

UNIVERSIDADE DE SOROCABA

PRÓ-REITORIA ACADÊMICA

CURSO DE CIÊNCIA DA COMPUTAÇÃO

Paulo Tiago Castanho Mariano

Teste de Aceitação Automatizado usando ferramenta com interface  
do usuário

Sorocaba/SP

2012

Paulo Tiago Castanho Mariano

Teste de Aceitação Automatizado usando ferramenta com interface  
do usuário

Trabalho de Conclusão de Curso apresentado  
como exigência parcial para obtenção do  
certificado de Bacharel em Ciência da  
Computação, da Universidade de Sorocaba.

Orientador: Prof. Ms. Walter Masson

Sorocaba/SP

2012

## **Resumo**

**Palavras chave:** Teste de aceitação. Automação de teste de software. Automação de interface do usuário.

Teste de aceitação é um conceito chave para tomada de decisão em projetos de desenvolvimento de sistemas, prática que valida o comportamento do software, para confirmar se está de acordo com o que foi solicitado. Esse trabalho descreve os conceitos relacionados com o teste de aceitação, considerando que o desenvolvimento de software usa histórias de usuário e critérios de aceitação para determinar os testes. O trabalho tem como foco demonstrar como uma ferramenta, nesse caso Autoit, pode ser aproveitada para essa prática. São criados algoritmos em scripts que são interpretados pela ferramenta para ações da interface de usuário, recebendo o requisito do usuário e garantindo o comportamento do sistema que foi solicitado pelo cliente. Como forma de validação foram utilizadas duas aplicações, passando as histórias de usuário, verificando que as ações foram realizadas e emitindo um relatório do resultado obtido.

## **Abstract**

**Keywords:** Acceptance testing. Software testing automation. Requirements. User interface automation.

Acceptance testing is a key concept for decision making in systems development projects, this practice validates that the software behavior is consistent with what was requested. This paper describes the concepts related to the acceptance test, which considers software development using user stories and acceptance criteria to determine the tests. The work has focused demonstrate how the tool Autoit can be used to this practice, algorithms being created in scripts that are interpreted by the tool to the user interface actions. Receive the requirement of the user and ensures the system behavior that was requested by the client is that these scripts do. Of validation was used two applications, passing the user stories and checking that the actions out and export a report of the expected result.

## **Lista de Figuras**

Figura 1 – Correspondente entre desenvolvimento e processo de teste .....	12
Figura 2 – Requisitos solicitados diretamente dos usuários.....	15
Figura 3 – Ciclo de automação de teste direcionado a estórias .....	16
Figura 4 – Exemplo de estória de usuário .....	17
Figura 5 – Exemplo de critério de aceitação .....	18
Figura 6 – Arquivo de internacionalização para português .....	20
Figura 7 – Estória de usuário sistema simulação crédito .....	22
Figura 8 – Estória de usuário de sistema de comercio eletrônico .....	23
Figura 9 – Estória de usuário na língua inglesa .....	24
Figura 10 – Arquivo de internacionalização língua inglesa .....	24

## **Sumário**

Introdução .....	6
Capítulo 1 – Qualidade e teste de software e a definição desses termos .....	8
Capítulo 2 – O Teste de aceitação .....	11
Capitulo 3 – Estórias de usuários e critérios de aceitação .....	14
Capitulo 4 – Ferramenta Autoit e o framework Acceptanceit .....	19
Capitulo 5 – Resultados obtidos.....	22
Capítulo 6 – Conclusão .....	25
Referências .....	27
Apêndice .....	28

## Introdução

Para Bezerra (2006) um sistema de informação é uma combinação de pessoas, dados, processos, interfaces, redes de comunicação e tecnologia que interagem com o objetivo de dar suporte e melhorar o processo de negócio de uma organização empresarial com relação às informações que nela fluem. O desenvolvimento do sistema também usa essa combinação gerando complexidade, que pode ser gerenciada com auxílio de técnicas.

Nesse trabalho de conclusão de curso do bacharelado em ciência da computação, apresenta-se uma análise dos processos de geração de requisitos no desenvolvimento de software e a criação de uma estrutura de um sistema para aplicação de uma técnica em teste de software, denominada Teste de Aceitação. O trabalho prossegue demonstrando como utilizar a ferramenta Autolt3 para que o teste de aceitação seja realizado.

O tema foi escolhido devido à relevância na atualidade, para os meios acadêmicos, científicos e comerciais, de se desenvolver sistemas confiáveis e o mais rápido possível. O cenário de desenvolvimento de sistema pode estar imprevisível, onde uma funcionalidade adicionada ao sistema pode gerar erros (Chelinsk 2010). Muitos gestores parecem não estar conscientes do arsenal de práticas e ferramentas disponíveis para enfrentar esses desafios (Melnik 2009). Justifica-se, nessas ocasiões, aplicar os conceitos para uma cobertura de testes automatizados que garantam mudanças estáveis.

O objetivo do trabalho é descrever como a estrutura de script para automação de interface de usuário pode interagir com testes de aceitação automatizados, sob os critérios de aceitação das histórias de usuário como requisitos de sistemas, garantindo a funcionalidade e documentando o comportamento.

Para servir à prática dos Testes de Aceitação foi utilizada a ferramenta freeware Autoit3, para a qual serve de interpretador dos algoritmos desenvolvidos para testes de aceitação. Essa ferramenta é para automação de interface do usuário sendo usada aplicações de tecnologia web.

No primeiro capítulo o trabalho mostra o contexto em que as equipes de desenvolvimento passaram durante as décadas e quais definições formais de testes e qualidade de software existe atualmente.

No segundo capítulo descreve o teste de aceitação e as considerações do trabalho a respeito dessa técnica.

O terceiro capítulo demonstra o que são histórias de usuário e critérios de aceitação, e como estes artefatos são usados para a técnica de teste de aceitação automatizado.

O quarto capítulo descreve sobre a ferramenta de scripts para automação de interface de usuário, o Autoit, e como é usado o framework Acceptanceit desenvolvido para interpretar as histórias de usuário.

O quinto capítulo apresenta a conclusão e os resultados obtidos ao utilizar o framework para realizar ações em aplicações web, também demonstra o resultado na interpretação de histórias de usuário em língua portuguesa e inglesa.



## Capítulo 1 - Qualidade e teste de software e as definições desses termos

Durante as diversas décadas de desenvolvimento de sistemas, este foi se modificando conforme cada contexto. Bezerra (2006) define o histórico abaixo de como os sistemas eram modelados:

**Décadas de 1950/60:** os sistemas de software eram bastante simples. O desenvolvimento desse sistema era feito de forma “direto ao assunto”. Os sistemas eram significativamente mais simples e, conseqüentemente, as técnicas de modelagem também eram mais simples: era a época dos fluxogramas e diagramas de módulos.

