1. Ambiente de teste

O que é ambiente de teste?

Conceito: o Ambiente de testes é o "local" que o sistema será testado. Ele deve incorporar tudo que envolve o software e seu funcionamento. Segundo Bastos 2007 a definição do ambiente envolve: ambiente físico, pessoas, hardware, documentação, software, suprimentos, rede e interface.

Preparação: o Ambiente de testes deve ser planejado na definição da estratégia de testes ou no plano de testes. Quanto maior o nível dos testes a serem aplicados, mais o ambiente de testes deverá ser capaz de reproduzir as características do ambiente de produção.

Criação: a Criação, pela equipe de testes, de um ambiente isolado, organizado, representativo e mensurável garante a descoberta de erros reais, ou seja, aqueles que realmente ocorreriam na produção e que não foram descobertos em tempo de desenvolvimento e o mais importante, oferece a garantia de que não houve influência externa.  
Outra vantagem é que a preparação desse ambiente isolado, libera a equipe de desenvolvimento para continuar produzindo novos códigos, sem prejuízo à integridade do ambiente, mesmo durante a execução dos testes e possibilita a realização dos de iteração e sistema, de modo a permitir a integração das diferentes camadas e ou ambientes.

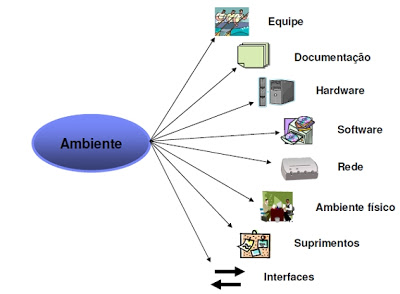
Principais características: ambiente isolado, com processamento independente e com componentes similares ao ambiente de desenvolvimento e produção; Ambiente restrito à equipe de testes; Criação de massa de teste com dados conhecidos e representativos quantitativamente qualitativamente, de modo a atender aos cenários de teste a serem executados e garantir a abertura do código.

Elementos do ambiente de testes

O ambiente de testes deve ser definido pelo nível de testes a ser executado, ou seja, quanto maior o nível, mas, o ambiente de teste deverá ser capaz de reproduzir as características do ambiente de produção.

O ambiente de testes deve ser isolado, com processamento independente e características similares ao ambiente de desenvolvimento e produção e deve ser restrito à equipe de testes para garantir a integridade dos testes realizados. [4] Logo adiante no item 5 é feita uma abordagem sobre o funcionamento e o ambiente de uma fábrica de testes.

Apresentamos a seguir os elementos do ambiente de testes que devem ser levados em consideração para o planejamento do ambiente de testes.



Maneiras de testar o software

Existem diferentes maneiras de se testar uma aplicação, com cada uma destas levando em conta aspectos como quais profissionais estarão executando o processo de validação ou ainda, a extensão do que estará sendo verificado. Tomando por base estes critérios, os diferentes tipos de testes de software podem ser classificados em:

* **Teste de unidade** (também conhecido como teste unitário): é uma modalidade que se concentra na verificação das menores unidades em um projeto de software. O teste é realizado em uma unidade lógica, utilizando dados suficientes apenas para verificar a lógica da estrutura em questão. Unidades em uma linguagem de programação orientada a objetos podem ser identificadas como um método, uma classe ou ainda um objeto;
* **Teste de integração**: procura apontar erros verificando os relacionamentos (interfaces) entre as diferentes partes (módulos) que compõem uma aplicação. Testes de integração costumam auxiliar na construção da estrutura de um programa, considerando para isto os requisitos definidos dentro do projeto correspondente;
* **Teste de sistema**: conduzido de maneira a simular a operação de uma aplicação por usuários finais, baseando-se para isto num ambiente similar ao do sistema já em produção e na manipulação de dados e informações próximas àquilo que será processado no dia-a-dia;
* **Teste de aceitação**: normalmente um grupo de usuários finais do sistema participa desse tipo de teste, visando com isto simular operações cotidianas que determinem se a aplicação está em conformidade com o que se espera para a mesma;
* **Teste de regressão**: realizado quando do lançamento de novas versões de um software, de forma a checar se as mudanças introduzidas na aplicação não produziram efeitos colaterais na execução de funcionalidades pré-existentes.

