Ministério da Agricultura, Pecuária e Abastecimento

Comunicado Técnico 14

Dezembro, 2001
Campinas, SP

ISSN 1677-8464

Desenvolvimento e Execução Automática de Testes Funcionais do SIGI

João Francisco Gonçalves Antunes[1]
Moacir Pedroso Júnior[2]
Marcos Cezar Visoli[3]
Flávia Gobet de Aguiar[4]
Marcos Geraldo Tozzi[5]
Maria Paz Palma Abedrapo[6]

O desenvolvimento de software envolve uma série de atividades de produção em que a oportunidade de ocorrer falhas humanas são grandes. Enganos podem acontecer tanto no início do processo de desenvolvimento, nas fases de análise e especificação, como no decorrer dele, nas fases de projeto e implementação. Devido a esta característica, o desenvolvimento de software é acompanhado por atividades de testes que visam garantir a qualidade do software (Rocha et al., 2001).

Teste é o processo de executar um programa com a intenção de descobrir a presença de defeitos. Casos de testes são então projetados para descobrir sistematicamente diferentes categorias de defeitos em uma quantidade de tempo e esforço mínimos. O teste, em especial, é um elemento usado para fornecer evidências da confiabilidade do software em complemento a outras atividades, sendo relevante para identificação e eliminação de defeitos que persistem no software (Pressman, 1997).

Por isso, os casos de teste desenvolvidos e submetidos ao programa têm o objetivo de produzir uma saída que esteja em desacordo com a especificação. Quando isto ocorre, diz-se que o teste foi bem sucedido (Myers, 1979). Caso defeitos não sejam detectados, o testador tem aumentada a sua confiança quanto à qualidade do software. Porém, para que isto ocorra, é preciso que os testes tenham sido realizados de maneira rigorosa e sistemática.

Umas das técnicas de teste sistemática e rigorosa é o teste funcional que preconiza o desenvolvimento de casos d e teste com o objetivo de testar aspectos importantes da especificação do sistema. Neste trabalho é apresentada a experiência de desenvolvimento e automação de execução de casos de testes no SIGI - Sistema de Informação Gerencial do *Instituto Nacional de Investigaciones Agrícolas de Venezuela* (Pedroso Júnior, 2001).

A arquitetura do SIGI pode ser chamada de cliente/servidor off-line, isto é, o cliente não opera em comunicação direta com o servidor. A parte cliente trata do planejamento, acompanhamento, reprogramação e avaliação final de projetos de pesquisa em uma base de dados local trazida do servidor. A criação de projetos,

[1] *Bsc. em Matemática Aplicada e Estatística, Pesquisador da Embrapa Informática Agropecuária, Caixa Postal 6041, Barão Geraldo – 13083-970 – Campinas, SP. (e-mail: joaof@cnptia.embrapa.br)*
[2] *Ph.D. em Pesquisa Operacional, Pesquisador da Embrapa Informática Agropecuária. (e-mail: pedroso@cnptia.embrapa.br)*
[3] *Bsc. em Ciência da Computação, Pesquisador da Embrapa Informática Agropecuária. (e-mail: visoli@cnptia.embrapa.br)*
[4] *Consultora da Embrapa Informática Agropecuária na área de testes do SIGI. (e-mail: flaviag@cnptia.embrapa.br)*
[5] *Consultor da Embrapa Informática Agropecuária na área de testes do SIGI. (e-mail: tozzi@cnptia.embrapa.br)*
[6] *Estagiária da Embrapa Informática Agropecuária na área de testes do SIGI. (e-mail: mppab@cnptia.embrapa.br)*

Embrapa

dentre outras funcionalidades, é feita diretamente no servidor através de uma aplicação Web. As máquinas clientes locais (executando sistema operacional Windows) trazem da base central projetos criados no estado inicial. Esses projetos, por sua vez, são alterados localmente e, ao final de cada etapa do ciclo de vida, reenviados ao servidor para atualização. Dessa maneira, os dados dos projetos na base local estão sempre sincronizados com a base central no servidor e disponíveis aos interessados com a devida autorização de acesso.