**Década de 1970:** nessa época, computadores mais avançados e acessíveis começaram a surgir. Houve uma grande expansão do mercado computacional. Sistemas mais complexos começaram a surgir. Por conseguinte, modelos mais robustos foram propostos. Nesse período, surgem a programação estruturada e o projeto estruturado. Os autores Larry Constantine e Edward Yourdon são grandes colaboradores nessas técnicas.

**Década de 1980:** nessa fase os computadores se tornaram ainda mais avançados e baratos. Surge a necessidade por interfaces mais sofisticadas, o que originou a produção de sistemas mais complexos. A Análise Estruturada surgiu no início deste período com os trabalhos de Edward Yourdon, Peter Coad, Tom DeMarco, James Martin e Chris Gane.

**Início da Década de 1990:** esse período em que surge um novo paradigma de modelagem, a Análise Orientada a Objetos. Grandes colaboradores deste paradigma são Sally Shlaer, Stephen Mellor, Rebecca Wirfs-Brock, James Rumbaugh, Grady Booch e Ivan Jacobson.

**Fim da Década de 1990:** o paradigma da orientação a objetos atinge sua maturidade. Os conceitos de padrões de projeto, frameworks, componentes e qualidade começam a ganhar espaço.

Nas três primeiras décadas da era do computador, o principal desafio era desenvolver hardware que reduzisse o custo de processamento e armazenagem de dados. Efetivado isto, o foco voltou-se para o software, e o desafio tornou-se melhorar a qualidade de reduzir o custo de soluções baseadas em computador. (PRESSMAN, 1995).

Diante deste contexto, o presente trabalho de conclusão de curso, trata de aceitação de software, que integra os Testes Funcionais, compondo a visão atual da Qualidade de Software.

Para Molinari (2008) a engenharia de software é mostrada como uma grande ciência, uma área de construção e gerência de projetos de software. Quando olhamos mais a fundo, veremos que esta tem uma área de Qualidade. Dessa forma, a Engenharia de Software, por sua vez, tem internamente o que é chamado de Garantia de Qualidade ou QA (Quality Assurance).

Quando falamos de software, estamos falando de GQS (Garantia da Qualidade de Software) ou SQA (Software Quality Assurance).

Teste é uma área importante devido ao seu crescente destaque durante os últimos anos. O teste é uma área da Engenharia de Software e tem como objetivo aprimorar a produtividade e fornecer evidências da confiabilidade e da qualidade do software em complemento a outras atividades de garantia de qualidade ao longo do processo de desenvolvimento de software.

E Molinari (2008) descreve definições formais de qualidade conforme a seguir.

- IEEE 610.12 1990 (Glossário de Termos de Engenharia de Software) – qualidade são sistemas, componentes ou processos que seguem os requisitos e atendem a necessidade do usuário.
- ISO 9003-3-1991 (Guia de Aplicação da ISO 9001 para Desenvolvimento, Entrega e Manutenção de Software): “totalidade de características e funções de um produto ou serviço que habilitem a satisfação de necessidades implícitas ou específicas”.

Kaner (2001): “O teste de software é feito para localizar informações, decisões críticas sobre o projeto ou o produto são tomadas com base nessas informações”.

O teste é tratado diversas vezes como meio para as decisões de um projeto de software, os resultados são dados para compor as tomadas de decisão. Sendo o teste de aceitação realizado pelo cliente este tem uma relação direta entre os resultados e as decisões.

## **Capítulo 2 – O Teste de Aceitação**

Teste de aceitação determina se um sistema satisfaz o critério de aceitação e permite ao cliente determinar se o software será aceito ou não.

As decisões tomadas em um projeto de software, levam em consideração os dados recebidos em testes realizados, para a entrega de partes do sistema o teste de aceitação é fundamental para tais decisões. Segundo o Melnik (2009) a aceitação de software envolve teste de aceitação, mas esse processo é maior, sendo elaborada para cada equipe e cada contexto enfrentado. Também define teste de aceitação como o conjunto de atividades para reunir informações necessárias para tomar a decisão: “Esse software está preparado para mim (ou para meu cliente) e cumpri meu requisito?”.

Para Mayers (2004) existe uma referência direta entre o que o desenvolvedor faz durante o desenvolvimento do software e o que o testador faz durante o teste de software, representado pela figura 01. E afirma que teste de aceitação é o processo que compara o programa com os requisitos iniciais e as necessidades atuais dos usuários finais. Evidenciando, um comum acordo entre a equipe e o cliente, para decisões sobre o produto.

