

Aula 9: Controle de bugs

Apresentação

Nesta aula, você verá como instalar o Mantis BugTracker, entenderá como deve ser feita a configuração e a utilização desse programa, identificar como cadastrar os defeitos e conhecer o gerenciamento do ciclo de vida de um defeito nele.

Objetivos

- Descrever como ocorre a instalação, configuração e utilização do Mantis BugTracker;
- Identificar como realizar o cadastro de defeitos e o gerenciamento do ciclo de vida de um defeito no Mantis.

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

O que é um BugTracker?

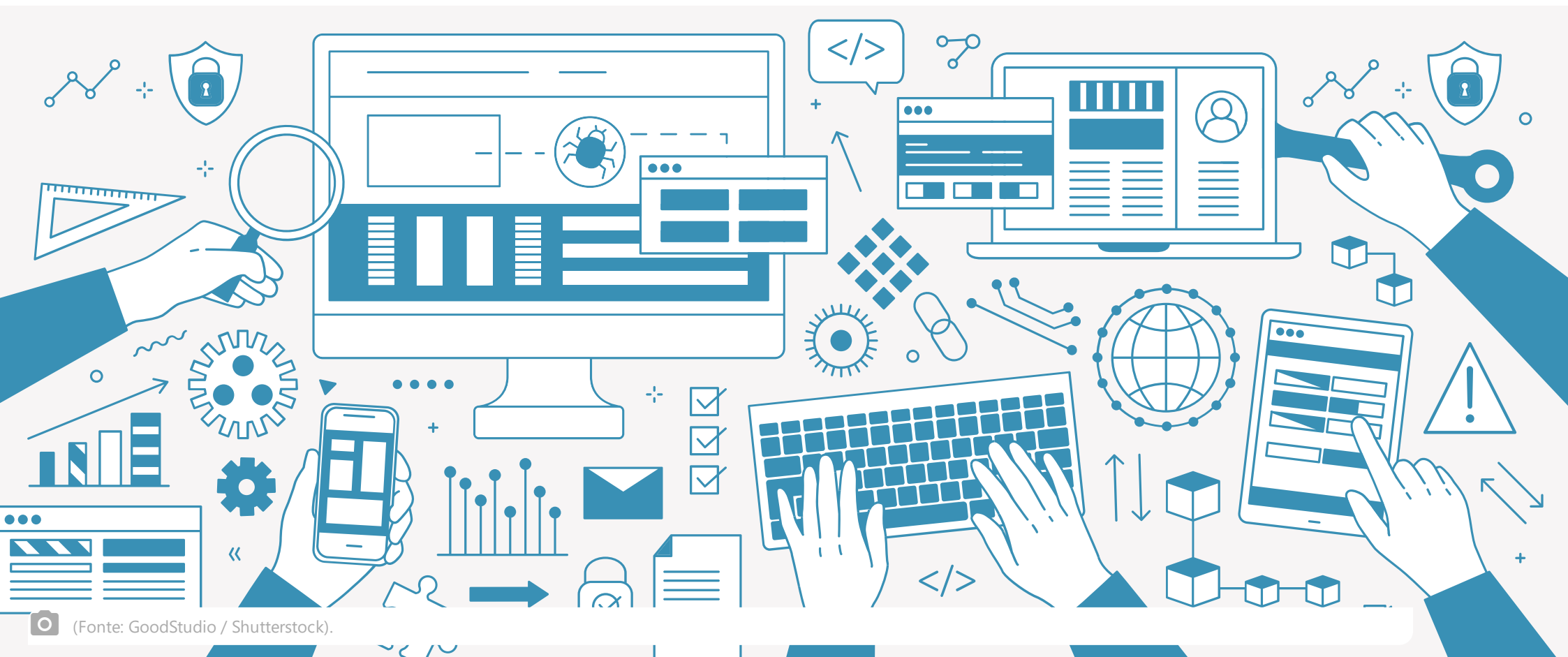
É um sistema de rastreamento de bugs, ou seja, de defeitos. É um aplicativo de software que controla os erros de software relatados em projetos de desenvolvimento de software. Pode ser considerado também um tipo de sistema de rastreamento de problemas.

O tester (isto é, o analista de teste) - é o responsável por encontrar erros, falhas, bugs e outros tipos de problemas que não foram detectados durante o desenvolvimento de um software.

Ao encontrar um bug, ele pode avisar o desenvolvedor, de várias formas. Exemplo: pessoalmente, msn, por e-mail, documentos, ferramentas de bug tracking, etc.

A melhor maneira de relatar a existência de bugs por meios formais e informais dependerá da dinâmica da equipe de testes e do seu processo dentre essas alternativas apresentas.