O processo de desenvolvimento de software adotado no SIGI é baseado no paradigma *eXtreme Programming* (Beck, 2000). Do ponto de vista de qualidade, neste paradigma, o teste sistemático é uma atividade essencial porque ocorre repetidamente a cada alteração no sistema. Isso envolve altos custos em relação a recursos humanos e tempo dispendido na atividade de teste que, mesmo com o auxílio de ferramentas, já não é pequeno. Para facilitar esse processo, foi utilizada no desenvolvimento do SIGI a ferramenta Vermont HighTest (Vermont Creative Software, 2001) que automatiza a execução dos testes disparando scripts de eventos.

O trabalho a seguir apresenta as diretrizes para a elaboração de casos de teste adotadas no SIGI, bem como a experiência na utilização da ferramenta de automação de testes.

Desenvolvimento de Testes Funcionais

Técnicas de teste

As técnicas de teste dividem-se em dois níveis: teste estrutural (caixa branca) e teste funcional (caixa preta).

Os testes estruturais focalizam a implementação do programa (código, funções, procedimentos e fluxo de dados e controle). Os casos de testes podem ser elaborados, por exemplo, para garantir que a maior abrangência possível das instruções do programa tenham sido exercitadas pelo menos uma vez.

Os testes funcionais estabelecem os requisitos de teste com base na especificação do software, não levando em consideração o funcionamento interno do programa. Os casos de teste derivados com o teste funcional visam detectar defeitos dos seguintes tipos: funções incorretas, defeitos de interface, defeitos em estrutura de dados, defeitos de acesso às bases de dados, problemas de desempenho e defeitos de iniciação e de finalização. Esse processo não necessita dos códigos fonte do programa, apenas de seu executável.

Os testes de regressão têm o papel de verificador das atualizações de uma versão para a outra do software, servindo para validar uma correção ou verificar um possível problema por conseqüência de uma alteração. Nesse caso, os casos de teste (estruturais ou funcionais) devem se repetir para que a nova versão seja adequadamente testada e para garantir que as partes do software que permaneceram inalteradas continuem funcionando dentro das especificações (Crespo et al., 2001).

A seguir, é descrita a metodologia para desenvolvimento de casos de teste funcionais no SIGI.

Casos de teste

Um caso de teste é composto por um conjunto de entradas, condições de execução e os resultados esperados, tendo como objetivo verificar os requisitos especificados do sistema.

Os casos de testes criados para o SIGI tiveram como base os casos de uso, uma técnica de modelagem utilizada para descrever o que um sistema deve fazer, especificar o contexto, capturar os requisitos, validar a arquitetura e direcionar a implementação (Heap & Richards, 2000).

Para a elaboração dos casos de teste a partir dos casos de uso deve-se considerar o seguinte:

- identificar todos os cenários contidos nos casos de uso;
- para cada cenário, identificar um ou mais casos de teste;
- para cada caso de teste, identificar condições execução;
- adicionar os dados para as condições nos casos de teste.

A criação de casos de teste foi baseada no método *Step-by-Step* (Iberle, 2000). O objetivo principal do método *Step-by-Step* é produzir rapidamente casos de teste completos para especificações individuais do sistema. Resumidamente consiste das seguintes passos:

1. Listar os requisitos de teste baseado nas especificações do sistema: identificar os requisitos óbvios do teste lembrando que os requisitos não são indicações exatas das entradas e saídas mas sim idéias do que deve ser testado.
2. Adicionar requisitos de teste para cobrir adequadamente o domínio de cada entrada: consiste em testar valores médios, limites e que extrapolem

as condições limites, entradas ilegais e condições de erro.