### The correspondence between development and testing processes.

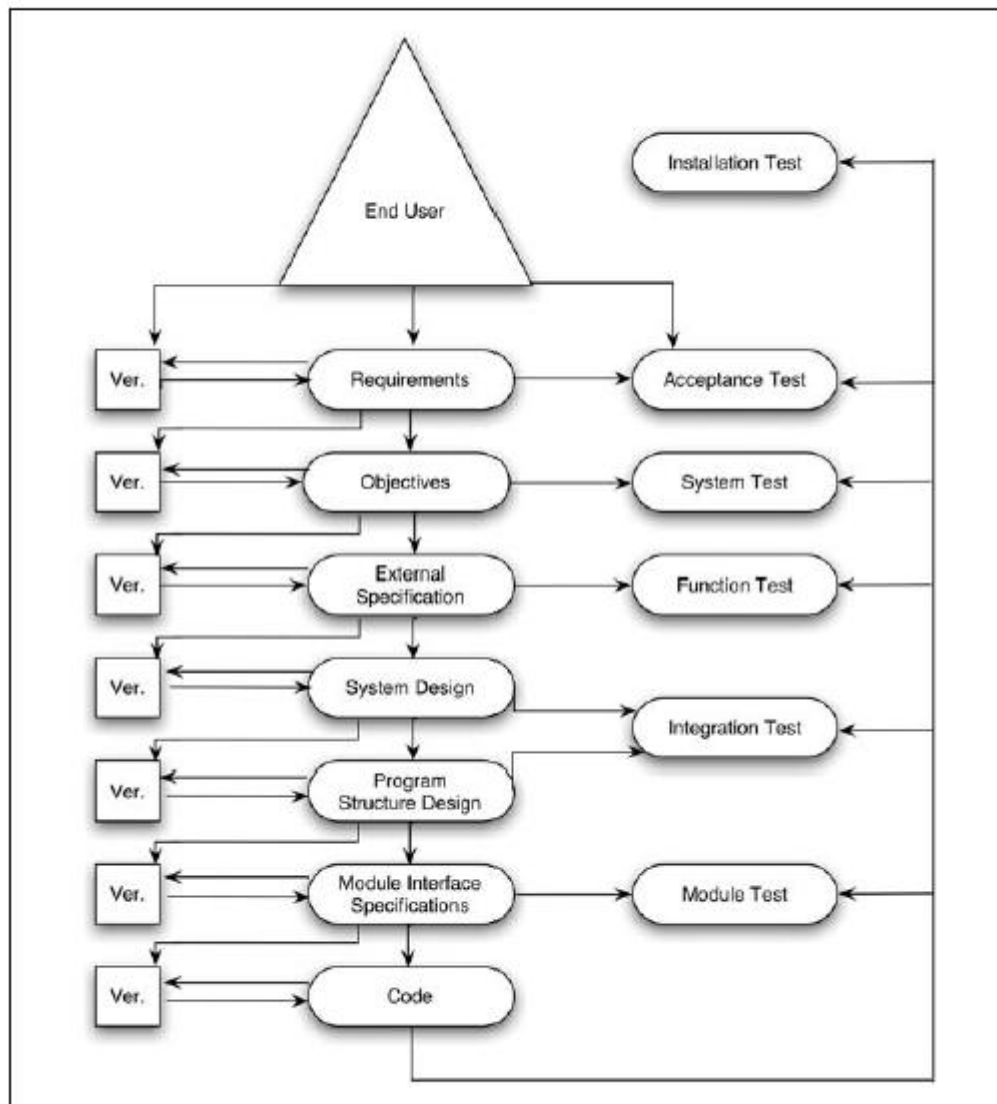


Figura 1 – Correspondente entre desenvolvimento e processo de teste. MAYERS (2004).

O teste de aceitação sendo uma decisão tomada pelo cliente (ou representante dos clientes) diante do produto apresentado pela equipe que desenvolveu, faz com que seja necessário uma reunião para o cliente apresentar o que deseja da equipe, e outra para a equipe mostre o que foi desenvolvido para o cliente. Essa reunião está muito relacionada com o contexto da equipe, do produto e dos clientes, portanto esse trabalho está direcionado a um modelo genérico, sendo que ao utilizar a ferramenta demonstrada será necessária adaptação dela.

Esse trabalho considera sistemas que tem interface com usuário com tecnologia web. Os testes automatizados abordados no trabalho foram direcionados para esse tipo de interface, aplicações web que processa dados em um servidor, mas também pode ter lógica na camada de apresentação (cliente). Portanto usando uma ferramenta de automação com interface de usuário são garantidas também as lógicas que ocorrem na camada cliente.

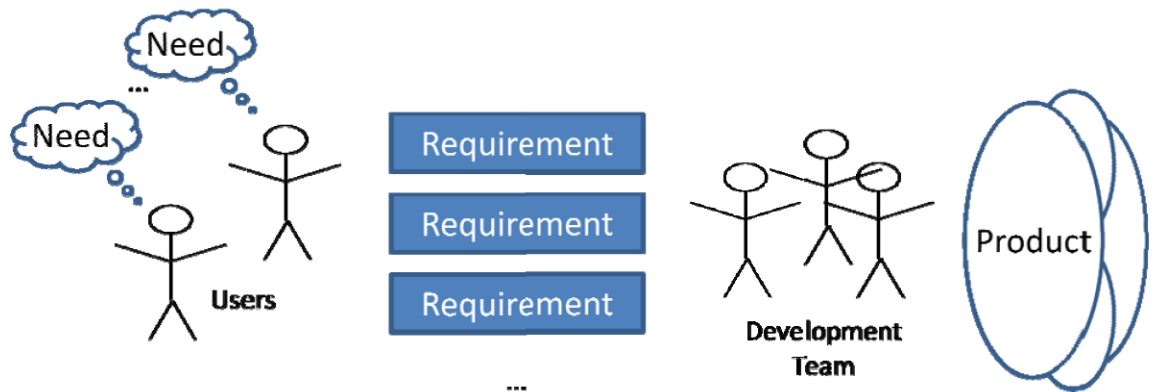
Considerando que o cliente vai documentar a solicitação de comportamentos para o sistema, esse trabalho implementa a abordagem de User Stories (estórias de usuário), sendo que para cada estória de usuário pode ter um ou mais critérios de aceitação. Esses artefatos são descrito no próximo capítulo.

## Capítulo 3 – Estórias e critérios de aceitação

Quem quer um novo software, seja para usar ou para vender, deve comunicar-se com quem vai construir o novo software. Diz Cohn (2004, pg 3) que para a execução de um projeto, depende das informações das cabeças de pessoas muito diferentes: de um lado os clientes, usuários, algumas vezes analistas ou especialistas de domínio e outros que veem o software a partir de uma empresa ou perspectiva organizacional; e do outro lado a equipe técnica que constrói o software. Para fazer isso, certifique-se de que temos um processo que nos leva a informação frequentemente e mais rápido possível. E é nesse sentido que as histórias de usuário vêm contribuir.

Segundo Chelimsky (2010), muitas pessoas usam a palavra feature e user stories de forma equivalente, mas existe uma diferença substancial. Uma feature (que vou chamar de funcionalidade) é algo que entrega valor coeso para um interessado. Uma estória é uma peça de uma funcionalidade demonstrável que não deve consumir mais de poucos dias para ser implementada.

Em Cohn (2010) Estórias de usuário é uma forma para direcionar o foco de escrever sobre as características do software (em inglês feature) e as discussões sobre elas. Uma estória de usuário é uma pequena e simples descrição de uma feature elaborada de uma perspectiva de uma pessoa que precisa de uma nova capacidade, tipicamente é um usuário ou cliente do sistema. A figura 2 abaixo, ilustra como Chelkins (2009) demonstrou as características do software que partem do usuário em direção a equipe até transformar-se em produto.



*Gathering Requirements directly from users*

Figura 2 – Solicitação de requisitos diretamente dos usuários. Adaptado de Chelkins (2009).

Os testes de aceitação visam validar o comportamento determinado na história de usuário. Mas para ter detalhes as histórias recebem critérios de aceitação, onde são determinados os passos com que o sistema deve se comportar. Sobre essa determinação que os testes de aceitação automatizados vão ser elaborados pela equipe técnica.

