Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

Maiara Martins Gonçalves

Teste e Qualidade de Software

Assis

2012

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

Maiara Martins Gonçalves

Teste e Qualidade de Software

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso de Análise e Desenvolvimento de Sistemas

Orientador: Prof. Dr. Luiz Carlos Begosso
Área de Concentração: Informática.

Assis, SP
2012

FICHA CATALOGRÁFICA

GONÇALVES, Maiara Martins.

Teste e Qualidade de Software / Maiara Martins Gonçalves. Fundação Educacional do Município de Assis – FEMA – Assis, 2012.

52 p.

Orientador: Prof. Dr. Luiz Carlos Begosso.

Trabalho de Conclusão de Curso – Instituto Municipal de Ensino Superior de Assis – IMESA.

CDD 001.61

Biblioteca FEMA

Fundação Educacional do Município de Assis
Instituto Municipal de Ensino Superior de Assis
Campus "José Santilli Sobrinho"

TESTE E QUALIDADE DE SOFTWARE

MAIARA MARTINS GONÇALVES

Trabalho de Conclusão de Curso apresentado ao Instituto Municipal de Ensino Superior de Assis, como requisito do Curso Análise e Desenvolvimento de Sistemas, analisado pela seguinte comissão examinadora:

Orientador: Prof. Dr. Luiz Carlos Begosso

Analisador: Prof.Esp. Domingos de Carvalho Villela Júnior

Assis, SP

2012

DEDICATÓRIA

Dedico este trabalho aos meus pais, Uraci e Maria, a minha irmã Emanuela, meu cunhado Carlos e ao meu noivo Raphael, que a todo o momento me apoiaram, incentivaram, quando desanimei me deram forças para seguir e me mostraram que sou capaz. Oraram por mim, sorriram a cada nova conquista e sofreram a cada obstáculo, sem eles não teria conseguido, a eles agradeço por ter chego ate aqui.

AGRADECIMENTOS

Primeiramente a **Deus**, me abençoou, pois sem ele nada é possível e a **Virgem Maria** que me cobriu com seu manto sagrado;

O meu orientador Prof. Dr. Luiz Carlos Begosso, pela orientação assertiva, mostrando com paciência, agilidade e dedicação onde deveria melhorar para a conquista da formação. Muito obrigado por compartilhar seu conhecimento;

Aos meus familiares e amigos pelo companheirismo, motivação, compreensão pelos momentos que não pude estar com eles e por me apoiaram nas horas mais complicadas e difíceis;

Aos meus professores do Curso de Análise e Desenvolvimento de Sistemas da FEMA, pelos ensinamentos durante o curso;

A todos que direta e indiretamente, contribuíram para a realização deste trabalho.

RESUMO

Ultimamente têm surgido sofwares extensos e complexos que exigem mais dedicação durante todo o processo. Devido ao constante crescimento, na área de tecnologia e a busca incessante por software com qualidade, este trabalho tem por objetivo desenvolver pesquisas na área de engenharia de sotware e teste de software. É também, objetivo desse trabalho a criação de um cenário para geração de testes automatizados sobre aplicação gerada em C#.

Palavras Chave: Automação de Teste. Engenharia de Software. Qualidade.

ABSTRACT

Lately it has arise large and complicated softwares that require more dedication throughout process. Due to frequent growth in tecnology and the unceasing search for a quality software, this work has for objective developer research in software engineering and software testing. Also, the objective of this work is the creation of a scenario to generate automated tests on application generated in C #.

Keywords: Test Automation. Software Engineering. Quallity.

Lista de Ilustrações

Sumário

1. Introdução

Nos dias de hoje é difícil acreditar, mas na década de 60 o Software entrou em crise. Pouco tempo antes do ocorrido, o foco era a produção de Hardware e Sistemas Operacionais. Quando o mercado recebeu os Sistemas Operacionais de multiprogramação, com ele chegou a grande crise do Software, considerado crise devido à falta de conhecimento, experiência e carência de adequação para os programas. É interessante destacar o fato de que um dia houve falta de conhecimento de informática, em contra partida observa-se que atualmente, a maior parte das crianças e adolescentes esbanja conhecimento e prática nessa área.

Todo desenvolvimento de software, trata de um processo complexo e na época não havia conhecimento, prática, modelo a ser seguido e etapas a serem alcançadas. Com a crise se viu necessário uma solução, permitindo-se a criação do termo Engenharia de Software (MAZZOLA,2003).

Engenharia de Software é uma área ampla, com objetivos de aplicar modelos, teorias, técnicas para o desenvolvimento do software. Pode-se citar alguns dos seus focos: a Organização e Programação do Desenvolvimento, Gerenciamento, Teste e Qualidade do Software. A proposta desse projeto está circunscrita na área de Teste e Qualidade do Software (PRESSMAN,2006) .

Sendo uma das áreas de conhecimento da Engenharia de Software a Qualidade do Software tem como objetivo garantir que o software seja de acordo com necessidade e as expectativas do cliente, foca-se em garantir a melhor qualidade do produto final, sem margem de erros, tanto na parte funcional quanto em atender exatamente a regra de negócio proposta.

Dantas (2009) ressalta que o teste de software se tornou incalculavelmente importante, devido ao resultado eficaz, aumento da qualidade do software, evidenciando os erros e possibilitando a correção antes de chegar ao consumidor final.

Ao definirmos a qualidade de software, podemos definir automação dos processos de teste, evidenciando se o software está ou não atingindo a qualidade necessária. A margem de erro humano é maior se comparado com o sistema. Vemos hoje em dia, que boa parte das empresas efetua manualmente testes básicos como validação de data, campo numérico ou caractere, correndo riscos de cair em rotina e passar despercebido. Para efetuar a automação, o desenvolvedor fará a rotina apenas uma vez, depois esse processo gravado não será mais necessário.

Pensando em facilitar o teste, diminuir o tempo e maximizar a qualidade, é que se propõe o presente projeto de Automação de Teste com fácil usabilidade. O homem irá criar um cenário de teste, gravar uma rotina e o sistema fará exatamente o mesmo processo, quando for executado.

1.1 Objetivo

Este trabalho de conclusão de curso tem como objetivo pesquisar sobre engenharia de software, sua história, evolução, importância, aplicação. Além disso, estudar métodos de teste e qualidade e a partir dos estudos realizados, criar um cenário de cadastro de fornecedor e aplicar um método de teste que será submetidos ao cenário. O cenário de teste será modelado, de acordo com as validações necessárias da aplicação.

1.2 Justificativa

A justificativa por esse tema é pelo grande interesse por parte dos desenvolvedores na área de Teste e Qualidade do Software, devido ao aumento da demanda e complexidade dos softwares desenvolvidos. O Teste e Qualidade têm acompanhado esse aumento, não na mesma proporção e tempo, mas minimizando ou até mesmo evitando surpresas indesejáveis. Quando um cliente busca um sistema para sua empresa, sempre esperam que o software seja fácil de usar e que traga solução rápida e eficaz, caso software seja mal desenvolvido irá resultar em atraso na entrega, não seja de acordo com o esperado, ocasionando em um grande desconforto ao desenvolvedor e ao cliente.

1.3 Motivação

A motivação para a escolha do tema surgiu pela importância e carência dessa área. Produzir e manter o software dentro de custos adequados é essencial para o funcionamento da economia nacional e internacional (SOMMERVILLE,2007).
O mundo é movido por tecnologia, computadores, software e a tendência é aumentar ainda mais essa dependência dessa ferramenta. Ao pensarmos o quão importante é um software, é inevitável que a qualidade dele se torne ainda mais indispensável, por esse motivo esse projeto aborda a área de Qualidade e Teste do Software. O interesse é pesquisar métodos de teste e seguimento de qualidade de software, usar o conhecimento para agregar e enriquecer aos métodos de desenvolvimento, evidenciando o erro antes que chegue ao produto final.

1.4 Perspectivas de Contribuição

As perspectivas de contribuição desse projeto são pesquisar e evidenciar métodos de teste e qualidade da engenharia de software, sua importância ao processo de entrega do produto, minimizando esforços e margem de erro, maximizando a qualidade. Tornar mais dinâmico o teste, suprindo a carência do conhecimento do homem. Ao gravarmos uma rotina, o analista que irá apenas executá-la não precisa saber exatamente a regra do negócio, pois a automação estará programada passo a passo, com o fluxo correto.

1.5 Metodologia de Pesquisa

Para a condução desse trabalho de conclusão de curso, serão utilizados livros, artigos e sites especializados na área de Engenharia, Testes e Qualidade de Software. Para a realização do experimento será utilizada a tecnologia de automação de teste, ferramenta Visual Studio e uma aplicação de cadastro de fornecedor desenvolvido em C#.

1.6 Estrutura do Trabalho

Este trabalho está dividido em cinco capítulos. O primeiro capítulo, esta Introdução, são apresentados os objetivos, justificativa e motivação para a execução da presente pesquisa.

No segundo capítulo, apresentam os conceitos, aplicações e objetivos da engenharia de software.

No terceiro capítulo, discutido os métodos, modelos e a importância do Teste e Qualidade de software.

No quarto capítulo, apresentam-se informações sobre automação de teste, a criação do cenário, documentação e gravação da automação do processo de teste.

Finalmente, no quinto capítulo, apresentamos as conclusões reflexões para trabalhos futuros.

2. Engenharia de Software

Engenharia de software trata-se de um ramo da engenharia, cujo foco é desenvolvimento de software, de acordo com as expectativas esperadas, dentro dos custos adequados e desenvolvimento de software de alta qualidade.

Com seu surgimento, resultou em grandes soluções, para projetos que apresentavam algumas vezes anos de atraso, software que não atendia a necessidade do cliente, não era de acordo com o acordado e software de baixa confiabilidade.

Disciplina relacionada a todos aspectos de produção de software, desde estágios iniciais de especificação de sistema ate sua manutenção, depois que este entrar em operação (SOMERVILLE, 2007).

Nesse capitulo são abordadas as definições sobre Engenharia de Software. Tendo como objetivo, sua história, conceitos de engenharia, software e suas áreas de conhecimento.

2.1 História

A evolução dos sistemas computadorizados tem origem nos anos 40, devido à carência de conhecimento e prática, o foco e investimentos na época eram maior parte concentrados em desenvolvimento e evolução do hardware. Nos anos 50 já havia amadurecimento, conhecimento e prática na criação de hardware, a preocupações e atenção se voltaram para desenvolvimento dos sistemas operacionais, surgindo as primeiras evoluções sobre os sistemas, como as programações de alto nível como FORTRAN e COBOL. Cada vez mais o usuário passou a se preocupar com o funcionamento do software deixando um pouco de lado a parte externa o hardware.

Os anos 60 destacam-se pelo surgimento dos sistemas operacionais de multiprogramação, havendo considerável crescimento na necessidade de contar com um sistema computacional e constante quedas de preço do hardware. Ocasionando uma grande demanda ao desenvolvimento de grandes sistemas em

substituição a pequenos programas conhecido ate então. Com essa demanda desenvolvedores e contratantes se depararam com um grande problema conhecido como ‘crise do software’, devido à falta de conhecimento, prática e a não adequação dos métodos de desenvolvimento existentes, permitindo a criação do termo “Engenharia de Software” (MAZZOLA,2003).

2.2 Engenharia

Engenharia é a ciência junto a esforços para o empreendimento de resultados inovadores, nas áreas especificas, contando com amplo conhecimento e calculando os impactos que possa ser ocasionado (COUTINHO, 2010).

Sua existência vem desde a época dos primórdios, que necessitava de objetos para complementar e auxiliar em suas atividades no cotidiano, como as pedras pontiagudas usadas na época para caça, tornando a engenharia indispensável para sobrevivência e evolução humana (SOMERVILLE, 2007).

Trata-se de uma abrangente ciência que envolve vários ramos, específicos nos campos de aplicação e varias opções de tecnologia. Reúnem técnica e arte de fazer engenhocas, utilizar descobertas para criação de maquinas e produtos úteis (EUCLIDES, 2010).

2.3 Software

Software é uma sequência de comandos, a serem interpretadas por um processador ou máquina virtual, realizando tarefas planejadas de acordo com o comportamento desejado para qual o software foi projetado.

Um conjunto de instruções que, quando executadas, produzem a função e o desempenho desejados, estrutura de dados que permitam que as informações relativas ao problema a resolver sejam manipuladas adequadamente e a documentação necessária para um melhor entendimento de sua operação e uso. (PRESSMAN, 1995).

Na Engenharia de Software, o software é visto como um produto com ou sem fins lucrativos, que não inclui apenas o programa para computador e sim todas as

documentações, parametrizações e configurações necessária para que o software funcione adequadamente.

O surgimento do desenvolvimento do software era considerado como uma arte, sem utilizar métodos, modelos, sem especificação e planejamento. Ao constatarem que no desenvolvimento de software, poderia resultar em inúmeros problemas, que influenciam diretamente na qualidade do produto, se viu necessário um amadurecimento da visão desenvolvimento, se tornando um processo com pré-requisitos para chegar ao desenvolvimento propriamente dito. Tornando crucial o teste do software para uma entrega de qualidade, encontrando antecipadamente e possibilitando a correção, vendo que os usuários muitas vezes não tem conhecimento para detectar e corrigir eventuais erros (SOMMERVILLE, 2007).