Há vários meios de relatar os bugs apresentados. Ao se deparar com algum resultado imprevisto, o usuário deve conversar com o desenvolvedor, para que juntos possam buscar a real causa daquele problema.



Na maioria das vezes, isso ocorre devido à complexidade do sistema e para obter um melhor detalhamento do bug.

O passo seguinte da coleta de todas as informações necessárias sobre o bug é cadastrar na ferramenta bug tracking. Podemos usar o Mantis ou o Eventum.

Saiba mais

Veja um [tutorial do Eventum](#).


Para quem faz isso pela primeira vez, um software de bug tracking tem como objetivo principal, ajudar as equipes de Teste e a de Desenvolvimento a manterem um histórico dos bugs do sistema.

Nesse histórico, encontram-se todos os dados do bug, ajudando todas as equipes, pois isso ele facilita o gerenciamento dos bugs.

Utilizando ferramentas para gerenciar defeitos

Existem no mercado muitas ferramentas para gerenciar defeitos. Vamos apresentar várias, mas aqui vamos estudar apenas o **Mantis BugTracker**.

Essas ferramentas permitem que informações sejam coletadas e acompanhadas, possibilitando uma visão mais realista para a tomada de decisão.

 Clique nos botões para ver as informações.

[Bugzilla](#)



É considerada a mais famosa e usada ferramenta de bug tracking open source, baseando-se em tecnologia web. Por ser open source, é mantida por voluntários, sendo utilizada por diversos projetos e empresas, dentre os principais: Mozilla, Gnome, Nasa, NBC e Wikipedia.

[Eventum](#)



Foi criada pela MySQL e disponibilizada, também gratuitamente, e usada pela equipe de suporte técnico do MySQL Lab. Atualmente se encontra na versão 2.1.1. Esta ferramenta é boa porque fornece uma interface amigável e sistema de emissão flexível de rastreamento, facilitando o uso.

Pode ser usado tanto por um departamento de suporte técnico, para monitorar solicitações de suporte, quanto por uma equipe do desenvolvimento de software, a fim de organizar tarefas e bugs.

[Jira](#)



Essa ferramenta é uma aplicação J2EE de acompanhamento e gestão dos problemas. Outra funcionalidade importante dessa ferramenta é de gestão de projetos. Pode ser baixado de graça, porém em versão trial, que expira após 30 dias.

[Mantis](#)



Uma ferramenta de bug tracking livre. Foi escrita em PHP e trabalha com MySQL, MS SQL e PostgreSQL, sendo baseada em web. Roda em Windows, Linux, Mac OS, OS/2, dentre outros, e está sobre os termos da licença GPL.

[Trac](#)



Essa ferramenta possui a funcionalidade de wiki para documentação. É um ótimo sistema de gerenciamento de bugs. Tem interface web e também está sob a licença **GPL**. (General Public License se refere à licença de software livre criada e idealizada por Richard Matthew Stallman para o projeto GNU (sistema operacional livre, quase completo, mas que não tinha um kernel [núcleo] estável), utilizada por muitos desenvolvedores. A GPL tem como objetivo garantir que todos tenham o direito de usar, copiar, modificar e distribuir o software, ou seja, o software (e seu uso) é livre.) Dentre os seus usuários, está o Laboratório de Propulsão a Jato da NASA.

Essa ferramenta é específica para equipes de testes. É mais uma aplicação com interface web, feita em Flash. Sua licença é gratuita para até três usuários.

Principais funcionalidades: execução de testes, criação de testes suítes, geração de relatórios e gerenciamento de bugs.

Instalação do Mantis BugTracker



Kenzaburo Ito e um amigo criaram originalmente um bug tracker como uma ferramenta interna para o seu projeto. Em 2002, Ken foi auxiliado por Jeroen Latour, Victor Bockor e Julian Fitzell para serem os administradores e a equipe central de desenvolvimento do MantisBT.

Essa ferramenta é simples e fácil de usar. Tem todos os recursos de que o desenvolvedor e o administrador necessitam.

Comentário

O Mantis não é apenas uma aplicação web. O programa acompanha não só os tempos de mudanças, como também tem sua própria versão para celular. A sua implementação é feita em PHP e é gratuito para uso.

Principalmente os grandes softwares requerem muito conhecimento dos desenvolvedores e a execução desses nos prazos necessários para alcançar a conclusão do projeto.

Repare que cada software pode ter **mais de uma versão**, e a utilização de ferramentas que beneficiem a identificação, organização e controle de versões é de extrema importância.

Independentemente dos testes efetuados, todo software implementado tem bugs, defeitos e imperfeições nos códigos, que podem causar algum dano ao produto desenvolvido, podendo ser identificados pela equipe de desenvolvimento ou até mesmo pelo próprio usuário.

