**<<Projeto nome- nome da instituição>>**

***Plano de Teste***

Histórico de alteração de documentos

|  |  |  |  |
| --- | --- | --- | --- |
| **Número da versão** | **Data** | **Contribuinte** | **Descrição** |
| V1.0 |  |  | Quais mudanças (adições e exclusões) foram feitas para esta versão? |
|  |  |  |  |
|  |  |  |  |

**\*\* Nota para Autor de Documentos –** Texto vermelho e azul (com exceção do título e nome do documento acima) neste documento é direcionado ao usuário do modelo para descrever processos, criar padrões e ajudar a construir o documento a partir do modelo. Todo esse texto vermelho e azul deve ser removido antes de enviar qualquer documentação formal, incluindo rascunho e/ou final, entregues.  **\*\*\*\***

**Tabela de Conteúdos**

1 Introdução 3

1.1 Scope 3

1.1.1 Em escopo 3

1.1.2 Out of Scope 3

1.2 Objetivo de Igualdade 3

1.2.1 ObjetivoPrimário 3

1.2.2 Ocondary Objective 4

1.3 Roles e Responsabilidades 4

1.3.1 Developer 4

1.3.2 Adopter 4

1.3.3 Testando equipe de gerenciamento de processos 4

1.4 Assumptions para execução de teste 5

1.5 Constrações para execução de teste 5

1.6 Definições 6

2 Metodologia TestE 6

2.1 Purpose 6

2.1.1 Visão geral 6

2.1.2 Testes de capacidade 6

2.1.3 Teste de unidade (múltiplo) 7

2.1.4 Iteração/Teste de regressão 7

2.1.5 Sa lançamento final 7

2.1.6 Teste critérios de completude 8

2.2 Teste níveis 8

2.2.1 Testes debuild 8

2.2.1.1 Level 1 - Testes de aceitação de construção 8

2.2.1.2 Level 2 - Testes de Fumaça 8

2.2.1.3 Level 2a - Teste de Regressão de Insetos 8

2.2.2 Milestone Tests 9

2.2.2.1 Level 3 - Testes de caminho crítico 9

2.2.3 Lerelease tests 9

2.2.3.1 Level 4 - Testes Padrão 9

2.2.3.2 Level 5 - Teste sugerido 9

2.3 Bug Regressão 9

2.4 Bug Triagem 9

2.5 Critérios de suspensão e requisitos de retomada 10

2.6 Teste completude 10

2.6.1 Condições padrão: 10

2.6.2 Bug Relatórios & Condições de Triagem: 10

3 Test Deliverables 11

3.1 Deliverables Matrix 11

3.2 Documentos 12

3.2.1 Teste documento de aproximação 12

3.2.2 Test Plan 12

3.2.3 Teste agenda 13

3.2.4 Test especificações 13

3.2.5 Requirements Traceability Matrix 13

3.3 Defect Tracking & Debugging 13

3.3.1 Testando fluxo de trabalho 13

3.3.2 Defetantes usando G FORGE 14

3.4 Reports 16

3.4.1 Teste relatórios de status 16

3.4.2 Relatórios de conclusão da pasta 16

3.4.3 Teste relatório final - Aprovação 16

3.5 Matriz de responsabilidade 16

4 Resource & Environment Needs 16

4.1 Testando ferramentas 16

4.1.1 Ferramentas de rastreamento 16

4.1.1.1 Gestão de configuração 17

4.2 Teste ambiente 17

4.2.1 Hardware 17

4.2.2 Software 17

4.3 Bug Gravidade e Definição Prioritária 17

4.3.1 Lista 17

4.3.2 Lista deprioridade 18

4.4 Bug Reporting 18

5 Terms/Siglas 18

# Introdução

Este documento de abordagem de teste descreve as estratégias, processos, fluxos de trabalho e metodologias adequadas utilizadas para planejar, organizar, executar e gerenciar testes de projetos de software dentro do caBIG.

## Âmbito

Descreva o escopo atual da abordagem do teste com base em sua função e objetivos do projeto.

### Em Escopo

O <nome sistema> < de <seaçado do caBIG define a abordagem de teste de unidade, integração, sistema, regressão e aceitação do cliente. O escopo de teste inclui o seguinte:

* Teste de todos os requisitos funcionais, de desempenho do aplicativo, segurança e de uso listados no documento *Caso de Uso* .
* Requisitos de qualidade e métricas < nome do sistema>.
* Testes e testes de interfaces de todos os sistemas que interagem com o <nome do sistema>.
* Fora de escopo

Os seguintes são considerados fora de escopo para caBIG < nome do espaço de trabalho> <istema nome> plano de teste do sistema e escopo de teste:

* Testes de requisitos funcionais para sistemas fora < nome de aplicação>
* Testes de SOPs de Negócios, recuperação de desastres e Plano de Continuidade de Negócios.

## Objetivo de Qualidade

### Objetivo Primário

Um objetivo primordial dos sistemas de aplicação de teste é: ***assegurar que o sistema atenda aos requisitos completos, incluindo requisitos de qualidade (também conhecidos: requisitos não funcionais) e ajustar métricas para cada requisito de qualidade e satisfaz os cenários de caso de uso e mantenha a qualidade do produto.*** Ao final do ciclo de desenvolvimento do projeto, o usuário deve descobrir que o projeto atendeu ou excedeu todas as suas expectativas conforme detalhado nos requisitos.