2.4 Áreas de Conhecimento

Engenharia de software é união de um conjunto de atividade e processo, aplicados a informatização. Seu amadurecimento, atualização, definição, manutenção e melhorias resultam em um produto final. Atividades que se completam de acordo com a evolução do processo, relacionadas e indispensável para um final com sucesso, estabelecendo regras, padrões, ordem e modelos no desenvolvimento de sistemas computacionais. Devido o constante aumento da demanda de software, tem dificultado ainda mais a administração do processo (PRESSMAN, 2006).

A seguir, são citadas algumas atividades que compõe o processo da Engenharia de Software:

- Levantamento de Requisito

É a identificação, análise, negociação, validação, especificação do que se faz necessário e documentação conforme acordado. Entendimento do que o cliente deseja e do que realmente é necessário, verificando omissões, conflitos e ambiguidade, expondo soluções mais ágil e eficaz possível. Explorar os dois extremos, do que há possibilidade fazer e o extremo da necessidade do cliente para criação do escopo.

- Estimativa hora

Trata-se de um processo complexo, grandes dados a serem coletadas e inúmeras surpresas indesejáveis, que pode acontecer durante o projeto acarretando em alteração na estimativa de horas. O básico a ser medido seria o tamanho do software a ser desenvolvido, os esforços que serão necessários para projeto medido em quantidade de recursos (homens) + horas gasta, a qualidade do software, considerando que o erro pode ser evidenciado em qualquer processo do projeto, não propriamente dito nos testes (GARCIA, 2009)

- Planejamento

Processo responsável por quantificar o tempo e o orçamento que o projeto custará. Agendando antecipadamente, possibilitando o gestor de projeto acompanhar o processo de evolução, se esta ou não de acordo com o planejado (FERNANDES, 2004).

- Especificação

Processo técnico do levantamento de requisitos, onde é colocado em linguagem técnica passo a passo, do que será realizado no desenvolvimento do software.

- Desenvolvimento

Criação, fabricação do software de acordo com a especificação.

- Teste e Qualidade

Validar que o desenvolvimento esteja de acordo com a especificação, atendendo todos os requisitos, layout, validações, parâmetros e regras de negócio (FERNANDES, 2004).

- Manutenção

Se software não estiver de acordo com o que foi especificado, se faz necessária manutenção e validação para que atenda exatamente o que fora proposto e

acordado (FERNANDES, 2004). A figura 1 ilustra as sete áreas que compões a Engenharia de Software.

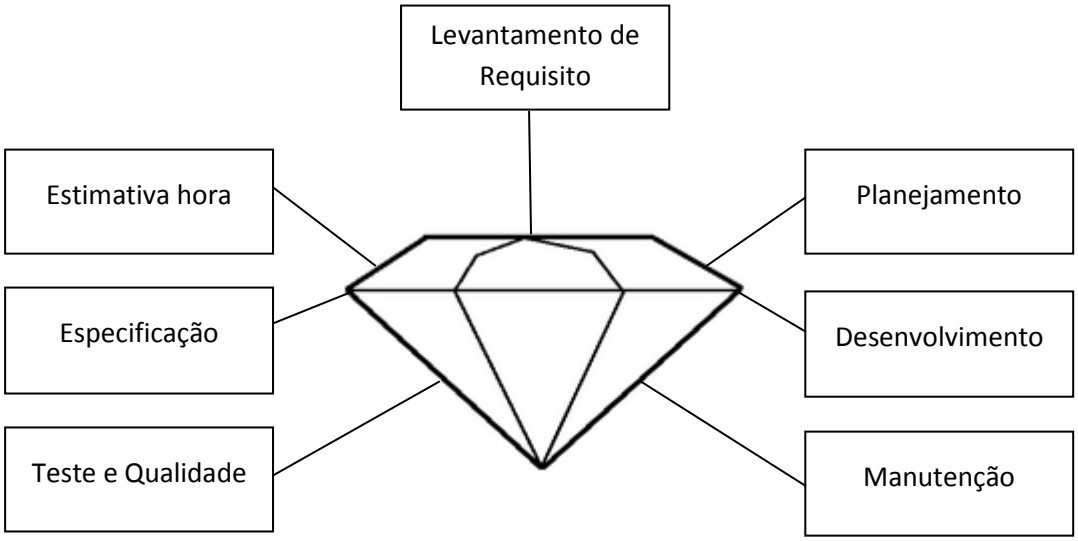


Figura 1 - Áreas da Engenharia de Software

3. Teste e Qualidade de Software

Sendo uma das áreas de conhecimento da Engenharia de Software, o Teste e Qualidade de Software têm se aprimorando significativamente nos últimos 15 anos, tornando indispensável para entrega de software de qualidade, confiável, íntegro e que atenda as necessidades e expectativas do cliente. Aprimorando constantemente, evidenciando erros antecipadamente, adotando novas técnicas e métodos, evitando cada vez mais, surpresas indesejáveis e garantindo a qualidade desde o planejamento até manutenção, depois do produto entregue.

Nesse capítulo, são abordadas definições sobre Qualidade de Software, requisitos necessários para que o software seja de qualidade, tipos e métodos para realização de testes dos softwares.

3.1 Software de qualidade

Embora tenha aumentado constantemente o interesse na área de teste e qualidade, por parte das empresas, ainda há uma porcentagem muito grande de desenvolvedores que desenvolvem sem nenhuma preocupação com a qualidade e consequentemente surgem grandes conflitos com prazos, manutenção, metas, orçamento e entrega.

McCALL (1997) organiza os critérios de qualidade de um software em três pontos. A figura 2 é ilustra tal situação:

Revisão: Capacidade do produto de ser modificado e evoluído (Manutenbilidade, flexibilidade, estabilidade).

Figura 2 Critério Organização McCall.

Transição: Adaptabilidade a novos e diferentes ambientes (Portabilidade, reusabilidade, interoperabilidade).

Operação: característica relativa ao uso do produto (Correção, confiabilidade, eficiência, integridade e usabilidade).

Podemos considerar um software com qualidade aqueles que atendam à: estar em conformidade com os requisitos solicitados e especificados; ser íntegro, confiável e com característica de fácil utilização. Além da parte interna do desenvolvimento, que se faz necessária a modularidade, legibilidade e portabilidade.

Conformidade com os requisitos funcionais e de desempenho explicitamente declarados, padrões de desenvolvimento explicitamente documentados e características implícitas, que são esperadas em todo software desenvolvido profissionalmente (PRESSMAN, 2002).

Segundo ISO 9000 (2000), qualidade é o grau de um conjunto de características inerentes a um produto, processo ou sistema cumpre os requisitos inicialmente estipulados para estes.

Para garantia de um software de qualidade, inúmeros testes, revisões e inspeções durante o processo do desenvolvimento devem estar envolvidos. É indispensável que todo desenvolvimento tenha uma série de documentações, para que possa assegurar que o desenvolvido seja de acordo com o que foi definido.

Cada vez mais empresas têm optado por realizar os testes, através de profissionais especializados, em atividade paralela ao desenvolvimento diminuindo o prazo de entrega e minimizando o custo, que boa parte era gerada pela manutenção.

Nos anos 60 e 70, a qualidade de software era focada nas linhas de código e tempo de execução. Na década de 80 toda qualidade do projeto estava voltada para a qualificação dos usuários e apenas nos anos 90 é que os testes e a qualidade se toraram mais formais, com normas ISO 9000-3 e 9126 (PRESSMAN, 2006).

Até pouco tempo atrás, os testes eram realizados pelo próprio desenvolvedor, conhecido como teste unitário, esse teste não permitia que todos os erros fossem detectados, e só evidenciavam quando já tinham sido entregues, gerando grande desconforto, além altos custos de manutenção, o que resultou atualmente na grande demanda e procura por pessoas especializadas na área de teste.

3.2 Teste de Software

Teste de software diz respeito a comparação do comportamento de um software com o comportamento previsto na documentação. De um modo geral, apenas o teste pode garantir se o produto desejado foi obtido, após a fase de implementação. Os testes de software contribuem para validação do levantamento de requisitos. Ou seja, ele confronta o objeto especificado com o produto desenvolvido. Se, a qualquer momento, for constatada a necessidade de alterar algo no escopo do projeto, será repetido todo processo de desenvolvimento: levantamento de requisito, especificação, desenvolvimento e teste.

Testar é percorrer o software, com o objetivo de encontrar erros. Todos os testes devem ser relacionados diretamente ao levantamento requisito do cliente.

Durante as primeiras atividades de engenharia de software, o engenheiro tenta construir um software a partir de um conceito abstrato até um produto tangível. Depois vem o teste. O engenheiro cria uma série de casos de teste, que são destinados a demolir o software que foi construído (PRESSMAN, 1995).

Existem vários métodos de teste a serem realizados. A seguir apresentam-se alguns deles:

- **Teste de componente (ou unidade):** também conhecido como teste unitário, onde são realizados os testes individuais. Testa-se toda interface das classes, a menor parte do código, garantindo que operem corretamente e com qualidade (SILVA, 2008). É o primeiro teste a ser realizado, normalmente é realizado pelo desenvolvedor. Este profissional testa a menor funcionalidade existente no software, separa uma parte do código e analisa suas funcionalidades. São verificados: se o resultado está de acordo com o esperado, se possui falhas no funcionamento das partes separadas e também do funcionamento independente.
- **Teste de integração:** Teste frequentemente aplicado após os testes unitários. Como os componentes são construídos separadamente, ao serem unidos se faz necessária a verificação se eles interagem. Normalmente não é

analisado o escopo do projeto. É a validação se as uniões das partes, que foram separadas anteriormente para o teste de unidade, se integram corretamente. O erro mais comum encontrado durante os testes de integração são os erros de interface, normalmente devido a incompatibilidade de interface entre os módulos que devem interagir.

- **Teste de Sistema:** é realizada a validação para sabe se o sistema atende ou não os requisitos funcionais e não funcionais. Dependendo do porte do sistema, esse processo pode ser dividido em várias etapas, tratando cada caso individualmente nos mínimos detalhes. Esse teste analisa e valida se todos os elementos que constituem o sistema estão corretos, não apenas o software, mas o hardware a ele integrado. Para agilizar o processo de identificação do local do erro, se faz necessário que o desenvolvedor faça tratativas de erros, que informem quando um determinado dispositivo de entrada ou saída está com problemas.
- **Testes Funcionais:** diz respeito à investigação se o software funciona de acordo como tem que funcionar, usando a especificação como geração de casos de teste. (SAUVÉ, 2003). Avalia o comportamento da aplicação e verifica se está conforme a sua documentação, se a implementação das regras de negócio foram desenvolvidas.
- **Teste de Performance:** avalia a capacidade, velocidade, estabilidade e disponibilidade do software sob uma determinada carga de trabalho. Pode ser analisada seu desempenho, por exemplo, com a alta quantidade simultânea de conexões, quantidade excessiva de dados inseridos simultaneamente, com intenção de causar problemas com memória, processamento ou espaço de disco.
- **Teste de Regressão:** no final dos testes, é feito um teste no software geral, abertura de telas, validações do tipo de cada campo e regras de negócio. Analisa se alguma alteração realizada gerou inconsistência no software, verifica se algumas das funções testadas e aprovadas anteriormente permanecem em funcionamento.
- **Teste de Aceitação:** antes da implantação do software em ambiente de produção no cliente, o sistema será testado e validado se está apto a

executar as funções que se propôs. Nessa fase, observa-se se está de acordo com o resultado esperado pelo cliente, deverá ser utilizado na base de homologação com dados reais do cliente, normalmente sob aprovação do usuário final.

Existem vários processos dentro de um projeto de desenvolvimento de software, a seguir apresentam-se algumas etapas a serem realizadas.

3.2.1 Entendimento Requisitos e Especificação

Durante essa etapa, é realizado o estudo, entendimento das necessidades do cliente expectativa do comportamento do software, o que foi especificado e o que foi desenvolvido.

Ambos os processos devem estar em comum acordo, levantamento de requisitos será tratado a regra de negócio, a especificação será como tecnicamente é para ser desenvolvido e o software como foi desenvolvido. A figura 3 ilustra como deve estar o Entendimento, Requisito e Especificação.

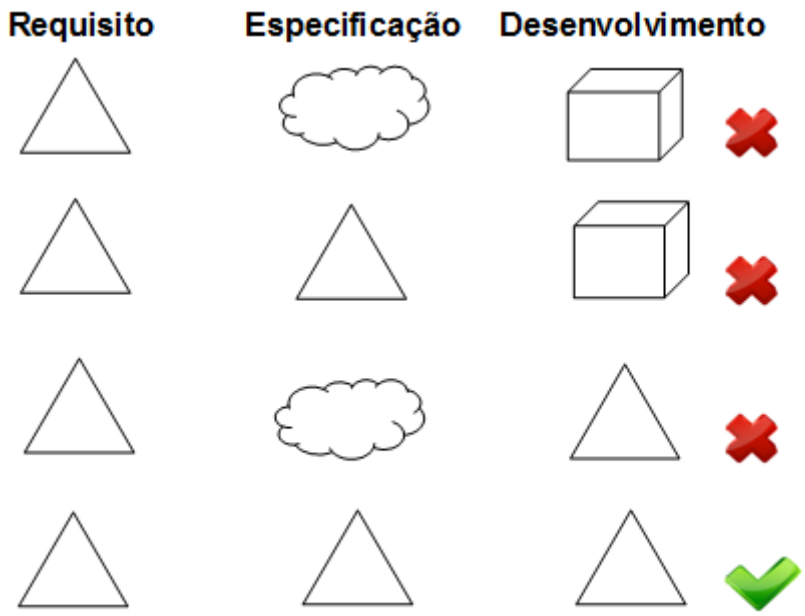


Figura 3 Requisito, Especificação e Desenvolvimento.