Preparação do ambiente

Alguns atributos do ambiente de testes precisam ser analisados e planejados para a realização dos testes. São estes:

* Escopo
* Equipe
* Volume de dados
* Origem dos dados
* Interfaces
* Ambiente

Escopo  
O escopo do ambiente sempre dependerá do nível de testes que será executado.

* Testes unitários
  + Unidade Individual
* Testes de integração
  + Agrupamento de unidades
* Testes de sistema
  + Toda a aplicação
* Aceitação
  + Toda a aplicação

Equipe

* Testes unitários
  + Desenvolvedores
* Testes de integração
  + Desenvolvedores / Analistas de sistema
* Testes de sistema
  + Analistas de sistema / Testadores
* Aceitação
  + Analistas de sistema / Testadores / Usuários

Volume dos dados

* Testes unitários
  + Volume pequeno
* Testes de Integração
  + Volume pequeno
* Testes de sistema
  + Volume grande
* Aceitação
  + Volume grande

Origem dos dados

* Testes unitários
  + Criação manual
* Testes de integração
  + Criação manual
* Testes de sistema
  + Criação automática e/ou dados reais
* Aceitação
  + Dados reais

Interfaces

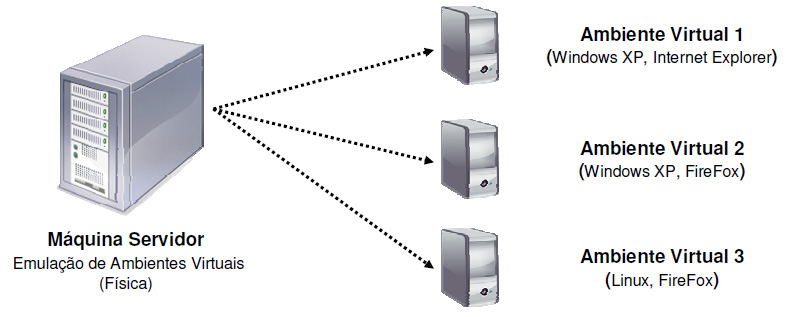
* Testes unitários
  + Não existem
* Testes de integração
  + Não existem
* Testes de sistema
  + Simuladas / Reais
* Aceitação
  + Reais

Ambientes virtuais

Infelizmente, na realidade atual, a maioria das empresas não prevê orçamento para preparação dos ambientes de testes na contratação de novos projetos de software.

Uma solução que vem ganhando espaço, por ser mais econômica, é a criação de ambientes virtuais - também são conhecidos como ‘máquinas virtuais’.

Máquina virtual é um software que permite ao arquiteto de testes criar vários ambientes de testes, com diferentes configurações de software, hardware, sistemas operacionais, etc., utilizando na realidade a mesma máquina física.

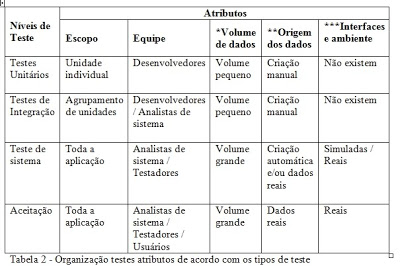


Preparação do ambiente de teste

O ambiente de testes deve ser planejado na definição da estratégia de testes ou no plano de testes. Quanto maior o nível dos testes a serem aplicados, mais o ambiente de testes deverá ser capaz de reproduzir as características do ambiente de produção.

Nessa fase do projeto, deve-se estimar o esforço gasto para montar o ambiente, ordenar e priorizar os casos de teste e preparar a massa de dados para testes. Dependendo do projeto, essa fase pode ser feita durante a execução, o que não é aconselhável, claro, pois devido a isso um grande atraso para o projeto passaria despercebido. Em grande parte dos projetos que participei e acompanhei, a massa de dados é criada somente durante a execução dos testes (com exceção dos projetos de performance que participei na EDS, que são outros quinhentos. Até projetos de automação, já vi criando esses dados on-the-fly. Portanto, já dá pra ver que o sucesso esperado não foi obtido...  
Todavia, boa parte desse processo de preparação do ambiente de testes eu adaptei do processo da RSI, empresa de consultoria e treinamentos especializada em testes, muito séria e profissional.