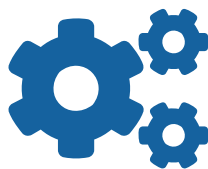
De acordo com Kalani, o Mantis Bug Tracking é uma ferramenta web open source, desenvolvida em PHP, personalizável e totalmente balanceada entre simplicidade e desempenho para controle de bugs.

Essa ferramenta suporta muitos sistemas de banco de dados. **Exemplos:** MySQL, MS SQL, PostgreSQL e DB2. (Kalani, 2005)



Vantagens do Mantis

- Pode ser executado em qualquer servidor que tenha suporte PHP, rodando Windows, Linux, OS/2, Mac OS X e uma variedade de sistemas operacionais executando Unix;
- Serve para auxiliar na construção e no relacionamento entre as fases de testes, proporcionar a gestão das atividades relacionadas a testes de software e controle de bugs e disponibilizar mecanismos de exportação de artefato e extração de métricas;
- É útil em todos os projetos Java, aumentando a produtividade do time de teste e dando maior visibilidade ao restante da equipe de projeto.



Características do Mantis BugTracking

- Tem interface simples, facilitando muito a experiência do usuário;
- Suporte para projetos, subprojetos e categorias;
- Usuários podem ter diferentes níveis de acesso por projeto;
- Busca e filtro para os resultados;
- Sistema de reporte com gráficos e logs;
- Notificação por e-mail;
- Usuários podem monitorar bugs específicos;
- Habilidade de enviar mensagens para os demais usuários de um bug específico;
- Projeto público ou privado public , isto é, público quando o projeto é acessível para todos os usuários e privado quando é acessível somente ao usuário específico adicionado;
- Access de grupo: as ações podem ser aplicadas a múltiplos bugs;
- Compatibilidade com o Eclipse;
- Possibilidade de instalação de plug-ins.

Infraestrutura

De acordo com o Guia de administração: referência para administradores, por MantisBT Development Team, são necessários tanto requisitos de sistema quanto de cliente.

Saiba mais

Acesse o [guia do MantisBT Development Team](#).

Requisitos

Requisitos de hardware do servidor

O MantisBT tem requisitos simples de hardware e requer um computador capaz de executar o software e do servidor.

- **Tipo de servidor**

O servidor pode ser um servidor da web público compartilhado ou uma caixa colocalizada dedicada.

- **CPU e Memória**

Quanto a qualquer aplicativo da web, você deve dimensionar seu servidor com base no tráfego no site.

- **Disco**

O código do aplicativo é menor que 30 MB.

Requisitos de software do servidor

Todo software necessário tem que ser gratuito para uso comercial e não comercial (código aberto).

- **Sistema operacional**

O MantisBT roda no Windows, MacOS, OS / 2, Linux, Solaris, os BSDs e praticamente qualquer coisa que suporte o software de servidor necessário.

- **Servidor web**

O MantisBT é testado principalmente com o [Microsoft IIS](#) e o [Apache](#). No entanto, espera-se que funcione com qualquer software de servidor da web recente. **Extensões de arquivos:** o MantisBT usa apenas arquivos .PHP . Se o seu servidor está configurado para outras extensões (exemplo: .PHP3, .PHTML), você terá que solicitar ao administrador para adicionar suporte para arquivos .PHP.

- **[PHP](#)**

O servidor da web deve suportar PHP. Pode ser instalado como CGI (Common Gateway Interface: Tecnologia que permite gerar páginas dinâmicas, permitindo a um navegador passar parâmetros para um programa alojado num servidor web.) ou qualquer outra tecnologia de integração.

- **Extensões PHP**

O MantisBT é projetado para funcionar em tantos ambientes quanto possível. Portanto, as extensões necessárias são mínimas e muitos deles são opcionais, afetando apenas um recurso.

Extensões obrigatórias

- A extensão para o RDBMS sendo usado (mysqli, pgsql, oci8, sql-srv);
- Mbstring - Requerido para suporte a Unicode (UTF-8).

Extensões opcionais

- Curl - necessário para o recurso de integração do Twitter;
- GD - necessário para o recurso captcha;
- Fileinfo - necessário para anexos de arquivos e a maioria dos plug-ins.
- Sem essa extensão, as visualizações e downloads de anexos de arquivos não funcionam com o MantisBT, que não será capaz de enviar o cabeçalho Content-Type para um navegador solicitando um anexo.

Base de dados

O MantisBT requer um banco de dados para armazenar seus dados. Os RDBMS suportados são:

- MySQL (ou um dos seus garfos, como por exemplo MariaDB);
- PostgreSQL.

