UNICURITIBA – CENTRO UNIVERSITÁRIO

HENRIQUE MAKUCH DA SILVEIRA

**VISÃO E ANÁLISE DE PROJETOS**

**DESIGN E ARQUITETURA**

CURITIBA

2023

HENRIQUE MAKUCH DA SILVEIRA

**VISÃO E ANÁLISE DE PROJETOS**

**DESIGN E ARQUITETURA**

Trabalho prático apresentado ao curso de Análise e Desenvolvimento de Software ADS como requisito parcial à obtenção de nota em Modelos Métodos e Técnicas Engenharia de Software.

Professor: Rubem Koide

CURITIBA

2023

**ANÁLISE COMPARATIVA DE MODELOS DE PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE**

|  |  |
| --- | --- |
| **MODELO CASCATA** | |
| **Tempo** | Rígido e sequencial, o que pode resultar em prazos mais longos. |
| **Erros** | Menos flexibilidade pode levar a descoberta tardia de erros. |
| **Funcionários** | Maior necessidade de especialistas em fases específicas. |
| **Custo** | Pode ser mais previsível, mas mudanças são dispendiosas. |

|  |  |
| --- | --- |
| **MODELO INCREMENTAL** | |
| **Tempo** | Desenvolvimento por partes, permitindo entregas mais rápidas. |
| **Erros** | Identificação precoce de problemas em fases iniciais. |
| **Funcionários** | Possibilidade de reutilização de equipes em cada incremento. |
| **Custo** | Custos mais distribuídos ao longo do ciclo. |

|  |  |
| --- | --- |
| **MODELO ESPIRAL** | |
| **Tempo** | Iterativo, permitindo ajustes contínuos ao longo do ciclo. |
| **Erros** | Abordagem iterativa favorece a identificação antecipada de problemas. |
| **Funcionários** | Maior flexibilidade, possibilitando adaptação rápida. |
| **Custo** | Controle contínuo de riscos pode aumentar a eficiência. |

|  |  |
| --- | --- |
| **PROTOTIPAÇÃO** | |
| **Tempo** | Rápido feedback devido à criação de protótipos. |
| **Erros** | Identificação precoce de requisitos mal compreendidos. |
| **Funcionários** | Envolvimento contínuo do cliente pode exigir colaboração estreita. |
| **Custo** | Menor custo de correção de erros no início do ciclo. |

|  |  |
| --- | --- |
| **RUP** | |
| **Tempo** | Estruturado em fases, com ênfase na qualidade. |
| **Erros** | Abordagem iterativa permite melhorias contínuas. |
| **Funcionários** | Papéis definidos, proporcionando clareza nas responsabilidades. |
| **Custo** | Controle rigoroso pode resultar em menor risco de desvios. |

**CONCLUSÃO**

Cada modelo apresenta vantagens e desvantagens, sendo a escolha dependente das características do projeto. A flexibilidade do modelo incremental e espiral pode ser crucial em ambientes dinâmicos, enquanto o cascata pode ser apropriado para projetos mais estáveis. A prototipação destaca-se na validação contínua de requisitos, enquanto o RUP proporciona um equilíbrio entre estrutura e adaptabilidade. A escolha deve considerar as necessidades específicas do projeto e a dinâmica do ambiente de desenvolvimento.

Nos primeiros “tick cloks”, o passo inicial foi colocar todos os funcionários para aprenderem os códigos para que pudessem dividir as tarefas. Um plano de lançamento foi elaborado. Em seguida, analisando os perfis dos funcionários na aba “employees”, foi designado aos funcionários mais experientes a função de elaborar o “coding”, enquanto os outros funcionários elaboravam o design e codgo. As atividades que levaram mais tempo para serem executadas foram o código e design. Após finalizados, testes de aceitação foram aplicados e o diagnóstico foi de que o código havia muitos bugs e que precisava ser consertado. Nesse momento, o cliente reclamou do atraso na entrega e todos os funcionários foram designados a consertar o bug no código a fim de acelerar a realização da tarefa. Tal atividade demorou mais tempo que o esperado e quando enfim foi finalizada e submetida ao teste de aceitação, o tempo limite acabou, encerrando a simulação com uma pontuação de 25.

O modelo cascata segue uma abordagem sequencial, em que a etapa seguinte depende da anterior. Foi possível observar tal abordagem na simulação, principalmente nas etapas de coleta de requisitos e análise para definir as funcionalidades do sistema, no projeto do sistema que inclui sua estrutura e design, na sua codificação, em testes para identificar e corrigir possíveis falhas ou erros e, por fim, na manutenção e correções de bugs e atualizações conforme necessário.