Quaisquer alterações, adições ou exclusões ao documento de requisitos, Especificação Funcional ou Especificação de Design serão documentadas e testadas no mais alto nível de qualidade permitido dentro do tempo restante do projeto e dentro da capacidade da equipe de teste.

### Objetivo Secundário

O objetivo secundário dos sistemas de aplicação de teste será: ***identificar e expor todos os problemas e riscos associados, comunicar todos os problemas conhecidos à equipe do projeto e garantir que todas as questões sejam tratadas em um assunto apropriado antes da liberação.*** Como objetivo, isso requer testes cuidadosos e metódicos da aplicação para primeiro garantir que todas as áreas do sistema sejam examinadas e, consequentemente, todas as questões (bugs) encontradas sejam tratadas adequadamente.

## Papéis e Responsabilidades

As funções e responsabilidades podem diferir com base no SOW real. As funções listadas abaixo são para a fase de teste.

### Revelador

Um Centro de Câncer designado pelo NCI selecionado e financiado pelo NCICB para participar de um Espaço de Trabalho específico para realizar atividades de desenvolvimento de software ou soluções. Responsável por:

a Desenvolver o sistema/aplicação

b Desenvolver casos e requisitos de uso em colaboração com os Adotantes

c Teste de Unidade de Conduta, sistema, regressão e integração

d Suportar testes de aceitação do usuário

### Adotar

Um Centro de Câncer designado pelo NCI selecionado e financiado pela NCICB para realizar adoção formal, testes, validação e aplicação de produtos ou soluções desenvolvidos pelos Desenvolvedores do Espaço de Trabalho. Responsável por:

a Contribuir para o caso de uso, desenvolvimento de requisitos através de revisão

(b) Contribuir para o desenvolvimento e execução dos scripts de teste de desenvolvimento através da revisão

c Realizar a aceitação completa do usuário, a regressão e os testes completos; isso inclui identificar cenários de teste, construir os scripts de teste, executar scripts e relatar resultados de testes

### Equipe de gerenciamento de processos de teste

Inclua NCI, BAH e Cancer Center Leads alocados no nome <workspace>. Grupo responsável por gerenciar todo o processo de teste, fluxo de trabalho e gestão da qualidade com atividades e responsabilidades para:

a Monitorar e gerenciar atividades de teste de integridade e suporte de testes

b Coordenar atividades em centros de câncer

Adicione mais conforme apropriado ao escopo de teste

## Suposições para execução de testes

Abaixo estão algumas premissas mínimas (em preto) que foram completadas com alguns exemplos (em vermelho). Qualquer exemplo pode ser usado se for considerado apropriado para o projeto específico. Novas suposições também podem ser adicionadas que são fundamentadas para serem adequadas ao projeto.

* Para testes de aceitação do usuário, a equipe do Desenvolvedor concluiu testes de unidade, sistema e integração e cumpriu todos os requisitos (incluindo requisitos de qualidade) com base na Matriz de Rastreabilidade de Requisitos.
* Os testes de aceitação do usuário serão realizados pelos usuários finais
* Os resultados dos testes serão informados diariamente usando Gforge. Scripts com falhas e lista de defeitos do Gforge com evidências serão enviados diretamente ao Desenvolvedor.
* Os casos de uso foram desenvolvidos por adotantes para testes de aceitação do usuário. Os casos de uso são aprovados por chumbo de teste.
* Os scripts de teste são desenvolvidos e aprovados.
* A Equipe de Teste apoiará e fornecerá orientações adequadas aos adotantes e desenvolvedores para realizar testes
* Grandes dependências devem ser relatadas imediatamente após a reunião de início dos testes.

## Restrições para execução de testes

Abaixo estão algumas premissas mínimas (em preto) seguidas de restrições de exemplo (vermelho). Qualquer exemplo pode ser usado se for considerado apropriado para o projeto específico. Novas restrições também podem ser adicionadas que são fundamentadas para serem adequadas ao projeto.

* Os adotantes devem entender claramente sobre os procedimentos de teste e registrar um defeito ou aprimoramento. A Equipe de Gerenciamento de Processos de Teste agendará uma teleconferência com desenvolvedores e adotantes para treinar e resolver quaisquer problemas relacionados aos testes.
* O desenvolvedor receberá uma lista consolidada de solicitação para configuração do ambiente de teste, configuração de contas de usuário, conjunto de dados (dados reais e falsos), lista de defeitos, etc. através do **GForge** após a reunião inicial de início do teste do Adotador.
* Desenvolvedor apoiará atividades de teste contínuas com base em prioridades
* Os scripts de teste devem ser aprovados pela execução prévia do teste de teste
* Scripts de teste, ambiente de teste e dependências devem ser abordados durante a reunião de início de teste na presença de uma PME e a lista de solicitações deve ser enviada dentro de 3 dias após a reunião de início
* O Desenvolvedor não pode executar os scripts de teste de Aceitação do Usuário e fim a fim. Após a depuração, o desenvolvedor pode realizar seu teste interno, mas nenhum resultado desse teste pode ser registrado/relatado.
* Os adotantes são responsáveis por identificar dependências entre scripts de teste e enviar solicitação clara para configurar o ambiente de teste

## Definições

Bugs: Qualquer erro ou defeito que cause o problema do software/aplicativo ou hardware. Isso também está incluído nos requisitos e não atende ao fluxo de trabalho, processo ou ponto de função necessários.

