Perfil:

Emerson Antonio Klisiewicz.

Possui graduação em Ciências da Computação, pela Pontifícia Universidade Católica do Paraná (1994), Especialização em Sistemas de Informações Gerenciais , pela Pontifícia Universidade Católica do Paraná (1995), Especialização em Redes e Sistemas Distribuídos pela Pontifícia Universidade Católica do Paraná (1997), com experiência de mais de 20 anos em áreas de tecnologia de empresas estatais e financeiras. Currículo Lattes:

http://buscatextual.cnpq.br/buscatextual/visualizacv.do?id=K4282839T3

AULA 01 – Introdução a Analise de Sistemas

Introdução:

Nessa nossa primeira aula, vamos abordar a evolução da Análise de Sistemas, desde os anos 50 até os dias de hoje. Veremos como vários fatores do dia a dia em tecnologia levou-nos a chamada Crise do Software e por fim conheceremos os Métodos Clássicos de análise e desenvolvimento de software.

Contextualizando:

Software é o conjunto dos programas e dos meios não materiais que possibilitam o funcionamento do computador, na execução das diversas tarefas.

A Análise de Sistemas é a atividade de identificar os problemas do domínio, apresentar alternativas de soluções e o estudo da viabilidade de um software.

No final dos anos 40 até os anos 60, quando se iniciou a evolução dos sistemas computadorizados, grande parte dos esforços, e consequentes custos, eram concentrados no desenvolvimento do hardware, em razão, principalmente das limitações e dificuldades encontradas na época. À medida que a tecnologia de hardware foi sendo dominada, as preocupações se voltaram, para o desenvolvimento de software. Surgiram então os primeiros sistemas operacionais, assim como as chamadas linguagens de programação de alto nível, como FORTRAN e COBOL.

Na década de 70 houve uma grande expansão do mercado computacional. Sistemas complexos começavam a surgir e por consequência, modelos mais robustos

foram propostos. Nesse período surge a programação estruturada e no final da década a análise e o projeto estruturado.

Nos anos 80 surge a necessidade por interfaces homem-máquina mais sofisticadas, o que originou a produção de sistemas de software mais complexos. A análise estruturada se consolidou na primeira metade dessa década e em 1989 Edward Yourdon lança o livro Análise Estruturada Moderna, tornando-o uma referência no assunto.

No período da década de 1990 surge um novo paradigma de modelagem, a Análise Orientada a Objetos, como resposta a dificuldades encontradas na aplicação da Análise Estruturada a certos domínios de aplicação.

Do final da década de 90 até o momento atual o paradigma da orientação a objetos atinge a sua maturidade. Os conceitos de padrões de projetos (design patterns), frameworks de desenvolvimento, componentes e padrões de qualidade começam a ganhar espaço. Nesse período surge a Linguagem de Modelagem Unificada (UML), que é a ferramenta de modelagem utilizada no desenvolvimento atual de sistemas.

Com as situações criadas pelo desenvolvimento das tecnologias entramos na CRISE DE SOFTWARE. Refere-se a um conjunto de problemas encontrados no desenvolvimento de software e na etapa de Manutenção. Entre elas destacamos:

1- As estimativas de prazo e de custo frequentemente são imprecisas.
2- Insatisfação do cliente com o sistema concluído.
3- A qualidade de software às vezes é menos que adequada.
4- O software existente é muito
5- Difícil de manter (Sem Manutibilidade).

Todos esses itens tiveram causas atreladas aos seguintes itens:

1- Caraterísticas próprias do software.
2- Falhas das pessoas responsáveis pelo desenvolvimento de software.

Com isso chegamos na aplicação de uma abordagem sistemática, disciplinada e possível de ser medida para o desenvolvimento, operação e manutenção do software que abrange um conjunto de três elementos fundamentais: Métodos, Ferramentas e Procedimentos para projetar, construir e manter grandes sistemas de software de forma profissional. As etapas que constituem cada elemento compõem o que

chamamos de Ciclo de Vida de Software, onde temos alguns ciclos mais conhecidos, Ciclo de Vida Clássico, Prototipação, Modelo Espiral.

Ciclo de Vida Clássico

Modelo mais antigo e o mais amplamente usado da engenharia de software. Requer uma abordagem sistemática, sequencial ao desenvolvimento de software.

Prototipação

Processo que possibilita que o desenvolvedor crie um modelo do software que deve ser construído. Idealmente, o modelo (protótipo) serve como um mecanismo para identificar os requisitos de software sendo apropriado para quando o cliente definiu um conjunto de objetivos gerais para o software, mas não identificou requisitos de entrada, processamento e saída com detalhes.

Ciclo de Vida em Espiral

Engloba as melhores características do ciclo de vida Clássico e da Prototipação, adicionando um novo elemento: a Análise de Risco. Segue a abordagem de passos sistemáticos do Ciclo de Vida Clássico incorporando-os numa estrutura iterativa que reflete mais realisticamente o mundo real e pode usar a Prototipação, em qualquer etapa da evolução do produto, como mecanismo de redução de riscos.

Síntese

Por Sommerville, software compreende tudo que é necessário para um sistema computacional funcionar: Programa de computador, documentação, arquivos de configuração, entre outros e existe por causa das necessidades de clientes. Como transformar necessidades em software? Devem ser consideradas as atividades de como entender as necessidades do cliente, planejar a solução, implementar a solução, validar esta solução, entregar o produto ao cliente. Estas atividades são executadas ordenadas ou não, formalmente ou informalmente.

Todo processo de transformação tem início e fim. Essa variável temporal, denominada de ciclo de vida, determina as fases do desenvolvimento de software.