O suporte experimental também está disponível para:

- Microsoft SQL Server;
- Oracle.

Suporte experimental significa que a intervenção manual por um administrador de banco de dados qualificado pode ser para concluir a instalação e/ou que possa haver problemas ou limitações conhecidas ao usar os programas.

Atenção: A equipe de desenvolvimento do MantisBT trabalha principalmente com o MySQL, então testar outros guias não é tão amplo como confiamos, principalmente nas contribuições da comunidade para melhorar o suporte e correção de problemas com outros RDBMS (*Relational Database Management System* - Designação dos sistemas de gerenciamento de banco de dados relacionais [DB2, Oracle, Sybase]). Portanto, é recomendado o MySQL para armazenar seu banco de dados.

Tabela de compatibilidade de versões

Categoria	Pacote	Mínima Versão	Recomendada	Comentários
RDBMS	MySQL	4.1.x	5.0.x ou superior	Extensão PHP: mysql/mysqli
PostgreSQL	7.0	8.0 ou superior	Extensão PHP: pgsql	
IBM DB2			Extensão PHP: ibm-db2	
MS SQL Server	2005	2005 ou superior	Extensão PHP: mssql or sqlsrv	
Oracle	8i	11gR2	Extensão PHP: oci8	
Web Server	Apache	1.3.x	2.2.x	–
lighttpd	1.4.x	1.4.x	–	
IIS	6.0	6.0	–	
PHP	PHP	5.1.x	5.2.x ou superior	–

Requisitos do Cliente

O MantisBT deve ser executado em todos os navegadores recentes no mercado, incluindo, mas não limitado a:

- Firefox 45 e acima;
- Internet Explorer 10 e acima;
- CromadaChrome;
- Safari;

- Opera.

Tarefas de pré-instalação/atualização

Essas tarefas abrangem o download e a implantação do MantisBT e devem ser executadas antes de qualquer nova instalação ou atualização.

1º. Fazer o download do MantisBT;

2º. Transferir o arquivo baixado para o seu servidor.

Isso pode ser feito usando o método que você mais gosta (ftp, scp etc). Você precisará de telnet/ ssh na máquina do servidor para as próximas etapas;

3º. Extrair o lançamento.

É altamente recomendável manter um diretório separado para cada lançamento. Isso não só evita erros de correspondência entre versões (arquivos podem ter sido adicionados ou removidos), mas também fornece um caminho. Faça o downgrade de sua instalação, caso seja necessário.

O comando usual é (um passo): **tar -xzf filename.tar.gz**

Ou (duas etapas):

- **Gunzip filename.tar.gz**
- **tar -xf filename.tar**

Saiba mais: Outras ferramentas de arquivamento de arquivos, como o [7-Zip](#), também devem ser capazes de lidar com a compressão do arquivo.

O processo de extração deve criar um novo diretório, como o mantisbt-1.3.x;

4º. Renomeie o diretório.

Para novas instalações, você pode querer renomear o diretório recém-criado para algo mais simples. Ex.: mantisbt
mv mantisbt-1.3.x mantisbt.

Atenção: Existe também uma documentação para o desenvolvedor, mostrando que o código-fonte e os plugins do MantisBT estão hospedados no GitHub.

Configuração e utilização

O Mantis é uma ferramenta de bug tracking, open source, muito utilizada para gerir o projeto como um todo, mas que também pode ser utilizada pelas equipes de qualidade e testes de software na gestão e controle dos bugs encontrados no sistema.

Uma facilidade desse programa é que conseguimos incluir todas as pessoas que trabalham na construção de um determinado software, de forma que todos terão a visibilidade do que os demais estão fazendo.

Para que isso ocorra, é necessário que cada membro envolvido na construção do software seja cadastrado e posteriormente associado a um projeto, e todas as pessoas que estejam trabalhando na mesma tarefa, dentro da ferramenta, sejam avisadas por e-mail sobre qualquer alteração.

Exemplo

A resolução de uma tarefa ou de um bug encontrado.

O Mantis possui tem, em sua tela principal, uma legenda com descrições e cores distintas, que ajuda o usuário a identificar o status dos bugs cadastrados. São eles:

<u>novo</u>	<u>retorno</u>	<u>atribuído</u>	<u>em execução</u>	<u>pendente</u>	<u>concluído</u>	<u>fechado</u>
-------------	----------------	------------------	--------------------	-----------------	------------------	----------------

New (novo) – nessa legenda, o bug assumirá este status assim que for reportado.

Feedback (retorno) – aqui será mostrado toda vez que a pessoa a quem o bug foi atribuído retornar algo a quem fez a reportagem.

Acknowledged (atribuído) – este sempre mostrará que um bug é reportado sem ter sido atribuído a alguém.