Melhoria:

1) Qualquer alteração ou modificação no sistema existente para melhor fluxo de trabalho e processo.

2) Um erro ou defeito que faz com que o software/aplicativo ou hardware funcione mal.

Quando 1) e 2) NÃO estão incluídos nos requisitos podem ser categorizados como um aprimoramento.

O aprimoramento pode ser adicionado como um novo requisito após o processo de gerenciamento de alterações apropriado.

# Metodologia de teste

## Propósito

### Visão geral

A lista abaixo não se destina a limitar a extensão do plano de teste e pode ser modificada para se tornar adequada para o projeto específico.

O objetivo do Plano de Teste é alcançar o seguinte:

1. Defina estratégias de teste para cada área e subárea para incluir todos os requisitos funcionais e de qualidade (não funcionais).
2. Divida a especificação de design em áreas e subáreas testáveis (não confunda com especificações de teste mais detalhadas). Certifique-se também de identificar e incluir áreas que devem ser omitidas (não testadas) também.
3. Defina procedimentos de rastreamento de bugs.
4. Identificar riscos de teste.
5. Identificar recursos necessários e informações relacionadas.
6. Forneça o cronograma de testes.

### Teste de usabilidade

O objetivo dos testes de usabilidade é garantir que os novos componentes e recursos funcionem de forma aceitável para o cliente.

O desenvolvimento normalmente criará um protótipo não funcional dos componentes de interface do usuário para avaliar o design proposto. Os testes de usabilidade podem ser coordenados por testes, mas os testes reais devem ser realizados por não-testadores (**o mais próximo possível dos usuários finais).** Os testes revisarão os resultados e fornecerão à equipe do projeto sua avaliação do impacto que essas mudanças terão no processo de teste e no projeto como um todo.

### Teste unitário (Múltiplo)

O Teste unitário é realizado pelo Desenvolvedor durante o processo de desenvolvimento de código para garantir que a funcionalidade adequada e a cobertura de código tenham sido alcançadas por cada desenvolvedor durante a codificação e em preparação para a aceitação em testes de iterações.

A seguir, as áreas de exemplo do projeto devem ser testadas e assinadas por unidade antes de serem passadas para testes de regressão:

1. Bancos de dados, procedimentos armazenados, gatilhos, tabelas e índices
2. Serviços NT
3. Conversão de banco de dados
4. . OCX, . DLL. EXE e outros executáveis binários formatados

### Teste de iteração/regressão

Durante os ciclos repetidos de identificação de bugs e recebimento de novas compilações (contendo alterações no código de correção de erros), existem vários processos que são comuns a essa fase em todos os projetos. Estes incluem os vários tipos de testes: funcionalidade, desempenho, estresse, configuração, etc. Há também o processo de comunicação dos resultados dos testes e a garantia de que novas gotas/iterações contenham correções estáveis (regressão). O projeto deve planejar um mínimo de 2-3 ciclos de testes (gotas/iterações de novas construções).

Em cada iteração, um interrogatório deve ser realizado. Especificamente, o relatório deve mostrar que, no melhor grau alcançável durante a fase de teste de iteração, todos os erros identificados de gravidade 1 e gravidade 2 foram comunicados e abordados. No mínimo, todos os bugs prioritários 1 e prioritário 2 devem ser resolvidos antes de entrar na fase beta.

Abaixo estão exemplos. Qualquer exemplo pode ser usado se for considerado apropriado para o projeto específico. Novos conteúdos também podem ser adicionados que são fundamentados para serem adequados ao projeto.

Os resultados importantes necessários para a aceitação nos testes de versão final incluem:

1. CONFIGURAÇÃO DO aplicativo. EXE
2. Instruções de instalação
3. Toda a documentação (scripts de teste beta, manuais ou guias de treinamento, etc.)

### Teste final de lançamento

A equipe de testes com usuários finais participa desse processo de marco, bem como fornece feedback de confirmação sobre novos problemas descobertos e entrada com base em problemas idênticos ou semelhantes detectados anteriormente. A intenção é verificar se o produto está pronto para distribuição, aceitável para o cliente e resolver possíveis problemas operacionais.

Supondo que os bugs críticos sejam resolvidos durante os testes de iterações anteriores- Durante todo o ciclo de teste de liberação final, as correções de bugs serão focadas em bugs menores e triviais (gravidade 3 e 4). Os testes continuarão seu processo de verificação da estabilidade da aplicação através de testes de regressão (bugs conhecidos existentes, bem como casos de teste existentes).

O objetivo desta fase é estabelecer que o aplicativo em teste atingiu um nível de estabilidade, apropriado para seu uso (usuários numégios, etc.), que pode ser liberado para os usuários finais e comunidade caBIG.

### Critérios de completude de teste

A liberação para produção só pode ocorrer após a conclusão bem sucedida da aplicação em teste durante todas as fases e marcos anteriormente discutidos acima.

