UNINOVE

Conceito e aplicação de desenvolvimento tradicional. Exemplos

O PRESENTE TÓPICO TEM POR OBJETIVO VIABILIZAR O CONTATO DO ALUNO COM OS CONCEITOS E COM A APLICAÇÃO DOS MÉTODOS TRADICIONAIS DE DESENVOLVIMENTO DE SOFTWARE, A FIM DA CONSTRUÇÃO DE COMPETÊNCIAS INERENTES AO TEMA EM QUESTÃO.

AUTOR(A): PROF. ANDRE RONALDO RIVAS

Conceito e aplicação de desenvolvimento tradicional

Uma metodologia de desenvolvimento de software pode ser definida por uma estrutura utilizada para organizar, planejar e controlar o processo de desenvolvimento de programas responsáveis pela operação de computadores, a fim de que estes executem tarefas específicas; sendo assim, inclui a predefinição de produtos e artefatos característicos, criados e praticados por equipes envolvidas neste trabalho. Neste sentido, Rosa e Massukado (2015) preconizam que o desenvolvimento de software exige controles e processos organizacionais eficientes para a sinergia de competências entre diferentes indivíduos de uma equipe.

Há uma grande variedade de metodologias para desenvolvimento de software, cada uma destas é mais adequada a tipos específicos de demandas, para estes distintos cenários há de se ponderar aspectos técnicos, organizacionais, de projeto e de equipe. Essas estruturas de desenvolvimento de software costumam estar vinculadas a algum tipo de organização, que desenvolve, suporta o uso e promove a estrutura da metodologia, registrando-a em algum tipo de documentação formal.

As metodologias tradicionais de desenvolvimento de software são baseadas em fases e/ou estágios pré-organizados do ciclo de vida para a produção do software. Neste contexto, o fluxo de desenvolvimento é unidirecional, dos requisitos ao design e, em seguida, ao desenvolvimento, aos testes e por fim, manutenção. Em abordagens clássicas como é o caso do modelo Cascata (também conhecido por Waterfall), cada fase possui resultados e documentação específica que são submetidos a um processo completo de revisão.

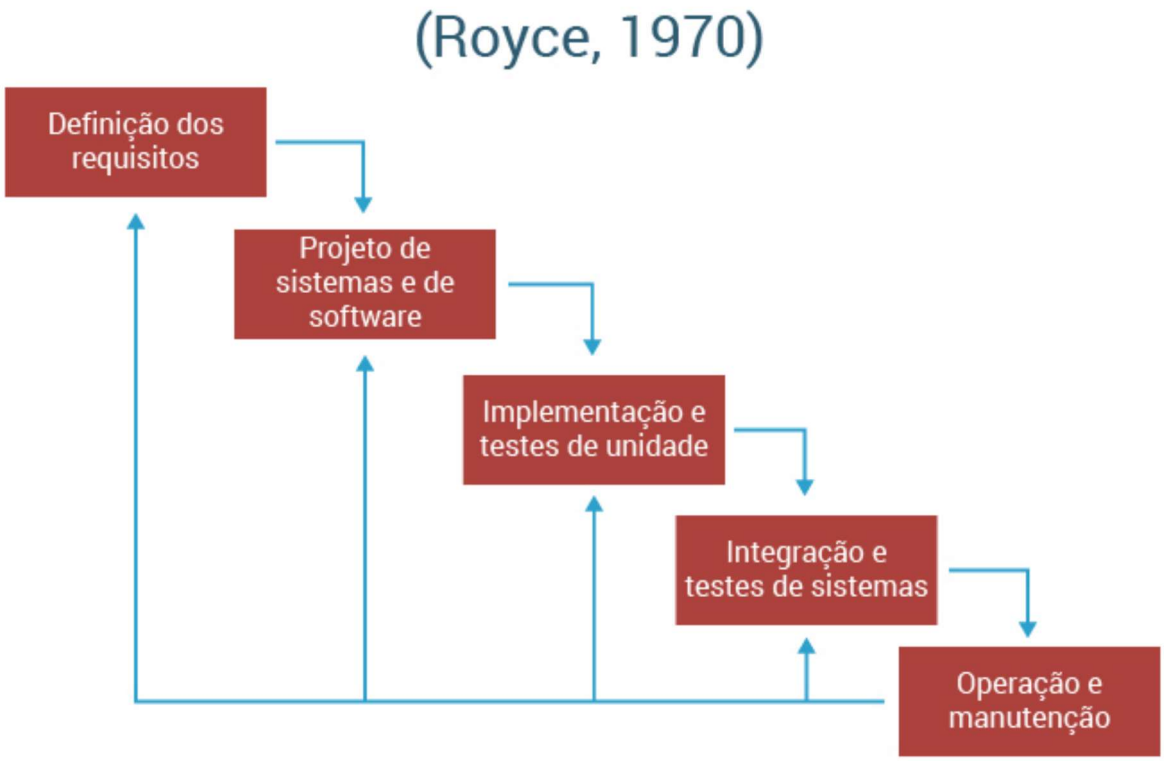
Modelo Cascata (Waterfall)