3.2.2 Criação do Plano de Teste

Elaboração documento de teste, estratégias, validações de acordo com a regra de negocio, criando situações que simule o cotidiano da produção. Normalmente a elaboração e a execução dos testes são realizadas simultaneamente, evidenciando com captura das telas e o passo a passo descrito.

3.2.3 Preparação e Execução

Consiste em preparar o ambiente que será realizado os testes (hardware e software), possibilitando a execução dos testes.

Executar os testes de acordo com o planejado e necessidade de cada projeto. Respeitando sempre testes básicos como validação datas, tipo dos campos, opções de pesquisas, preenchimento de campo inválido, etc.

3.2.4 Entrega

Depois de feito o teste e documentação, caso o software esteja de acordo com a especificação e o levantamento de requisito, será encaminhado para finalização e entrega do projeto. Caso seja evidenciado algum erro, o software será encaminhado à manutenção para correção.

3.3.5 Manutenção do Software

Se constatado erro no software, será avaliado se o erro foi no levantamento de requisito, especificação ou desenvolvimento. Ao evidenciar que etapa foi responsável, será refeito o planejamento e repetir todas as etapas novamente.

4. Automação de Teste

O teste investiga, detecta e evidencia o máximo possível de erros, porém não é possível assegurar que no software não exista erro. Para assegurar total qualidade seria necessário testar todas as entradas válidas possíveis, que normalmente é impossível devido à alta quantidade de dados a ser testada.

Para alguns softwares, preparar a bateria de testes é simples e rápido, quando se trata de aplicações pequenas, com poucas telas e validações. Já os softwares maiores, complexos e integrados exigem maior conhecimento da regra de negócio, demanda maior tempo dedicado aos testes, conforme aumenta a possibilidade de erro, consequentemente aumenta a quantidade de horas com testes.

Grande parte das empresas efetuam manualmente testes básicos (inclusão, alteração, validações de datas, tipo dos campos, etc.). Tais testes são cansativos e de difícil repetição, se fizermos esses testes básicos automaticamente, teremos mais tempo para dedicar aos testes mais complexos.

Além do tempo gasto durante esses testes básicos, há uma grande probabilidade de passar despercebida alguma validação, entrando em rotina. Feita a automação de testes, será apenas executado a gravação e a própria automação certificará essas validações. A automação facilita os testes e aumenta a qualidade do software.

Testes automatizados são feitos com auxílio de uma ferramenta apropriada, a princípio demanda maior tempo, devido à implementação e documentação dos testes, mas fornecem maior segurança na manutenção, e está disponível a qualquer momento para execução.

Entretanto, é necessário verificar quando realmente vale a pena automatizar o teste. Nem sempre é vantajoso, pois se a interface do software sofrer alteração num futuro próximo, qualquer automação realizada nessa interface se tornará inútil após a alteração.

Nesse capítulo, são abordadas documentações de cenário de teste. Foi elaborado um passo a passo de como é realizado a automação de teste usando ferramenta Coded UI Test Visual Studio.

4.1 Documentação Cenário de Teste

Cenário de teste é o passo a passo a ser seguido durante a etapa de teste, escrito detalhadamente para que possa realizar a automação. Quem escreve a documentação do cenário de teste é necessário ter conhecimento das regras de negócio, conhecer as funcionalidades requeridas para o software. Esse conhecimento já não é tão importante para a pessoa que realiza a gravação da automação, pois será necessário apenas simular o passo a passo da documentação elaborada do cenário de teste. Não é recomendado que seja a mesma pessoa que escreve o cenário de teste e que grava a automação, para evitar a rotina e que passe despercebido alguma validação. O analista irá criar um cenário de teste, outro analista irá gravar uma rotina e o sistema fará o processo de teste quando for executado.

O layout do cenário de teste varia de acordo com cada empresa, assim como especificação, é indispensável que esteja descrito o passo a passo a ser seguido na automação, que seja claro e se possível evidenciado com telas para facilitar o entendimento.

É importante que para a gravação de teste, tenha disponível uma base congelada, ou seja uma base que não possua movimentações para evitar conflitos na execução da automação. Ao gravar uma inclusão e exclusão de um cadastro, e se houver movimentação na base de dados, corre-se o risco de que a chave primária já tenha sido cadastrada e ao executar a automação, a inclusão não será realizada.

4.2 Automações de Teste

Ao realizarmos testes durante o desenvolvimento de software adicionamos valor ao produto, uma vez que o teste corretamente executado tende a descobrir defeitos, que devem ser corrigidos, aumentando assim a qualidade e confiabilidade de um sistema (Pressman 2006).

Automação de teste agiliza o processo de teste, tornando mais dinâmico e eficaz, assim que automatizado diminui o tempo gasto em testes básicos, maximizará a mão de obra disponível para testes mais minuciosos e demorados.

A quantidade de ferramentas para automação de teste tem aumentado significativamente, devido ao interesse pelos desenvolvedores e empresas em melhorar a qualidade de seus produtos. Testes automatizados tendem a reduzir as falhas introduzidas pela intervenção humana.

A automação de teste, realizada nesse trabalho, utiliza ferramenta Visual Studio (Ambiente de Desenvolvimento Integrado) da Microsoft, de desenvolvimento de software em diversas linguagens (VB, C, C++, C#, J#, etc.). Foi utilizada a Versão 2010 Ultimate. Esta funcionalidade de teste também está disponível na versão Premium 2010 e Teste Professional.

Abaixo, encontra-se o passo a passo da automação de teste do processo de inclusão de um cadastro de fornecedor, conforme documentação apresentada no Anexo A.

4.2.1 Criação Projeto e Método de Teste

- Criar projeto de teste: Visual Studio> New Project > Visual C# >Teste Project

Definir o nome e local a ser salvo o projeto.

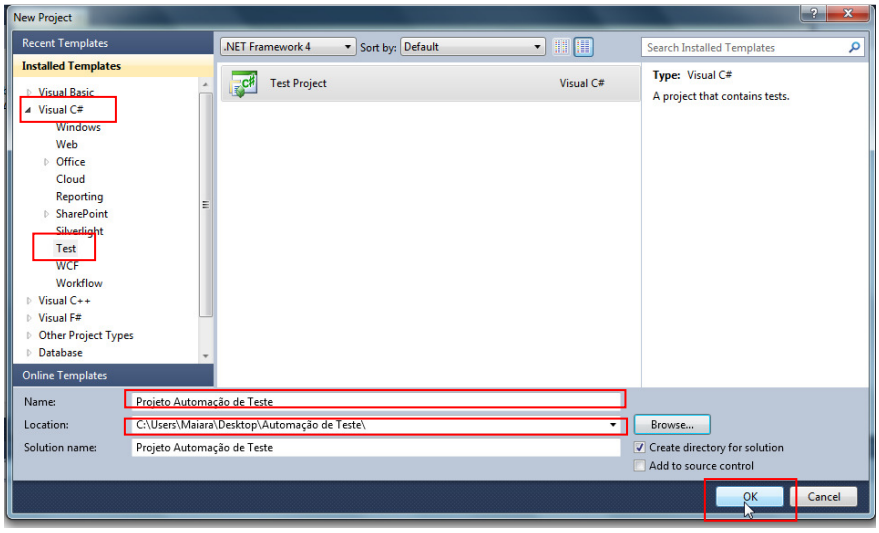


Figura 4 Criação Projeto

- Clique no Menu: Test > New Test

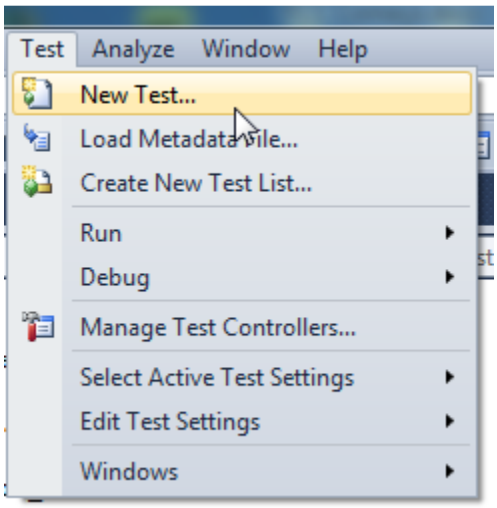


Figura 5 Criação Teste

Como o teste deste trabalho é do tipo interface, selecionar a opção “Coded UI Teste”.

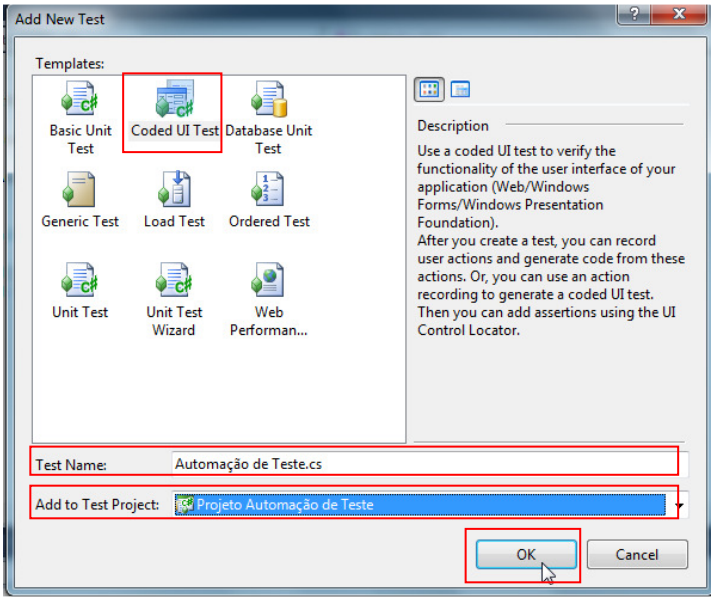


Figura 6 Nomeação de teste

O campo “Test Name” nomeia o teste que será gerado e o campo “Add to Test Project” escolhe o projeto que será incluso o teste. Clicar em “OK”.

Automaticamente abrirá a janela como abaixo, habilitando “Record action, Edit UI map or add assertions”, iniciar a gravação dos testes, manter essa opção e clicar em “OK”.

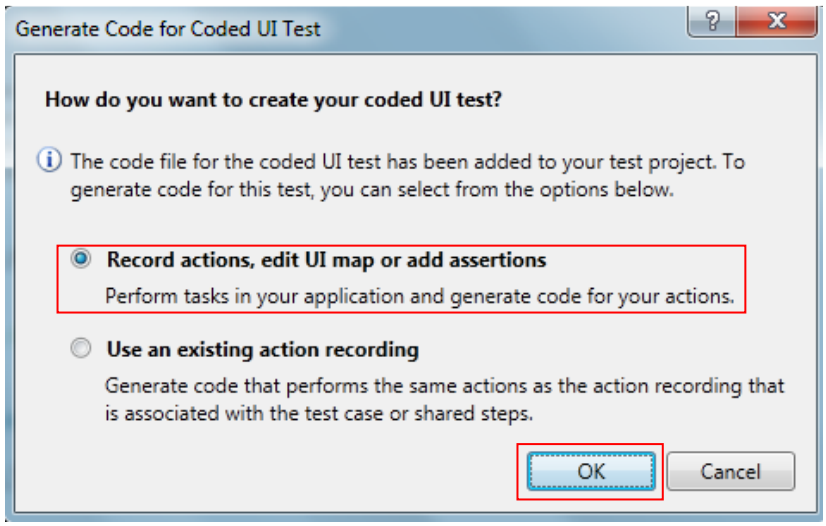


Figura 7 Generate Code for Coded UI Test

- A seguir aparecerá a ferramenta Coded UI Test Builder para administrar os testes.