O objetivo marco é colocar o release/app (build) em produção depois que for demonstrado que o aplicativo atingiu um nível de estabilidade que atende ou excede as expectativas do cliente conforme definido nos Requisitos, Especificação Funcional e caBIG Production Standards.

## Níveis de teste

Os testes de uma aplicação podem ser divididos em três categorias primárias e vários subníveis. As três categorias primárias incluem testes realizados em cada build (Build Tests), testes realizados em cada grande marco (Testes de Marco) e testes realizados pelo menos uma vez em cada ciclo de lançamento do projeto (Testes de Liberação). As categorias de teste e os níveis de teste são definidos abaixo:

### Construir testes

#### Nível 1 - Testes de aceitação de construção

Testes de aceitação de compilação devem levar menos de 2-3 horas para serem concluídos (15 minutos é típico). Esses casos de teste simplesmente garantem que o aplicativo possa ser construído e instalado com sucesso. Outros casos de teste relacionados garantem que os adotantes receberam o documento de liberação de desenvolvimento adequado mais outras informações relacionadas à compilação (ponto de queda, etc.). O objetivo é determinar se novos testes são possíveis. Se algum caso de teste nível 1 falhar, a compilação será devolvida aos desenvolvedores não testados.

#### Nível 2 - Testes de Fumaça

Os testes de fumaça devem ser automatizados e levar menos de 2-3 horas (20 minutos é típico). Esses casos de testes verificam a funcionalidade principal de alto nível.

O objetivo é determinar se novos testes são possíveis. Esses casos de teste devem enfatizar a amplitude mais do que a profundidade. Todos os componentes devem ser tocados, e todos os principais recursos devem ser testados brevemente pelo Teste de Fumaça. Se algum caso de teste nível 2 falhar, a compilação será devolvida aos desenvolvedores não testados.

#### Nível 2a - Teste de Regressão de Insetos

Todos os bugs que foram "Abertos" durante a compilação anterior, mas marcados como "Fixo, Precisa retestar" para a compilação atual em teste, precisarão ser regredidos ou retestados. Uma vez que o teste de fumaça é concluído, todos os insetos resolvidos precisam ser regredidos. Deve levar entre 5 minutos a 1 hora para regredir a maioria dos insetos.

### Testes de Marcos

#### Nível 3 - Testes de Caminho Crítico

Os casos de teste do Critical Path são direcionados aos recursos e funcionalidades que o usuário verá e usará todos os dias.

Os casos de teste do Caminho Crítico devem passar até o final de cada ciclo de teste de compilação de 2-3. Eles não precisam ser testados a cada gota, mas devem ser testados pelo menos uma vez por marco. Assim, os casos de teste do Caminho Crítico devem ser todos executados pelo menos uma vez durante o ciclo de Iteração e, uma vez, durante o ciclo de Liberação Final.

### Testes de liberação

#### Nível 4 - Testes Padrão

Casos de teste que precisam ser executados pelo menos uma vez durante todo o ciclo de teste para esta versão. Esses casos são executados uma vez, não repetidos como são os casos de teste em níveis anteriores. Testes funcionais e testes de design detalhados (casos de teste de especificações e especificações de design funcionais, respectivamente). Estes podem ser testados várias vezes para cada Ciclo de Teste de Marco (Iteração, Versão Final, etc.).

Os casos de teste padrão geralmente incluem instalação, dados, GUI e outras áreas de teste.

#### Nível 5 - Teste Sugerido

Estes são casos de teste que seriam agradáveis de executar, mas podem ser omitidos devido a restrições de tempo.

A maioria dos casos de teste de desempenho e estresse são exemplos clássicos de casos de teste sugeridos (embora alguns devem ser considerados casos de teste padrão). Outros exemplos de casos de teste sugeridos incluem testes de WAN, LAN, Rede e Carga.

## Regressão de Insetos

A Regressão do Bug será um inquilino central durante todas as fases de teste.

Todos os bugs que forem resolvidos como "Corrigidos, Necessidades Retestando" serão regredidos quando a equipe de testes for notificada da nova gota contendo as correções. Quando um bug passa a regressão será considerado "Fechado, Fixo". Se um bug falhar na regressão, a equipe de testes dos adotantes notificará a equipe de desenvolvimento inserindo notas no GForge. **Quando um bug Severity 1 falha na regressão, a equipe de testes dos adotantes também deve enviar um e-mail imediato para o desenvolvimento.** O Test Lead será responsável por acompanhar e relatar ao desenvolvimento e gerenciamento de produtos o status dos testes de regressão.

## Triagem de Insetos

As Triagens de Bugs serão realizadas em todas as fases do ciclo de desenvolvimento. As triagens de bugs serão de responsabilidade do Test Lead. As triagens serão realizadas regularmente, com o prazo sendo determinado pela taxa de achado de bugs e horários do projeto.

Assim, seria típico realizar poucas triagens durante a **fase de Planejamento**, depois talvez uma triagem por semana durante a **fase de Projeto**, aumentando para duas vezes por semana durante as últimas **etapas da fase de Desenvolvimento**. Em seguida, a fase de Estabilização deve ver uma redução substancial no número de novos bugs encontrados, portanto, algumas triagens por semana seriam o máximo (para lidar com o status em bugs existentes).