3. Listar um caso de teste para cada requisito de teste: pensar nos vários requisitos de teste em várias perspectivas de forma a detectar diferentes tipos de defeitos. Essa abordagem ajuda a testar a confiabilidade do sistema sob o uso constante.
4. Revisar os casos de teste completando o que for necessário: listar todos os casos de teste verificando a aplicabilidade a cada especificação do sistema.
5. Escrever um caso de teste: para cada teste considerar as entradas e saídas, entradas especiais, a configuração necessária para execução e definir como o testador saberá se o teste passou ou falhou.
6. Agrupar os casos de teste em scritps de teste: neste passo é possível ser mais eficiente agrupando os casos de testes com entradas e configurações comuns num mesmo script. Além disso, deve-se planejar os scripts de forma que sejam executados em no máximo 3 horas. O script pode ter sua execução automatizada conforme discussão na Seção Ferramenta para Automação de Testes.

A seguir é apresentado o exemplo de parte do script de teste **Fazer Planejamento de Projeto** (*Hacer Planificación de Proyecto*) do SIGI que testa vários requisitos de teste:

Descrição: edição das informações relacionadas com o planejamento do projeto pelo coordenador na base de dados local. A notificação da disponibilidade dos dados recentemente editados para os demais componentes da equipe envolvidos com o projeto ocorrerá somente depois de feita a sincronização com a base de dados central com os dados atualizados na base de dados local.

Tipo: funcional.

Pré-condição: o script de teste Descarregar Projeto (*Descargar Proyecto*) tenha sido executado com sucesso.

Configuração: sistema operacional: Windows 2000; Pentium III, HD 20 GB, 128 MB de RAM; área de tela High Color (16 bits) 800 por 600 pixels.

Testador: Flávia Gobet de Aguiar.

Script: resumen.inb.

Data da elaboração: 06/12/2001.

Plano de teste: realizar a validação e verificação dos requisitos da especificação do sistema apresentados no caso de uso *Hacer Planificación de Proyecto*. Os casos de teste testam entradas e saídas válidas e inválidas para os campos das interfaces *Resumen y palabras claves*, *Justificación*, *Orientación de la investigación*, *Objetivos*, *Metas e indicadores*, *Viabilidad económica*, *Impactos previstos*, *Clientes e Referencias bibliográficas*.

Requisitos de teste:

1. Testar entrada válida no campo *Resumen*: o campo tem que possuir algum conteúdo.
2. Testar entrada inválida no campo *Resumen*: o campo não possui conteúdo.
3. Testar entrada válida no campo *palabras claves*: incluir de 1 a 5 itens.
4. Testar entrada inválida no campo *palavras claves*: não incluir nenhum item ou mais que 5.

O script a seguir inclui casos de teste para testar os requisitos 1 e 3 citados. Ele começa quando o coordenador, utilizando a interface de acesso a base de dados local, seleciona um dos projetos da área de Negociação Tecnológica da lista que tem permissão para edição.

Acesso do coordenador do projeto:

- **Opção**: *Hacer planificación del proyecto*

Projeto: ProyectoNT.pry

Título: Organizacao, recuperacao e disponibilizacao de informacao tecnico-cientifica em bibliotecas - Desenvolvimento do AINFO

Coordenador: João Francisco Gonçalves Antunes

- **Entrada**: o coordenador acessará a base de dados local e receberá uma lista com seus respectivos projetos.
- **Resultado esperado**: projeto ProyectoNT.pry selecionado e aberto.
- **Entradas**:

  Campo *Resumen*: "Por determinacao da Diretoria Executiva, o software AINFO foi implantado na Embrapa no final de 1991, tendo por finalidades a organizacao, a recuperacao e a disseminacao da informacao tecnico-cientifica gerada e/ou adquirida pela Empresa. Inicialmente, os esforcos foram concentrados na formacao das bases de dados bibliograficas: Acervo Documental - livros, folhetos, separatas, artigos de periodicos, trabalhos apresentados em congressos, software, fitas de video, teses e outros materiais especiais - e Colecao de Periodicos. Posteriormente, com a implementacao do relacionamento entre bases de

dados para automacao das rotinas de emprestimo, foram iniciados os trabalhos com as bases de dados de Instituicoes de Recursos Humanos. Automatizou-se, tambem, o controle de processos bibliograficos Aquisicao, Comutacao e Servico de SDI, com a transferencia automatica de dados entre as bases. ..."