1) Objetivo  
Oferecer um ambiente com infraestrutura similar ao de produção, com bases de dados reduzidas, descaracterizadas e íntegras, utilizando processos automáticos de carga e validação de informações. O resultado é a realização de testes integrados e com total visualização dos processos de negócio da empresa.  
  
2) Artefatos de entrada  
Plano de testes  
Sistema em desenvolvimento  
Acesso ao Banco de Dados  
Quaisquer outras ferramentas periféricas ao sistema que necessitem estar configuradas/preparadas para a execução dos casos de teste  
  
3) Artefatos de saída  
Sistema pronto para ser testado  
  
4) Tarefas a serem realizadas   
a) Preparar a infraestrutura do ambiente de homologação e manter o controle sobre a utilização de seus recursos.   
b) Estabelecer infraestrutura de hardware, software e testware.   
c) Definir o tipo de ambiente (sistemas ativos e sistemas por demanda).  
d) Definir as políticas, normas e padrões do ambiente e dos processos.   
e) Estabelecer as regras e critérios para criação da Massa de Dados de Teste.  
e1. Se os dados vão ser populados manualmente  
e2. Se os dados vão ser populados automaticamente (através de uma ferramenta de automação dos dados do ambiente) e3. Se os dados vão ser populados diretamente no Banco de Dados (verificar se estes precisam estar associados à Regras de Negócio interdependentes ou se podem ser simplesmente adicionados sem afetar a estrutura do ambiente).  
  
5) Atores envolvidos na atividade e suas responsabilidades  
Administrador do Banco de Dados – executa  
Analista de Testes – executa  
Desenvolvedor – acompanha  
Gerente – acompanha  
6) Padrões   
N/A  
  
7) Procedimentos e Orientações  
Deve, idealmente, haver um ambiente exclusivo e dedicado para testes.  
  
Definição de requisitos de infraestrutura:   
Hardware: CPU e armazenamento  
Software: Básico e produtos   
Testware: Recursos humanos com especialização e foco em testes, e ferramentas de gerenciamento, documentação, automação, produtividade e geração de massa de testes.   
Segurança: Segregação física ou lógica   
Metodologia de teste: Processos estruturados   
Tipo de ambiente:  
Ativo: permanentemente disponível; sincronismo com a produção; processamento diário; controle rígido dos acessos, testes integrados.   
Por demanda: gerado pela necessidade; sem sincronismo; execução sob demanda; flexibilidade de acessos; testes integrados.   
Normatização:   
Definição de normas de controle de configuração e versão, gestão de mudanças e problemas.   
Definição de padrões de Nomenclaturas e Codificação (identificador de ambientes, estruturas de processamento, automação de processamento e scheduler).   
Massa de dados de teste:  
Definição da origem dos dados: laboratório, produção ou simulado   
Definição das ferramentas de produtividade  
  
Alguns atributos do ambiente de testes precisam ser analisados e planejados para a realização dos testes. Veja na tabela abaixo a organização destes atributos de acordo com os tipos de teste:



1.1 Automação de testes

Técnica x Ferramenta

A princípio será abordado dois conceitos importantes na automação de testes:

Técnica: é o procedimento ou o conjunto de procedimentos que têm como objetivo obter um determinado resultado.

Ferramenta: É um instrumento para aplicar a técnica.

Mas, afinal, o que é automação de testes?

É a utilização de ferramentas que executam os testes na aplicação sem intervenção humana, por meio da implementação de scripts. Dessa forma, a ferramenta simula uma utilização do software testado e verifica os resultados esperados.

A automação de testes traz diversas vantagens: é mais rápido, diminui a chance de erro humano (mas é preciso atentar para que os scripts não apresentem falhas) e reduz o esforço com tarefas repetitivas, liberando os testadores para atividades que exijam raciocínio humano.

É importante lembrar, também, que para optar pela automação dos testes é necessário ter mão-de-obra especializada para a ferramenta utilizada.

Enquanto pessoas são mais inteligentes, intuitivas e criativas que máquinas, computadores já se dão melhor com cálculos, tarefas repetitivas e que precisam ser feitas num curto espaço de tempo.

Quando utilizar ferramentas de automação de testes?

