**Prática 1:**

**Visão Geral dos Modelos de Processos de Desenvolvimento de Software**

**1. Objetivos da Prática.**

Esta prática tem como objetivo que o aluno aprenda e/ou aperfeiçoe as habilidades e desenvolva competências necessárias para identificar e resolver problemas e desafios técnicos relativo a seleção e utilização de processos de desenvolvimento de software.

Espera-se que ao final desta prática o aluno:

* Saiba identificar qual é o problema central em um estudo de caso apresentado.
* Conheça o conceito de paradigma.
* Saiba quais são os paradigmas dos processos de desenvolvimento de software.
* Saiba identificar qual é o processo de software que está sendo utilizado.
* Consiga argumentar e expor o porquê da caracterização deste processo.
* Consiga identificar e argumentar quais são as vantagens, desvantagens de cada tipo de processo de desenvolvimento.

**Introdução Conceitual.**

A seleção de um modelo de processo de desenvolvimento de software é uma atividade central e crítica durante o projeto. A escolha deste modelo irá definir quais atividades serão executadas, e em qual sequência elas serão executadas.

Podemos definir “processo” como sendo um conjunto de atividades que devem ser executadas em uma dada sequência ao longo do tempo. Estas atividades recebem um conjunto de dados e informações, e por sua vez ao final da execução da atividade são gerados novos dados e informações, os quais serão consumidos pela atividade seguinte do processo. A execução (realização) de uma atividade é feita com o uso de técnicas, métodos e/ou ferramentas computacionais.

Basicamente existem dois paradigmas de processos de software. Os dois paradigmas são os processos do tipo preditivo e os processos do tipo iterativo e incremental. Os processos do tipo preditivo são processos onde todo o planejamento é feito a priori, e nada deverá ou deveria alterar este planejamento a posteriori. Cada atividade do projeto é bem definida, e uma atividade somente pode ser executada se, e somente se, a sua atividade anterior foi executada em sua totalidade e foi considerada como concluída. Este tipo de processo foi criado no final dos anos 60, e o mais conhecido e famoso dos processos preditivos é o processo da cascata (também conhecido como sequencial ou linear), o qual é apresentado na figura 1 abaixo.

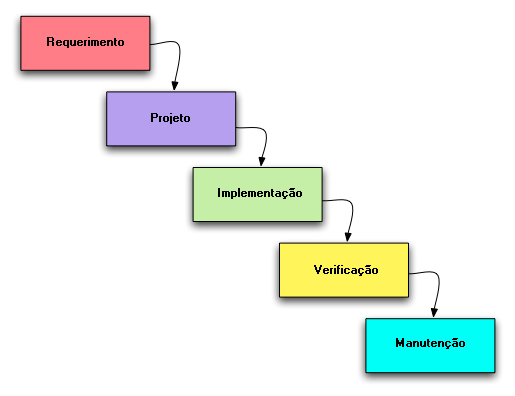


Figura 1 – Processo de desenvolvimento sequencial.

Neste tipo de processo ao lado, a fase de Projeto somente terá início após concluída a fase Requerimento. Isto significa que o projeto somente irá mover-se para a fase subsequente depois de todos os requisitos terem sido identificados, modelados, documentados e aprovados. A fase de Implementação somente será iniciada após a fase de projeto ter sido concluída totalmente, e assim por diante.

O processo da cascata foi proposto por Winston Royce em uma conferência acadêmica nos USA, e documentado na publicação: *Managing the Development of Large Software Systems*, Proceedings of IEEE Western Electronic Show and Convention (WesCon), Los Angeles, USA, 1970.

