

DATA TEST: GERAÇÃO DE DADOS PARA O APOIO DO PROCESSO DE TESTE

**Andressa R. Siqueira dos Santos, Emerson R. Eduardo, Judemir B. da Silva
Júnior, Michele Bravo**

Faculdade de Tecnologia (FATEC) – Av. Presidente Vargas, 2331 – CEP: 17.400-000 –
Garça – SP – Brasil.

michele@hotmail.com; andressaroberts@gmail.com ; Falta o e-mail dos meninos que eu não tenho.

Resumo.

Este artigo descreve o software Data Test, desenvolvido com o objetivo de agilizar a prática do teste de software através da geração de dados válidos aleatoriamente que servirão para preenchimento dos campos no software em que o profissional está efetuando verificação de erros de desenvolvimento do programa a ser entregue.

Abstract.

This article describes the Data Test software developed in order to streamline the practice of software testing by generating valid data at random which will be used to fill the fields in the software that the trader is making development of error-checking program to be delivered.

1. Introdução

A dependência tecnológica das organizações tem aumentado exponencialmente ao longo dos anos, conseqüentemente, suas operações internas passam a ser conduzidas por um número cada vez maior de sistemas informatizados. Os softwares que garantem a integração desses ambientes se tornam aplicações cada vez mais complexas tornando o processo de desenvolvimento cada vez mais difícil.

Assim, inibir o aparecimento de falhas torna-se um desafio na garantia da qualidade do software. Processos são então estruturados para que os vários estágios do

desenvolvimento da aplicação sejam avaliados, com o objetivo de garantir que defeitos sejam identificados o mais cedo possível, minimizando o número de falhas.

A qualidade é então resultado das decisões tomadas ao longo de todo o ciclo do projeto e não apenas consequência de uma fase. As melhores práticas de gerenciamento de projetos, como as descritas no PMBOK (Project Management Body of Knowledge), de autoria do Project Management Institute (PMI), reconhecem a importância do gerenciamento da qualidade do projeto.

Ainda que o conceito de qualidade seja abrangente, um software pode ser considerado com qualidade quando um conjunto de características atende determinados requisitos ou até mesmo quando foi construído de forma a poder ser mantido eficientemente depois de implantado (RIOS, 2006). Para se certificar que o software atende as especificações foram estabelecidos os processos de verificação e análise. Enquanto a validação determina se o produto correto está sendo construído, a verificação analisa se o produto foi construído corretamente.

Apesar de serem de natureza distinta, são complementares e visam fortalecer o processo de detecção de erros. Segundo Sommerville (2007), ambas visam estabelecer a confiança de que o sistema está adequado a seu propósito. Esta pesquisa em questão pretende auxiliar o processo de validação, cujos testes são baseados no comportamento do software, validando se este faz o que foi definido pelos requisitos e assegurando que este atende as expectativas do cliente.

Durante essa etapa, são executados testes com o objetivo de encontrar defeitos. Dados são inseridos e as saídas do software são examinadas para verificar se seu comportamento está em conformidade. Por ser uma fase custosa e trabalhosa, existem diversas ferramentas oferecendo uma variedade de recursos de forma a reduzir os custos de teste.

Uma das maiores necessidades na área de testes de software é o fornecimento de dados específicos e válidos para determinados campos do sistema. Sem uma ferramenta que disponibilize os dados necessários, esse trabalho se torna mais lento e complexo para o analista, pois terá que informar por si mesmo dados válidos em cada campo de informação.

O sistema descrito nesse artigo vem justamente suprir essa necessidade, fornecendo ao analista de teste os dados requeridos agilizando assim o procedimento da análise de testes e apoiando o processo de teste.

2. Metodologia Agil

A partir de entrevistas com profissionais da área de teste de software verificou-se que uma de suas maiores necessidades é a obtenção de dados válidos necessários para o preenchimento de campos de informação requeridos pelo sistema ao qual o teste se aplica. Além disso, foi realizado um brainstorming entre os analistas para coleta de múltiplas ideias relacionadas aos requisitos do projeto.

Assumindo um deadline do projeto de seis meses, optou-se por implementar um modelo de processo ágil, com foco na produção do software em si e não em processos e ferramentas ou documentação. Dentre as dezenas de metodologias, foram escolhidos os princípios do método Scrum com vistas a obter os melhores resultados em curto prazo, de forma incremental, onde cada membro da equipe recebeu um papel específico.