Cada vez mais, ferramentas de automação de teste estão sendo lançadas no mercado para automatizar as atividades de teste. Mas antes de pensar em uma ferramenta é preciso definir os processos. Veja alguns fatores que leva a automação de testes:

* Existem fortes pressões para melhorar a qualidade;
* O projeto tem situações que não possam ser testadas adequadamente pelos métodos tradicionais,
* Os perfis dos softwares desenvolvidos forem complexos e com impacto no negócio,
* Estudos de custo X benefício justificar o investimento,
* O tamanho do projeto ou do ambiente de teste justificar.

A maioria das empresas sempre avaliam a possibilidade do uso de ferramentas abertas e livres.

Ferramentas de teste

Existem basicamente três categorias de ferramentas de automação de testes, a saber:

* Ferramentas de gerenciamento
* Ferramentas de verificação de código-fonte
* Ferramentas de automatização na execução dos testes

Ferramentas de gerenciamento

Ferramentas de gerenciamento são as ferramentas gerenciais, normalmente utilizadas para fazer a gestão de testes e defeitos. Por exemplo, uma ferramenta que permite que sejam cadastrados os defeitos encontrados no software durantes os testes.

São ferramentas muito importantes, pois auxiliam o processo de teste como um todo.

Estão divididas em:

* Ferramentas de gerenciamento de defeitos
  + Exemplos de ferramentas: Jira, Mantis, BugZilla
* Ferramentas de controle de versionamento
  + Exemplos de ferramentas: SubVersion, SourceSafe
* Ferramenta de gerenciamento dos testes
  + Exemplos de ferramentas: RTH, TestLink, Mantis

Ferramentas de verificação de código  
São ferramentas utilizadas para:

* Verificar se o trabalho foi produzido dentro dos padrões de codificação
* Identificar pedaços de códigos não executados
* Identificar erros mais comuns, como problemas com inicialização e variáveis, estouro de memória, etc.

As ferramentas de verificação de código não têm a obrigação de verificar se o código realiza o que deveria realizar, ou seja, não têm a responsabilidade de testar as funcionalidades. Seu objetivo não é este. Esse tipo de ferramenta é muito utilizada nos testes unitários.

Ferramentas de execução de testes

* Unitários
  + JUnit
* Integração
* Sistema
  + TestComplet
  + JUnitPerf
* Aceitação
  + JMeter
  + JUnitPerf

Análise de riscos

O que é risco?  
É um evento ou condição incerta que, se ocorrer, provocará um efeito positivo ou negativo nos objetivos do projeto.

A Análise de Riscos e o Teste de Software estão diretamente ligados, afinal, todos os projetos de software possuem riscos. Peculiares ou não, estes riscos podem envolver desde a tecnologia utilizada até a complexidade do projeto. No entanto, independente da sua particularidade, um risco nada mais é do que um evento que pode ou não ocorrer, e que consequentemente pode causar alguma perda.

Como na maioria das vezes não é possível (nem recomendável) testar exaustivamente um software, a análise de risco voltada para o teste pode revelar uma vasta probabilidade de problemas, e evitá-los ao priorizar os testes.

Existe um consenso de que a atividade de teste é bastante cara, podendo custar até 2/4 do valor estimado para o projeto. Porém, fatores como competitividade, complexidade dos sistemas, e claro, a satisfação dos clientes, vêm sendo levados em consideração e fazem jus ao investimento. Isso sem falar da famosa Regra 10 de Myers, que indica que o custo da correção dos defeitos tende a ser cada vez maior quanto mais tarde ele for descoberto.

A análise de risco no processo de teste de software tem o objetivo de avaliar os fatores cujas ocorrências podem vir a acarretar falhas no sistema e perda para a empresa. É uma avaliação dos recursos de informação de uma organização, dos seus controles e das suas vulnerabilidades. A análise de riscos combina a probabilidade de ocorrência de um problema com a gravidade e o grau de prejuízos que os danos poderão acarretar.

Um risco apenas vira uma ameaça quando existe uma vulnerabilidade. Dentro de um o processo de teste de software, uma análise de risco envolve a identificação das ameaças mais prováveis em conjunto com o exame das vulnerabilidades relacionadas ao software. Os motivos que podem levar ao risco são muitos, como por exemplo, um processo de teste mal definido ou até mesmo não conduzido de forma planejada.

É importante observar que a análise de risco voltada para o teste de software segue basicamente as mesmas regras e metodologias utilizadas em projetos de software em geral, acrescida de algumas características próprias, como demonstrado ao longo do artigo.