O Líder de Teste, o Gerente de Produto e o Líder de Desenvolvimento devem estar todos envolvidos nessas reuniões de triagem. O Test Lead fornecerá documentação e relatórios necessários sobre bugs para todos os participantes. O objetivo da triagem é determinar o tipo de resolução para cada bug e priorizar e determinar um cronograma para todos os "Erros Fixos". O desenvolvimento atribuirá os bugs à pessoa apropriada para fixação e relatará a resolução de cada bug de volta ao sistema de rastreador de bugs GForge. O Test Lead será responsável por rastrear e relatar o status de todas as resoluções de bugs.

## Critérios de suspensão e requisitos de retomada

Esta seção deve ser definida para listar critérios e requisitos de retomada caso determinados níveis de grau e pré-definidos de objetivos e metas de teste não sejam atingidos.

Veja o exemplo abaixo:

- Os testes serão suspensos no módulo de software afetado quando os bugs do caso do teste Smoke Test (Nível 1) ou Critical Path (Nível 2) forem descobertos após a terceira iteração.

- Os testes serão suspensos se houver mudança de escopo crítica que impacte o Caminho Crítico

Um relatório de bugs deve ser arquivado pela equipe de desenvolvimento. Após a correção do bug, a equipe de desenvolvimento seguirá os critérios de queda (descritos acima) para fornecer sua última queda para testes adicionais. Nesse momento, os adotantes regredirão o bug e, se passarem, continuarão testando o módulo.

## Completude do teste

Os testes serão considerados completos quando as seguintes condições tiverem sido atendidas:

### Condições Padrão:

1. Quando adotantes e desenvolvedores, concordam que os testes estão concluídos, o aplicativo é estável e concordam que o aplicativo atende aos requisitos funcionais.
2. A execução do script de todos os casos de teste em todas as áreas já passou.
3. Casos de teste automatizados já passaram em todas as áreas.
4. Todos os bugs prioritários 1 e 2 foram resolvidos e fechados.
5. NCI aprova a conclusão do teste
6. Cada área de teste foi assinada conforme concluído pelo Test Lead.
7. 50% de todos os erros de gravidade 1 e 2 resolvidos foram regredidos com sucesso como validação final.
8. Os testes ad hoc em todas as áreas foram concluídos.

### Relatórios de bugs e condições de triagem:

Adicione as condições de relatórios de bugs e triagem que serão submetidas e avaliadas para medir o estado atual.

1. A taxa de achado de bugs indica uma tendência decrescente antes da Taxa de Bug Zero (nenhum novo sev. 1/2/3 bugs encontrados).
2. A taxa de achado de bugs permanece em 0 novos bugs encontrados (Severity 1/2/3) apesar de um esforço constante de teste ao longo de 3 ou mais dias.
3. A distribuição da gravidade dos insetos mudou para uma diminuição constante no Sev. 1 e 2 insetos descobertos.
4. Nenhum bug 'Deve corrigir' permanece antes, apesar dos testes sustentados.

# Resultados de teste

Os testes fornecerão entregas específicas durante o projeto. Essas entregas se enquadram em três categorias básicas: Documentos, Casos de Teste / Anotações de Bugs e Relatórios. Aqui está um diagrama indicando as dependências dos vários entregadores:



Como o diagrama acima mostra, há uma progressão de um entregue para o próximo. Cada entrega tem suas próprias dependências, sem as quais não é possível completar totalmente o entregue.

A página a seguir contém uma matriz que descreve todas as entregas que os testes usarão.

## Matriz de Entregas

Abaixo está a lista de artefatos que são conduzidos pelo processo e devem ser produzidos durante o ciclo de vida do teste. Certos resultados devem ser entregues como parte da validação do teste, você pode adicionar à lista abaixo de entregas que suportam os objetivos gerais e manter a qualidade.

Esta matriz deve ser atualizada rotineiramente durante todo o ciclo de desenvolvimento do projeto em você projetar plano de teste específico.

|  |
| --- |
| **Entrega** |
| **Documentos** |
| Abordagem de teste |
| 🡪 Plano de Teste |
| 🡪 Cronograma de testes |
| 🡪 Especificações de teste |
| **Caso de teste / Gravação de bugs** |
| Casos de teste / Resultados |
| Relatórios de cobertura de testes |
| Rastreador de bugs GForge para relatórios de bugs |
| **Relatórios** |
| Relatório de resultados do teste |
| Relatório Final do Teste - Aprovação |

## Documentos

### Documento de abordagem de teste

O documento de Abordagem de Teste é derivado dos documentos do Plano de Projeto, Requisitos e Especificação Funcional. Este documento define a abordagem geral do teste a ser feita para o projeto. O documento de abordagem padrão de teste que você está lendo atualmente é uma caldeira da qual o documento de aproximação de teste de projeto mais específico pode ser extraído.

Quando este documento for concluído, o Test Lead irá distribuí-lo ao Gerente de Produto, Líder de Desenvolvimento, Representante do Usuário, Gerente de Programa e outros, conforme necessário para revisão e aprovação.