Portanto Cohn (2004) reafirma que é importante entender as expectativas do usuário em um projeto. Essas expectativas são capturadas, formuladas e verificadas em testes de aceitação. Portanto temos uma sequência de processos sendo feitos para que dessa forma garanta o comportamento do sistema, representada na figura 3, pegar a história, escrever os testes, automatizar testes, implementar funcionalidade.



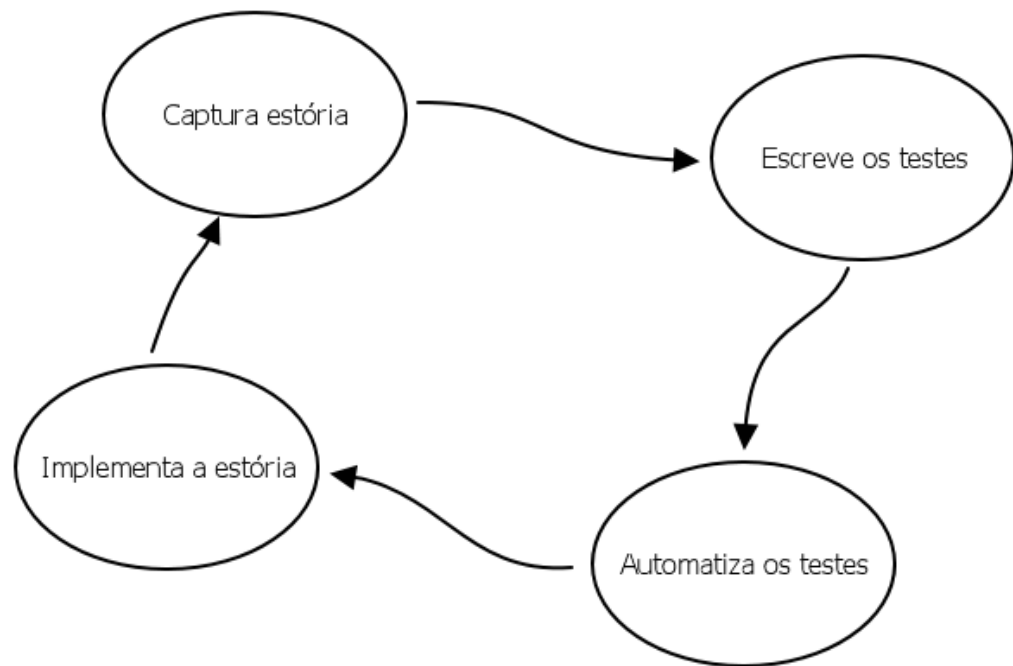


Figura 3 – Ciclo de automação de teste direcionado a histórias. Adaptado de KOSKELA (2007).

Os testes de aceitação automatizados interpretam as histórias de usuário, escritas em linguagem natural, porém algumas palavras chave são consideradas para que haja a interpretação e também auxilia na elaboração e entendimento do negócio.

A estrutura da história de usuário deve seguir o seguinte modelo: Eu como <um papel> quero que <fazer algo> de modo que <alguma razão>. Dessa forma a comunicação entre cliente e equipe de desenvolvimento consegue deixar claro o valor que o cliente necessita.

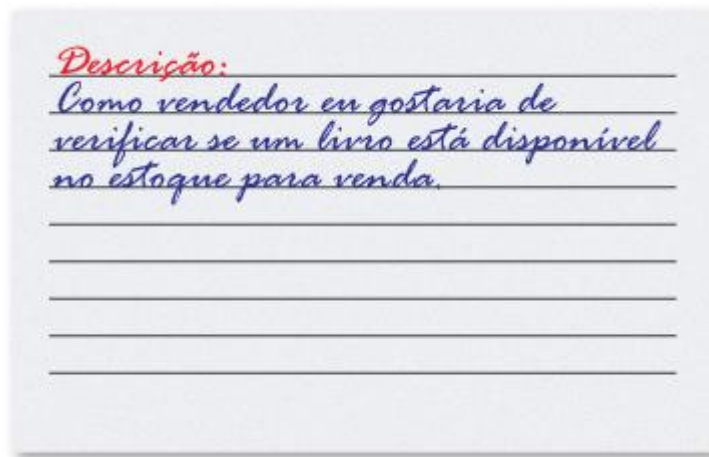


Figura 4 – Exemplo de estória de usuário. LIMA (2011).

Dentro dessa estrutura que os testes de aceitação vão ser definidos. O usuário escreve o que ele precisa do sistema e a equipe se encarrega de fazer, para a criação das estórias de usuário a equipe conversa com o cliente para saber os detalhes de cada estória, ao passar os detalhes é que são escritos os critérios de aceitação para cada estória, após esse processo é que a equipe planeja e estima quanto tempo será necessário para a execução das estórias. Em seguida da estimativa apresentada ao cliente e com a decisão firmada das estórias é que vai iniciar o desenvolvimento, são desenvolvidos os testes de aceitação automatizados seguindo os critérios de aceitação de cada estória.

Da forma que existe um modelo para criar estórias de usuário, convencionalmente se que os critérios de aceitação seguem o modelo:

Cenário:

Dado que <para um estado conhecido>

Quando <um determinado evento ocorre>

Então <resultado esperado>

Feature: code-breaker starts game

As a code-breaker

I want to start a game

So that I can break the code

Scenario: start game

Given I am not yet playing

When I start a new game

Then I should see *"Welcome to Codebreaker!"*

And I should see *"Enter guess:"*

Figura 5 – Exemplo de critério de aceitação. Adaptado de CHELIMSKY (2010).

Para o desenvolvimento dos critérios de aceitação automatizados a equipe pode optar por ferramentas, existem diversas ferramentas para teste de aceitação automatizado, por exemplo: [jbehave.org](http://jbehave.org), [cucumber](http://cucumber), [arc42.org](http://arc42.org), [fitnesse.org](http://fitnesse.org), [twist](http://twistframework.com) e o recente [Acceptanceit](http://acceptanceit.com).

Criando histórias e critérios de aceite como testes, o teste está junto com o seu requisito, ou seja, o requisito e o teste nasceram juntos e caso o requisito seja alterado teste o também será alterado. Portanto os testes não ficam desatualizados diante de mudanças nos requisitos. Para que uma história esteja pronta, o software deve ser aprovado por todos os testes de aceitação implementados anteriormente.