Na reunião de planejamento, foram definidos os números de Sprints - que são pequenos marcos com datas iniciais e finais com mais ou menos a mesma duração (DIMES, 2014) - e o Backlog do Produto, além das datas para entrega de resultados. Foram também definidas as ferramentas de trabalho explanadas na terceira parte desta pesquisa. O *Scrum Master* – responsável por evitar os impedimentos durante a realização do trabalho - a cada reunião certificava-se que os desenvolvedores estivessem concentrados nas questões técnicas, tais como a definição da linguagem a ser utilizada, ferramentas e modelagem do banco de dados. O *Product Owner*, conhecedor das regras de negócio do cliente foi o responsável por definir junto aos *stakeholders* quais os requisitos do produto final, além de definir suas prioridades, traduzidas no backlog do produto.

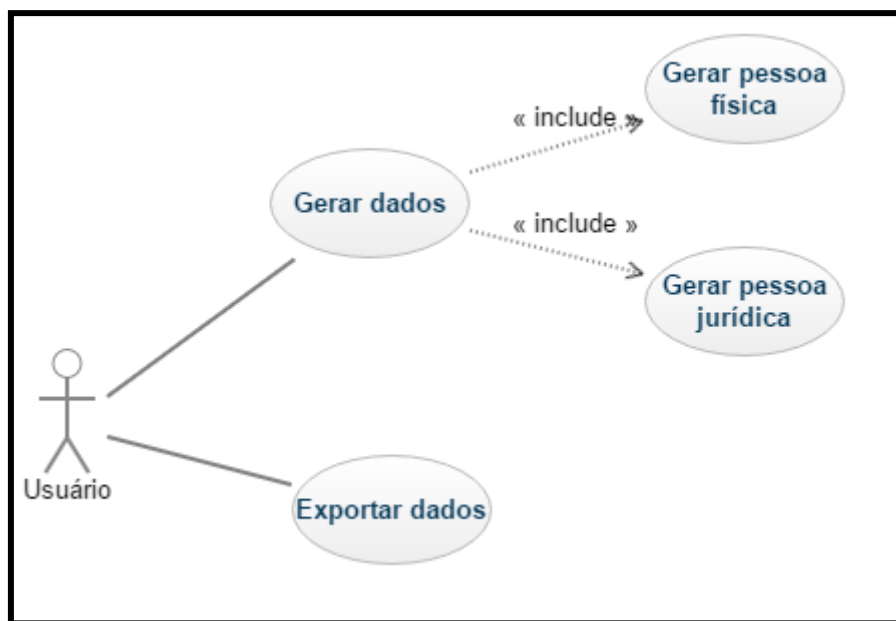
Para gerenciamento da evolução do projeto utilizou-se o Trello como ferramenta de apoio, onde os integrantes envolvidos interagiram ativa e continuamente conforme suas tarefas designadas por setor. Periodicamente foi-se documentando a evolução do sistema, tais como o esboço de telas, construção do banco de dados, documentações teóricas e correções a serem executadas.

3. Desenvolvimento

Partindo da definição do software que seria desenvolvido pelo grupo, iniciou-se a elaboração de diagramas de uso e entidade relacionamento utilizando a ferramenta da Microsoft, Visio.

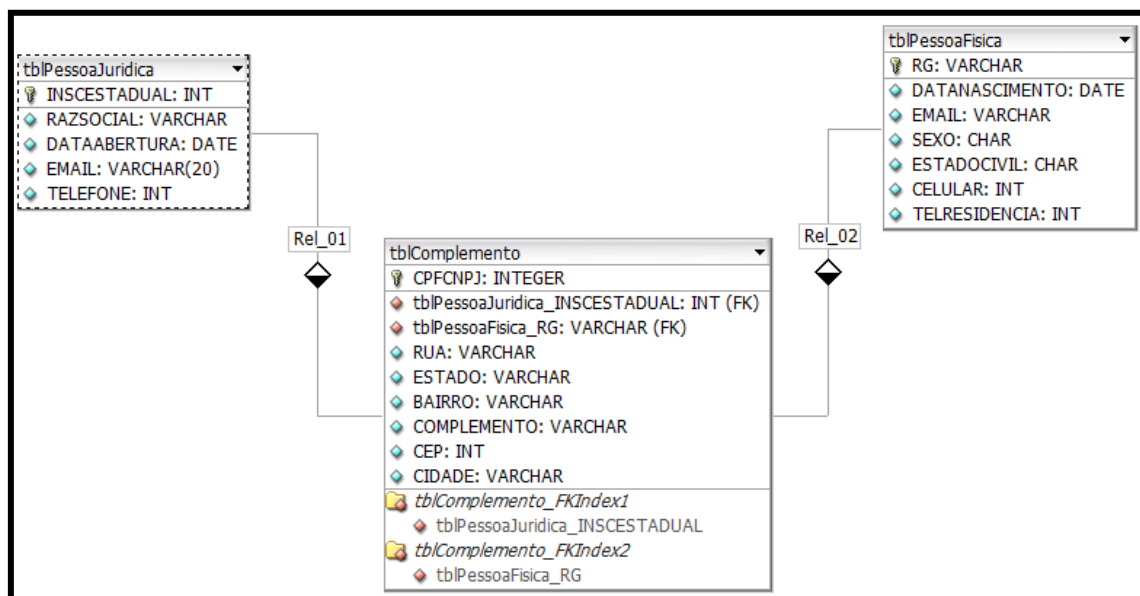
A Figura 01 mostra o diagrama de caso de uso, criado para mostrar as formas de funcionamento do projeto, apresentando a maneira de interação do usuário com o sistema.

