**Teste e Análise de Software/Pézze e Young**

**Capítulo 1**

# Roteiro de Leitura

1.  Todo desenvolvimento de produto exige atividades de verificação para aferir a qualidade? Quase todo produto exige atividades de verificação, porém, normalmente possuem atividades diferentes ou que são executadas em contextos diferentes, sendo assim adaptadas ao âmbito do problema que o produto tenta resolver.

2.  As atividades de verificação são sempre as mesmas para produtos desenvolvidos em série? Por quê? Nem sempre, para produtos desenvolvidos em série cujo produção é idêntica a análise de um único produto de um grande lote pode ser satisfatório, quando o intutio não é medir a possibilidade de defeito. Em contrapartida, quando produtos são feitos em série mas possuem características voltadas para aplicações diferentes, ou seja, possuem diferenças entre si, um teste específica para cada uma destas versões é necessário para garantir que a proposta inicial do produto não é violada pelos recursos adicionados.

3.  Qual a diferença entre a verificação de produtos desenvolvidos em série para produtos únicos? Como comentado antteriormente, a verificação para produtos únicos requer que cada produto seja avaliado individualmente, ou até mesmo, de formas diferentes, quando alguns recursos são adicionados para resolver problemas além da motivação base de um produto mais generalista, que apresenta as mesmas funcionalidades base.

4.  Como se caracterizam os produtos de software neste contexto de teste e análise? Os produtos de software são os produtos mais complexos que são gerados diariamente em nossa sociedade. A variadade dse problemas que estes produtos abordam faz com que não haja uma receita de bolo padrão para sua verificação e validação.

5.  Qual o percentual do custo da verificação de software frente ao custo total do produto? Em geral o custo de teste e validação de software chega a atingir metade dos custos de um projeto inteiro. Isso se dá pois o esforço computacional de analisar o comportamento de toda a aplicação em conjunção com todas as ferramentas acopladas em seu desenvolvimento necessita traçar padrões de performance e geração de erros que level ao motivo do problema ou de baixa perfomance em determinadas situações, até chegar na refatoração do código neceśsaria.

6.  Técnicas avançadas de desenvolvimento impactam na verificação? O uso de tecnológias avançadas e ferramentas poderosas que suportam o processo de verificação não só eliminam alguns erros conhecidos apenas a partir do seu uso, como também mostram atalhos e enfatizam os comportamentos essenciais do programa ou sistema a partir de um erro ou fluxo não esperado, facilitando o processo de testagem e pavimentando o software já funcional para torná-lo mais produtivo e fluido.

7.  Quando se começa e termina o processo de verificação e validação? Em geral, a partir do termino de um protótipo funcional, o processo de verificação e validação de software nunca para. Isso se dá pois mesmo depois de solucionado o problema e feita a entrega final do produto, sempre surgirão maneira novas e mais otimizadas para resolver aquele mesmo problema, que deverão ser acopladas à gama de possibilidades daquele produto e serem testadas e validadas novamente. De outra forma, a partir da interação do software com o usuário, novos problemas surgirão dos quais verificação não foi priorizada ou simplesmente não apareceram no radar do time ou dos testadores. Além disso, é possível que em pouco tempo depois de uma release oficial do produto, novos problemas próximos ao original possam ser resolvidos de maneira a agregar valor ao produto, o que gera um novo ciclo de desenvolvimento e testes, que por sua vez ainda passará por evoluções durante seu uso público com seus usuários.

8.  Todo o produto de software passa, obrigatoriamente, pelas mesmas etapas de teste e análise? Para alguns produtos, diversas etapas de verificação podem ser consideradas obsoletas ou desnecessárias dado ao contexto do problema a ser resolvido, ou até mesmo falta de recursos para validação e verificação extensiva dos seus componentes e suas aplicações dentro da proposta do produto.

9.  Quais técnicas devem ser aplicadas? Várias técnicas deve ser aplicadas, enfatizando algumas dentro do contexto do produto. Depois da fase de prototipação este metaproduto deve chegar o mais perto possível de aceitação de teste, aleḿ de validar os requisitos e deicisões de design, assim como checar a falta de rombos de segurança. Posteriormente o produto deve entrar em uma espécie de versão Alpha, onde a interação com seu público de usuários será monitorada para adquirir conhecimento sobre os problemas que seu uso diário poderá provocar

10.  Como se pode verificar se um produto está pronto? Um padrão de aceitação deve ser definido baseado no numero de ocorrencias de erro num teste geral do software. Por exemplo, na fase de prototipação se menos de 90% das declarções são executadas por testes funcionais isso siginifica que as especificações estão incompletas ou que alguma complexidade esta abstraída da interface. Dessa forma, mais testes são elaborados e quando há um base de teste grande o suficiente, uma meta é definida, por exemplo, um erro a cada mil testes. Ao chegar desta meta, o produto pode passar por uma fase de release.

11.  O que vem a ser teste estatístico? Para que serve? Qual sua relação com perfis operacionais? O teste estatístico é um teste que avalia toda a plataforma em questão baseado em um histórico específico ou dados que a empresa possui de outros softwares, ou de algum teste de caso específico. Eles servem para testar todos os componentes de um software e suas interações as quais foram mapeadas, gerando um perfil operacional a partir destes dados anteriores aplicado ào software em questão.

12.  O que são testes alfa e beta? Qual sua relação com testes estatísticos? O testes alfa e beta utilizados para estabelecer um interação entre o software em desenvolvimento e o usuário com intuito de mapear e desolver novos problemas antes ignorados ou que simplesmente não ocorriam em um ambiente isolado.

13.  Como se garante o sucesso de versões sucessivas? Os desenvolvedores e testadores de um software devem manter uma base de dados com todos os erros ou bugs conhecidos em cada versão ou release do produto. Dessa forma garantindo que caso um erro persista ou volte a acontecer no futuro, saibamos de onde ele partiu e quais aspectos devem ser focados pelo time de desenvolvimento para que este se resolva.

14.  Como o processo de desenvolvimento pode ser melhorado? Através da análise de acontecimentos passados que se repetem é possível mapear situações que geram erros ou comportamentos inesperados que tendem se manterem em determinado contexto ou forma de implementação, podendo assim , sucessivamente moldar a forma de pensar e codificar para eliminar tais dificuldades sem ter que filtrá-las em processo de verificação e validação de sofware intensivo, ou seja, aprendendo com os erros ou imperfeições e os evitando em implementações futuras.

15. Quais as vantagens do uso de uma arquitetura em camadas frente as etapas de teste? Uma arquitetura em camadas evita depencias muito fortes entre os componentes, possibilitando a retirada de uma camada que se tornou obsoleta sem a refatoração intensiva de outra.