## Capítulo 4 – Ferramenta Autoit e framework Acceptanceit

Autoit versão 3 é uma ferramenta freeware de linguagem script de formato BASIC desenhado para automatizar tarefas de interface com usuário Windows e de uso geral. Esta ferramenta usa uma combinação de simulações em teclas, movimentos do mouse, manipulação e controle de janelas a fim de automatizar tarefas em uma forma não possível ou fiável com outras linguagens (exemplo VBScript e SendKeys). Autoit é também muito pequena, autossuficiente e roda nas versões regulares do windows até hoje, sem exigir problemas em tempo de execução. Descrição do site dos desenvolvedores.

As características do Autoit é uma lista de benefícios a quem busca automação de interface com usuário, sendo as principais: Amplo conjunto de funções; aprendizado fácil da sintaxe; simula movimentos do mouse e pressionamento de teclas; manipula janelas e processos; interage com todos os controles de janela padrão; os scripts podem ser compilados em executáveis; suporta COM e expressões regulares.

O trabalho usa os benefícios proporcionados para automação de interface de usuário para gerar testes de aceitação com a abordagem caixa preta.

Essa ferramenta é comumente usada para tarefas repetitivas de interface com usuário. Testadores que precisam fazer uma rotina diversas vezes ao dia pode usar essa ferramenta para agilizar o processo.

A opção pela ferramenta Autoit e a estrutura de testes de aceitação, será favorável quando já houver um sistema com interface web, que o usuário ou cliente queria uma garantia de que mudanças no sistema não vão afetar o comportamento descrito em histórias de usuário.

Ao efetuar mudanças no sistema será necessário mudar as histórias de usuário, que é a documentação, assim os documentos não ficam desatualizados. Ao mudar o sistema também muda o teste para ser mais eficaz ou abrangente. A equipe fará as manutenções dos testes de aceitação automatizados em script com

o editor de código do Autoit, para que após uma alteração seja feita a execução e validação do resultado.

Para as equipes que já tem um sistema em andamento e desejam aplicar técnicas para garantir a qualidade, a ferramenta pode apoiar se utilizar os conceitos dos capítulos 2 e 3 em conjunto do framework elaborado nesse trabalho.

O framework que foi elaborado em linguagem script e distribuído como código livre para a contribuição, podendo ser encontrado em <<https://github.com/ptcmariano/AcceptanceIt>>. Esse repositório está em um formato de rede social, onde pode ser visualizado por qualquer pessoa e a contribuição é encorajada aos usuários do github.

A estrutura encontra-se em dois arquivos de scripts, um para receber a história de usuário e interpretar o texto e outro arquivo para gerenciar as ações realizadas no sistema alvo do teste. Com isso ao alterar a ação a ser realizada no sistema não necessita mudança em como é interpretado as histórias.

Existe um arquivo para internacionalização das histórias de usuário. O script que está no repositório interpreta esse arquivo contendo a paridade entre o que significa ao interpretar um dado e como é escrito na história. Veja a figura 6 onde é mostrado o arquivo para Português do Brasil.

```
1 who:Eu como
2 what:Quero
3 why:Para
4 scenario:Cenario
5 given:Dado que
6 when:Quando
7 then:Entao
8 navigate:navego
9 type:E preencho
10 find:vejo
11 click:E clico
12 select:E Selecciono
13 inputtxt:o campo
14 button:botao
```

Figura 6 – Arquivo de internacionalização pra português. Elaboração própria.

Existe no framework uma interpretação das histórias de usuário onde, em cada linha é verificada a existência das palavras-chave, caso ela exista é chamado um comando de ação que está no script de ações.

Para geração de relatório foi utilizado um conjunto de funções padrão do Autoit, que permite extrair relatórios em tela ou diretamente em arquivo. Para cada execução de critério de aceite são extraídas as situações que ocorreram.

## Capítulo 5 – Resultados obtidos

O comportamento que o sistema prove pode ser definido com as histórias de usuário e critérios de aceitação. Para ter um desempenho melhor a atividade de teste de aceitação é automatizada usando uma ferramenta de automação, que nesse trabalho utiliza a abordagem de Interface de Usuário.

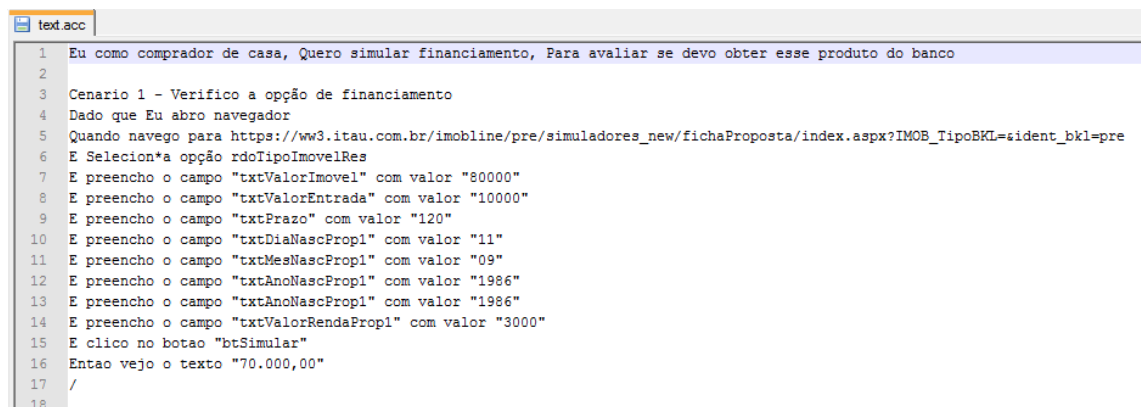
As ações que foram implementadas no script de ações, podem ser encontradas no Apêndice B, são listadas abaixo:

- Digitar em caixa de texto; com a função Action\_TextboxType
- Selecionar opção de botão radio; com a função Action\_Option
- Ação de pesquisa um texto na página; com a função Action\_SearchText
- Clicar em botão de um formulário da página; função Action\_ButtonClick
- Submeter formulário web; com a função Action\_FormSubmit

Com esses comandos é possível informar dados a um formulário e verificar se a informação retornada corresponde à esperada.

Para verificar os resultados foram utilizadas aplicações web de diferentes histórias de usuário, usando também histórias em português e em inglês.

Usando a história de usuário demonstrada na figura 7, foram aplicadas ações sobre um sistema de simulação de crédito imobiliário, disponível no portal do banco Itaú no endereço [https://ww3.itaubr.com.br/imobline/pre/simuladores\\_new/fichaProposta/index.aspx?IMOB\\_TipoBKL=&ident\\_bkl=pre](https://ww3.itaubr.com.br/imobline/pre/simuladores_new/fichaProposta/index.aspx?IMOB_TipoBKL=&ident_bkl=pre).