Conceituando riscos - Exemplos

Exemplo de Risco:

* Devido ao uso de uma nova tecnologia para a qual os programadores foram recém treinados, os mesmos poderão precisar de mais tempo para desenvolvimento dos programas, ocasionando um atraso nas atividades do cronograma.

Exemplo de Não-Risco:

* Desconhecimento dos programadores quanto à tecnologia utilizada para a programação deste projeto.

Categoria de riscos

Quando analisamos riscos em projetos de teste de software, geralmente os categorizamos em:

* Riscos de projeto: são riscos ligados diretamente ao projeto.
* Riscos técnicos: são riscos relacionados à qualidade do software a ser desenvolvido.
* Riscos para o negócio do cliente: são riscos relacionados a defeitos no software, que podem causar prejuízos ao negócio do cliente.

Gerenciamento de riscos

É um conjunto de técnicas e ferramentas para identificar, estimar, avaliar, monitorar e administrar os acontecimentos que colocam em risco e execução do projeto.

Para Gerentes de Projeto, o Gerenciamento de Riscos é uma disciplina que busca, pró-ativamente, minimizar os efeitos de eventos futuros que possam causar perdas para o projeto, ou maximizar os efeitos de eventos futuros que possam gerar ganhos para o projeto.

É o processo de identificação, avaliação, priorização, desenvolvimento de estratégias de tratamento e acompanhamento dos riscos.

Identificação dos riscos:

A identificação dos riscos pode ser feita através de um “Checklist” que focaliza subconjuntos de riscos conhecidos:

1) Tamanho do produto  
2) Impacto no negócio  
3) Características do cliente  
4) Definição do processo  
5) Tecnologia a ser utilizada  
6) Tamanho experiência da equipe

 Além disso, é importante fazer analogia com casos conhecidos (dados históricos; relatórios de experiências anteriores; experiências pessoais) e fazer a identificação baseada no julgamento de pessoas e em suas experiências anteriores.

**1) Risco Tamanho do Produto**  
São riscos associados ao tamanho do produto a ser desenvolvido ou modificado. O risco do projeto é diretamente proporcional ao tamanho do produto.

Checklist : Formule as seguintes perguntas e compare as respostas com experiências anteriores. Se o percentual de desvio for elevado ou se os números obtidos são similares a experiências anteriores cujos resultados não foram considerados satisfatórios, o risco é elevado.

* Tamanho estimado em LOC ou PF?
* Grau de confiabilidade da estimativa?
* Tamanho do produto em número de transações, programas ou arquivos?
* Percentual de desvio do tamanho estimado observado em projetos anteriores?
* Número de usuários?
* Número de mudanças nos requisitos do produto, antes e depois da entrega?
* Quantidade de reuso?

**2) Risco Impacto no Negócio**  
Conflito entre as considerações de negócios do departamento de marketing e a realidade técnica.

 Checklist: formule as seguintes perguntas e compare as respostas com experiências anteriores. Se o percentual de desvio for elevado ou se os números obtidos são similares a experiências anteriores cujos resultados não foram considerados satisfatórios, o risco é elevado

* Efeito do produto sobre o lucro da empresa?
* Visibilidade do produto para a gerência sênior?
* Número de clientes que irão utilizar o produto e consistência entre o produto e suas necessidades?
* Número de produtos com o qual irá interagir?
* Sofisticação dos usuários finais?
* Quantidade e qualidade da documentação que deverá ser entregue ao cliente?
* Restrições governamentais quanto à construção do produto?
* Custos associados a entrega em atraso?
* Custos associados a entrega com defeitos?

**3) Risco Características do Cliente**  
Um mau cliente provoca fortes impactos negativos na capacidade da equipe entregar o produto no prazo e dentro do orçamento. Clientes geralmente possuem diferentes necessidades (uns sabem bem o que querem; outros não. Uns são detalhistas e outros se contentam com promessas vagas), personalidades (uns gostam de ser clientes – a tensão, as negociações e as recompensas de um bom produto; outros aceitam qualquer produto e acham excelente um produto pobre. Uns irão reclamar quando a qualidade for baixa e elogiar quando for alta. Outros não irão se importar), relações com seus fornecedores (alguns conhecem bem o produto e o fornecedor. Outros são reservados e formais, comunicando-se apenas por correspondência escrita e curtas e escassas ligações telefônicas)

Além disso, clientes frequentemente são contraditórios, representando ameaças ao plano do projeto e riscos substanciais à gerência do projeto.