Campo *Palabras claves*: *Palabra 1*, *Palabra 2*.

- **Resultados esperados**: dados inseridos nos respectivos campos e o botão de inserção de *Palabras claves* estará habilitado.

Ferramenta para Automação de Testes

A ferramenta utilizada para automação dos casos de teste do SIGI é a Vermont HighTest Plus 3.50, da Vermont Creative Software (2001) que auxilia na execução de testes de aplicações Windows e simplifica tarefas repetitivas. É uma ferramenta da classe **captura** e **reproduz** (*capture and playback*), isto é, grava os eventos gerados pelo testador através dos dispositivos de entrada e saída e gera um script que pode ser reproduzido inúmeras vezes.

A Vermont oferece as seguintes funcionalidades:

- gravação orientada a objetos e reprodução dos eventos especificados, tais como seleção de menus, movimentos do Windows, cliques de botões e eventos do mouse;
- captura e compara imagens, funções internas e controles do Windows e conteúdo de arquivos;
- mascara áreas de imagens ou arquivos que se deseja excluir da comparação;
- construção e adaptação dos scripts de testes através de uma linguagem própria de programação;
- criação de uma coleção organizada de scripts de teste através de um gerenciador de conjunto de scripts.

Por exemplo, através desses recursos fornecidos pela ferramenta, é possível popular bases de dados sem ter que digitar todo o conteúdo dos campos, testar o tamanho limite de campos, comparar componentes de interface e executar testes de regressão.

A Vermont foi escolhida para automação dos testes do SIGI porque é uma ferramenta de fácil uso e de baixo custo (U$195,00). Além disso, não é necessário ser um programador experiente para usá-la efetivamente. A versão atual pode ser executada no Windows 9x, NT, Me e 2000. O suporte pode ser obtido através do site http://www.vtsoft.com onde existe um fórum de discussão, uma base de conhecimento sobre anotações técnicas e uma área de atualizações da ferramenta.

A utilização da Vermont não precisa aguardar que todo o desenvolvimento do software esteja finalizado. Para cada módulo liberado em que existem funcionalidades a serem testadas, é possível elaborar um script e executá-lo. Hoje, chegando praticamente ao final do desenvolvimento do SIGI com praticamente todos os módulos implementados, é possível executar automaticamente scripts que testam todo o ciclo de vida de um projeto de pesquisa, cobrindo a fase de planejamento, acompanhamento, reprogramação e avaliação final.

Diretrizes para utilização da Vermont

A utilização da Vermont é dividida basicamente em:

- Criação dos scripts: a primeira coisa a fazer é indicar o nome do programa executável que está sendo testado. Depois disso, o testador baseado no caso de teste cria os scripts de eventos através da execução do sistema a partir da Vermont, determinando objetos de interface (menus, janelas, botões, etc.) que devem ser comparados e o caminho a ser seguido através das telas, baseado no caso de teste.
- Reprodução dos scripts: para repetir o teste, o script criado anteriormente deve ser reproduzido. Os dados de entrada durante a gravação servem como referência na fase de reprodução. Nesta fase, os objetos de interface capturados são comparados com os do software ativo e se houver alguma diferença na interface, com relação a posição, fontes, cores, etc., a execução da Vermont é interrompida imediatamente mostrando a linha do script onde ocorreu o problema.
- Scripts genéricos: o script gerado pode ser alterado através de comandos que a ferramenta oferece para criar casos de teste mais complexos. Por exemplo, se é necessário inserir 100 cadastros, então gera-se um script que insere um 1 cadastro e depois, através do comando *FOR*, altera-se o código do script para cadastrar os outros 100. Dessa forma, os scripts podem ser alterados de acordo com a necessidade para testar a confiabilidade do sistema.
- Scripts para testar defeitos complexos: no SIGI, a Vermont também é utilizada para verificar se defeitos complexos de difícil reprodução foram corrigidos. Nestas situações, o script de eventos que detecta o defeito é capturado utilizando a ferramenta. Quando a próxima versão com os

