineering-traduaoprof-pedro-fernandes#:~:text=Slide%206-,Especifica%C3%A7%C3%A3o%20Formal,s%C3%A3o%20formalmente%20definidos%20e%20r%C3%ADgidos>.
* Descreva Computação Móvel, Pervasiva e Ubíqua. Dê um exemplo de cada.

“A ideia principal da **Computação Móvel** é a de que o seu **dispositivo está sempre em movimento**, você pode usá-lo em **praticamente qualquer lugar que você esteja**. Este conceito se difundiu muito devido a **aparição dos Smartphones e dos avanços computacionais** que permitiram que os dispositivos ficassem cada vez menores.

O conceito de **Computação Pervasiva** implica que os meios de computação estão distribuídos no ambiente de forma perceptível e imperceptível ao usuário. Basicamente a Computação Pervasiva é o que a **define como invisível ao olho nu, mas sabe-se que ela está presente no espaço (Wi-Fi)**.

Como se pode observar na figura abaixo, a **fusão dessas tecnologias resulta na Computação Ubíqua** que possui como suas **principais características a descentralização**, **diversidade** e a **conectividade**. Basicamente surge da **integração da mobilidade com a presença distribuída, imperceptível, inteligente e altamente integrada de computadores e suas aplicações (Alexa e Google Assistente)**”.

Diagrama

Descrição gerada automaticamente<https://pt.stackoverflow.com/questions/193102/o-que-%C3%A9-computa%C3%A7%C3%A3o-m%C3%B3vel-e-ub%C3%ADqua>.

* Exemplos de éticapara os desenvolvedores de software.

“Os gestores responsáveis devem aderir e promover uma abordagem ética no ambiente de trabalho no que concerne ao desenvolvimento e manutenção de software”

“Os engenheiros devem manter a integridade e independência nos seus julgamentos profissionais”

<https://gaea.com.br/afinal-o-desenvolvedor-de-software-deve-seguir-um-codigo-de-etica/>.

* Por que os desenvolvedores de software se preocupam com o projeto lógico do sistema?

Pois o Projeto lógico é o escopo do trabalho, uma vez que as ideias e requisitos do sistema e dos clientes são levados a sério na aplicação do Projeto Físico. Sendo elaborado relatórios para obter a visão detalhada da solução, dos produtos e das integrações sistêmicas. Desse modo, ajudando a desenvolver um sistema complexo sem problemas sistêmicos ou bugs.

<https://trabsi.wordpress.com/2011/06/02/desenvolvimento-de-projeto-fisico-e-projeto-logico/>.

* O que são **fases do projeto lógico e fases do projeto físico**, no desenvolvimento do software?

**Projeto Lógico**

1. Refinar o modelo de informações empresariais ou organizacionais;
2. Descrever a lógica das informações: detalhar a lógica de cada informação do sistema proposto, descrevendo “como” as informações serão construídas e disponibilizadas aos interessados
3. Esboçar entradas e saída das informações: desenhar as telas ou relatórios do sistema de informação proposto;
4. Definir tecnologia de base para o projeto físico: relatar as configurações necessárias para hardware, software, sistemas de telecomunicações, gestão de dados e informações;
5. Elaborar plano logístico e de contingência: relatar as necessidades de materiais, móveis, instalações elétricas, pessoal, obras civis e demais infraestruturas necessárias para o sistema proposto;
6. Determinar planejamento das fases seguintes: projeto físico e projeto de implantação;
7. Analisar custos, benefícios, riscos e viabilidade;
8. Aprovar o projeto lógico: parecer ou termo de compromisso, relatando a avaliação da conformidade dos requisitos da qualidade, de produtividade e de efetividade do projeto de sistema de informação

**Projeto Físico**

1. Especificar o modelo de dados: reestruturar os dados, eliminar redundâncias, analisar as dependências funcionais;
2. Especificar o modelo de processos: identificar e reestruturar os processos (tarefas) do sistema;
3. Definir arquitetura e plano de segurança:
4. Definir os arquivos físicos e métodos de acesso do software.
5. Definir procedimentos do plano de segurança das informações (Backup).
6. Construir o sistema de informação: analisar as linguagens de programação e executar o sistema ou programar o software.
7. Finalizar o sistema de informação: testes dos módulos, dos programas e arquivar os resultados.
8. Aprovar o projeto físico.