### Plano de Teste

O Plano de Teste é derivado da Abordagem de Teste, Requisitos, Especificações Funcionais e Especificações de Design detalhadas. O Plano de Teste identifica os detalhes da abordagem do teste, identificando as áreas de caso de teste associadas dentro do produto específico para este ciclo de lançamento.

O objetivo do documento do Plano de Teste é:

1. Especifique a abordagem que o Teste usará para testar o produto e os produtos (extrato da abordagem de teste).
2. Desbre o produto em áreas distintas e identifique características do produto a ser testado.
3. Especifique os procedimentos a serem usados para testar a aprovação e a liberação do produto.
4. Indique as ferramentas usadas para testar o produto.
5. Liste os planos de recursos e agendamento.
6. Indique os contatos dos responsáveis por diversas áreas do projeto.
7. Identificar riscos e planos de contingência que possam impactar os testes do produto.
8. Especifique os procedimentos de gerenciamento de bugs para o projeto.
9. Especifique critérios para aceitação de gotas de desenvolvimento para testes (de compilações).

### Cronograma de testes

Esta seção não é vital para o documento como um todo e pode ser modificada ou excluída se necessário pelo autor.

O Cronograma de Teste é de responsabilidade do Gerenciador de Testes (ou Programador de Departamento, se existir) e será baseado em informações do Programador de Projetos (feito pelo Gerente de Produto). O cronograma de teste específico do projeto pode ser feito no Projeto MS.

### Especificações de teste

Um documento de especificação de teste é derivado do Plano de Teste, bem como dos documentos Requisitos, Especificações Funcionais e Especificação de Design. Ele fornece especificações para a construção de Cases de Teste e inclui listas(s) de áreas de caso de teste e objetivos de teste para cada um dos componentes a serem testados conforme identificado no Plano de Teste do projeto.

### Matriz de rastreabilidade de requisitos

Uma RtM (Requirements Traceability Matrix, matriz de rastreabilidade de requisitos) que é usada para vincular os cenários de teste aos requisitos e casos de uso é uma parte necessária da documentação do Plano de Teste para todos os projetos. A rastreabilidade dos requisitos é definida como a capacidade de descrever e seguir a vida útil de um requisito, tanto em direção para frente quanto para trás (ou seja, desde suas origens, através de seu desenvolvimento e especificação, até sua subsequente implantação e uso, e através de períodos de refinamento e iteração contínuos em qualquer uma dessas fases). [[1]](#footnote-1)

Anexado é uma amostra rtm básica que poderia fornecer um ponto de partida para esta documentação. O importante é escolher um modelo ou base documental que atinja uma rastreabilidade completa ao longo da vida do projeto.



## Rastreamento de defeitos e depuração

### Testando fluxo de trabalho

O fluxo de trabalho abaixo ilustra o processo de fluxo de trabalho de teste para desenvolvedores e adotantes para testes de aceitação do usuário e fim a fim.

Pl. note o processo amarelo destacado onde o Adotante é obrigado a enviar diretamente lista de defeitos com evidências para o Desenvolvedor. Da mesma forma, o Desenvolvedor é obrigado a confirmar diretamente com o Adotante após correções de bugs, juntamente com a atualização no Bugzilla.



### Relatório de defeitos usando G FORGE

TODOS os defeitos devem ser registrados usando 'G FORGE', para resolver e depurar defeitos. Os adotantes também são solicitados a enviar um relatório de defeito diário para o desenvolvedor. Os desenvolvedores atualizarão a lista de defeitos em G Forge e notificarão o solicitante após a resolução do defeito.

Desenvolvedores e adotantes são obrigados a solicitar uma conta no G Forge para o espaço de trabalho relativo. A depuração deve ser baseada em Prioridade – Alta > Média > Baixo, essas prioridades são definidas pelos Adotantes e baseiam-se em quão crítico é o script de teste em termos de dependência e principalmente com base no cenário de caso de uso.

Abaixo a tela captura exibe tela 'Adicionar novo defeito', campos marcados com ( \* ) são campos obrigatórios e os adotantes também devem carregar o arquivo de evidência para todos os defeitos listados.

Todos os defeitos de **alta prioridade** devem ser resolvidos no prazo de 1 dia após a solicitação e resolvidos/encerrados dentro de 2 dias após a solicitação inicial

Todos os defeitos prioritários médios devem ser resolvidos no prazo de 2 dias a partir da solicitação e resolvidos/encerrados dentro de 4 dias após a solicitação inicial