defeitos corrigidos é liberada, este script é reproduzido. O resultado obtido deverá ser diferente do esperado porque a comparação da tela na qual aparecia o problema deve ser diferente da tela de referência previamente gravada. A partir disso, o defeito específico é considerado corrigido somente com a posterior validação visual do testador para certificar que o fluxo correto do caso de teste foi finalizado.

Problemas e soluções

Durante a utilização da Vermont no processo de automação de testes do SIGI surgiram alguns problemas que foram contornados da seguinte maneira:

Identificadores de janelas: como a Vermont funciona com a captura de eventos, cada janela capturada tem um identificador. Quando um script que utiliza a função *WaitWindow* é reproduzido, os identificadores das janelas novamente gerados são diferentes dos identificadores das janelas já gravadas. Isso acarreta um problema de conflito na identificação da janela referente a tela que está sendo testada. Para evitar esse problema, é necessário ignorar os identificadores durante a comparação. Isso deve ser configurado desabilitando a opção *Commands IDs* nas opções de execução.

Caracteres acentuados: a Vermont não suporta caracteres acentuados. Por isso, o conteúdo de teste de campos não pode ter acentuação. Também existem alguns caracteres como as aspas (") que são reservados da linguagem de programação e não podem ser utilizados.

Aparência do Windows: um dos maiores problemas encontrados na Vermont é a diferença que a configuração de aparência entre as diversas versões do Windows provoca na execução de um script. Um script que faça a comparação de telas gerado no Windows 2000 não poderá ser executado diretamente no Windows 9x ou NT devido a diferenças de cores e fonte, provocando diferenças nas imagens capturadas.

Para minimizar este problema, foi criada a seguinte configuração padrão para a execução da Vermont que deve ser aplicada às diversas versões do Windows antes da reprodução dos scripts:

Item	Tamanho	Cor	Cor 2	Fonte	Tamanho	Cor	Negrito
Menu	18	cinza	---	MS Sans Serif	8	preto	–-
Janela ativa	18	azul*	azul *	MS Sans Serif	8	branca	X
Janela inativa	18	cinza*	cinza *	MS Sans Serif	8	cinza claro ***	X
Caixa de mensagem	---	–-	---	MS Sans Serif	8	preta	–-
Texto da janela	---	branca	---	---	–-	preta	–-
Objetos 3D	---	cinza	---	---	–-	preta	–-
Botões legenda	18	–-	---	---	–-	---	–-
Bordas	1	cinza	---	---	–-	---	–-

* **azul**

Matiz	149	Verm	10
Sat	199	Verde	36
Lum	55	Azul	106

** **cinza**

Matiz	160	Verm	128
Sat	0	Verde	128
Lum	120	Azul	128

*** **cinza claro**

Matiz	27	Verm	212
Sat	29	Verde	208
Lum	194	Azul	200

A Fig. 1 apresenta o exemplo de parte do script gerado a partir do caso de teste Fazer Planejamento de Projeto (*Hacer Planificación de Proyecto*), na qual todos os eventos foram gravados no momento em que o SIGI foi executado.

A tela mostrada na Fig. 2 foi capturada durante a gravação e servirá como referência no momento em que o script for reproduzido novamente. Pode-se observar que existe um defeito nesta tela porque os botões *Modificar* e *Eliminar* não poderiam estar habilitados já que não existe nenhum registro na grade para ser editado. Os retângulos apontados pelas setas destacam onde vão ocorrer as diferenças.