Como todos e qualquer processo de desenvolvimento o processo da cascata possui pontos fortes e pontos fracos.

|  |  |
| --- | --- |
| **Pontos Fortes** | **Pontos Fracos** |
| * + Fácil de entender, fácil de usar.   + Fornece um processo de trabalho estruturado para uma equipe inexperiente.   + Marcos das mudanças de uma fase para outra são bem definidos.   + Define e estabiliza os requisitos desde o início do projeto. | * + Os requisitos devem ser estáveis, o que muitas vezes não ocorre.   + Alterações geram impactos grandes, uma vez que as fases anteriores devem ser refeitas para alterar a mudança que ocorreu.   + Pouca oportunidade para que o cliente visualize o software durante o seu desenvolvimento, e quando e consegue visualizar pode ser tarde demais para solicitar correções e alterações.   + Não prioriza o ataque aos riscos desde o início do projeto. Todos os requisitos devem ser concluídos ao final da fase de Requisitos, e todos os requisitos serão modelados em conjunto na fase de Projeto etc. Isto impões dificuldades em lidar com requisitos que representam riscos. |

O processo iterativo e incremental caracteriza-se por seus dois elementos: a iteração e o incremento, os quais são definidos a seguir.

* Definição de Iterativo:
  + Fazer de forma semelhante repetidamente (n vezes).
* Definição de incremental:
  + Que vai aumentando gradativamente por partes, ou seja, incrementos.

É iterativo na medida em que envolve o refinamento sucessivo da definição e implementação da solução pela aplicação repetitiva das atividades de desenvolvimento. É incremental no que cada passagem através do ciclo iterativo aumenta a compreensão do problema e a capacidade oferecida pela solução.

Modelo iterativo e incremental é uma alternativa para solução de problemas enfrentado no modelo cascata. Nesse modelo, considera-se o desenvolvimento do software em ciclos iterativos onde uma pequena porção dos requisitos passa por todas as etapas de desenvolvimento como em um modelo cascata. Ao final de cada ciclo de iteração têm-se uma nova versão do software, a qual será incrementada a cada novo ciclo que ocorrer. Isso reduz a ansiedade e a expectativa do usuário em relação ao software. Embora pareça que o modelo imprima uma forma mais ágil de desenvolvimento, um planejamento formal não é desprezado.

A ideia das iterações desse modelo é permitir que o sistema seja melhor compreendido e refinado a cada etapa. A figura 2 ilustra esse paradigma de processo de desenvolvimento.

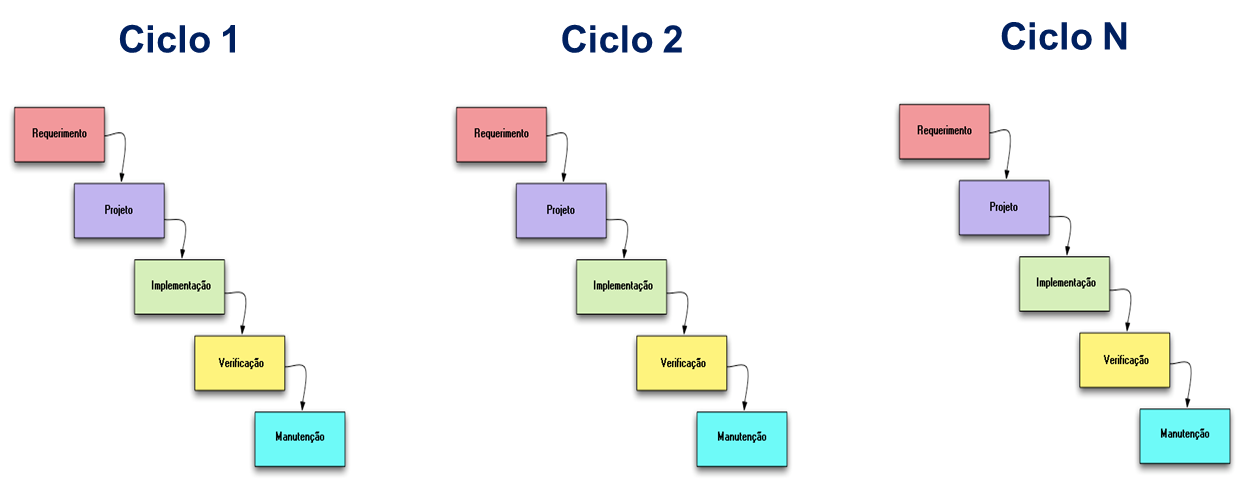


Figura 2 – Processo de desenvolvimento iterativo e incremental.