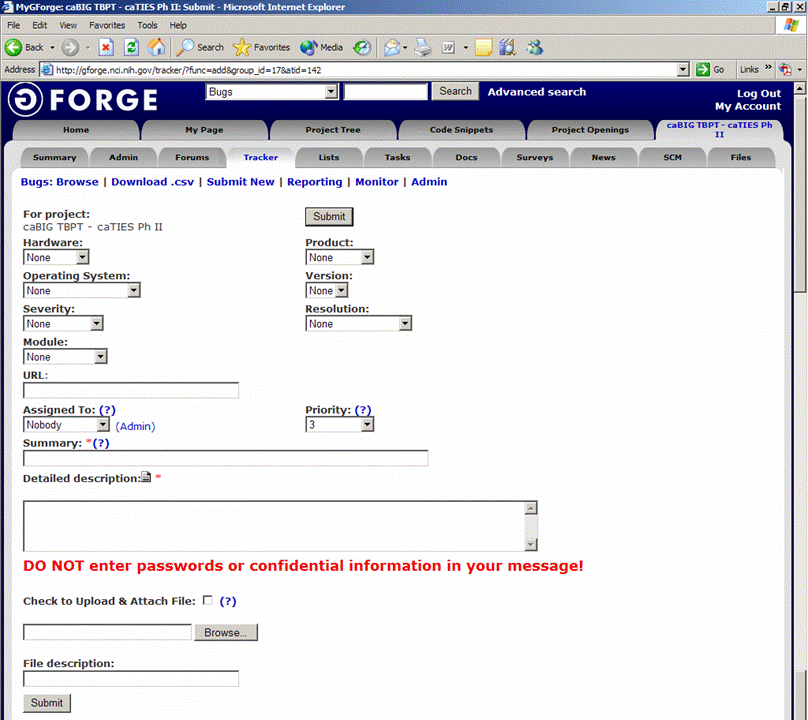
Todos os defeitos de **baixa prioridade** devem ser resolvidos/fechados no máximo 5 dias da solicitação inicial.

G Forge URL - <http://gforge.nci.nih.gov>

O usuário pode procurar espaço de trabalho ou selecionar a partir da lista do projeto recente do lado inferior direito da janela. Por exemplo, procurando por "caties".

No espaço de trabalho, o usuário pode solicitar aos administradores que configurem sua conta de usuário para esse espaço de trabalho.

Após o login, o usuário pode selecionar a guia 'Tracker' para 'Enviar novo' defeito. O usuário pode adicionar informações de defeito. Como mostrado na tela abaixo.



**FieldComments**

Hardware - Liste o hardware do ambiente de teste. por exemplo, PC, MAC

Produto - Adicionar padrão ao nome do espaço de trabalho, por exemplo, caTIES

Sistema Operacional - Win 98, Win 2k, Win XP, MAC, etc.

Versão - Versão do aplicativo

Gravidade - Se for um aprimoramento ou um bug com gravidade menor a grande que pode não interferir com mais testes ou bloquear completamente quaisquer esforços futuros de teste.

Resolução - Somente o DESENVOLVEDOR será atualizado com base no defeito

Módulo - Para uma aplicação com vários módulos, selecione o módulo apropriado para o defeito relatado.

URL - Adicionar URL se houver

Atribuído a - A ser atualizado pelo Desenvolvedor, para quem o bilhete é atribuído

Prioridade - Definir uma prioridade com base na gravidade

Resumo - Resumo do defeito, bug ou problema

Desc detalhado. - Detalhes do defeito, bug ou problema

Arquivo de upload - Anexar arquivo de evidência

Enviar - Enviar o bilhete bug

## Relatórios

O Test Lead será responsável por escrever e divulgar os seguintes relatórios para o pessoal do projeto apropriado, conforme necessário.

### Relatórios de status de teste

Um relatório de status semanal ou quinzenal será fornecido pelo Test Lead para o pessoal do projeto. Este relatório resumirá atividades semanais de testes, problemas, riscos, contagem de bugs, cobertura de casos de teste e outras métricas relevantes.

### Relatórios de conclusão de fase

Quando cada fase de teste for concluída, o Test Lead distribuirá um Relatório de Conclusão de Fase ao gerente de produto, líder de desenvolvimento e gerente de programa para revisão e aprovação.

As balas abaixo ilustram um exemplo do que o documento pode incluir.

O documento deve conter as seguintes métricas:

1. Total de casos de teste, número executado, passes de número / falhas, número ainda para executar
2. Número de bugs encontrados até agora, número resolvido e número ainda aberto
3. Quebra de Bugs por Gravidade / Matriz Prioritária
4. Discussão de Riscos Não Resolvidos
5. Discussão do Progresso do Cronograma (estamos onde deveríamos estar?)

### Relatório Final do Teste - Aprovação

Um relatório final de teste será emitido pelo Test Lead. Ele certificará até que ponto os testes foram efetivamente concluídos (relatório de cobertura de caso de teste sugerido) e uma avaliação da prontidão do produto para liberação à produção.

## Matriz de Responsabilidade

Insira a matriz de recursos de teste – quem está envolvido e qual é o papel.

# Necessidades de recursos & ambientes

## Ferramentas de teste

### Ferramentas de rastreamento

O rastreador de bugs GForge é usado pelo caBIG para inserir e rastrear todos os bugs e problemas de projeto. O Test Lead é responsável pela manutenção do banco de dados GForge.

#### Gerenciamento de configuração

Por favor, inclua seu plano de gerenciamento de configuração

## Ambiente de teste

### Hardware

Inclua os requisitos mínimos de hardware que serão usados para testar o Aplicativo.

Os testes terão controle de acesso a um ou mais servidores de aplicativo/banco de dados separados de qualquer usado por membros não-testados da equipe do projeto. Os testes também terão controle de acesso a um número adequado de estações de trabalho de PC configuradas de várias estrias para garantir o teste de uma faixa do mínimo às configurações de hardware recomendadas do cliente listadas nos documentos requisitos, especificações funcionais e especificação de design do projeto.