Checklist: Se a maioria das respostas a essas perguntas for “não”, investigações futuras devem ser realizadas, para avaliar o risco em potencial

* Você já trabalhou com o cliente no passado?
* O cliente possui uma sólida ideia sobre suas necessidades? Ele reserva algum tempo para descrevê-las?
* Os clientes concordam em reservar tempo para participar de sessões de elicitação de requisitos?
* Os clientes estão dispostos a estabelecer vínculos de comunicação com a equipe de desenvolvimento?
* Os clientes concordam em participar de revisões?
* Os clientes são tecnicamente sofisticados na área do produto?
* Os clientes entendem os processos de software?
* Os clientes resistirão a participar de trabalhos técnicos detalhados?

**4) Risco do Processo**Se os processos de software não estão definidos, ou estão definidos, mas não são utilizados, o risco do projeto é elevado.

Checklist: Se a maioria das respostas a essas perguntas for “não”, o risco é elevado

* A gerência sênior apoia a definição de políticas formais que enfatizem a importância de padrões de processos de software?
* A organização desenvolveu um processo de software por escrito para ser utilizado no projeto?
* A equipe concorda com o processo como está documentado e está disposta a utilizá-lo?
* Processos de software foram utilizados em projetos anteriores?
* A organização promove treinamentos técnicos e gerenciais em engenharia de software?
* A equipe técnica e gerencial dispõe de padrões de engenharia de software publicados?
* Os produtos intermediários associados ao processo estão definidos e existem exemplos disponíveis?
* Revisões técnicas formais são realizadas regularmente em todas as fases do processo?
* Casos de testes e procedimentos de testes formais são realizados regularmente?
* Os resultados das revisões técnicas são documentados, incluindo erros e recursos utilizados?
* Existe algum mecanismo estabelecido para garantir que os padrões de engenharia de software estão sendo observados?
* A gerência de configuração está sendo utilizada para manter consistência entre os requisitos, projeto, código e casos de testes?
* Existem mecanismos para controlar mudanças nos requisitos dos usuários?
* O trabalho subcontratado está formalmente documentado, os requisitos estão especificados e existe um plano para cada subcontratado?
* Existem procedimentos definidos para acompanhar e rever os trabalhos subcontratados?
* Estão sendo utilizadas técnicas para facilitar a comunicação entre a equipe e usuários?
* Estão sendo utilizados métodos específicos de análise?
* Estão sendo utilizados métodos específicos de projeto de dados e de arquitetura?
* Está sendo utilizada linguagem de alto nível em mais de 90% do código?
* Existem convenções para código documentadas e são utilizadas?
* Métodos específicos de teste são utilizados?
* Estão sendo utilizadas ferramentas para apoiar o planejamento e acompanhamento do projeto?
* Estão sendo utilizadas ferramentas de gerência de configuração?
* Estão sendo utilizadas ferramentas CASE?
* Estão sendo utilizadas ferramentas para auxiliar na criação de protótipos?
* Estão sendo utilizadas ferramentas de teste?
* Estão sendo utilizadas ferramentas para auxiliar na produção da documentação?
* Métricas de qualidade estão sendo coletadas e analisadas?
* Métricas de produtividade estão sendo coletadas e analisadas?

##### **5) Risco da Tecnologia** Atuar no limite da tecnologia é excitante e desafiador, mas oferece muitos riscos. Além de ser muito difícil prever esses riscos e mais ainda elaborar um planejamento para mitigá-los.

Checklist: Se a maioria das respostas a essas perguntas for “sim”, investigações futuras devem ser realizadas, para avaliar o risco em potencial

* A tecnologia é nova para a organização?
* Os requisitos exigem o desenvolvimento de novos algoritmos ou novas tecnologias de entrada e saída?
* Existem interfaces com hardware novo ou não testado?
* O software irá fazer interface com produtos de software de outros fornecedores, ainda não testados?
* O software irá fazer interface com sistemas de bancos de dados ainda testados na área do software?
* Os requisitos demandam interfaces com usuários especializadas?
* Os requisitos exigem a construção de componentes que nunca foram desenvolvidos pela organização?
* Os requisitos exigem novos métodos de análise, projeto e teste?
* Os requisitos demandam a utilização de novos métodos de desenvolvimento como os baseados em inteligência artificial, métodos formais ou redes neurais?
* Os requisitos estabelecem restrições excessivas de desempenho?