Quando a reprodução estiver em curso, a Vermont faz a comparação da tela atual com a tela de referência capturada na gravação do script, conforme mostrado na Fig. 3:

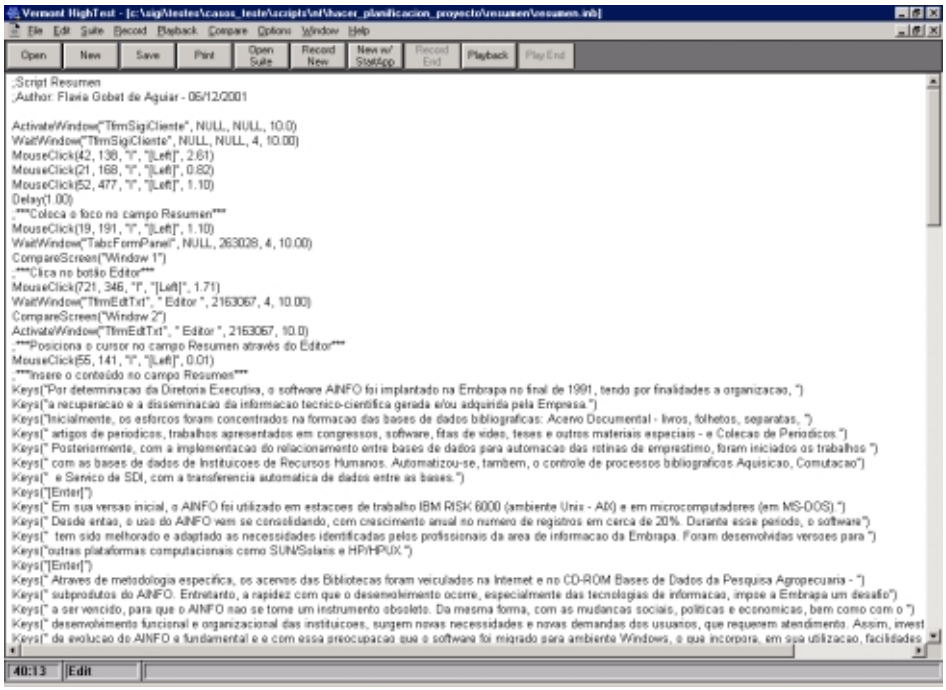

Fig.1. Script com os comandos da linguagem de programação gerados pela Vermont.

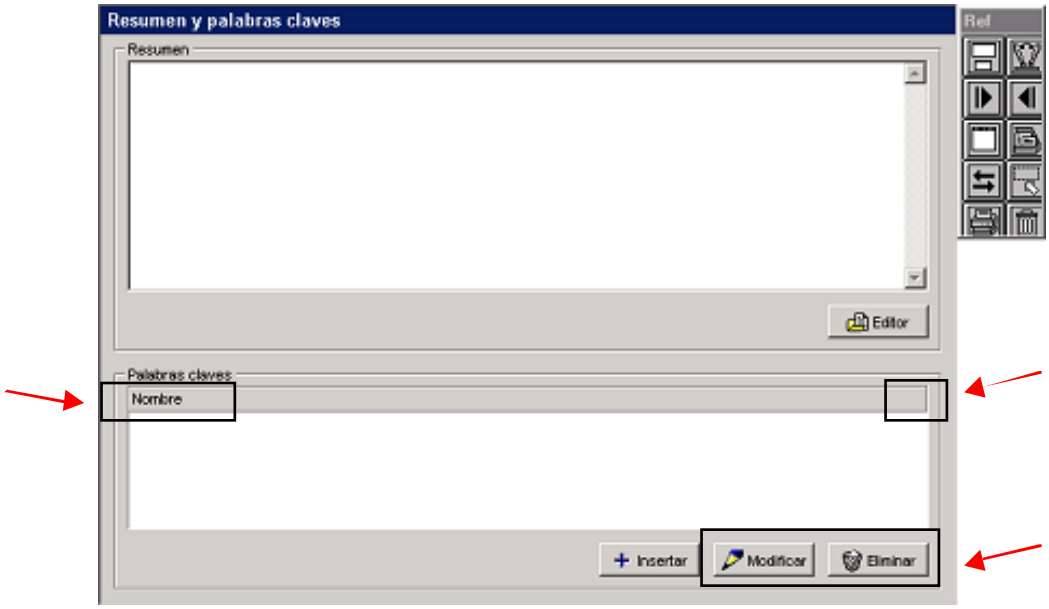

Fig. 2. Tela capturada durante a gravação do script.

