Trabalho de Conclusão de Curso

Protótipo para implementação de teste de software segundo as recomendações da ABNT

Aluno:

Marcio Tomelin

Orientador:

Paulo César Rodacki Gomes

Roteiro da apresentação

1. Introdução
    1.1. Objetivos
2. Teste de Software
    2.1 Objetivos e Limites do Teste de Software
    2.2 Metodologias de teste de Software
    2.3 Baterias de Teste
    2.4 Atividades de Teste de Software
    2.5 Técnicas de Teste de Software
    2.6 Casos de Teste
    2.7 Norma Brasileira ABNT NBR 12207
    2.8 A Ferramenta *Visual Test*

Roteiro da apresentação(Cont.)

3. Desenvolvimento
4. Conclusões
   - 4.1 Limitações
   - 4.2 Extensões

1. Introdução

Através deste trabalho de pesquisa pretende-se explicar como o software pode ser testado com eficiência. E de que maneira a automatização dos testes de software, pode contribuir para a qualidade final do produto.

1.1 Objetivos

- Principal
  - Implementar uma solução automatizada para a atividade de teste de software
- Específicos
  - Identificar como vem sendo realizada esta atividade em uma software house de Blumenau, e fazer uma avaliação.

2. Teste de Software

Processo de executar um programa com o objetivo de revelar a presença de defeitos.

"Os testes não provam que um programa está correto, somente contribuem para aumentar a confiança de que o software desempenha as funções especificadas." (Souza, 1996)

2.1 Objetivos e limites do Teste de Software

Descobrir sistematicamente diferentes classes de erros com uma quantidade de tempo e esforço mínimos.

2.2 Metodologias de teste de software

Existem basicamente duas maneiras de se construir os testes:

- Método da Caixa Branca
- Método da Caixa Preta

2.3 Baterias de Teste

Segundo o IEEE, os principais tipos de baterias de teste são:

- Teste de Unidade
- Testes de Regressão
- Testes de Integração
- Testes de Aceitação

2.4 Atividades de Teste de Software

O processo de testes tem dois grandes grupos de atividades para cada bateria de testes: **preparação** e **realização** dos testes.

2.4 Atividades de Teste de Software (cont.)

	Entradas
Testes de Aceitação	Especificação dos requisitos Plano de desenvolvimento Plano da qualidade
Testes de Integração	Descrição do Projeto (projeto de alto nível) Descrição dos Testes (testes de aceitação)
Testes de Unidade	Descrição do Projeto (projeto detalhado) Descrição dos Testes (testes de integração) Código Fonte da Unidade

2.4.1 Planejamento dos Testes

A atividade de planejamento foi dividida em duas tarefas separadas, **Planejamento Inicial** e **Identificação dos Itens a Testar**.

Item	Descrição
Identificar o Plano de Testes	Identificador único para o Plano
Introdução	Objetivos, Histórico, escopo, referências a documentos
Itens a testar	Conjunto dos itens coberto pelo plano
Aspectos a testar	Conteúdo dos testes que serão feitos
Aspectos que não serão testados	Aspectos significativos que não serão testados
Abordagem	Opções metodológicas que são aplicáveis ao conjunto de testes
Critérios de completeza e sucesso	Condições que devem ser satisfeitas e resultados que devem ser atingidos para que o conjunto de testes seja considerado bem sucedido
Resultado do teste	Artefatos que serão produzidos durante a realização da baterias de testes
Tarefas de teste	Lista detalhada das tarefas que serão cobertas por este plano
Ambiente	Hardware e Software utilizados para o conjunto de teste
Responsabilidades	Responsabilidades de cada participante da equipe de teste
Agenda	Data de início e fim para cada plano de teste
Riscos e contingências	Riscos e contingências aplicáveis ao plano de testes
Aprovação	Nome e assinatura dos responsáveis pela criação do plano

2.4.2 Desenho de testes

É a especificação completa desta atividade.

Especificações dos testes

Detalhes dos testes

Desenho de procedimentos de teste

Seqüência de passos necessárias para ser executar uma variação de teste.

Revisão do caso de teste

Grande quantidade de informação.