Confirmed (em execução) – É importante porque indica que a pessoa a quem o bug foi atribuído já o recebeu e acessou o mesmo.

Assigned (pendente) – Ssempre que um bugs já foiram atribuídos a alguém , ele é exibido aqui.

Resolved (concluído) – temos que ilnformar se um bug já foi resolvido. Aqui mostra que o seu status passa para *resolved*.

Closed (fechado) – Quando um bug é resolvido, ele deve ser re-testado por quem fez a reportagem dele. Caso o erro persista, seu status deve ser marcado como feedback e retornado a quem o resolveu. Se o erro foi realmente corrigido, o status deverá ser marcado como “closed”, indicando que a funcionalidade está livre daquele erro.

Cada **nível** de usuários do Mantis, possui tem diferentes responsabilidades dentro da ferramenta. São elas:

 Clique nos botões para ver as informações.

1º. Visualizador



Neste nível, os usuários têm apenas permissão de consulta. Este nível geralmente é atribuído a pessoas que pertencem a outros projetos e que precisam de certa forma consultar algo na ferramenta. Exemplo: como um determinado bug foi resolvido.

2º. Relator



São usuários que estão habilitados a reportar bugs e atribuí-los aos responsáveis pelo desenvolvimento da funcionalidade na qual o erro foi encontrado.

3º. Atualizador



Os usuários deste nível do Mantis podem atualizar bugs e/ou tarefas já atribuídas a alguém. Neste nível, os relatores e desenvolvedores têm este acesso, para que possam incluir comentários durante a resolução dos bugs e/ou tarefas.

4º. Desenvolvedor



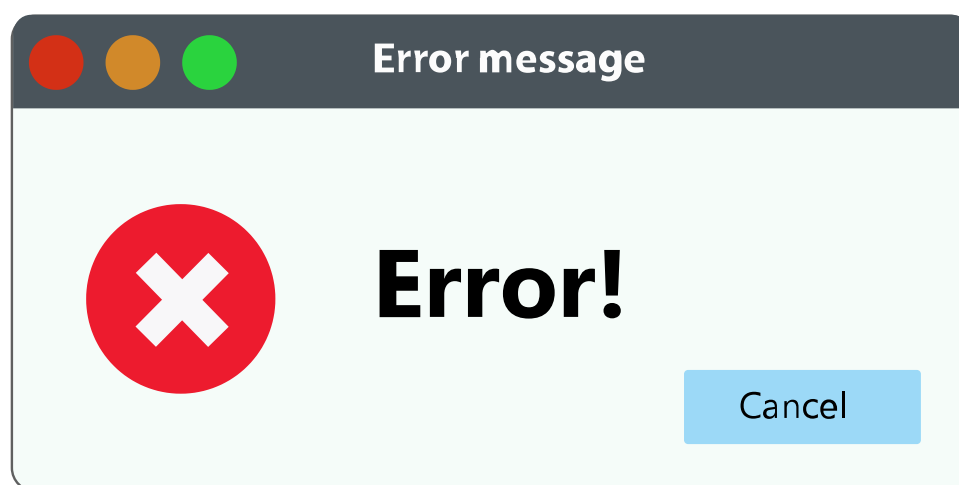
Este nível é especial, reservado às pessoas que resolverão os bugs propriamente ditos, mas o desenvolvedor não poderá alterar a descrição do bug ou da tarefa atribuída a ele; porém, poderá incluir comentários e alterar seu status para resolvido.

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

Processo de gestão de defeitos

Objetivo desse processo: minimizar os riscos de um projeto e definir práticas para prevenir os defeitos. As ferramentas automatizadas oferecem um meio para fomentar a integração entre o time de desenvolvimento e o time de testes, além de uma base comum para a entrada de informações.

Por meio dos relatórios de gestão e métricas geradas por essas ferramentas, os gestores do projeto poderão promover a melhoria contínua do processo estabelecido.



 (Fonte: 1000s_pixels / Shutterstock).

A definição genérica do termo Erro (**Error**) é utilizada normalmente para indicar uma diferença entre os valores: computado, observado ou medido em relação ao esperado.

No entanto, o padrão IEEE 610.12-1990 (*IEEE Standard Glossary of Software Engineering Terminology*) distingue a terminologia da seguinte forma:



Defeito (*Fault*)

Passo, processo ou definição de dados incorretos. Exemplo: uma instrução incorreta no código ou uma falta num artefato estático.



Engano (*Mistake*)

Ação humana que produz um resultado incorreto. Exemplo: uma ação incorreta tomada pelo desenvolvedor ou analista.