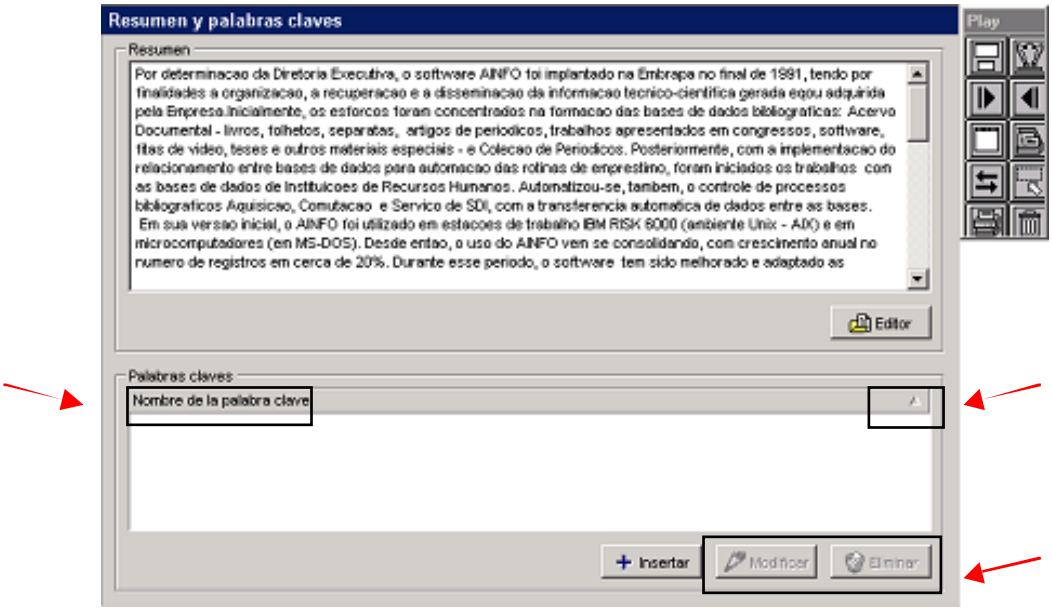

Fig. 3. Tela capturada na reprodução do script.

Assim, com as duas imagens podemos ver onde aparecem as diferenças de interface de uma versão para outra, conforme mostrado na Fig. 4.

Conforme detectado pela Vermont, pode-se observar que o rótulo do campo *Palabras Claves* foi alterado, foi acrescentado um componente de ordenação na barra de títulos da grade e que o defeito referente a habilitação dos botões *Modificar* e *Eliminar* foi corrigido.

A diferença mostrada no campo *Resumen* refere-se aos dados inseridos pela Vermont. Entretanto, como é um campo textual livre, interessa apenas testar os limites mínimo e máximo e não os seus valores propriamente ditos. Campos desse tipo recebem uma máscara durante a captura da imagem para que o seu conteúdo seja excluído da comparação.

Fig. 4. Diferenças entre os componentes da tela.

Conclusões e Comentários

A atividade de teste é complexa porque software é complexo, intangível e altamente modificável. É uma atividade que exige planejamento, projeto, execução, acompanhamento, integração com outras áreas e recursos como equipe, processo, treinamento e ferramentas adequadas. Por isso, é responsável pela maior porcentagem de esforço técnico no processo de desenvolvimento do software.