2.4.3 Implementação dos testes

Descrição	Implementação dos testes
Insumos	Plano e Especificações dos testes Recursos para os testes Itens a testar Ferramentas de teste Dados de atividades anteriores, se houver Estruturas provisórias de testes anteriores, se houver
Atividades	Configurar ambientes de teste Disponibilizar os recursos necessários Instalar os itens a testar, estruturas e ferramentas
Resultados	Itens a testar, instalado e configurados Ferramentas e estruturas provisórias instaladas e configuradas

2.4.4 Execução dos testes

Esta atividade executa os testes da bateria, produzindo os relatórios correspondentes. Destas, podem ser solicitadas correções dos itens, assim como alterações nos planos de teste.

2.4.5 Verificação do término dos testes

Determina se estão satisfeitas as condições para completeza e sucesso dos testes.

2.4.6 Balanço Final

Nesta atividade são registradas as lições aprendidas, bem como, se possível, são identificados artefatos reutilizáveis em outros testes.

2.5 técnicas de teste de software

- **Testes de integração**
- **Testes de aceitação**
  - ***Testes Funcionais***
    - Análise do valor limite
    - Testes de comparação
    - Testes de tempo real
  - ***Testes Não Funcionais***
    - a seguir, o quadro demonstra os tipos de testes funcionais.
- **Testes de regressão**

2.5 técnicas de teste de software (cont.)

Tipos de requisitos	Tipos de teste
Desempenho	Números de terminais suportados Números de usuários simultâneos Volume de informação que deve ser tratado.
Dados Persistentes	Freqüência de uso Restrições de acesso Restrições de integridade Requisitos de guarda e retenção de dados
Outros atributos de qualidade	Funcionalidade Confiabilidade Usabilidade Manutenibilidade Portabilidade

2.6 Casos de teste

Casos de teste são um conjunto específico de dados de teste juntamente com os resultados esperados.

2.6 Casos de teste(cont.)

É conveniente classificar os casos de teste em categorias bem definidas, como é mostrado abaixo:

- Casos Funcionais
- Casos Estruturais
- Casos de dados
- Casos Aleatórios
- Casos Extraídos
- Casos Anormais ou Extremos

2.6.1 Identificação de Casos de teste

Os casos de teste são descritos por dois elementos:

- Um conjunto de ações ou valores de entrada
- Um conjunto de ações ou resultados esperados

2.7 Norma Brasileira ABNT NBR 12207

Estrutura comum para os processos de ciclo de vida de software.

Porém neste trabalho será dado enfoque apenas às partes que dizem respeito à teste de software.

2.7.1 Recomendações da norma referente à teste de software

PROCESSO DE DESENVOLVIMENTO

Este processo contém as seguintes atividades relacionadas à testes de software:

1. **Codificação e teste de software**
2. **Integração do software**
3. **Teste de qualificação do software**

PROCESSO DE DESENVOLVIMENTO(cont.)

1. Codificação e teste de software

O desenvolvedor deve:

- Desenvolver e documentar cada unidade de software e procedimentos de teste e dados para os testes
- Deve testar cada unidade
- Atualizar a documentação do usuário
- Avaliar o código do software e os resultados dos testes

PROCESSO DE DESENVOLVIMENTO(cont.)

2. Integração do software

Consiste nas seguintes tarefas para o desenvolvedor:

- Deve desenvolver um plano de integração para as unidades e componentes de software. Este deve conter requisitos de teste, dados, responsabilidades e cronograma.
- Deve integrar as unidade e componentes de software e testar estas integrações à medida que forem sendo integradas.
- Atualizar a documentação do usuário.

PROCESSO DE DESENVOLVIMENTO(cont.)

2. Integração do software(cont.)

- Deve desenvolver e documentar, um conjunto de testes, casos de testes(entradas, saídas e critérios de teste e procedimentos de testes).

- Deve avaliar o plano de integração, projeto, código, testes, resultados dos testes e documentação do usuário.

PROCESSO DE DESENVOLVIMENTO(cont.)

3. Qualificação do software

Consiste nas seguintes tarefas para o desenvolvedor:

- Deve conduzir o teste de qualificação
- Deve atualizar a documentação do usuário
- Deve avaliar o projeto, código, testes, resultados dos testes e a documentação do usuário.
- Deve apoiar auditorias

PROCESSO DE OPERAÇÃO

Teste operacional

O operador deve executar o teste operacional e, satisfazendo os critérios especificados, liberar o produto de software para uso operacional.