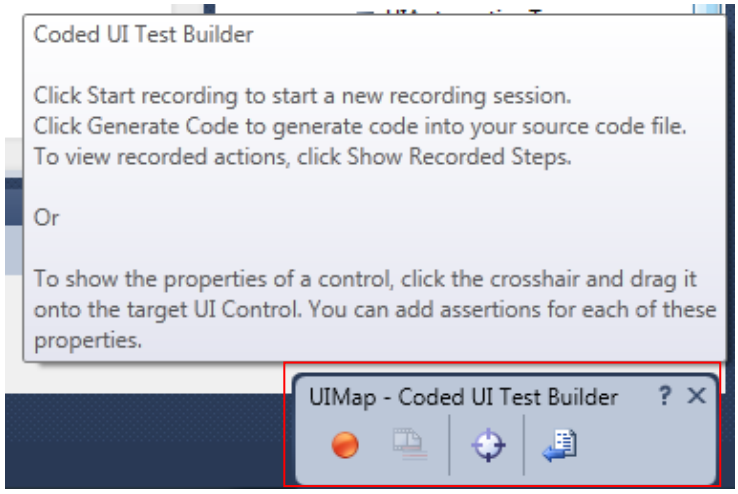


Figura 8 Coded UI Test Builder

Se em algum momento a aplicação for fechada a ferramenta administrativa de teste poderá reabri-lá:

Teste View> CodedUITestMethod1 (Duplo clique):

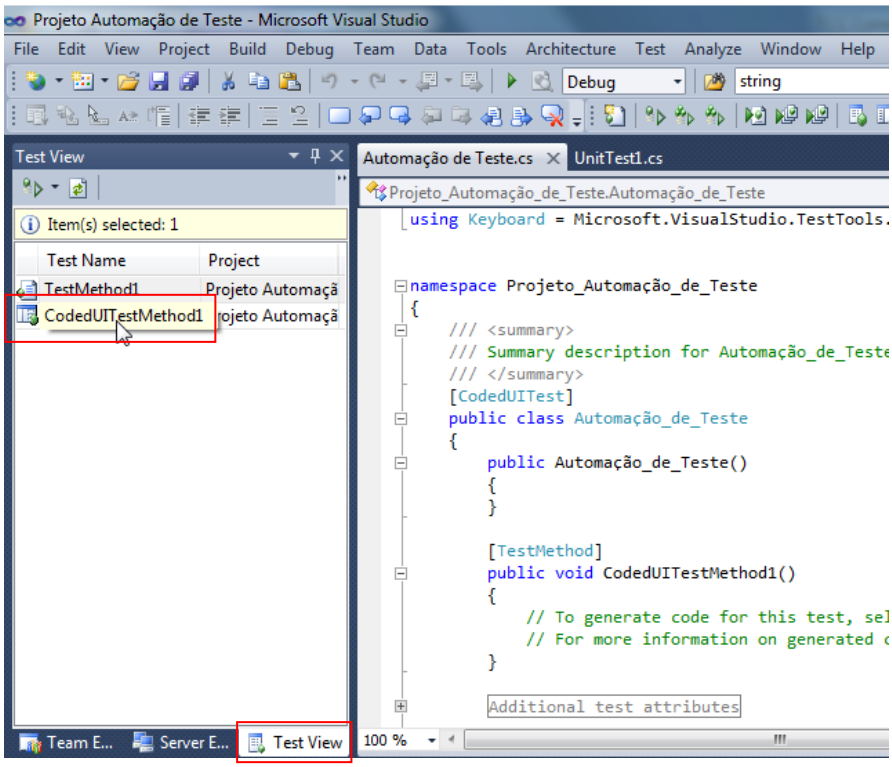


Figura 9 CodedUITestMethod1

Botão direito do mouse entre: Public cod is teste Methodl () {....}

Generate Code for Coded UI Teste> Use Coded UI Test Builder;

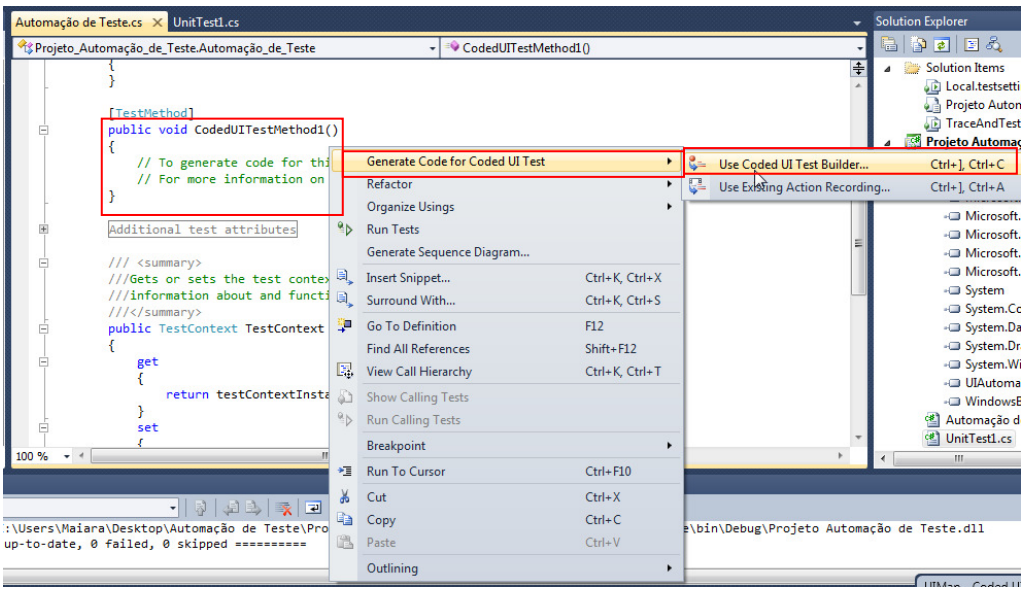


Figura 10 Generate Code for Coded UI Teste

4.2.2 Funções Coded UI Test Builder

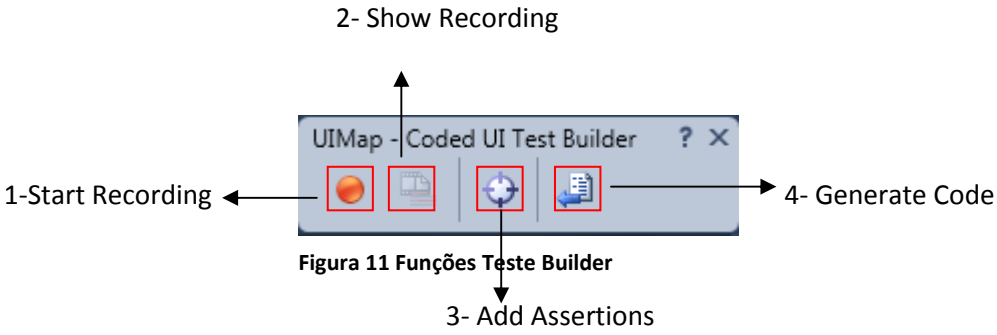


Figura 11 Funções Teste Builder

1- Start Recording:

Inicia-se e pausa a gravação da automação de teste, atalho Alt+R.

2- Show Recording:

Histórico do processo realizado na automação, passo a passo do que vai sendo realizado, atalho Alt+S.

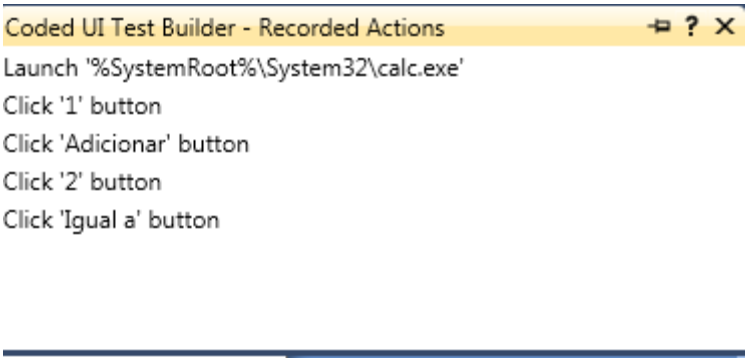


Figura 12 Histórico do Processo

Possibilita a exclusão de algum passo casa seja necessário.

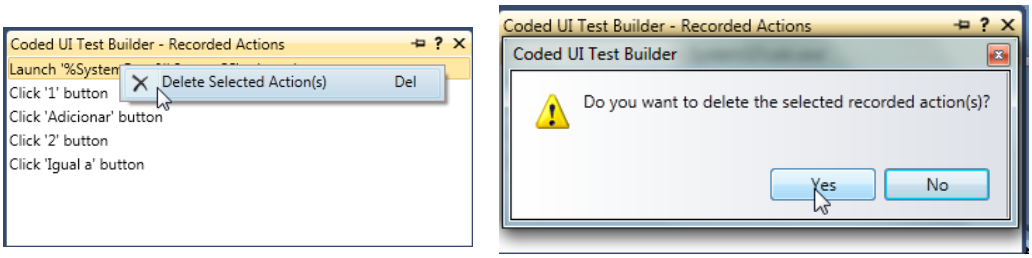


Figura 13 Delete Selected Action

3- Add Assertions:

Ao clicar e arrastar até um objeto abre uma tela com informações do objeto:

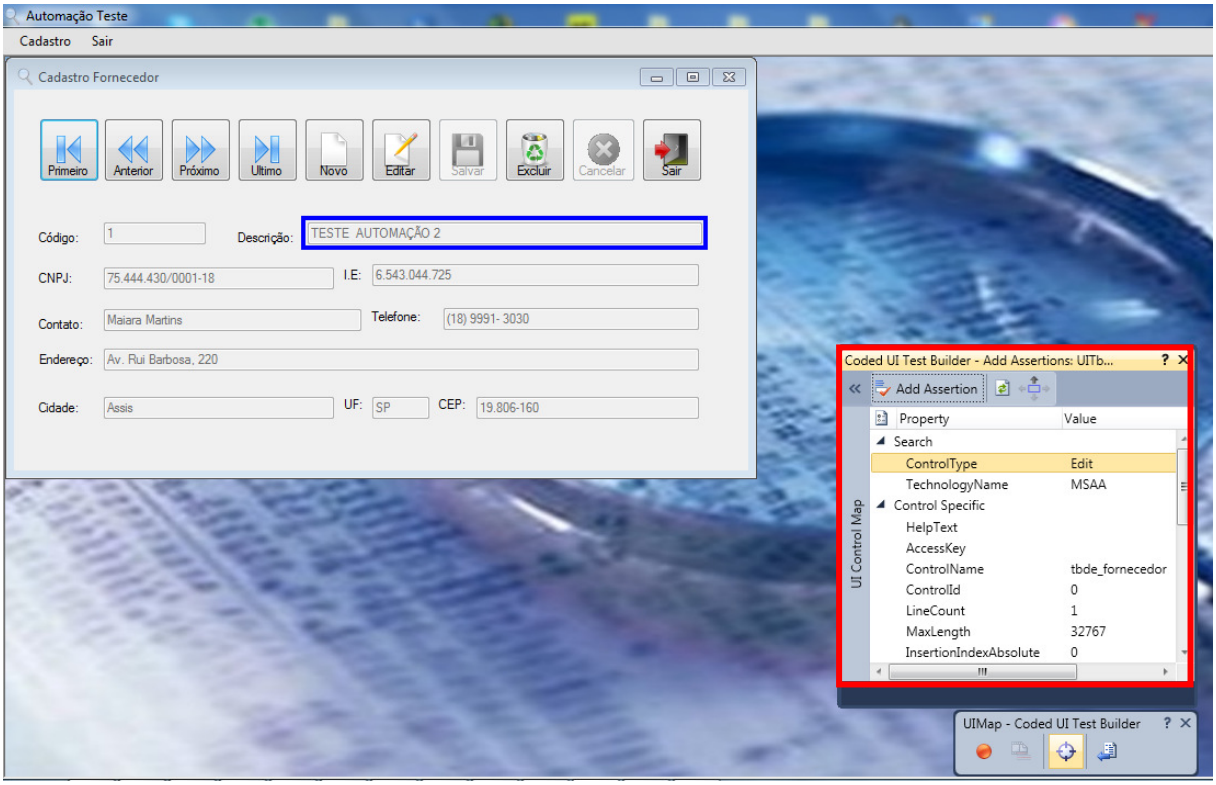


Figura 14 informação Objeto

Possibilita fazer validações, se um campo da tela esta com o valor correto:

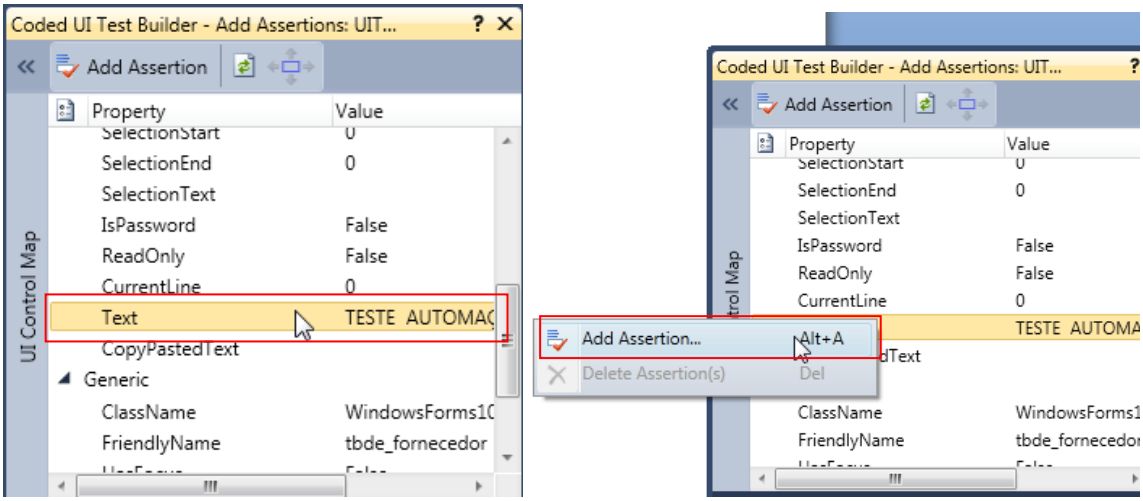


Figura 15 Add Assertions

Por padrão, automaticamente vem preenchido AreEqual, mas existem várias outras funções para comparação:

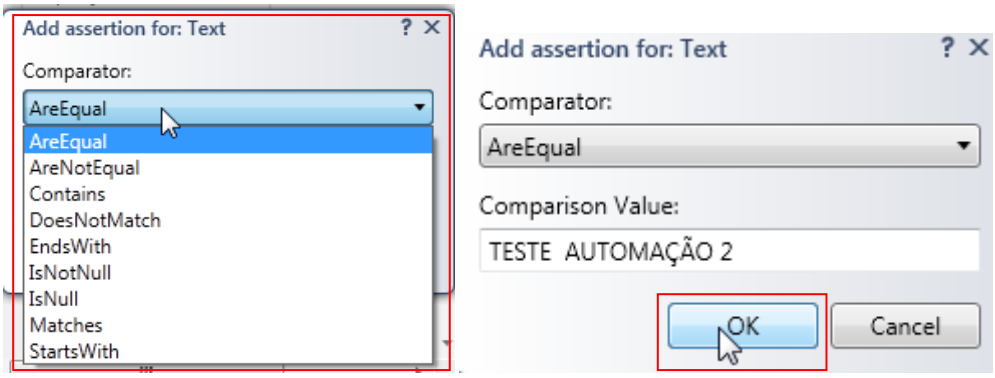


Figura 16 Comparator

4- Generate Code:

Feita a gravação da tela, se faz necessário gerar esse código através do botão Generate Code, atalho Alt+G.