No SIGI, os casos de teste foram derivados com a técnica funcional utilizando o método *Step-by-Step* baseado em casos de uso. A execução automática dos casos de teste foi realizada com auxílio da ferramenta Vermont. Apesar das suas particularidades, a Vermont está sendo de grande utilidade para realização dos testes, não só para parte cliente em Windows como para a parte servidor Web. Além disso, ela ainda foi utilizada para auxiliar na verificação dos padrões de interface estabelecidos no desenvolvimento como parte da garantia de qualidade do software e para a execução dos testes de regressão que devem ser repetidos após as modificações no sistema.

A partir da experiência no SIGI, observou-se que normalmente criar, verificar e documentar um caso de teste leva de três a dez vezes o tempo necessário para executar o caso de teste correspondente manualmente. A manutenção dos scripts é outro ponto a ser observado porque, quando a interface gráfica é modificada, ou será necessário criar um novo caso de teste ou refazer o script de eventos.

Finalmente, a automação de teste exige boa experiência da equipe, adquirida com muito tempo e esforço. O gerente deve manter as expectativas da equipe sob controle porque os resultados demoram um pouco a aparecer. Geralmente são obtidos na próxima versão do software que está sendo testado.

Agradecimentos

Os autores apresentam seus especiais agradecimentos a Roseane Marcílio de Freitas que trabalhou como consultora no treinamento da equipe de testes na utilização ferramenta Vermont. Destacamos, também, a harmoniosa integração entre as equipes o que ajudou bastante no alcance dos resultados relatados neste trabalho.

Referências Bibliográficas

BECK, K. *Extreme programming explained*: embrace change. Boston: Addison Wesley, 2000. 224 p.

CRESPO, A. N.; FANTINATO, M.; MARTINEZ, M. R. M.; JINO, M.; ARGOLLO JÚNIOR, M. de T. e. *Teste de software*. [Campinas: Instituto Nacional de Tecnologia da Informação, 2001]. 58 transparências.

HEAP, K.; RICHARDS, T. *Object-oriented analysis and design using the Unified Modeling Language*: student guide. Redwood Shores: Oracle Corporation, 2000. Paginação irregular. (Oracle University).

IBERLE, K. A. Step-by-step test design: how to quickly produce thorough tests for individual product features. *The Software Testing & Quality Enginering Magazin*e, v. 2, n. 5, Sept./Oct. 2000.

MYERS, G. J. *The art of software testing*. New York: John Wiley Professional, 1979. 192 p. (Business Data Processing: a Wiley Series).

PEDROSO JÚNIOR, M. *Sistema de Informação Gerencial de Projetos de Pesquisa Agropecuária para o Instituto Nacional de Investigaciones Agrícolas da Venezuela - SIGI*. Campinas: Embrapa Informática Agropecuária, 2001. (Embrapa. Programa 12 – Automação Agropecuária. Projeto 12.2001.950). Projeto em andamento.

PRESSMAN, R. S. *Software engineering*: a practitioner's approach. 4th ed. New York: McGraw-Hill, 1997. 852 p.

ROCHA, A. R. C.; MALDONADO, J. C.; WEBER, K. C. *Qualidade de software*: teoria e prática. São Paulo: Makron, 2001. 320 p.

VERMONT CREATIVE SOFTWARE. *Vermont high test plus*: version 3.50 for Windows – the comprehensive testing tool that´s easy to use. Disponível em: <http://www.vtsoft.com>. Acesso em: 10 dez. 2001.

Comunicado Técnico, 14

MINISTÉRIO DA AGRICULTURA, PECUÁRIA E ABASTECIMENTO

GOVERNO FEDERAL
Trabalhando em todo o Brasil

Embrapa Informática Agropecuária
Área de Comunicação e Negócios
Av. Dr. André Tosello s/nº
Cidade Universitária - "Zeferino Vaz"
Barão Geraldo - Caixa Postal 6041
13083-970 - Campinas, SP
Telefone/Fax: (19) 3789-5743
E-mail: sac@cnptia.embrapa.br

1ª edição