O operador deve garantir que o código de software e as bases de dados sejam iniciados, executados e finalizados, como descrito no plano.

PROCESSO DE APOIO AO CICLO DE VIDA

Processo de validação

Este processo determina se os requisitos e o produto final, sistema ou produto de software construído, atendem ao uso específico pretendido. Este processo consiste em duas etapas, Implementação do Processo e Validação.

2.8 A ferramenta *Visual Test*

Automação de teste de software

- Encontrar rapidamente os *Bugs;*
- Emula ações de um usuário;
- Não substitui o testador humano;
- Forte na re-execução dos testes;
- Perfeita para os testes de regressão.

2.8 A ferramenta *Visual Test*(cont.)

Interface do *Visual Test*

Possui uma plataforma semelhante ao do Developer Studio Visual C++, dividida em três partes principais:

- **Workspace**
- **Output**
- **Source File**

2.8 A ferramenta *Visual Test*(cont.)

Linguagem do Visual Test

É uma versão robusta do *Basic*

Utilitários

Gravador de Cenário(*Scenario Recorder*),

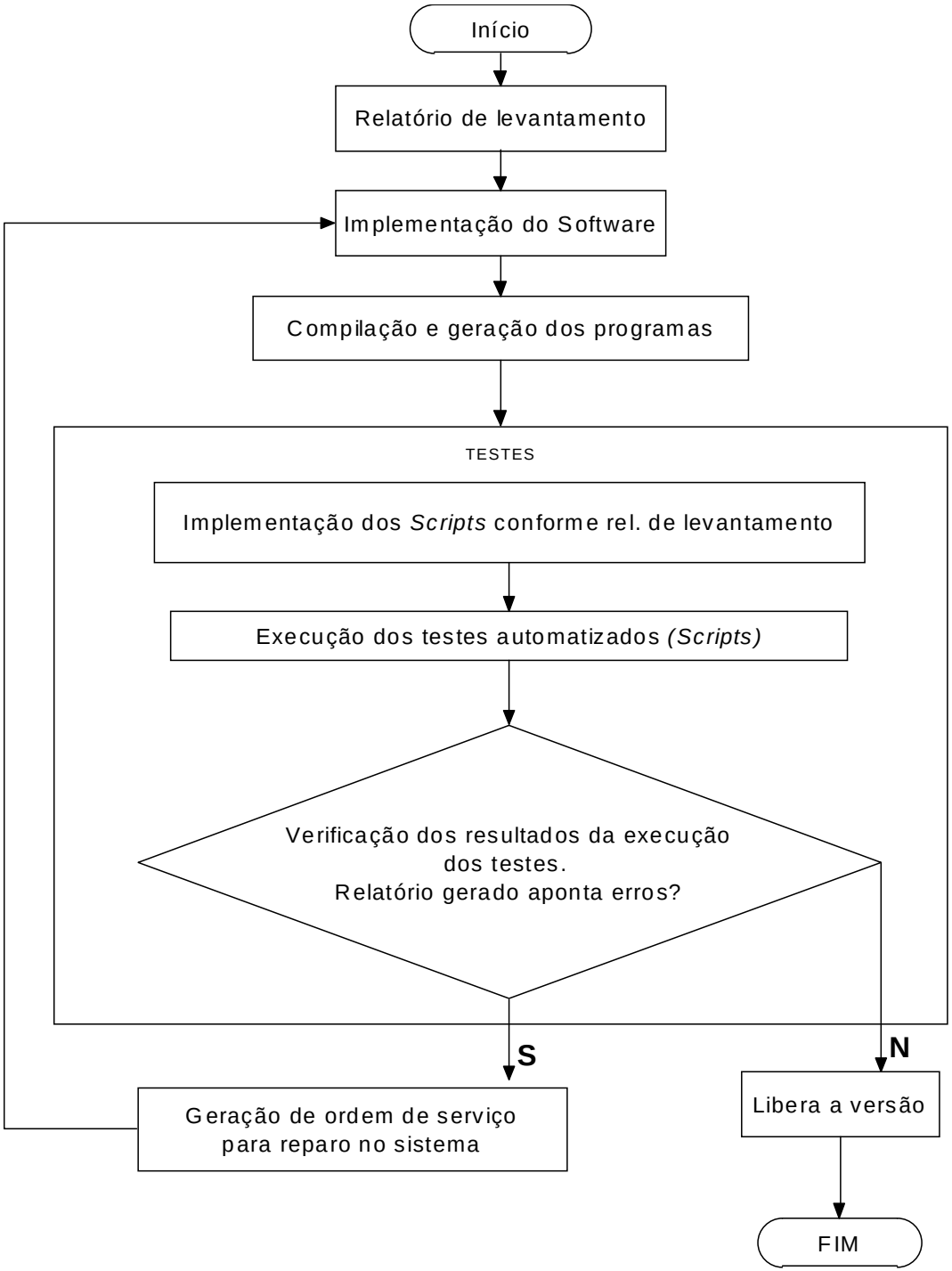
Utilitário de informações de janela(*Window Information Utility*) e