O primeiro documento público capaz de evidenciar crédito na criação do Modelo Cascata é de 1970, este confere tal mérito a Winston Walker Royce (15 de agosto de 1929 - 7 de junho de 1995), cientista da computação americano, diretor do Lockheed Software Technology Center de Austin, Texas.

O Modelo Cascata em seu formato original é caracterizado por uma divisão das atividades em fases sequenciais (lineares), em que cada uma destas depende das entregas da etapa anterior, viabilizadas diante de certa especialidade (levantamento/definição de requisitos, design/projeto de sistemas e de software, implementação e testes de unidade, integração e testes de sistemas, operações e manutenção); trata-se de uma abordagem típica de um projeto de engenharia.

> A falta de um método estruturado (como o Modelo Cascata) ou etapas formalizadas do processo de trabalho acarretou em processos caóticos de desenvolvimento, nos quais havia muito desperdício de recursos, pouca eficiência dos programas dando origem, em inúmeros casos, a programas que apenas o programador original conseguia manter ou atualizar.
>
> AMORIM, H.; MAURÍCIO, R.G. TAYLORIZAçãO E AUTO-TAYLORIZAçãO DO TRABALHO: AS METODOLOGIAS áGEIS NA INDúSTRIA DE SOFTWARE. SECULO XXI, SANTA MARIA, V. 8, N. 2, 2018.

Considerando o propósito do desenvolvimento de software, torna-se fácil a percepção de que este é um modelo limitado quanto a adaptação ou iterações, sobretudo porque o progresso flui em uma única direção (descendente como uma cascata), percorrendo cada uma das etapas, sem oportunidade de ajustes, sem voltar à fase anterior para revistar a concepção do produto de software. Ainda assim, consolidou-se como uma das mais importantes iniciativas da Engenharia de Software; cabe mencionar a aderência do modelo ao cenário vigente da época de sua criação.



Legenda: MODELO CASCATA (ROYCE, 1970)

Etapas do Modelo Cascata: Definição de Requisitos

Os requisitos são um conjunto de necessidades dos envolvidos (stakeholders), definidos nas fases iniciais de um projeto e servem como especificação do que deve ser implementado. São descrições de como o sistema deve se comportar. Em suma, é algo que a aplicação deve possuir para satisfazer um contrato, dentro do prazo, custo e qualquer outra característica ou comportamento acordados.

A fase de Definição de Requisitos é certamente a etapa mais delicada do Modelo Cascata, e claro, é a mais propensa a erros; como dela decorre as demais, aqui não se deve economizar tempo, por vezes se caracteriza como a fase mais cara de toda a empreitada. Neste momento, é criado um documento de especificação de requisitos, que serve ao objetivo de orientação para a próxima fase do modelo.

Etapas do Modelo Cascata: Projetos de Sistema e de Software

O documento de especificação de requisitos produzido no estágio anterior serve como entrada desta fase. Antes de começar de fato a codificação, é muito importante entender o que será criado e como. O Projeto de Sistema e de Software auxilia a definição da arquitetura geral do sistema (design) e como os requisitos serão nela articulados.

Alguns recursos (ferramentas) frequentemente encontrados nesta etapa do modelo são:

- Design por Contrato;
- Arquitetura por Modelo;
- Design Patterns;
- Refatoração.

Etapas do Modelo Cascata: Implementação e Testes de Unidade

Na fase de Implementação e Testes de Unidade, logo ao receber os documentos de projeto do sistema, o trabalho é dividido em módulos/unidades e a codificação real é iniciada. O sistema é desenvolvido pela primeira vez em pequenos programas chamados unidades, que são integrados em seguida. Cada unidade é desenvolvida e testada por sua funcionalidade; isso é chamado de teste de unidade. O teste de unidade verifica principalmente se os módulos/unidades atendem às suas especificações.

Etapas do Modelo Cascata: Integração e Testes do Sistema

Conforme especificado na etapa de Implementação e Testes de Unidade, o sistema é dividido primeiramente em unidades que são desenvolvidas e testadas por suas funcionalidades. Na fase de Integração e Testes do Sistema, essas unidades são integradas a um sistema completo e testadas a fim de verificar se todos os módulos/unidades interagem de modo esperado entre si, fazendo com que o sistema (como um todo) se comporte de acordo com as especificações.