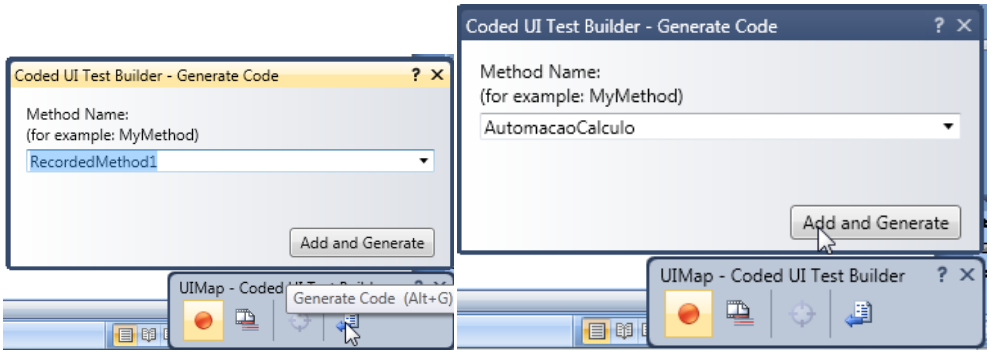


Figura 17 Add and Generate

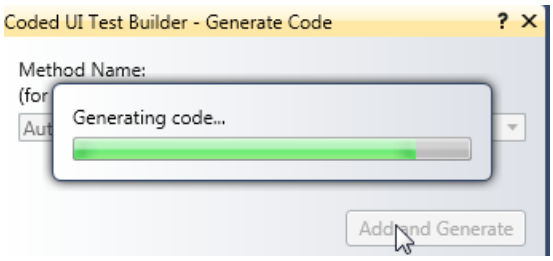


Figura 18 Add and Generate

4.2.3 Gravação da Automação de Teste

Criando automação de teste, de acordo com a documentação do cenário de teste no Anexo A:

1. Criar o projeto e o método de teste:

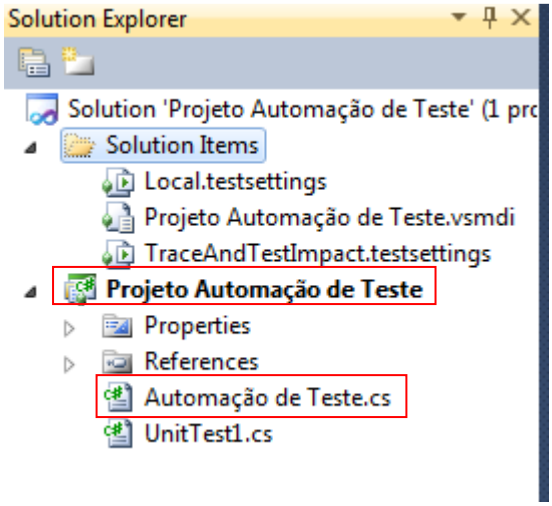


Figura 19 Projeto e Teste

2. Clique no Start Recording e comece a realizar os testes:

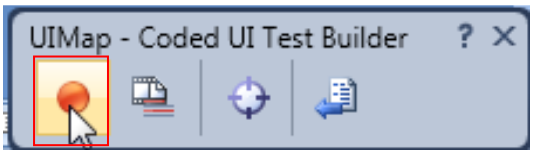


Figura 20 Start Recording

Fazer o passo a passo de acordo com a documentação do cenário de teste, no Anexo A.

Feito gravação, pausar o teste. Usaremos as outras ferramentas do Test Builder.

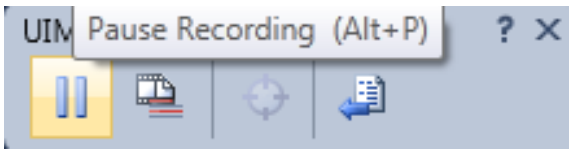


Figura 21 Pause Recording

3. Clique no botão Generate Code, para gerarmos o teste gravado.

No campo Method Name, defina o nome da automação e clique em Add and Generate.

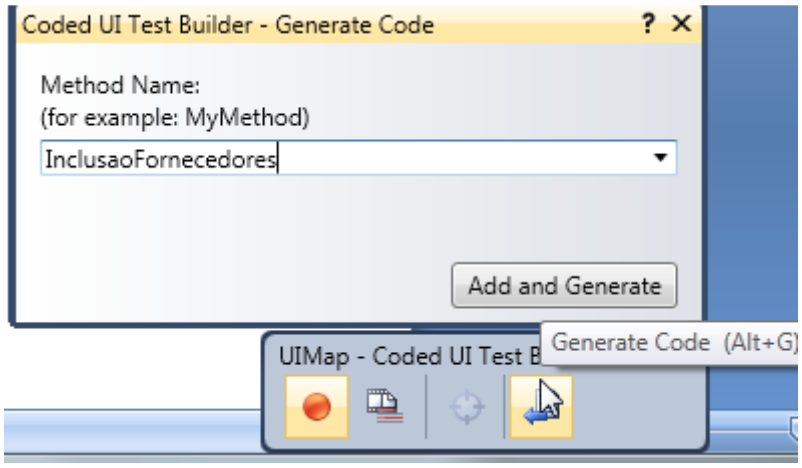


Figura 22 Generate Code

4. Gerado a automação, verifique que habilitou o ícone de Assertions. Clique e arraste até um componente da tela que queira fazer validação.

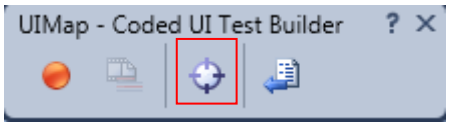


Figura 23 Assertions

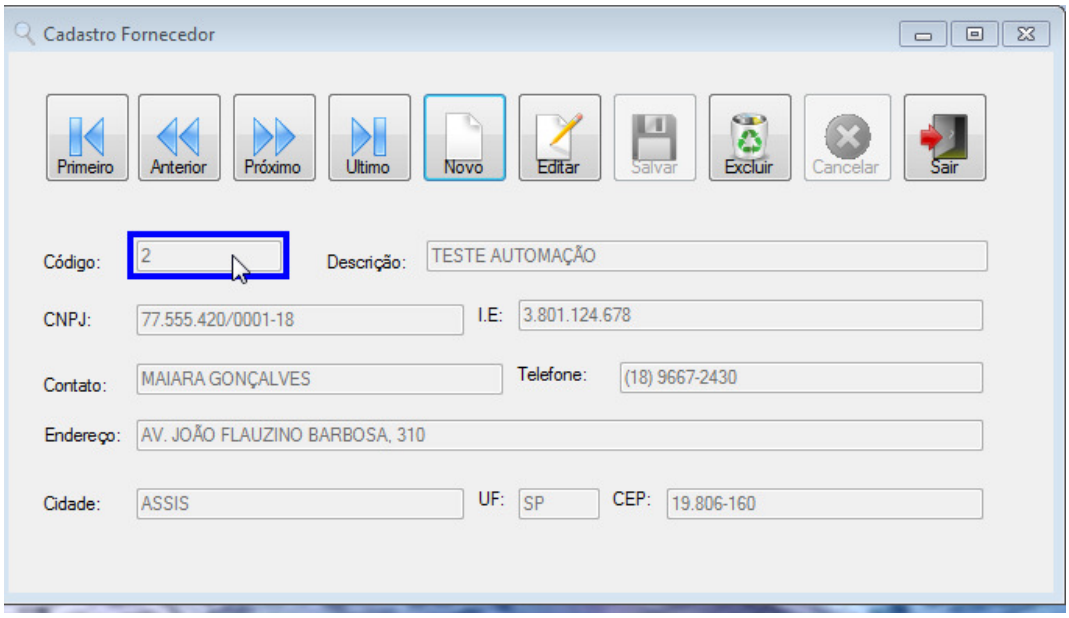


Figura 24 Assertion Text I

Automaticamente aparecerá a tela abaixo. Sua funcionalidade é verificar se realmente a propriedade texto, está atribuído o valor 2.

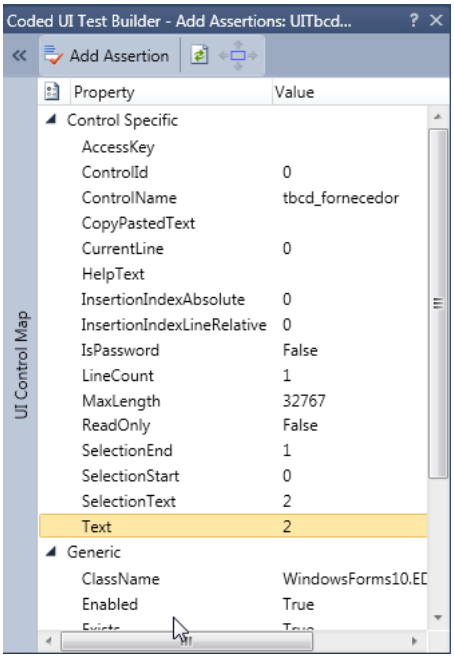


Figura 25 Add Assertions UItbcd_fornecedorWindow

5. Clique no Text, Add Assertion. Abrirá a tela de comparação, escolha a função necessária, no exemplo usaremos AreEqual, ou seja o componente da tela selecionado deverá ser igual a 2:

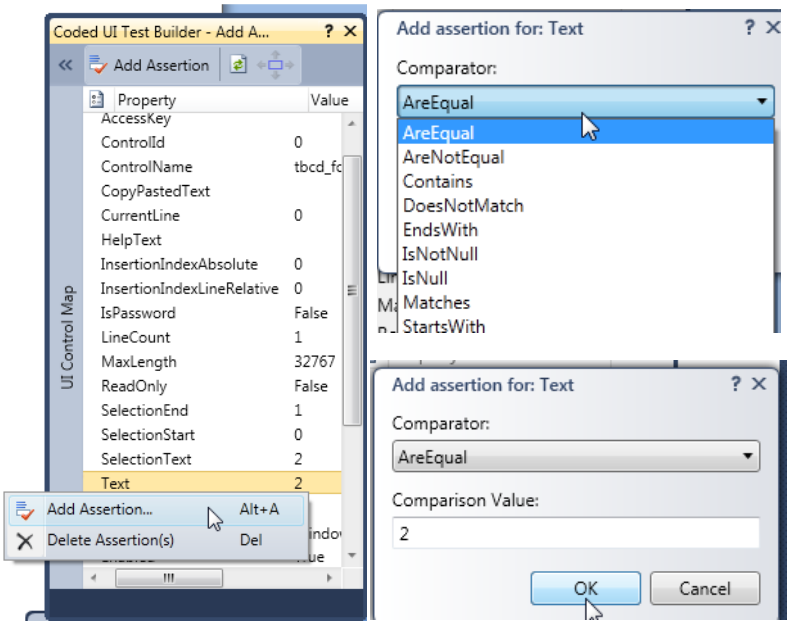


Figura 26 Add assertion for: Text

Adicionando nova Asserção, agora no campo Descrição:

Seleciona o componente a ser validado.