Suite Manager.

Execução dos *Scripts*

3. Desenvolvimento

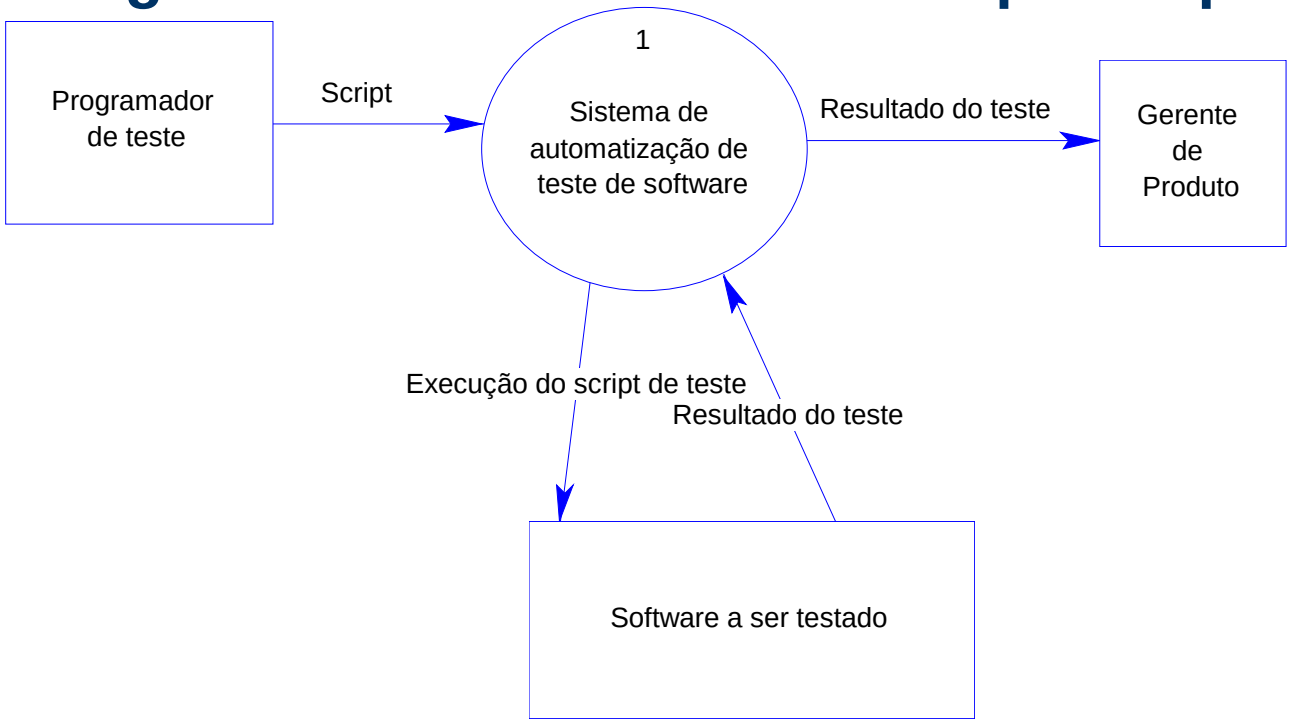
Como ponto de partida para o desenvolvimento do protótipo foi estudado as etapas de desenvolvimento de software na WK Sistemas que veremos a seguir em forma de um fluxograma.