Figura 01 – Diagrama de caso de uso.



A figura 02 mostra o diagrama de entidade e relacionamento, desenvolvido para representar todo o modelo conceitual do projeto e ajudar na organização das tabelas criadas na base de Dados, descrevendo o modelo de dados do sistema com alto nível de abstração.

Figura02 – Diagrama de entidade e relacionamento.



Seguiu-se o uso do DbDesigner para que se pudesse concretizar a construção do banco de dados relacional, com base nos diagramas, que alimentará os respectivos campos do software.

Foram também apresentadas e corrigidas as telas do sistema, de modo a se ter uma visão inicial do produto a ser entregue. Para tal, recorreu-se ao Netbeans como IDE (*Integrated Development Environment* ou Ambiente de Desenvolvimento Integrado) no qual através da linguagem de programação Java será estabelecida a conexão com a base de dados, previamente modelada e estruturada.

Ao realizar o acesso ao sistema o usuário irá se deparar com a tela inicial, onde terá a opção de gerar os dados apenas através de um botão de seleção, indicando se deseja dados de pessoa física ou jurídica, assim que o usuário realiza a seleção o sistema apresenta uma segunda tela já com os dados gerados. A tela gerada contém as mesmas informações básicas, como por exemplo: telefone, endereço, entre outras informações que podem conter em um cadastro tanto de pessoa física como jurídica, esses dados são buscados na base de dados em uma mesma tabela, independente da seleção do usuário, os demais dados, como CPF ou CNPJ, Nome ou Razão social, entre outros dados que diferem de um cadastro físico para jurídico são buscados em outra tabela, dependendo da seleção que o usuário realizar. Por último a tela de dados possibilita ao usuário realizar uma atualização, gerando outros dados para o mesmo tipo de cadastro apenas com um clique de botão.

Os dados requeridos no programa serão buscados com auxílio da ferramenta Xampp, que irá auxiliar a instalação e visualização do banco de dados nas máquinas utilizadas para a conexão Sql e dessa forma os dados serão enviados ao sistema através dos comandos específicos já programados em cada tela.

4. Resultados até o momento

Durante o design das telas e criação do banco de dados, verificou-se a necessidade de uso de padrões de projeto para o desenvolvimento do software de forma a garantir que boas técnicas estavam sendo implementadas.

Constatou-se também a necessidade de definir regras para a geração dos dados de forma a moldar uma base persistente, evitando resultados redundantes ao longo das diversas gerações de dados que apesar de terem sido alimentados com dados pré-definidos serão entregues de forma aleatória ao usuário, possibilitando sua utilização em diversos cenários de teste.

5. Considerações e Trabalhos futuros

Apesar de existirem diversas ferramentas de apoio ao processo de software, o sistema desenvolvido visa atender a demanda específica dos analistas de teste consultados. Dessa forma, pretende-se avaliar a melhoria da eficácia dos testes executados após a utilização do software.

Além de verificar a variedade de combinações disponibilizada, analisaremos a rapidez quanto ao volume de dados gerado e a quantidade de vezes em que o teste é repetível. Assim, propõem-se como trabalhos futuros o aperfeiçoamento das funcionalidades do Data Test após a análise do seu uso pelos analistas de teste, avaliando o próprio método de geração de dados.

Referências

DIMES, T. Scrum Essencial. Babelcube Inc., 2014

PEZZE, Mauro; YOUNG, Michal. **Teste e análise de software**: processo, princípios e técnicas. Porto Alegre: Bookman, c2008.

SOMMERVILLE, Ian. **Engenharia de Software**, Addison-Wesley Brasil, 8ª Edição, 2007.

RIOS, Emerson. **Teste de software**. Alta Books Editora, 2006.