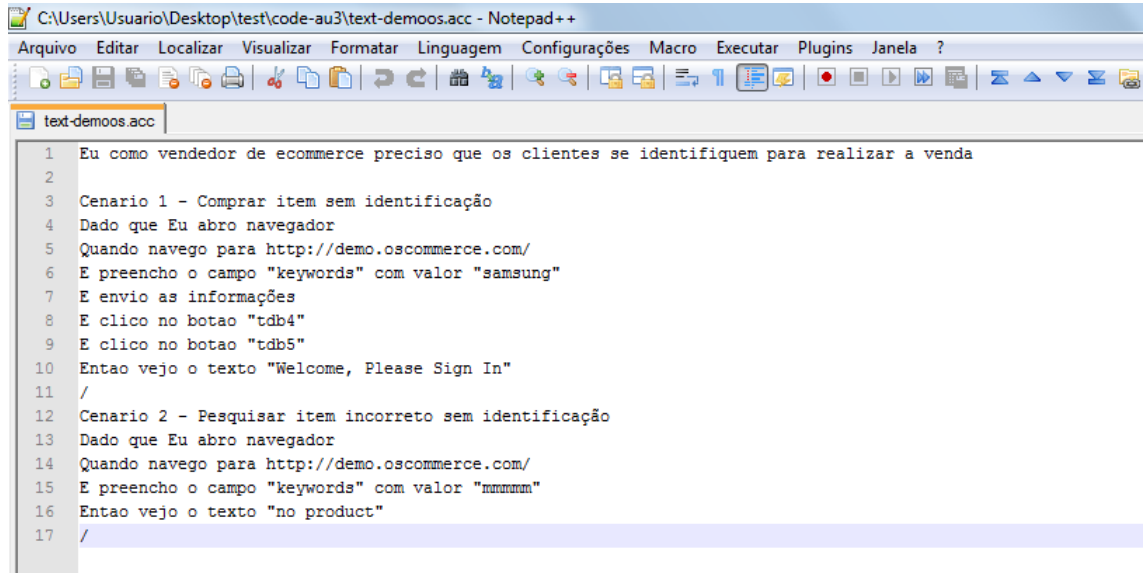
```

1  Eu como comprador de casa, Quero simular financiamento, Para avaliar se devo obter esse produto do banco
2
3  Cenario 1 - Verifico a opção de financiamento
4  Dado que Eu abro navegador
5  Quando navego para https://ww3.itaubr.com.br/imobline/pre/simuladores_new/fichaProposta/index.aspx?IMOB_TipoBKL=&ident_bkl=pre
6  E Selecciona opção rdoTipoImovelRes
7  E preencho o campo "txtValorImovel" com valor "80000"
8  E preencho o campo "txtValorEntrada" com valor "10000"
9  E preencho o campo "txtPrazo" com valor "120"
10 E preencho o campo "txtDiaNascProp1" com valor "11"
11 E preencho o campo "txtMesNascProp1" com valor "09"
12 E preencho o campo "txtAnoNascProp1" com valor "1986"
13 E preencho o campo "txtAnoNascProp1" com valor "1986"
14 E preencho o campo "txtValorRendaProp1" com valor "3000"
15 E cliço no botao "btSimular"
16 Entao vejo o texto "70.000,00"
17 /
18

```

Figura 7 – Estória de usuário sistema simulação crédito. Elaboração própria.

Usando a história de usuário demonstrada na figura 8, foram aplicadas ações sobre um sistema de comércio eletrônico, disponível no portal de endereço <<http://demo.oscommerce.com/>>.



```

1  Eu como vendedor de ecommerce preciso que os clientes se identifiquem para realizar a venda
2
3  Cenario 1 - Comprar item sem identificação
4  Dado que Eu abro navegador
5  Quando navego para http://demo.oscommerce.com/
6  E preencho o campo "keywords" com valor "samsung"
7  E envio as informações
8  E cliço no botao "tdb4"
9  E cliço no botao "tdb5"
10 Entao vejo o texto "Welcome, Please Sign In"
11 /
12 Cenario 2 - Pesquisar item incorreto sem identificação
13 Dado que Eu abro navegador
14 Quando navego para http://demo.oscommerce.com/
15 E preencho o campo "keywords" com valor "mumumum"
16 Entao vejo o texto "no product"
17 /
  
```

Figura 8 – Estória de usuário de sistema de comércio eletrônico

A internacionalização se dá através de uma relação entre cada palavra chave com uma variável no script. Relacionando a palavra chave com uma variável no script de interpretação das histórias. Em um laço é passado linha a linha do arquivo de internacionalização, interpretando qual palavra chave será enviada para a variável que depois será interpretada como ação da história de usuário.

Indicar arquivos de internacionalização diferente faz o script interpretar histórias de usuário em outra língua. Na figura 9 é a história de usuário do sistema de simulação de financiamento na língua inglesa, interpretada pelo arquivo de internacionalização mostrado na figura 10.



```

1 I as home buyer, Need simulate financiamento, For verify if can get the bank product
2
3 Scenario 1 - Verify type of financiamento
4 Given i open browser
5 When navigate para https://ww3.itaui.com.br/imobline/pre/simuladores_new/fichaProposta/index.aspx?IMOB_TipoBKL=#ident_bkl=pre
6 And fill the field "txtValorImovel" com valor "80000"
7 And fill the field "txtValorEntrada" com valor "10000"
8 And fill the field "txtPrazo" com valor "120"
9 And fill the field "txtDiaNascProp1" com valor "11"
10 And fill the field "txtMesNascProp1" com valor "09"
11 And fill the field "txtAnoNascProp1" com valor "1986"
12 And fill the field "txtAnoNascProp1" com valor "1986"
13 And fill the field "txtValorRendaProp1" com valor "3000"
14 And click on button "btSimular"
15 Then see the text "70.000,00"
16 /
17

```

Figura 9 – Estória de usuário na língua inglesa

```

1 who:I as
2 what:Need
3 why:For
4 scenario:Scenario
5 given:Given
6 when:When
7 then:Then
8 navigate:navigate
9 type:And fill
10 find:see
11 click:And click
12 selectRadio:And select
13 selectCombo:And choise
14 inputtxt:the field
15 button:button
16 submit:And send
17 option:option
18

```

Figura 10 – Arquivo de internacionalização língua inglesa

## Capítulo 6 – Conclusão

Como descritos nos capítulos, as equipes de desenvolvimento de software passaram por grandes mudanças desde o início do desenvolvimento de sistemas. O teste de software tem abordagens diferentes de acordo com o software alvo. Aceitação está relacionada em como é solicitado o software e como será cobrado o que deve ser feito.