Etapas do Modelo Cascata: Operação e Manutenção

A fase de Operação e Manutenção do Modelo Cascata é praticamente uma fase sem fim (muito longa). Geralmente, os problemas com o sistema desenvolvido (que não são encontrados durante o ciclo de vida do desenvolvimento) surgem após o início do uso prático; portanto, os problemas relacionados ao

sistema são resolvidos após a implantação do sistema. Nem todos os problemas aparecem diretamente, mas surgem de tempos em tempos e precisam ser resolvidos; portanto, esse processo é chamado de manutenção.

VANTAGENS DO MODELO CASCATA

- Este modelo é simples, de fácil compreensão e utilização.
- É também mais fácil de gerenciar devido à rigidez do modelo - cada fase possui entregas específicas e um processo de revisão.
- Neste modelo, as fases são processadas e concluídas uma de cada vez; neste sentido, é mais fácil prever custos de produção.
- O Modelo Cascata funciona bem para projetos menores, onde os requisitos são claramente definidos e muito bem compreendidos.

DESVANTAGENS DO MODELO CASCATA

- Quando um software está no estágio de teste, é muito difícil voltar e alterar algo que não foi bem pensado no estágio de conceito.
- Níveis de risco e incerteza são altos durante o ciclo de vida.
- Não é um bom modelo para projetos complexos.
- Não se mostra como boa opção para projetos longos.
- Não é adequado para projetos em que os requisitos estão sujeitos às mudanças frequentes de especificação.

ATIVIDADE

Selecione a alternativa que não corresponde ao Modelo Cascata:

A. O modelo em cascata é um modelo de desenvolvimento de software sequencial no qual o desenvolvimento é visto como um fluir constante para frente (como uma cascata) através das fases de análise de requisitos, projeto, implementação, testes (validação), integração, e manutenção de software.

B. O modelo em cascata move-se para a próxima fase somente quando a fase anterior está completa e perfeita.

C. O desenvolvimento de fases no modelo em cascata são discretas, e não há pulo para frente, para trás ou sobreposição entre elas.

D. Baseia-se na ideia de uma implementação inicial, expondo o resultado aos comentários do usuário e refinando-o por meio de versões até que um sistema adequado venha a ser construído.

ATIVIDADE FINAL

Metodologias de desenvolvimento de software se baseiam em um modelo de ciclo de vida ou modelo de processo, tais como cascata, espiral e prototipagem; sendo assim, é correto afirmar que:

A. Metodologias que seguem o modelo em espiral normalmente possuem um maior potencial de risco, uma vez que esse modelo não lida explicitamente com isso.
B. Metodologias que seguem o modelo de prototipagem devem, necessariamente, descartar os protótipos construídos; dessa forma, essas metodologias costumam ser mais custosas.
C. Metodologias que seguem o modelo em cascata possuem fases bem definidas, que podem ser desenvolvidas incrementalmente, em diferentes ciclos de desenvolvimento, isto é, a fase seguinte pode ser executada, ainda que a fase anterior não tenha sido finalizada completamente.
D. Metodologias que seguem o modelo em cascata possuem fases bem definidas e executadas sequencialmente. Além disso, não há sobreposição entre as fases, isto é, a fase seguinte somente pode ser executada após a finalização da fase anterior.

REFERÊNCIA

1. AMORIM, H.; MAURÍCIO, R.G. Taylorização e auto-taylorização do trabalho: as metodologias ágeis na indústria de software. Seculo XXI, Santa Maria, v. 8, n. 2, 2018. Disponível em: https://search.proquest.com/docview/2163363358/abstract/ABB1B81F97894B0CPQ/7?accountid=43603 (https://search.proquest.com/docview/2163363358/abstract/ABB1B81F97894B0CPQ/7?accountid=43603) . Acesso em: 11 junho 2020.
2. ROSA, G.D.A.; MASSUKADO, L.M.; STUMPF, E.R.T. Análise de um time de trabalho à luz do framework scrum: o caso de uma organização pública federal de educação profissional e tecnológica/Analysis of a teamwork in light of scrum framework: the case of a public federal organization of technological and profesional education. HOLOS, Natal, v. 31, n. 5, p. 338-349, 2015. Disponível em: https://search.proquest.com/docview/1725004794/9872E136AB3A44A5PQ/1?accountid=43603 (https://search.proquest.com/docview/1725004794/9872E136AB3A44A5PQ/1?accountid=43603)?. Acesso em: 11 junho. 2020.
3. ROYCE, Winston W. Managing the development of large software systems: concepts and techniques. In: Proceedings of the 9th international conference on Software Engineering. 1987. p. 328-338.