**Projeto de Software Lógico x Físico**

* Projeto lógico (independente de implementação) é executado para produzir um projeto que poderia ser implementado em diferentes plataformas (hardware, linguagem de programação, SGBD)
* Projeto físico (implementação específica) é executado para produzir um projeto que é específico para a plataforma escolhida
* Algumas vezes se a plataforma é conhecida quando começa o projeto, não haverá o estágio de projeto lógico.

<https://trabsi.wordpress.com/2011/06/02/desenvolvimento-de-projeto-fisico-e-projeto-logico/>.

* O que é design de um projeto de software?

**Design de Software** é a parte da engenharia de software que se **encarrega** de fazer todo o **planejamento anterior ao desenvolvimento**, incluindo a **definição da arquitetura do software**, e **transformar tudo em um documento** ou conjunto de documentos capazes de serem interpretados diretamente pelo programador.

Para atingir este objetivo, o projetista deve **mapear as estruturas e funcionalidades identificadas na análise de requerimentos** dentro do contexto e das restrições da arquitetura, de forma a tornar possível a construção do software. Ao longo do tempo e nos diversos processos de software existentes, várias ferramentas foram idealizadas para facilitar e atingir este objetivo:

1. Design por contrato
2. Model Driven Architecture (MDA) e Model Driven Design (MDD)
3. Design Patterns
4. Refatoração
5. Entre outras.

* Por que não usamos modelagem formal em desenvolvimento ágil?

Pois no desenvolvimento ágil têm funções diferentes dos métodos formais, SCRUM e XP são técnicas de aprimoramento de projetos e a modelagem formal têm como intuito de resolver problemas relacionados ao sistema e a parte prática da codificação. Portanto, não se faz necessário o uso obrigatório da modelagem formal.

* Em um projeto de desenvolvimento de software, os membros da equipe do projeto conversam, diariamente, numa rápida reunião, para verificar o andamento das tarefas e expor eventuais dificuldades. Essa equipe é multidisciplinar, composta predominantemente de profissionais experientes que trabalham em conjunto com, pelo menos, um representante do cliente. As iterações de trabalho são curtas e, ao final de cada uma delas, o produto ganha novas funcionalidades. Nesse momento, a versão atual é apresentada funcionando ao cliente, visto que ter o software funcionando é mais importante do que ter uma documentação detalhada. Qual o modelo de desenvolvimento de sistemas que se encaixa nesse cenário?
* Histórias de usuários na atividade de planejamento, encorajamento de uso de cartões CRC e de re-fabricação, reuniões em pé e programação em pares são características típicas de qual modelo de processo de software?

Em um projeto de desenvolvimento de software, os membros da equipe do projeto conversam, diariamente, numa rápida reunião, para **verificar o andamento das tarefas e expor eventuais dificuldades**. Essa equipe é multidisciplinar, composta predominantemente de **profissionais experientes que trabalham em conjunto com, pelo menos, um representante do cliente**. As **iterações de trabalho são curtas e, ao final de cada uma delas, o produto ganha novas funcionalidades**. Nesse momento, a versão atual é apresentada funcionando ao cliente, visto que ter **o software funcionando é mais importante do que ter uma documentação detalhada**. Qual o modelo de desenvolvimento de sistemas que se encaixa nesse cenário?

* SCRUM

Histórias de usuários na atividade de planejamento, encorajamento de uso de cartões CRC e de pré-fabricação, reuniões em pé e programação em pares são características típicas de qual modelo de processo de software?

* XP