O paradigma de processo de desenvolvimento iterativo e incremental foi formalizado e por Barry Bohem em seu artigo “*A Spiral Model of Software Development and Enhancement”* publicado na IEEE Software Magazine em 1988. Neste artigo Bohem também propôs um modelo de desenvolvimento iterativo e incremental chamado de “modelo espiral”.

O paradigma iterativo e incremental é também utilizado por todos os denominados “métodos ágeis”. Dentre estes métodos, talvez, o mais famoso dentre eles seja o método ágil “Scrum”. O conceito de iterativo e incremental está presente no Scrum no seus Sprints. Cada Sprint nada mais é do que um “uma iteração em um incremento do escopo (requisitos)” do projeto.

|  |  |
| --- | --- |
| **Pontos Fortes** | **Pontos Fracos** |
| * + Permitem que o cliente vá visualizando progressivamente o produto à medida que o projeto avança. Isto é importante para garantir que o projeto está desenvolvendo o produto correto.   + Mudanças são mais facilmente absorvidas, uma vez que afetarão somente a parte dos requisitos implementados até o ciclo do momento.   + Permite que o cliente vá amadurecendo melhor os requisitos, pois somente os requisitos que serão implementados no ciclo em questão precisam ser aprovados.   + Permite um melhor gerenciamento dos riscos, pois os requisitos que apresentem riscos podem ser selecionados para serem implementados log no primeiro ciclo. | * + Requer uma equipe com maior experiência, e maturidade em desenvolvimento de software.   + Impõe uma mudança cultural forte na empresa, pois o usuário deverá ter uma participação bastante intensa durante todo o projeto, e não somente no início no final como no processo da cascata.   + Quantos ciclos (iterações) serão necessários para desenvolver o produto? O planejamento do projeto utilizando este paradigma é mais complicado, o que gera um esforço de gerenciamento maior. |

**Estudo de Caso Winburg.**

Para reduzir o congestionamento de tráfego no centro da cidade de Winburg, em Indiana - USA, o prefeito convenceu a cidade a criar um sistema de transporte público. As ruas terão faixas exclusiva para os ônibus para poderem trafegar mais rapidamente, e os passageiros serão encorajados a "estacionar e andar"; isto é, estacionar seus carros em áreas de estacionamento suburbano e, em seguida, pegar ônibus a partir de lá para irem trabalhar e voltar com um custo de um dólar por viagem.

Cada ônibus deverá ter uma máquina para pagamento automático da passagem, a qual irá ler automaticamente as cédulas de dólares. Os passageiros irão inserir uma cédula de dólar no slot da máquina de leitura de cédulas enquanto eles entrarem no ônibus. Os sensores dentro da máquina irão escanear a cédula fazendo a sua leitura. O software na máquina de leitura de cédulas utiliza um algoritmo de reconhecimento de imagem para decidir se o passageiro realmente inseriu uma cédula de dólar válida no slot. É importante que a máquina de leitura seja precisa, porque assim poderá ser evitado fraude com o uso de cédulas de outros países, cédulas falsificadas etc etc. Por outro lado, se a máquina rejeitar regularmente as cédulas de dólar válidas, então os passageiros ficarão relutantes em usar os ônibus. Além disso, a máquina de tarifa deve ser rápida o suficiente para evitar a formação de filas, o que também desestimularia a utilização do novo serviço de ônibus. Os passageiros ficarão relutantes em usar os ônibus se a máquina demorar 15 segundos para ler e verificar a autenticidade das cédulas de dólares. Como o serviço de ônibus terá um horário pré-programado para partida dos ônibus, isto faria que um número relativamente pequeno de passageiros fosse levado a bordo de um ônibus. Portanto, os requisitos para o software da máquina de leitura de cédulas incluem o tempo de resposta médio de menos de 1 segundo, e uma precisão média acerto na validação das cédulas de pelo menos 98%.

*Episódio 1* – A primeira versão do software teve o seu desenvolvimento concluído.