Início
Relatório de levantamento
Implementação do Software
Compilação e geração dos programas
TESTES
Implementação dos *Scripts* conforme rel. de levantamento
Execução dos testes automatizados *(Scripts)*
Verificação dos resultados da execução dos testes.
Relatório gerado aponta erros?
S
N
Geração de ordem de serviço para reparo no sistema
Libera a versão
FIM

3. Desenvolvimento(cont.)

ESPECIFICAÇÃO

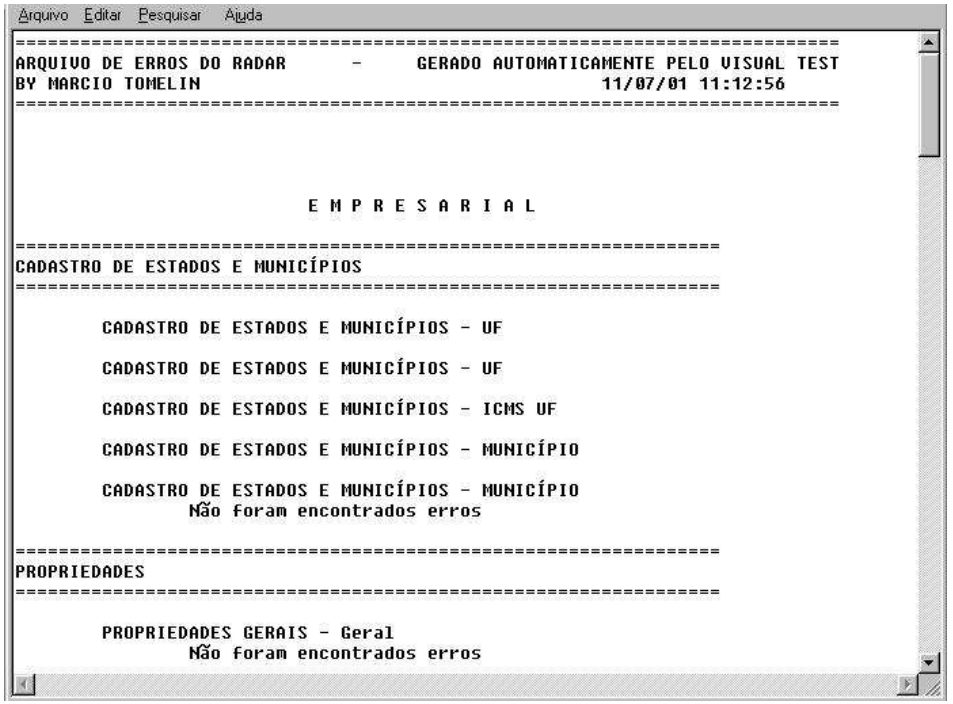
Diagrama do fluxo de dados do protótipo



3. Desenvolvimento(cont.)

ESPECIFICAÇÃO

Modelo de relatório gerado pelo protótipo durante a execução do teste.



Arquivo Editar Pesquisar Ajuda

```
================================================================================
ARQUIVO DE ERROS DO RADAR      -     GERADO AUTOMATICAMENTE PELO VISUAL TEST
BY MARCIO TOMELIN                                      11/07/01 11:12:56
================================================================================



                          E M P R E S A R I A L

================================================================
CADASTRO DE ESTADOS E MUNICÍPIOS
================================================================

       CADASTRO DE ESTADOS E MUNICÍPIOS - UF

       CADASTRO DE ESTADOS E MUNICÍPIOS - UF

       CADASTRO DE ESTADOS E MUNICÍPIOS - ICMS UF

       CADASTRO DE ESTADOS E MUNICÍPIOS - MUNICÍPIO

       CADASTRO DE ESTADOS E MUNICÍPIOS - MUNICÍPIO
              Não foram encontrados erros

================================================================
PROPRIEDADES
================================================================

       PROPRIEDADES GERAIS - Geral
              Não foram encontrados erros
```

3. Desenvolvimento(cont.)

IMPLEMENTAÇÃO

Conforme o proposto pelo trabalho a implementação baseou-se na medida do possível em normas e técnicas de teste de software estudadas neste trabalho.