Observa-se que as ações sobre aplicações web foram realizadas mesmo com histórias de usuário validando sistemas diferentes, sem a necessidade de alterar o código script. Portanto outra aplicação poderia ser validada também, apenas escrevendo outra história de usuário. Isso demonstra uma flexibilidade em histórias atender aplicações distintas no framework.

Outra demonstração de flexibilidade é a interpretação de histórias de usuário em línguas distintas.

A ferramenta Autoit e Acceptanceit atendeu a expectativa de interpretar histórias de usuário e executar os critérios de aceitação sobre aplicações web. Podendo garantir o comportamento do sistema através dos requisitos. Ainda com a flexibilidade de escrever as histórias de usuário na língua inglesa ou portuguesa de acordo com o arquivo de internacionalização.

Para aprofundar o tema tratado pode ser realizados outros trabalhos que vão complementar os fundamentos utilizados. Nos parágrafos abaixo alguns exemplos de próximos trabalhos.

Implementar uma aplicação escrevendo as histórias de usuário como requisitos do sistema, automatizando os testes de aceitação com o framework Acceptanceit e demonstrar os resultados obtidos como um estudo de caso.

Descrever sobre linguagem ubíqua que é aquela em que todos os envolvidos usam a mesma linguagem para a comunicação e demonstrar os benefícios dessa abordagem para documentação do sistema.

Aprimorar o framework Aceeptanceit para tratar também aplicações desktop e testar as histórias de usuário nessa outra plataforma.

Comparar ferramentas que existem no mercado, ou comparar as abordagens de testes de aceitação com unidade e testes de aceitação com interface de usuário.

## Referências

BENNETT, Jonathan. **Autolt tools**: Automation and scripting language, 1999. Disponível em: <<http://www.autoitscript.com/site/>>. Acesso em: 17 out. 2012.

BEZERRA, Eduardo. **Princípios de análise e projeto de sistemas UML**: Um guia prático para modelagem de sistemas. São Paulo, SP: Campus, 2006.

CHELIMSKY, David. **The RSpec Book Behavior-Driven Development with RSpec, Cucumber and Friends**. Raleigh, North Carolina, USA: Pragmatic Bookshelf, 2010.

COHN, Mike. **User Stories Applied**: For Agile Software Development. Boston, Massachusetts, USA: Addison-Wesley, 2004.

COHN, Mike. **Succeeding with Agile**: Software Development Using Scrum. Boston, Massachusetts, USA: Addison-Wesley, 2010.

KANER, Cem; BACH, James; PETTICHORD, Bret. **Lessons Learned in Software Testing**. Hoboken, New Jersey, USA: John Wiley & Sons, 2001.

KOSKELA, Lasse. **Practical TDD and Acceptance TDD for Java Developers**. Manning Publications, Shelter Island, New York, USA, 2007.

LIMA, Ester. **Critérios de Aceitação das User Stories**, 2011. Disponível em <<http://blog.myscrumhalf.com/2011/10/criterios-de-aceitacao-das-user-stories/>>. Acesso em 06 nov. 2012.

MELNIK, Grigori; Maszaros, Gerard; Bach, Jon. **Acceptance Test Engineering Guide**: How to decide if software is ready for you and your customers, Redmond, Washington, USA: Microsoft patterns & practices, 2009.

MYERS, Glenford. **The art of software testing**, Hoboken, New Jersey, USA: John Wiley & Sons, 2004.

PRESSMAN, Roger S.; **Engenharia de software**: tradução José Carlos Barbosa dos Santos, São Paulo, SP: Makron books, 1995.

## Apêndice

### Apêndice A – Script de interpretação das histórias de usuário

```
#cs -----

Autolt Version: 3.3.6.1
Author:      Paulo Tiago Castanho Mariano

Script Function:
    Interpreter user stories and send action on internet explorer

#ce -----
#include <File.au3>
#include <WinAPI.au3>
#include <Actions.au3>
#include <IE.au3>
#include <Debug.au3>

$arquivo = _WinAPI_GetOpenFileName("", "All Files (*.*)", ".", "text-us.acc")

;$arquivo = $CmdLine[1]

Dim $array
Dim $oIE

_FileReadToArray($arquivo[1]&"\"&$arquivo[2],$array)

_DebugSetup ("Acceptance: "&$arquivo[2], False, 5)

;read lines of internacionalization
Dim $who,$what,$why
Dim $scenario,$given,$when,$then
Dim $type,$find,$click,$selectRadio,$inputtxt,$button,$navigate,$submit,$option,$selectCombo
Local $aRecords

If Not _FileReadToArray("I18N-en.lng", $aRecords) Then
    MsgBox(4096, "Error", " Error reading log to Array >> error:" & @error)
    Exit
EndIf

For $x = 1 To $aRecords[0]

    Internationalization($aRecords[$x])
Next

;read lines of acceptance criteria
For $c = 1 To UBound($array)-1

    AnalyseLine($array[$c])

Next

Func AnalyseLine($line)

    ;See if have the user for story
    If StringInStr( $line , $who ,0) <> 0 Then
        ;here found info for create Log
    EndIf
EndFunc
```

```

        _DebugReport($line)

;See if have the propose for story
Elseif StringInStr($line, $scenario, 0) <> 0 Then

        _DebugReport(">> " & $line)

;See if need open browser
Elseif StringInStr($line, $given, 0) <> 0 Then
        $oIE = _IECreate()

;See if need navigate
Elseif StringInStr($line, $navigate, 0) <> 0 Then
        Local $arr
        $arr = StringSplit($line, " ", 1)

        _IENavigate($oIE, $arr[$arr[0]])
        _IELoadWait($oIE)

;See if have text to fill
Elseif StringInStr($line, $type, 0) <> 0 Then

        ;get element and text
        If StringInStr($line, $inputtxt, 0) <> 0 Then
                Local $arr
                $arr = StringSplit($line, "", 1)

                Action_TextboxType($oIE, $arr[2], $arr[4])
        EndIf

;See if have button to click
Elseif StringInStr($line, $click, 0) <> 0 Then

        If StringInStr($line, $button, 0) <> 0 Then
                Local $arr
                $arr = StringSplit($line, "", 1)

                Action_ButtonClick($oIE, $arr[2])
        EndIf

;See if have word Then for find anything
Elseif StringInStr($line, $then, 0) <> 0 Then

        If StringInStr($line, $find, 0) <> 0 Then
                Local $arr
                $arr = StringSplit($line, "", 1)
                _DebugReport("Result: " & Action_SearchText($oIE, $arr[2]))
        EndIf

;See if have radio button to select
Elseif StringInStr($line, $selectRadio, 0) <> 0 Then

        If StringInStr($line, $option, 0) <> 0 Then
                Local $arr
                $arr = StringSplit($line, " ", 1)

                Action_Option($oIE, $arr[$arr[0]])
        EndIf