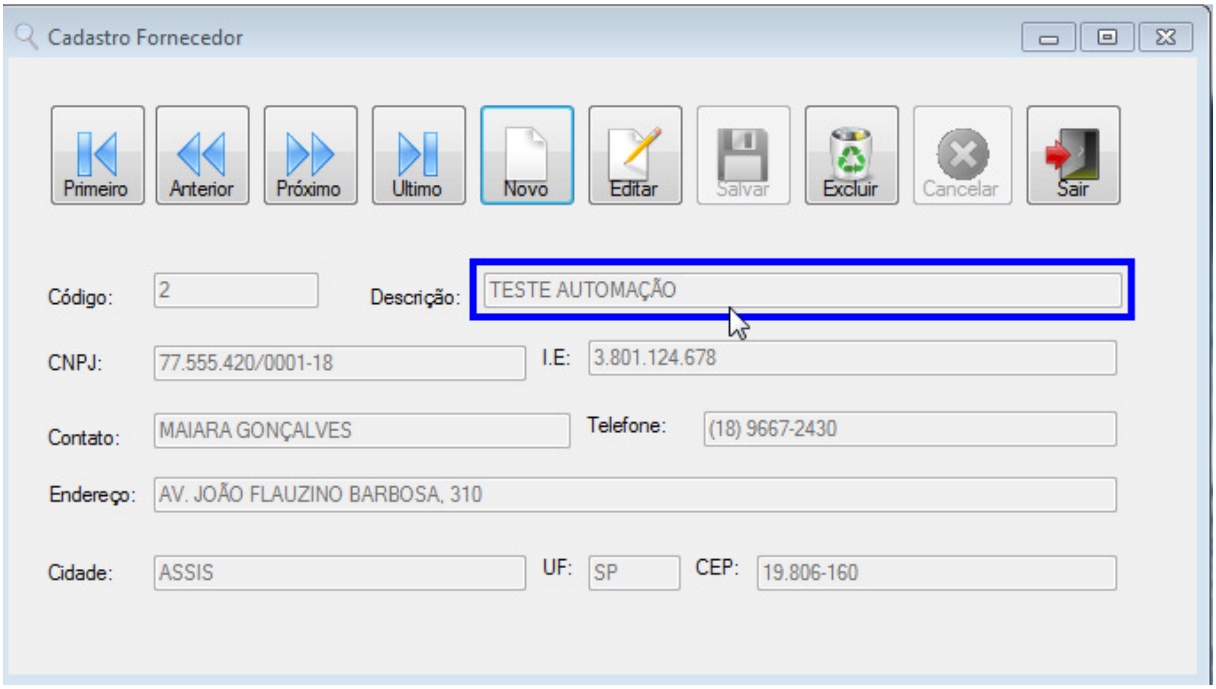


Figura 27 Assertion Test II

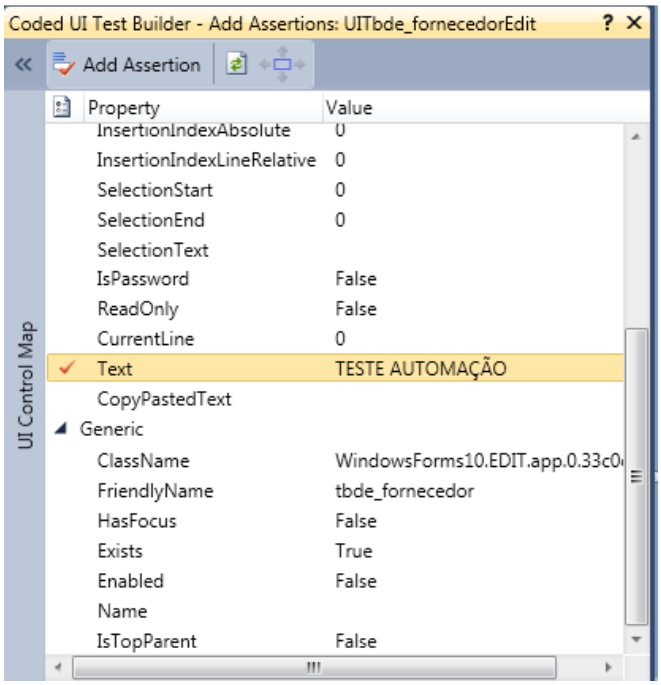


Figura 28 Add Assertions: UITbde_fornecedorEdit

Para concluir as validações, clicar em Generate Code na ferramenta Coded UI Teste Builder.

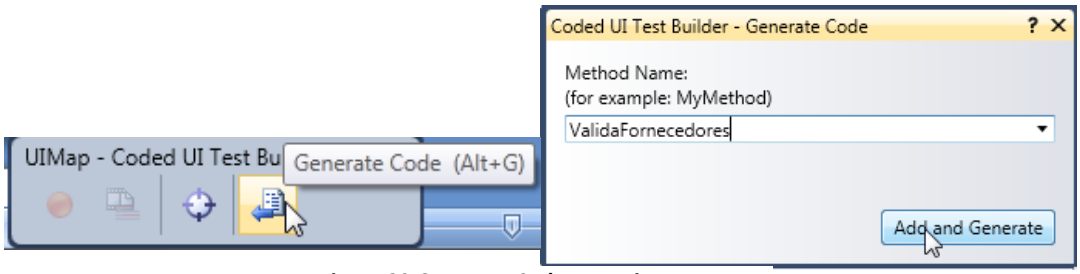
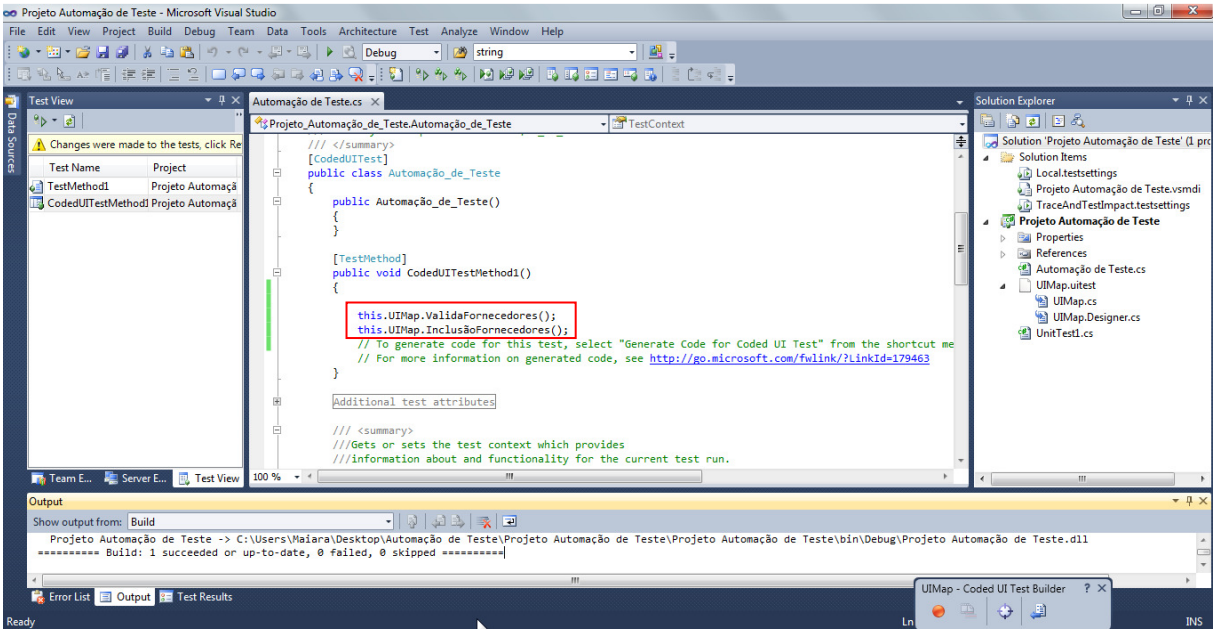


Figura 29 Generate Code Assertion

6. Os métodos que acabam de ser criados no script:



Verifique que a aba Test View, possui alteração que ainda não foi atualizada. Clicar e Refresh:

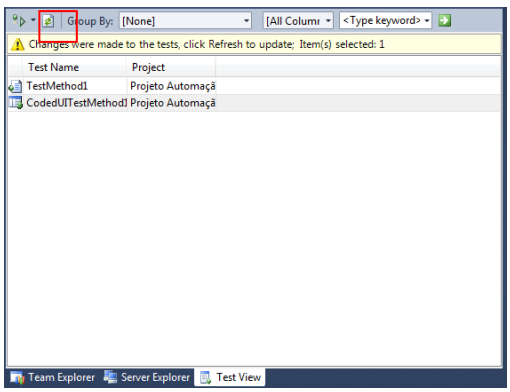


Figura 30 Test View Refresh

7. Para executar seus testes, acessar a aba Test View> Run Selection e selecione a opção Run Selection ou Debug Selection, como preferir.

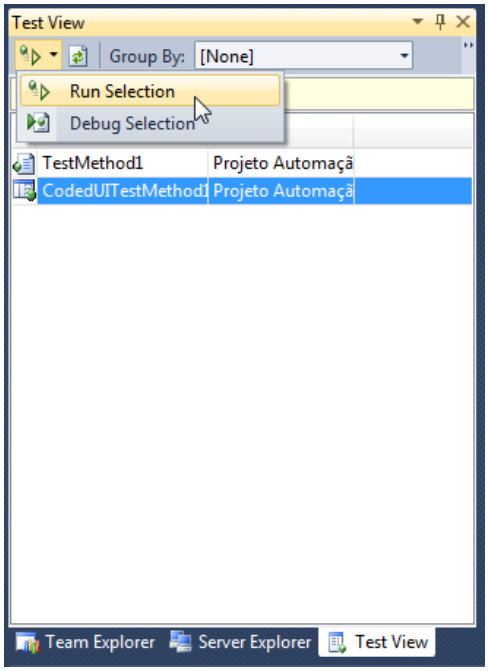


Figura 31 Run Selection

Ao executar essa automação, ocorrerá erro de asserção. Não será válida a comparação da inclusão gravada na automação, com a inclusão realizada na execução. Verifique no item Manutenção.

4.2.4 Manutenção

Ao executar a automação realizada no passo anterior, ocorrerá um erro de asserção, ou seja, erro de comparação.

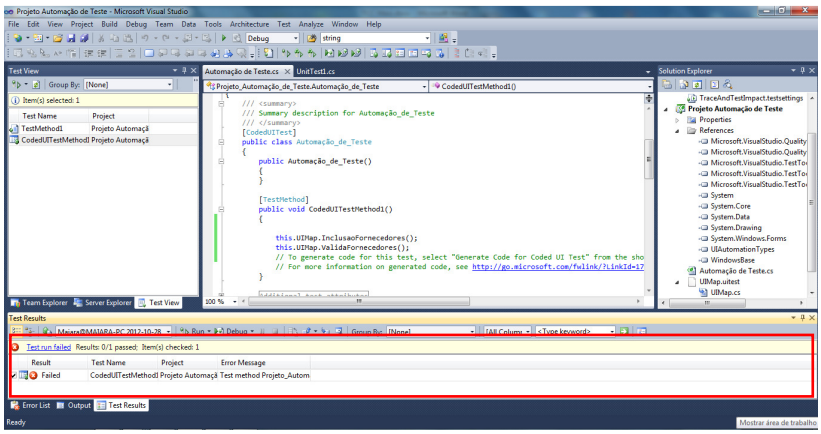


Figura 32 Erro Asserção I

A aplicação de cadastro de fornecedor está programada para quando clicarmos em novo, o código do fornecedor já vir preenchido automaticamente (último cadastro soma mais um).

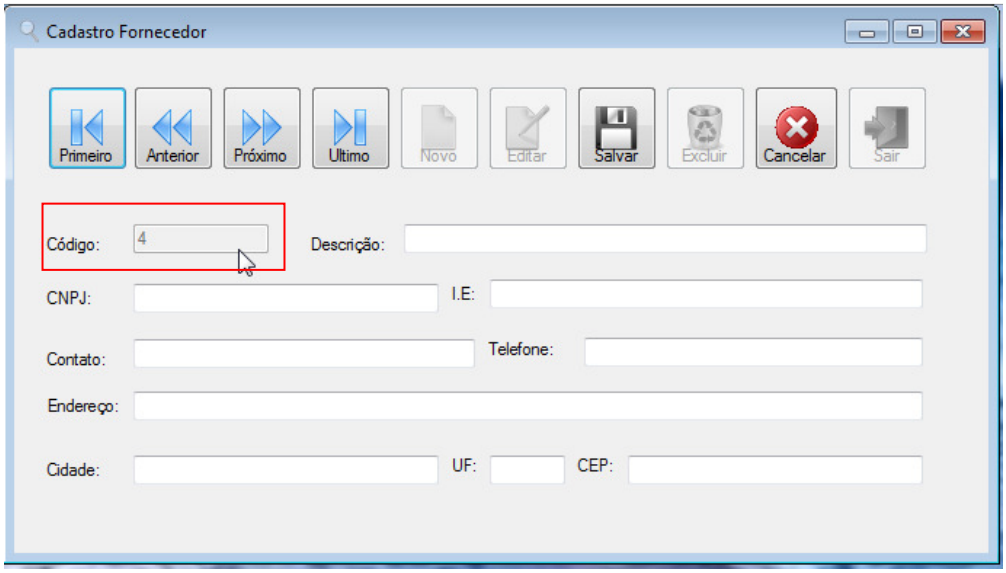
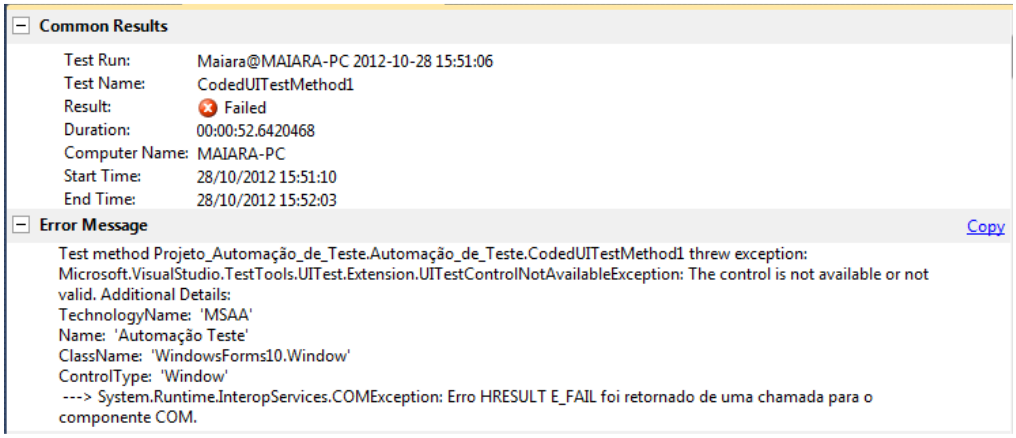


Figura 33 Função Novo Cadastro de Fornecedor

Na automação gravada, fizemos a inclusão do código 2 e fizemos a asserção validando que o campo código seja igual a 2. Quando executarmos a automação, ela fará a inclusão do código 3 (último código 2 + 1) e irá comparar se o 3 incluso é igual a 2 gravado, como são diferentes ocorrerá o erro.