Como podemos ver a seguir:

- **Testes de Caixa Branca**
- **Testes de Caixa Preta (Aceitação e Regressão)**

3. Desenvolvimento(cont.)

Codificação para o Teste de Aceitação

```
'VERIFICA SALDO DO ATIVO
WEditSetText("#1211","17") 'Conta
sleep 1
Play "{TAB}"
WEditSetText("#1212","01/01/99") 'Data Inicial
sleep 1
Play "{TAB}"
WEditSetText("#1213","31/12/01") 'Data Final
Sleep 1
Play "{TAB}"
WButtonClick("OK")
Sleep 1
IF ((StaticText("#1060") = "95.028,00") and (StaticText("#1061") = "107.724,00")) and (StaticText("#1062") = "237.304,00")
then
    certo1 = 1
else
    LogErros(Cabecalho,"O SALDO DO ATIVO NÃO CONFERE, algum lançamento não foi efetuado com  sucesso.",
Cabecalho)
```

3. Desenvolvimento(cont.)

Testes de Integração
Ordem Lógica de execução

3. Desenvolvimento(cont.)

- Testes Funcionais
  - Análise do valor limite
  - Testes de Comparação
  - Testes de Tempo Real
- Testes não funcionais
  - Desempenho
  - Dados Persistentes(Restrições de Integridade)
- Geração de Relatório com o resultado do teste
- ABNT NBR 12207
  - Processo de desenvolvimento
  - Processo de Operação
  - Processo ao Apoio ao Ciclo de Vida

3. Desenvolvimento(cont.)

Codificação para o Teste de Análise do Valor Limite

```
WMenuSelect ("&Editar\&Incluir")
WEditSetText("#1001","005")
play "{TAB}"
if EditText("#1003") = "Banco Internacionali" then ’20 caracteres
    resultado% = 1
else
    LogErros(Cabecalho,"O Campo FANTASIA não foi gravado corretamente",Cabecalho)
end if
if EditText("#1004") = "Banco Internacional S/A Pan-Americano do Desenvolv" then ’50 caracteres
    resultado% = resultado% + 1
else
    LogErros(Cabecalho,"O Campo DENOMINAÇÃO SOCIAL não foi gravado corretamente",Cabecalho)
end if
if resultado% = 2 then
   LogErros(cabecalho,"Não foram encontrados erros",Cabecalho)
   MsgBox "Todos os dados foram gravados corretamente!",MB_ICONINFORMATION,"Verificação"
Else
   VisualizaArquivo()
End If
```

3. Desenvolvimento(cont.)

Codificação para o Teste de Desempenho

```
for x = 1 to y
        for i=1 to m
                        If i=1 then mes$="01"
                        If i=2 then mes$="02"
                        If i=3 then mes$="03"
                        If i=4 then mes$="04"
                        If i=5 then mes$="05"
                        If i=6 then mes$="06"
                        If i=7 then mes$="07"
                        If i=8 then mes$="08"
                        If i=9 then mes$="09"
                        If i=10 then mes$="10"
                        If i=11 then mes$="11"
                        If i=12 then mes$="12"

                        dias$ = GeraDia()
                        WEditSetText("#1002","1") ' FILIAL
                        Sleep ENTRE
                        Play"{TAB}"
                        WEditSetFocus("#1003") 'DATA
                        Play (dias$) 'DIA
                        Play (mes$) 'MES
                        Play (ano$) 'ANO
                        Sleep ENTRE
                        Play"{TAB}"
```

3. Desenvolvimento(cont.)

Codificação para o Teste de Restrições de Integridade

```
'RESTRIÇÕES DE INTEGRIDADE
'CADASTRO DE TIPOS DE COBRANÇA
WEditSetText("#1003", "Teste Teste Teste") 'Descrição
WEditSetText("#1001", "1") 'Código
WButtonClick("OK")

Cabecalho = ("RESTRIÇÕES DE INTEGRIDADE")
LogErros(Cabecalho,"","2")

WmenuSelect("&Editar\&Incluir")
WEditSetText("#1001","1") 'Código
play "{TAB}"
Cabecalho = ("CADASTRO DE TIPOS DE COBRANÇA")
LogErros(Cabecalho,"","2")

if EditText("#1003") = "Cobrança Simples" then 'Descrição
LogErros(cabecalho,"Não foram encontrados erros",Cabecalho)
MsgBox "Todos os dados foram gravados corretamente!",MB_ICONINFORMATION,"Verificação"
else
LogErros(Cabecalho,"O Campo DESCRIÇÃO não foi gravado corretamente",Cabecalho)
VisualizaArquivo()
end if
```