```

```

ElseIf StringInStr($line, $selectCombo, 0) <> 0 Then

    If StringInStr($line, $option, 0) <> 0 Then
        Local $arr
        $arr = StringSplit($line, " ", 1)

        Action_Option($oIE, $arr[$arr[0]])
    EndIf

ElseIf StringInStr($line, $submit, 0) <> 0 Then
    Action_FormSubmit($oIE)

ElseIf StringInStr($line, "/", 0) Then
    If IsObj($oIE) Then
        _IEQuit($oIE)
    EndIf
EndIf

```

EndFunc

```

Func Internationalization($line18N)
    Local $arr
    $arr = StringSplit($line18N, ".:")

    Switch $arr[1]
        Case "who"
            $who = $arr[2]

        Case "why"
            $why = $arr[2]

        Case "what"
            $what = $arr[2]

        Case "scenario"
            $scenario = $arr[2]

        Case "given"
            $given = $arr[2]

        Case "when"
            $when = $arr[2]

        Case "then"
            $then = $arr[2]

        Case "type"
            $type = $arr[2]

        Case "find"
            $find = $arr[2]

        Case "click"
            $click = $arr[2]

        Case "selectRadio"
            $selectRadio = $arr[2]
    EndSwitch
EndFunc

```

```
Case "inputtxt"  
    $inputtxt = $arr[2]  
  
Case "button"  
    $button = $arr[2]  
  
Case "navigate"  
    $navigate = $arr[2]  
  
Case "submit"  
    $submit = $arr[2]  
  
Case "option"  
    $option = $arr[2]  
EndSwitch  
EndFunc
```



## Apêndice B – Script de ações a serem realizados no sistema

#cs -----

Autolt Version: 3.3.6.1

Author: Paulo.Mariano

Script Function:

Actions on the object

#ce -----

#include <IE.au3>

#include <Debug.au3>

Dim \$formNameUsed

; #FUNCTION#

=====

; Name.....: Action\_FormSubmit

; Description ...:

; Syntax.....:

; Parameters .....

; Return values ..:

; Author .....: Paulo.Mariano

; Modified.....:

; Remarks .....

; Related .....

; Link .....

; Example .....

;

;

;

;

;

=====

Func Action\_FormSubmit(\$oIE)

    \$oForm = \_IEFormGetObjByName(\$oIE, \$formNameUsed)

    ;MsgBox(0,"","Submitou: "& \$formNameUsed)

    \_IEFormSubmit(\$oForm)

EndFunc

; #FUNCTION#

=====

; Name.....: Action\_ButtonClick

; Description ...:

; Syntax.....:

; Parameters .....

; Return values ..:

; Author .....: Paulo.Mariano

; Modified.....:

; Remarks .....

; Related .....

; Link .....

; Example .....

;

;

;

;

```

;
;
=====
=====
Func Action_ButtonClick($oIE, $nObj)

    $vObj = _IEGetObjByName($oIE, $nObj)
    _IEAction($vObj, "click")
    _IELoadWait ($oIE)

EndFunc
; #FUNCTION#
=====
=====
; Name.....: Action_SearchText
; Description ...:
; Syntax.....:
; Parameters ....:
; Return values .:
; Author .....: Paulo.Mariano
; Modified.....:
; Remarks .....:
; Related .....:
; Link .....:
; Example .....:
;
;
; *
;
;
;
=====
=====
Func Action_SearchText($oIE, $text)
    _IELoadWait($oIE)

    $oElements = _IETagNameAllGetCollection ($oIE)

    For $oElement In $oElements

        If StringInStr( $oElement.innerText, $text, 0 ) <> 0 Then
            Return "Find text: " & $text
        EndIf

    Next

    Return "Not find: " & $text

EndFunc
; #FUNCTION#
=====
=====
; Name.....: Action_Option
; Description ...:
; Syntax.....:
; Parameters ....:
; Return values .:
; Author .....: Paulo.Mariano
; Modified.....:
; Remarks .....:
; Related .....:
; Link .....:
; Example .....:

```

```

*
*
*
*
*
*
*
=====
=====
Func Action_Option($oIE,$opt)

    $oForm = _IEFormGetCollection($oIE,0)

    _IEFormElementRadioSelect($oForm, $opt,"level")

EndFunc

; #FUNCTION#
=====
=====
; Name.....: Action_TextboxType
; Description ....: Click on button
; Syntax.....: Action_TextboxType($text)
; Parameters .....: $oIE, object needed to handle web page
;                  $idForm, form how find the inputs
;                  $id, name or id of input form element to type the text
;                  $text = "text", the text to type on input
;
;
; Author .....: Paulo.Mariano
; Modified.....:
; Remarks .....: Find on forms the field selected and fill with text
; Related .....: _IEFormElementSetValue, _IEFormElementGetObjByName
;
; Example .....:
;
;
; *
;
;
;
=====
=====
Func Action_TextboxType($oIE, $id, $text = "text")

    $oForms = _IEFormGetCollection($oIE)

    For $oForm In $oForms
        If _IEFormElementGetObjByName($oForm, $id) <> 0 Then
            $oQuery = _IEFormElementGetObjByName ($oForm, $id)
            $formNameUsed = $oForm.name
        EndIf
    Next

    _IEFormElementSetValue ($oQuery, $text)

EndFunc

```

**Apêndice C** – Conteúdo do arquivo de internacionalização língua portuguesa

who:Eu como

what:Quero

why:Para

scenario:Cenário

given:Dado que

when:Quando

then:Então

navigate:navego

type:E preencho

find:vejo

click:E clico

selectRadio:E Selecciono

selectCombo:E escolho

inputtxt:o campo

button:botão

submit:E envio

option:opção

**Apêndice D – Conteúdo do arquivo de internacionalização língua inglesa**

who:I as  
what:Need  
why:For  
scenario:Scenario  
given:Given  
when:When  
then:Then  
navigate:navigate  
type:And fill  
find:see  
click:And click  
selectRadio:And select  
selectCombo:And choise  
inputtxt:the field  
button:button  
submit:And send  
option:option