*Episódio 2* – Os testes mostraram que a restrição necessária de um tempo de resposta médio de 1 segundo para decidir sobre a validade de uma nota de dólar não foi alcançada. Na verdade, o software esta gastando, em média, 10 segundos para obter uma resposta. A gerência sênior investigou, e descobriu a causa do problema. Aparentemente, para obter a precisão de acerto necessária de 98%, um programador foi instruído por seu líder para utilizar números de precisão dupla para todos os cálculos matemáticos. Como resultado, cada operação leva pelo menos duas vezes o tempo que seria com os números habituais de precisão simples. O resultado é que o programa é muito mais lento do que deveria ser, resultando em um longo tempo de resposta. Os cálculos mostram que, apesar do que o líder disse ao programador, a precisão estipulada de 98% pode ser alcançada, mesmo que sejam utilizados números de precisão simples. O programador começou a fazer as mudanças necessárias no código do software.

*Episódio 3* – Antes que o programador possa completar seu trabalho de modificação na precisão de dupla para simples, outros testes do sistema mostram que, mesmo que as mudanças indicadas anteriormente na implementação fossem feitas, o sistema ainda teria um tempo de resposta médio de mais de 4,5 segundos, e não perto do 1 segundo estipulado nos requisitos iniciais do projeto. Uma nova análise foi feita, e identificou-se que o algoritmo de reconhecimento de imagem é bastante complexo. Felizmente, um algoritmo mais simples e rápido acabou de ser desenvolvido por um centro de pesquisas. Isto obrigará que o software da máquina de leitura de cédulas seja redesenhado, e reescrito usando o novo algoritmo. Isso resultará que o tempo médio de resposta seja alcançado com sucesso.

*Episódio 4* – No momento, o projeto está consideravelmente atrasado, e o custo atual está muito mais do que foi estimado no orçamento. O prefeito, um empreendedor bem-sucedido, teve a brilhante ideia de pedir ao time de desenvolvimento de software que tente aumentar a precisão do componente de reconhecimento de cédulas de notas de dólar do sistema tanto quanto possível, para vender o software para às empresas que fazem uso de máquinas de venda automática. Desta forma, a prefeitura poderia recuperar parte dos custos de desenvolvimento do software.

Para atender a este novo requisito, foi realizado um novo desenho do software, o qual melhorou a precisão média de acerto na leitura das cédulas para mais de 99,5%. A administração decidiu instalar essa nova versão do software nas máquinas tarifárias dos ônibus. Neste momento, o desenvolvimento do software está completo. A cidade já conseguiu vender o software para duas pequenas empresas de máquinas de venda automática, totalizando a recuperação dos custos de cerca de um terço do valor gasto no desenvolvimento do software.

*Epílogo* – Alguns anos depois, os sensores dentro da máquina de tarifa tornaram-se obsoletos e precisaram ser substituídos por um modelo mais novo. A administração da cidade sugeriu aproveitar a mudança, e ao mesmo tempo, atualizar o hardware. Os desenvolvedores do software avaliaram que mudar o hardware irá implicar, que um novo software também seja desenvolvido. Eles sugerem reescrever o software em uma linguagem de programação.

O status do projeto, informado pelo gerente de projetos, na data de hoje informa, que o projeto está atrasado 6 meses, e 25% acima do orçamento inicial. Entretanto, todos os envolvidos estão confiantes de que o novo sistema será mais confiável e de maior qualidade, apesar das "pequenas discrepâncias" em atender aos requisitos de tempo de resposta e precisão.

Atividade: Monte um texto explicando com a melhor solução (processo)para o projeto acima. Explique o porquê!

*Resposta:* *O modelo de processo preditivo pode ser a melhor escolha para o projeto acima, visto que o prazo para entrega do projeto ultrapassou bastante ao final da narrativa e, além disso, o custo do orçamento encareceu um valor considerável de 25%. Acredito que o principal obstáculo enfrentado pela equipe foi o tempo gasto no desenvolvimento do software, visto que a cada problema descoberto, os programadores refaziam o código do zero com a alteração aplicada. A repetição desse processo resultou no atraso de 6 meses devido à falta de organização da equipe de projetos. O recomendado seria utilizar um modelo para pessoas inexperientes que só possa avançar de etapa quando os requisitos forem alcançados para tal. Dessa forma, o projeto possivelmente estará completo dentro do prazo, visto que a descoberta de erros e seu tratamento serão mais rápidos.*