UNINOVE

Processo de software: fases e atividades essenciais

APRESENTAR AS FASES E ATIVIDADES ESSENCIAIS EM UM PROCESSO DE SOFTWARE.

Atividades essenciais no processo de desenvolvimento de software

Para que a Engenharia de Software possa ser aplicada como uma abordagem disciplinada para o desenvolvimento, operação e manutenção de um software, um processo deve ser definido.
De uma forma geral, um processo é caracterizado por fases/atividades. Existem várias propostas e denominações para as fases/atividades do ciclo de vida de um software. Uma proposta possível das fases a serem seguidas é a descrita a seguir, conforme apresentada por Pressman.

Fases de definição, desenvolvimento, manutenção

1. Fase de definição: essa fase se concentra no "quê" o sistema de software irá realizar, isto é, identifica que informação deve ser processada, que função e desempenho são desejados, que comportamento deve ser esperado do sistema, que interfaces devem ser estabelecidas, que restrições de projeto existem, que critérios de validação são necessários. Nessa fase, os requisitos-chave do sistema e do software são identificados. Esta fase engloba três importantes etapas: engenharia de sistemas ou de informação, planejamento do projeto, análise de requisitos.

2. Fase de desenvolvimento: focaliza "como" o desenvolvimento será realizado, isto é, define como os dados devem ser estruturados, como as funções devem ser implementadas, como os detalhes procedimentais devem ser implementados, como as interfaces devem ser caracterizadas, como o projeto deve ser traduzido em uma linguagem de programação, como o teste vai ser realizado.Nessa fase, 3 etapas técnicas específicas ocorrerão: projeto do software, geração de código, teste de software.
3. Fase de Manutenção: tem como alvo as modificações e manutenções que o software sofrerá. Quatro tipos de modificações são encontradas durante essa fase:

- Manutenção corretiva: modifica o software para corrigir defeitos.

- Manutenção adaptativa: modifica o software para acomodar mudanças no seu ambiente externo (processador, sistema operacional, etc.).
- Manutenção de aperfeiçoamento: aprimora o software além dos requisitos funcionais originais (cliente/ usuário reconhece e solicita funcionalidades adicionais que trarão benefícios, à medida que o software é usado).
- Manutenção preventiva: faz modificações nos programas de modo que eles possam ser mais facilmente corrigidos, adaptados e melhorados.

Estas 3 fases são complementadas por atividades "guarda-chuva":

- Controle e Rastreamento do Projeto.
- Gestão de Riscos.
- Revisões Técnicas Formais.
- Garantia de Qualidade.
- Gestão de Configuração de Software.

- Produção e Preparação de Produtos do Trabalho (documentos).
- Gestão de Reusabilidade.
- Medição.

Essas atividades são aplicadas ao longo do processo de software.

Atividades de especificação, desenvolvimento, validação, evolução

Em relação às atividades a serem executadas no processo de desenvolvimento de software, basicamente, podemos descrevê-las da seguinte maneira [de acordo com Leite (2007)],:

1. Especificação

Nesta etapa, os profissionais devem realizar uma tarefa de extrema importância, que é fazer o levantamento dos requisitos de software e modelos de domínio. Tais requisitos são fundamentais para que o engenheiro de software possa elaborar um plano de desenvolvimento de software, indicando em detalhes os recursos necessários (humanos e materiais), bem como as estimativas de prazos e custos (cronograma e orçamento).

2. Desenvolvimento

A etapa de desenvolvimento ou de produção do software inclui todas as atividades que têm por objetivo a construção do produto. Esta etapa inclui as atividades de projeto e implementação do software.

- Projeto:: A atividade de projeto compreende todo o esforço de concepção e modelagem que tem por objetivo descrever como o software será implementado. O projeto inclui:
    - Projeto conceitual: envolve a elaboração das ideias e conceitos básicos que determinam os elementos fundamentais do software a ser desenvolvido.
    - Projeto da interface com o usuário: envolve a elaboração das formas como ele vai interagir para realizar suas tarefas, a escolha dos objetos de interfaces (botões, menus, caixas de texto, etc.), o layout de janelas e telas, dentre outros. A interface deve garantir uma usabilidade satisfatória do software e é um fator fundamental de seu sucesso.
    - Projeto da arquitetura do software: refere-se à elaboração de uma visão macroscópica do software em relação aos componentes que interagem entre si. São exemplos de visões arquitetônicas, a visão conceitual, visão de módulos, visão de código e visão de execução.
    - Projeto dos algoritmos e estruturas de dados: objetiva determinar, de maneira independente da linguagem de programação a ser adotada, as soluções algorítmicas e as estruturas de dados associados.
- Implementação: A implementação envolve principalmente as atividades de codificação e compilação.

3. Validação

São atividades que se destinam a mostrar que o sistema está de acordo com a especificação e que ele atende às expectativas de clientes e usuários. A validação, por exemplo, visa assegurar que o programa está fazendo o que foi definido na sua especificação. A verificação, por sua vez, visa certificar se o programa está correto, isto é, se não possui erros de execução e está fazendo de forma correta suas funcionalidades. Existem diferentes formas de verificação e validação. Os testes de correção, desempenho, confiabilidade, robustez, usabilidade, dentre outros, podem ser usados para avaliar diversos fatores de qualidade a partir da execução do software.

4. Evolução (ou retirada)

Durante sua vida, um software pode sofrer vários tipos de manutenção. Ela pode ocorrer de forma corretiva, adaptativa, de aperfeiçoamento e evolutiva. Também podem ser relativas à resolução de problemas referentes à qualidade do software (falhas, baixo desempenho, baixa usabilidade, falta de confiabilidade, etc.), à produção de novas versões do software de forma a atender aos novos requisitos dos clientes, ou adaptar-se às novas tecnologias que surgem (hardware, plataformas operacionais, novas linguagens e paradigmas, etc). Mudanças no domínio de aplicação implicam novos requisitos e incorporação de novas funcionalidades. Surgimento de novas tecnologias de software e hardware e mudanças para uma plataforma mais avançada também requerem evolução.

A fase de retirada ou desuso de um software é um grande desafio para os tempos atuais. Há diversos sistemas estão em funcionamento em empresas e possuem ótimos níveis de confiabilidade e de correção. No entanto, estes sistemas precisam evoluir para novas plataformas operacionais ou para permitirem a incorporação de novos requisitos e funcionalidades. A retirada desses sistemas legados em uma empresa é sempre uma decisão difícil: como se desfazer daquilo que é confiável e ao quais funcionários estão acostumados após anos de treinamento e utilização?

Nestes casos, processos de reengenharia podem ser aplicados para viabilizar a transição ou a migração de um software legado para um novo software de forma a proporcionar uma retirada mais suave.

Agora que você já estudou esta aula, resolva os exercícios e verifique seu conhecimento. Caso fique alguma dúvida, leve a questão ao Fórum e divida com seus colegas e professor.

EXERCÍCIOS (https://ead.uninove.br/ead/disciplinas/impressos/_g/pdsoft80_100/a03ex01_pdsoft80_100.pdf)

REFERÊNCIA

PRESSMAN, R. S. *Engenharia de software*. 7ª ed. São Paulo: McGraw-Hill, 2010.

SOMMERVILLE, Ian. *Engenharia de software*. São Paulo: Addison-Wesley, 2007.