3. Desenvolvimento(cont.)

Codificação para geração do relatório com o resultado do teste

```
Function LogErros(Msg1$,Msg2$,Verifica$) as String
DIM ARQUIVO%
DIM Nome, Cabecalho as String
DIM Erro as String
ARQUIVO% = FREEFILE
Cabecalho = Msg1
Erro = Msg2

if EXISTS ("D:\TCC\Resultado.TXT") then
OPEN "D:\TCC\Resultado.TXT" FOR APPEND AS #ARQUIVO

if verifica = "1" then 'quando tem que escrever o M O D U L O
PRINT #ARQUIVO, Cabecalho
end if

if verifica = "2" then 'quando tem que escrever um título
PRINT #ARQUIVO , ""
PRINT #ARQUIVO , "================================================================="
PRINT #ARQUIVO, Cabecalho
PRINT #ARQUIVO , "================================================================="
end if

if verifica = "3" then 'quando tem que escrever um título ou sub-título
PRINT #ARQUIVO , ""
PRINT #ARQUIVO, "    ",Cabecalho
end if

if Cabecalho = Verifica then 'quando tem que escrever somente o erro
PRINT #ARQUIVO, "          ",erro
end if

else
OPEN "D:\TCC\Resultado.TXT" FOR OUTPUT AS #ARQUIVO
print "nao existe arquivo"
PRINT #ARQUIVO , “=============================================================================="
PRINT #ARQUIVO , "ARQUIVO DE ERROS DO RADAR    -    GERADO AUTOMATICAMENTE PELO VISUAL TEST"
PRINT #ARQUIVO , "BY MARCIO TOMELIN                              " + DATE$ + " " + TIME$
PRINT #ARQUIVO , “==============================================================================="
PRINT #ARQUIVO, Cabecalho
PRINT #ARQUIVO, ""
Verifica = Cabecalho
end if
Close ARQUIVO
End Function
```

3. Desenvolvimento(cont.)

FERRAMENTAS UTILIZADAS

Visual Test 6.5 da Rational que já foi visto com mais detalhes anteriormente.

OPERACIONALIDADE DA IMPLEMENTAÇÃO

Foi implementado para ser aplicado ao sistema de contabilidade Radar Contábil da WK Sistemas.

3. Desenvolvimento(cont.)

RESULTADOS E DISCUSSÃO

Com o desenvolvimento deste trabalho, pode-se chegar em seu objetivo, elaborar um estudo entre as propostas de testes sugeridas pelas normas e técnicas especificadas neste projeto, com as atividades de teste de software praticadas pela WK Sistemas, conforme veremos a seguir.

3. Desenvolvimento(cont.)

Características da atividade de teste empregada hoje na WK Sistemas:

- **Testes de Regressão**
- **Testes de Aceitação (Funcionais e não funcionais)**
- **Testes de Integração**
- **Testes de Unidade**
- **Desenho de Casos de Teste**

4. Conclusões

Foi atingido o objetivo, além disso este trabalho pode servir como orientação à implantação da atividade de teste de software em empresas que atuam na área de desenvolvimento de software, de uma forma eficiente e planejada.

O protótipo funciona perfeitamente às condições que foi submetido, e ao referido sistema a ser testado para qual ele foi implementado.

4.1 Limitações

O protótipo atende apenas ao teste do sistema para qual ele foi implementado.

A ferramenta utilizada para a geração dos *Scripts,* é voltada para sistemas desenvolvidos em *Visual C++.*

4.2 Extensões

Avaliações de outras ferramentas para automatização de testes de software.

Além disso, uma possível extensão seria a aplicação dos testes propostos neste trabalho em empresas que não desenvolvem atividades de teste de uma forma organizada e metódica

FIM

OBRIGADO!