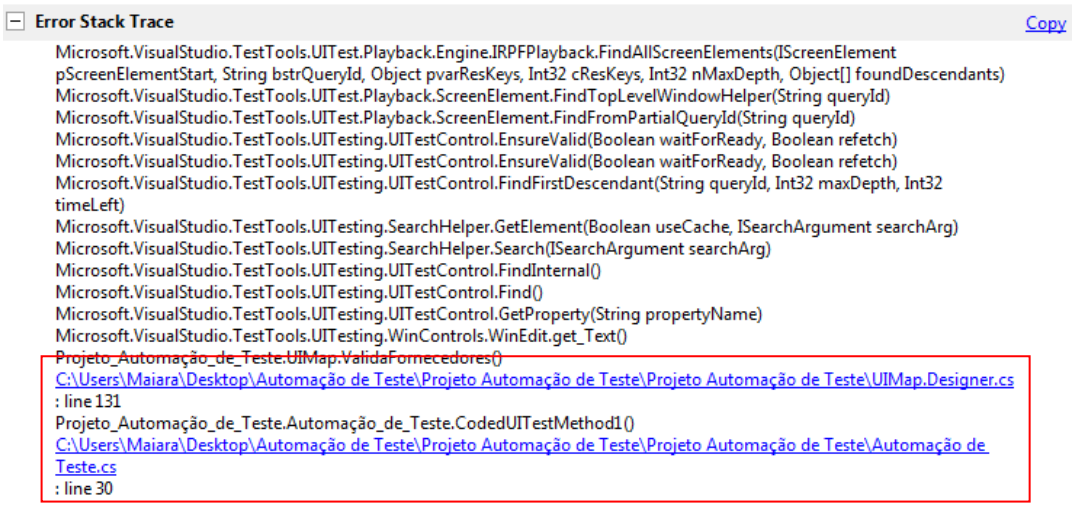


Figura 34 Erro Asserção II

Local referenciado erro:

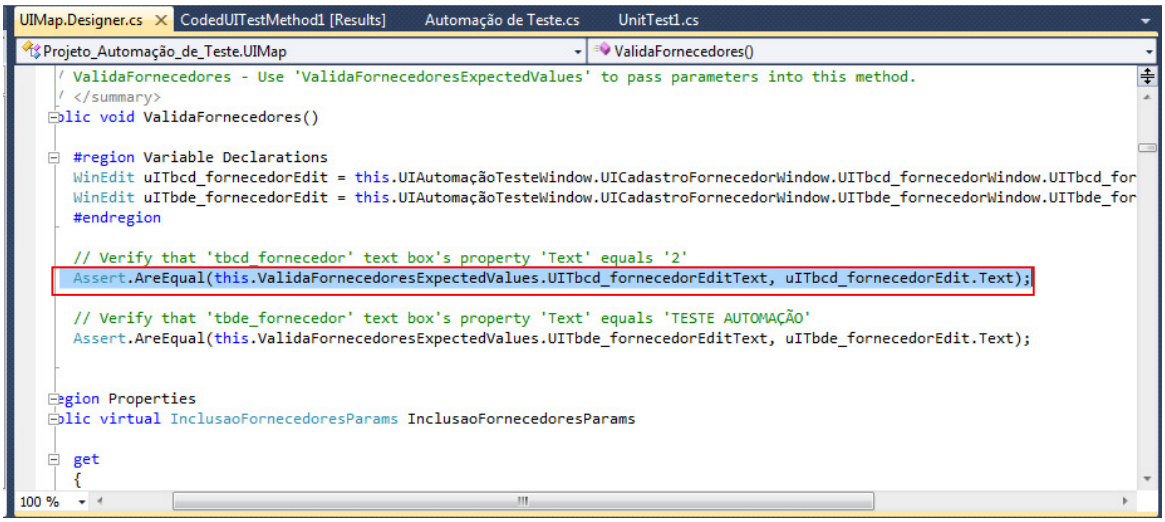


Figura 35 Localização Erro de Asserção

Por este motivo, é importante ter disponível uma base congelada, que não possua movimentação e é fundamental que toda automação de inclusão de registro, tenha também automatizado a exclusão desse mesmo registro. Ou seja, para solução desse problema, antes de executar a automação de teste de inclusão que gravamos, devemos gravar a automação de exclusão.

Feito a automação de exclusão, já pode ser executada a automação de inclusão.

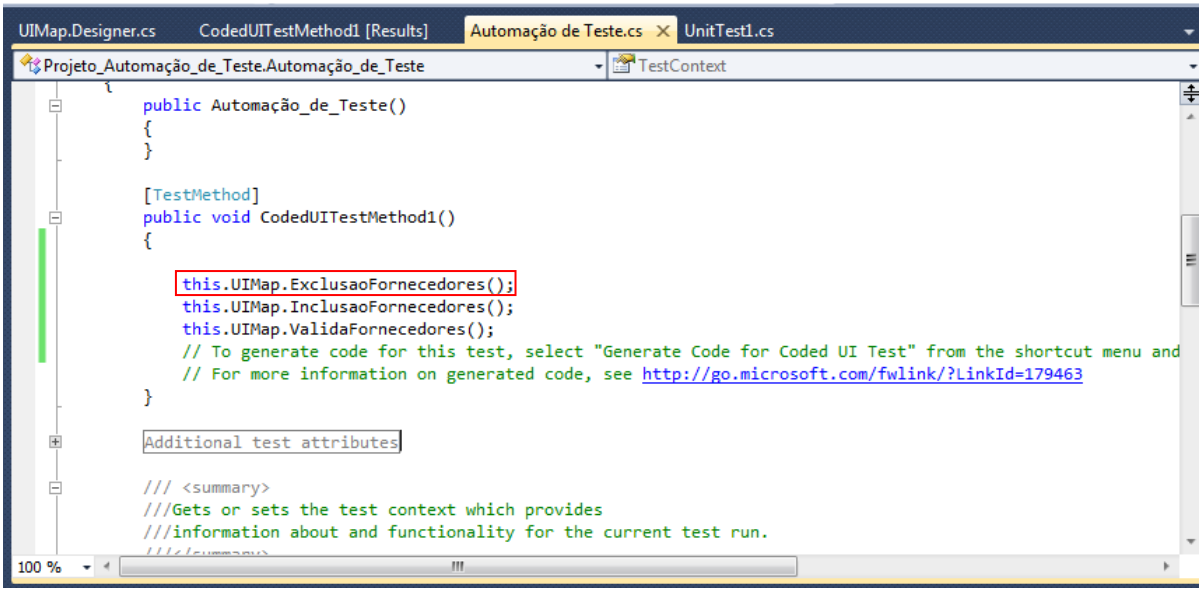


Figura 36 Exclusão Fornecedores

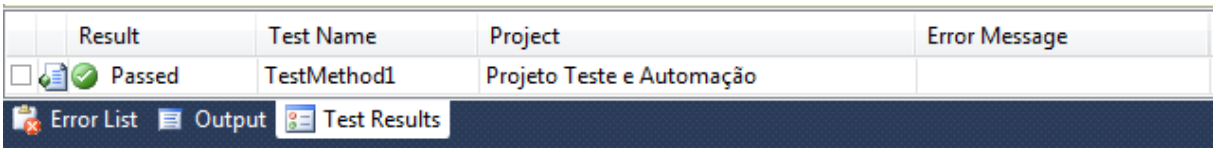


Figura 37 Resultado

4.2.5 Dicas e Informações

- Quebra de funcionalidade

Vamos usar como exemplo o teste de cadastro de fornecedor: não é viável fazer um teste único de todas as funcionalidades da tela (inclusão, alteração e exclusão). O ideal seria dividir a automação em partes, evitando sobrecarregá-la e consequentemente travar durante a execução.

Ao dividirmos a automação em partes, temos a flexibilidade de mudar a ordem da execução, além de facilitar a manutenção caso seja necessária.

- Identificação dos Componentes

Para que os testes automatizados funcionem, é fundamental que os componentes visuais estejam mapeados, ou seja, que possuam algum método de identificação.

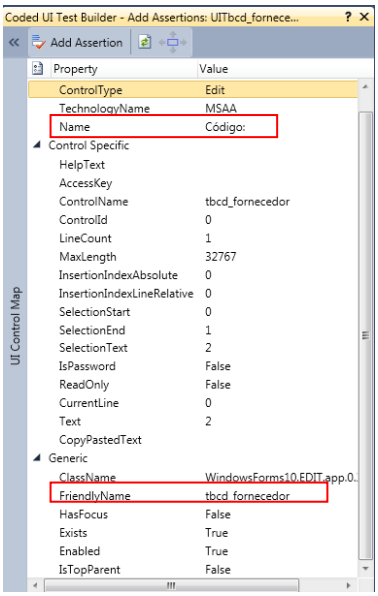


Figura 38 Identificação dos Componentes

- Ordered Test

Durante um projeto de automação de teste, existem vários projetos em uma mesma *solution*, onde nem sempre a ordem que você deseja testar é na ordem a qual foi gravada. No Visual Studio, possui a ferramenta Ordered Test, que possibilita ter a flexibilidade de mudar a posição dos projetos, de acordo com a necessidade.

5. Conclusão

Ao pensarmos na necessidade e dependência do software, é impossível não nos preocuparmos com a qualidade do mesmo.

A presente pesquisa analisou ramos da engenharia de software, suas etapas e necessidade, métodos de teste e aspectos de qualidade. Para conquistarmos a satisfação do contratante, é indispensável testar o software, assegurar a qualidade do produto a ser entregue. Foi também, objetivo desse trabalho, a descrição de um exemplo de automação de teste utilizando ferramenta do Visual Studio.

A gravação da automação de teste demanda mão de obra extra e horas dedicadas à documentação e gravação. Os benefícios a médio e longo prazos, podem ser caracterizados como: maximização da qualidade do produto; os testes são mais dinâmicos e ágeis; diminuição dos erros resultantes da influência humana. Uma vez gravada a automação, não será necessário refazer o processo, apenas a sua manutenção. Ao automatizarmos testes básicos, teremos mais tempo para dedicar a testes mais complexos.

Como trabalho futuro, novos estudos e implementações podem ser realizadas no tocante assunto, como promover treinamentos com a finalidade de capacitar profissionais na área, assim como divulgar a necessidade da realização de testes para garantia da qualidade do produto final.

Referências Bibliográficas

DANTAS, Kívia.- **A importância do Teste de Software-** Disponível em: <http://www.testexpert.com.br/?q=node/1265> Acesso em: 14 de abr. 2012.

FERNANDES, Jorge H.C. - **Porque Engenharia de Software?** Disponível em: <www.cic.unb.br/~jhcf/MyBooks/iess/Intro/PorqueEngenharia.pdf > Acesso em: 07 de abr. 2012.

EUCLIDES – **O que é Engenharia e quais suas especialidade** Disponível em: <http://pir2.forumeiros.com/t8992-o-que-e-engenharia-e-quais-as-sua-especialidades> Acesso em: 01 de jun. 2012.

GARCIA, Franciele P. - **Como fazer estimativas num Projeto de Software?** Disponível em: <http://www.dsc.ufcg.edu.br/~patricia/pct/aula9/estimativas.PDF> Acesso em: 08 de jun. 2012.

FERNANDES, Jorge H. C. – **As 10 Áreas da Engenharia de Software, Conforme o SWEBOK** Disponível em:<http://www.cic.unb.br/~jhcf/MyBooks/iess/Intro/10AreasDaEngenhariaDeSoftware.pdf> Acesso: 11 de jun. 2012.

ÁVILA, Marcio - **Modelo de Qualidade de Software de McCall** Disponível em: <http://blog.mhavila.com.br/2010/06/> Acesso: 15 de jun. 2012.

COUTINHO, Rafael – **Engenharia: Uma Ciência e Muitas Profissões** Disponíveis em: < http://www.culturamix.com/cultura/curiosidades/engenharia-uma-ciencia-e-muitas-profissoes > Acesso em: 15 de jun. 2012.

SAUVÉ, Jacques - **Projeto em Computação II** Disponível em: < http://www.dsc.ufcg.edu.br/~jacques/cursos/apoo/html/impl/impl3.htm> Acesso: 18 de jun. 2012.

SILVA, Alexandre M. – **Teste de Unidade** Disponível em: < http://www.dsc.ufcg.edu.br/~jacques/cursos/apoo/html/impl/impl3.htm> Acesso: 18 de jun. 2012.

ROESSLER, Rosangela G. – **Casos de Testes Funcionais** Disponível em: <http://testersoftware.blogspot.com.br/2010/08/casos-de-testes-funcionais.html> Acesso: 06 de out. 2012.

SILVA, Fernando R. – **Teste de Software- Teste Unitário** Disponível em: <http://www.devmedia.com.br/testes-de-software-teste-unitario/22284> Acesso: 20 de ut. 2012.

SALDANHA, Sara A. – **Qualidade de Software e a Importância dos testes nas empresas de desenvolvimento de Software** 2009. 60 pg.

MAZZOLA, Vitório B. **Engenharia de Software** 2003. 145 pg.

FALBO Ricardo, A. **Engenharia de Requisitos** 2012. 179 pg.

VASCONCELOS, Alexandre M. L. **Introdução à Engenharia de Software e a Qualidade de Software.** 2006. 163 pg.

VALDRICH, Poliane K. **Automação de Teste- Visual Studio C#** 2011. 48 pg.

PASSOS, Jocélio **Engenharia de Software** 2007. 18 pg.

PRESSMAN, Roger S. **Engenharia de Software**, 6. ed. Porto Alegre, Bookman, 2006.