### Software

Além do aplicativo e de qualquer outro software especificado pelo cliente, a seguinte lista de software deve ser considerada um mínimo:

1. NT Workstation 4.0 ou Windows 95.
2. MS Office 95 Profissional
3. Bolsa MS
4. TCM (Servidor de ferramentas de teste)
5. Gerenciador de tarefas (servidor de ferramentas de teste)

## Gravidade do bug e definição prioritária

Os campos de Gravidade e Prioridade do Bug são muito importantes para categorizar bugs e priorizar se e quando os bugs serão corrigidos. Os níveis de gravidade e prioridade do bug serão definidos conforme descrito nas tabelas a seguir. Os testes atribuirão um nível de gravidade a todos os bugs. O lead de teste será responsável por ver que um nível de gravidade correto é atribuído a cada bug.

O Gerenciador de Líderes de Teste, Líder de Desenvolvimento e Programa participará de reuniões de revisão de bugs para atribuir a prioridade de todos os bugs ativos atualmente. Este encontro será conhecido como "Encontros de Triagem de Insetos". O Test Lead é responsável por configurar essas reuniões de rotina para abordar o conjunto atual de bugs novos e existentes, mas não resolvidos.

### Lista de Gravidade

O testador que inseriu um bug no GForge também é responsável por inserir o erro Gravidade.

|  |  |  |
| --- | --- | --- |
| **ID de gravidade** | **Nível de gravidade** | **Descrição da gravidade** |
| 1 | Crítico | O módulo/produto trava ou o bug causa condições não recuperáveis. Falhas do sistema, falhas do GP ou corrupção de banco de dados ou arquivos, ou perda potencial de dados, travas de programa que requerem reinicialização são todos exemplos de um Sev. Um inseto. |
| 2 | Alto | Principal componente do sistema inutilizável devido a falha ou funcionalidade incorreta. Sev. 2 bugs causam problemas sérios, como falta de funcionalidade, ou mensagens de erro insuficientes ou pouco claras que podem ter um grande impacto para o usuário, impede que outras áreas do aplicativo sejam testadas, etc. Sev. 2 bugs podem ter um trabalho ao redor, mas o trabalho ao redor é inconveniente ou difícil. |
| 3 | Média | Funcionalidade incorreta do componente ou processo. Há um simples trabalho ao redor para o bug se for Sev. 3. |
| 4 | Menor | Erros de documentação ou corte de gravidade 3 bugs. |

### Lista de Prioridades

|  |  |  |
| --- | --- | --- |
| **ID prioritário** | **Nível de Prioridade** | **Descrição prioritária** |
| 5 | Deve corrigir | Este bug deve ser corrigido imediatamente; o produto não pode ser enviado com este bug. |
| 4 | Deve corrigir | Estes são problemas importantes que devem ser corrigidos o mais rápido possível. Seria uma vergonha para a empresa se este bug fosse enviado. |
| 3 | Corrigir quando tiver tempo | O problema deve ser corrigido dentro do tempo disponível. Se o bug não atrasar a data de envio, então conserte-o. |
| 2 | Baixa Prioridade | Não é importante (neste momento) que esses bugs sejam abordados. Corrija esses bugs depois que todos os outros bugs tiverem sido corrigidos. |
| 1 | Trivial | Aprimoramentos/ Bom ter recursos incorporados, basta estar fora do escopo atual. |

## Relatórios de bugs

A equipe de testes reconhece que o processo de emissão de relatórios de bugs é uma ferramenta de comunicação crítica dentro do processo de teste. Sem uma comunicação efetiva de informações sobre bugs e outros problemas, o processo de desenvolvimento e liberação será impactado negativamente.

O Test Lead será responsável pelo gerenciamento do processo de emissão de relatórios de bugs. Serão usadas as ferramentas e processos padrão de emissão de relatórios de bugs do teste. GForge é a ferramenta padrão de registro/ rastreamento de bugs em toda a empresa. Testes e desenvolvimento inserirão seus dados no banco de dados GForge seguindo as definições de entrada de campo definidas abaixo.

# Termos/Siglas

Os termos abaixo são usados como exemplos, por favor adicione/remova quaisquer termos relevantes para o documento.

| TERMO/SIGLA | DEFINIÇÃO |
| --- | --- |
| API | Interface do programa de aplicativos |
| BAH | Booz Allen Hamilton |
| BCP | Plano de Continuidade de Negócios |
| GATO | Teste de aceitação do cliente |
| Teste de ponta a ponta | Testa cenários de usuários e várias condições de caminho, verificando se o sistema executa e executa tarefas com precisão com o mesmo conjunto de dados do início ao fim, como pretendido. |
| ER;ES | Registros Eletrônicos; Assinaturas Eletrônicas |
| N/A | Não aplicável |
| NCI | Instituto Nacional do Câncer |
| QA | Garantia da qualidade |
| RTM | Matriz de rastreabilidade de requisitos |
| ARE | Especialista em Matéria |
| SOP | Procedimento operacional padrão |
| TBPT | Banco de Tecidos e Ferramentas patológicas |
| TBD | A ser determinado |
| TSR | Relatório de resumo do teste |

1. http://www.projectperfect.com.au/info\_requirements\_traceability.php [↑](#footnote-ref-1)