Falha (*Failure*)

Desvio entre o resultado/comportamento apresentado pela aplicação em relação aos requisitos. Exemplo: a falha ocorre em consequência de um erro, defeito ou engano, gerando um comportamento incorreto da aplicação.

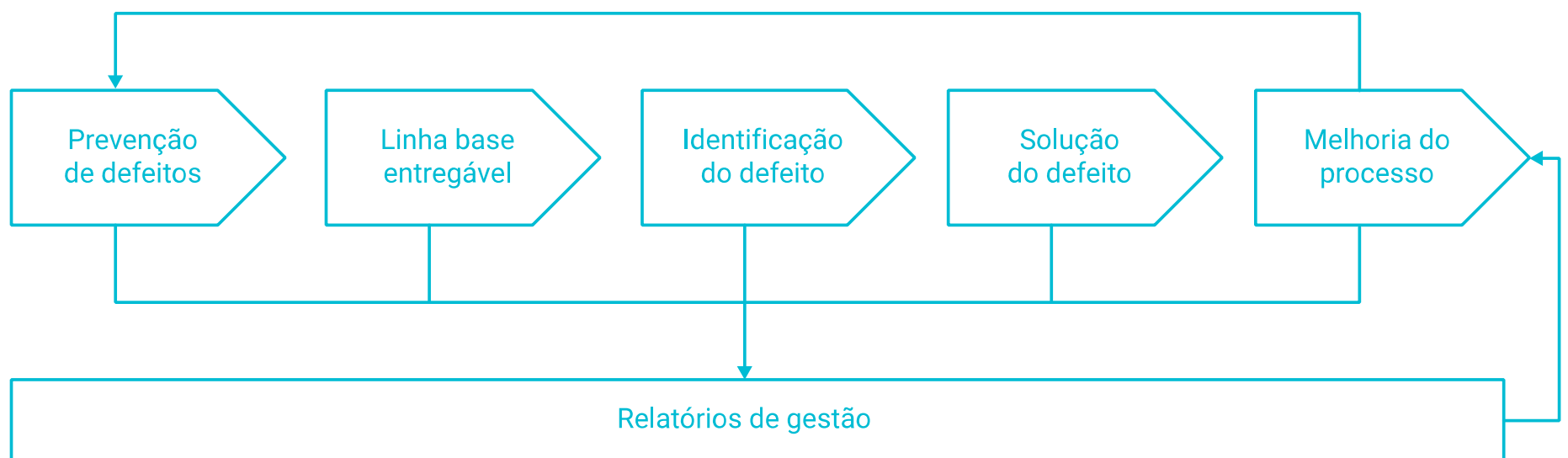
Leia o texto a seguir:

 Processo de gestão de defeitos

 Clique no botão acima.

Processo de gestão de defeitos

Quais são os elementos chave de um processo de gestão de defeitos?



O padrão IEEE 610.12-1990 (*IEEE Standard Glossary of Software Engineering Terminology*) define seis elementos chave:

- **Prevenção de defeitos:** Com base no levantamento dos riscos críticos do projeto, devem ser promovidas ações de prevenção e planejamento de contingências para minimizar o impacto caso os riscos tornem-se problemas;
- **Linha base entregável:** Estabelecimento formal de linhas base (baselines), por meio da gGerência de cConfiguração de Software. Cada linha base deve determinar quais requisitos/ artefatos serão liberados e submetidos ao teste;
- **Identificação do defeito:** Definição das técnicas necessárias para encontrar, reportar e classificar os defeitos, assim como, os critérios para reconhecê-los;
- **Solução do defeito:** Definição das atividades para a correção e posterior notificação da resolução do defeito;
- **Melhoria do processo:** Análise das métricas e relatórios de gestão para entender a causa raiz dos problemas e promover a melhoria contínua do processo;
- **Relatório de gestão:** geração de relatórios com dados relevantes para acompanhar o progresso dos testes e a qualidade do sistema, assim como a geração de métricas para alimentar a atividade de melhoria do processo.

Atenção! Aqui existe uma videoaula, acesso pelo conteúdo online

Gerenciando o ciclo de vida de um defeito no Mantis

Ciclo de vida de um defeito

Sabemos que a qualidade de um sistema pode ser medida pelo número de defeitos encontrados durante a execução dos testes.

Por meio da execução formal dos testes (estruturais ou funcionais), durante a utilização do sistema em produção ou por acidente, esses defeitos podem ser encontrados.

Como classificamos os defeitos por categorias?

Faltante (*Missing*): o defeito ocorre em virtude da falta parcial ou total de um requisito.

Errado (*Wrong*): ocorre porque o requisito foi implementado incorretamente.

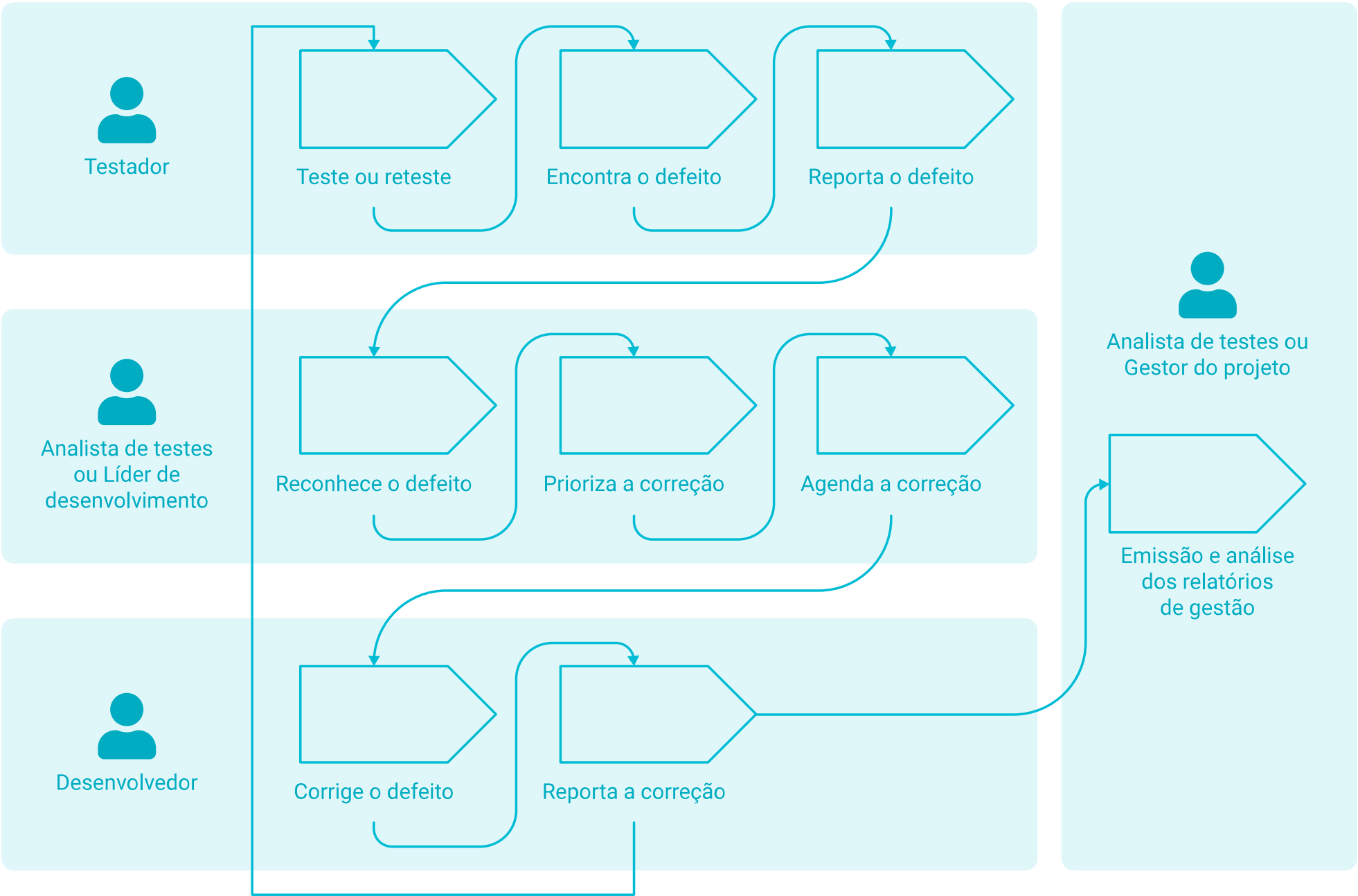
Acréscimo (*Extra*): ocorre em virtude de um comportamento ou elemento que foi implementado, mas foi não especificado no requisito.

Seja por intenção ou por acidente, já que o defeito foir encontrado, no passo seguinte devemos fazer o seu relato (ou reporte) desse defeito por meio de algum mecanismo estabelecido no processo de gestão de defeitos.

Atenção

Podemos relatar com uma simples planilha ou uma ferramenta automatizada. Assim que o defeito for relatado, ele deverá ser submetido a um ciclo de vida predefinido pelo processo de gestão de defeitos. Este ciclo de vida define os fluxos que o defeito deverá percorrer até o seu fechamento.

Vejamos o ciclo de vida de um defeito referente ao processo de gestão de defeitos apresentado anteriormente:



Algumas recomendações para o relato de defeitos

Não podemos relegar ao segundo plano o relato de um defeito. Ele deve ser um dos passos fundamentais do processo de gestão de defeitos.

Diretrizes que devem ser seguidas durante o relato de um defeito com base nas recomendações descritas no livro *Base de conhecimento em teste de software* (Martins Fontes, 2007):

- **Resumir:** descreva claramente o defeito, mas de forma resumida;
- **Precisão:** certifique-se de que o defeito identificado realmente é um desvio do comportamento esperado, e não uma falha de entendimento;
- **Neutralizar:** relate apenas os fatos, evitando manifestações de humor, emoção etc.;
- **Generalizar:** procure entender o problema de forma genérica, em virtude de que ele também pode acontecer em outras situações ou funcionalidades;
- **Reproduzir:** garanta que o defeito seja reproduzível e descreva os passos necessários para a sua reprodução;
- **Evidenciar:** evidencie a existência do defeito encontrado por meio de arquivos de saída, printscreens (tecla PrtSc SysReq) das telas etc.;
- **Revisar:** revise a descrição e os passos para reproduzir o defeito. Lembre-se de que o relato do defeito é um documento do projeto, assim como um caso de uso, um plano de testes etc. Trate-o como tal.

Atividade

1. O que é um BugTracker?

- a) Um software que não tinha um kernel (núcleo) estável.
 - b) É uma aplicação J2EE de acompanhamento e gestão dos problemas.
 - c) É uma ferramenta que tem a funcionalidade de wiki para documentação.
 - d) Um sistema de rastreamento de defeitos.
 - e) É um software que elimina o histórico dos bugs do sistema.
-

2. Qual sistema de banco de dados não é suportado pelo Mantis BugTracker?

- a) MySQL.
 - b) MS SQL.
 - c) ADABAS.
 - d) PostgreSQL.
 - e) DB2.
-

3. Qual tarefa não é necessária para a pré-instalação do MantisBT?

- a) Fazer o download do MantisBT.
 - b) Transferir o arquivo baixado para o seu servidor.
 - c) Extrair o lançamento.
 - d) Renomear o diretório.
 - e) Visualizar os usuários que têm apenas permissão de consulta.
-

4. Cite os seis elementos chave de um processo de gestão de defeitos definidos pelo padrão IEEE 610.12-1990.

Referências

BARTIÉ, Alexandre. **Garantia de qualidade de software**. 1. ed. Rio de Janeiro: Campus, 2002.

Notas

BASTOS, Anderson *et al.* **Base de conhecimento em teste de software**, Martins Fontes, 2007.

BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. **UML**: guia do usuário. 2.ed. Editora

CAETANO, C. **Automação e gerenciamento de testes**: aumentando a produtividade com as principais soluções open source e gratuitas. Disponível em: <http://www.linhadecodigo.com.br/artigo/1566/automacao-e-gerenciamento-de-testes-aumentando-a-produtividade-com-as-principais-solucoes-open-source-e-gratuitas-2a-edicao.aspx>. Acesso em: 06 jun. 2019.

COCKBURN, A. **Escrevendo casos de uso eficazes**. Porto Alegre: Bookman, 2005.

KALANI, Permatan Laxminiwas. **Issue/Bug Tracking System**. Chico: California State University, 2005.

MANTISBT DEVELOPMENT TEAM. **Guia de administração**: referência para administradores. Disponível em: https://translate.google.com/translate?depth=1&hl=pt-BR&prev=search&rurl=translate.google.com.br&sl=en&sp=nmt4&u=http://www.mantisbt.org/docs/master/en-US/Admin_Guide/Admin_Guide.pdf&xid=17259,15700023,15700124,15700149,15700186,15700191,15700201,15700237,15700242. Acesso em: 5 jun. 2019.

PRESSMAN, R. S. **Engenharia de Software**. 6. ed. São Paulo: Makron Books, 1995.

SOMMERVILLE, Ian. **Engenharia de Software**. 9. ed. São Paulo: Pearson, 2011.

Próxima aula

- Instalando o TestLink:

- Instalando o TestLink;
- Configuração e utilização;
- Criando novos usuários e definindo papéis;
- Criando planos de teste e casos de teste no TestLink;
- Atribuindo e executando testes;
- Reportando o status do teste;
- Relatórios e métricas.

Explore mais

Assista aos vídeos:

- [Laboratório de testes com TestLink.](#)
- [Tutorial de instalação e utilização do TestLink.](#)

Leia os textos

- [Testlink - uma ferramenta de gerenciamento de testes de software.](#)
- Livro *Teste de software* - 3ª Edição revisada e atualizada, de Emerson Rios e Trayahú Moreira Filho.