**6) Risco tamanho e experiência da equipe**Checklist: Se a maioria das respostas a essas perguntas for “não”, investigações futuras devem ser realizadas, para avaliar o risco em potencial.

* Boas pessoas estão disponíveis?
* As pessoas possuem o perfil adequado?
* A equipe pode comprometer-se durante toda a duração do projeto?
* Parte da equipe está atuando em “part time”?
* A equipe possui expectativas adequadas sobre o projeto?
* A equipe recebeu o treinamento necessário?
* A rotatividade de pessoal deverá ser baixa?

Analisar os riscos

Uma vez identificados os riscos, precisamos analisá-los a fim de encontrar o que os causam, e detalhar o seu impacto e a probabilidade da sua ocorrência.

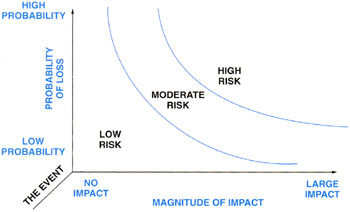
Cada risco deve ser analisado sob a perspectiva do seu impacto e probabilidade de ocorrência Impacto

Impacto

* Prejuízo que um risco pode causar ao projeto.
* Modelo de classificação: variação de 1 a 9.

Probabilidade

* Chance de um risco se concretizar.
* Modelo de classificação: variação de 1 a 9.



Priorizar os riscos

O objetivo desta atividade é comparar os riscos e identificar quais são os mais importantes e que consequentemente merecem maior atenção. A priorização é necessária porque normalmente os recursos do projeto são escassos, e devemos concentrar esforços primeiro nos riscos mais importantes.



Planejar resolução dos riscos

Uma vez priorizados e selecionados os riscos que serão atacados, para cada um deles deveremos preparar:

* Um plano de mitigação
* Um plano de contingência

Plano de mitigação

O plano de mitigação pode ser considerado uma atividade proativa, pois através dele tentamos minimizar o impacto e a probabilidade de um risco acontecer.

Características:

* Proativo
* Contempla atividades para minimizar impacto e probabilidade de ocorrência
* Deve ser executado antes que o risco ocorra
* Evita que o risco se transforme em um problema

Plano de contingência

Ao contrário do plano de mitigação, o plano de contingência é reativo, e deve ser acionado quando o risco virar um problema.

Características:

* Reatividade
* Sem chance de evitar que o risco ocorra

Bibliografia

**[4]** **Base de Conhecimento em Teste de Software.** 2ª Edição. Rios, Emerson; Cristalli, Ricardo; Moreira, Trayahú & Bastos, Aderson. – S. Paulo, Martins Fontes, 2007.  
  
Almeida, Carla. **Introdução ao Teste de Software**<<http://www.linhadecodigo.com.br/artigo/2775/introducao-ao-teste-de-software.aspx#ixzz3khJtHuvx>> acessado em: 11/09/2015

Gomes De Souza, Jesiane. **Ambiente de Testes de Software** <<http://www.webartigos.com/artigos/ambiente-de-testes-de-software/74582/#ixzz3khF0ruka>> acessado em: 11/09/2015

 Schroeder Viera, Luiz Gustavo. **Preparar Ambiente de Testes**<<http://www.testavo.com.br/2010/06/preparar-ambiente-de-testes.html>> acessado em: 11/09/2015

Eliza, Renata e Lagares, Vivian. [**Análise de Riscos e o Teste de Software - Revista Java Magazine 97**](Análise%20de%20Riscos%20e%20o%20Teste%20de%20Software%20-%20Revista%20Java%20Magazine%2097) <<http://www.devmedia.com.br/analise-de-riscos-e-o-teste-de-software-revista-java-magazine-97/22831#ixzz3lOX467Km>> acessado em: 11/09/2015

Groffe, Renato Jose. **Testes Unitários no Visual Studio 2012**   
<<http://www.devmedia.com.br/testes-unitarios-no-visual-studio-2012/27215#ixzz3khKnptAo>> acessado em: 11/09/2015