SOMMERVILLE, Ian **Engenharia de Software**, 8. ed, São Paulo,Pearson Addison, 2007.

PRESSMAN, Roger S. **Engenharia de Software**, 3. ed. São Paulo, Mackron Books, 1995.

Anexos

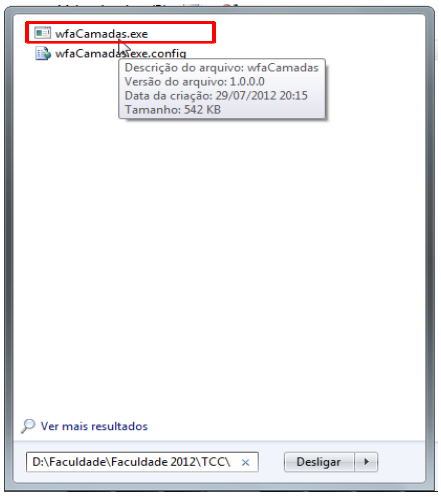
A. Cenário de Teste

CENÁRIO DE TESTES

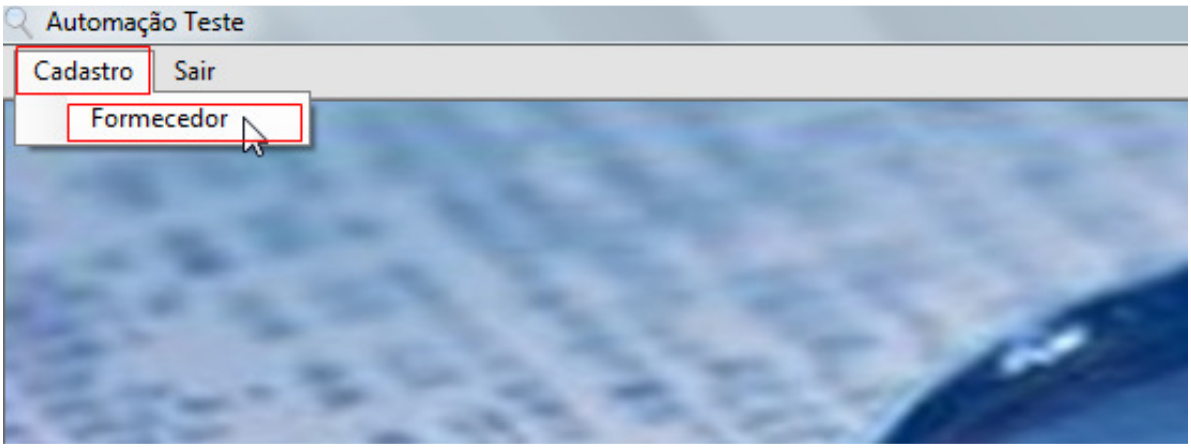
Data	22/10/2012	**Autor Cenário:**
Automação:	Sistema Automação de Teste	Maiara Martins Gonçalves

Sistema de Testes:	Projeto Conclusão de Curso		
Pré-Condição:	Possuir Base de Dados Congelada/ Arquivo programa do sistema mapeado de acordo com endereço passo 1.		
Pós-Condição:	(não se aplica)		
Caso de Teste:		**Linha**	**Status Gravação**
1	**Acessar: D:\Faculdade\Faculdade 2012\TCC\Programa\CamadasBase\wfaCamadas\wfaCamadas\bin\Debug\wfaCamadas.exe**	**20**	**OK**
2	**Acessar Menu Cadastro> Fornecedor**	**51**	**OK**
3	**Clicar no botão Novo**	**95**	**OK**
3.1	**Preencher os dados de acordo com print, a cada campo preenchido clicar com mouse no campo subsequente**	**125**	**OK**
3.2	**Clicar no botão Salvar**	**124**	**OK**
3.3	**Clicar no botão OK**	**150**	**OK**
3.4	**Clicar botão próximo**	**164**	**OK**
Analista Gravação:	Maiara Martins Gonçalves		
Evidências: ***(colar as telas evidenciando os testes)***			

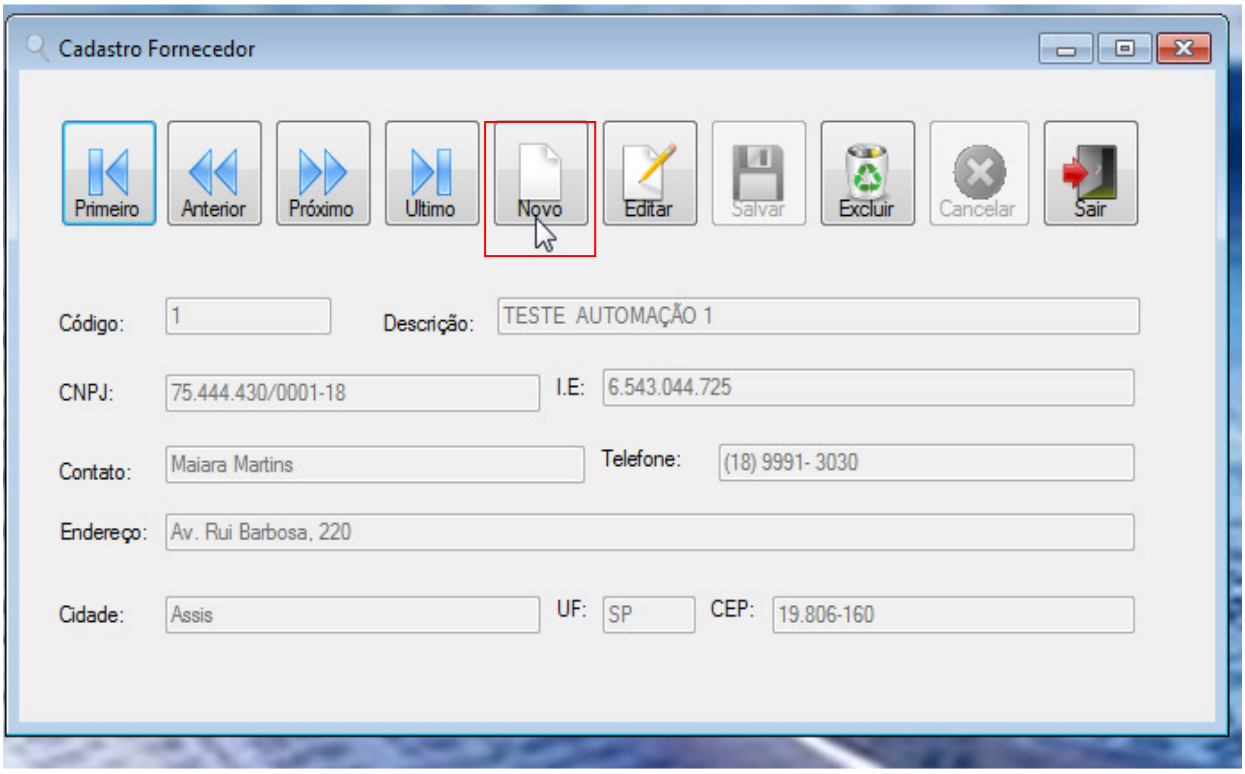
Passo: 1 Acessar: D:\Faculdade\Faculdade2012\TCC\Programa\CamadasBase\wfaCamadas\wfaCamadas\bin\Debug\wfaCamadas.exe



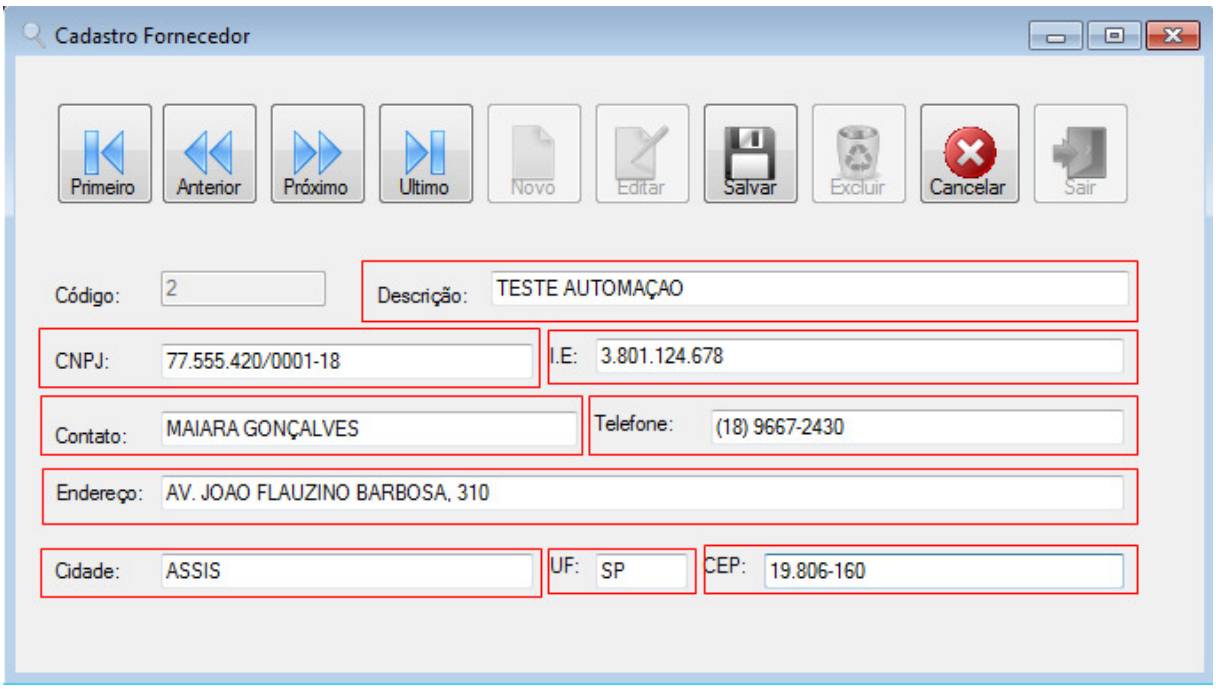
Passo 2: Acessar Menu Cadastro> Fornecedor



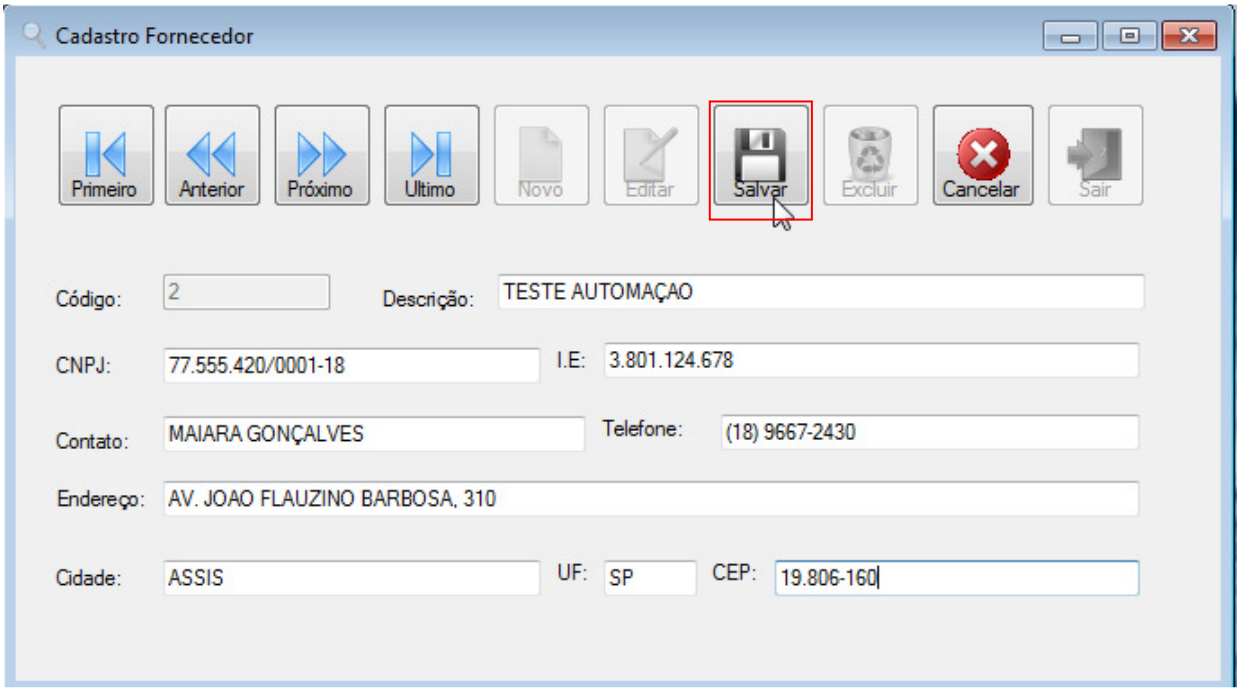
Passo 3: Clicar botão Novo:



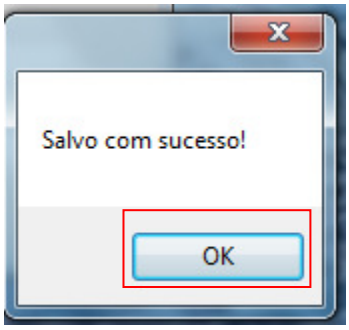
Passo: 3.1 Preencher os dados de acordo com evidência, a cada campo preenchido clicar com mouse no campo subsequente:



Passo 3.2: Clicar no botão Salvar:



Passo 3.3: Clicar no botão OK:



Passo 3.4: Clique no botão próximo: